

Herausgeber

H. SCHULTE

F. HOFFMANN

R. MIKUT



Berlin | 20. – 21. November 2025

PROCEEDINGS **35. WORKSHOP** COMPUTATIONAL INTELLIGENCE

H. Schulte, F. Hoffmann, R. Mikut (Hrsg.)

Proceedings. 35. Workshop Computational Intelligence

Berlin, 20. – 21. November 2025

PROCEEDINGS **35. WORKSHOP** COMPUTATIONAL INTELLIGENCE

Berlin, 20. – 21. November 2025

Herausgegeben von

H. Schulte

F. Hoffmann

R. Mikut

Impressum



Karlsruher Institut für Technologie

Karlsruher Institut für Technologie (KIT)
Kaiserstraße 12
76131 Karlsruhe

Institut für Automation und angewandte Informatik
www.iai.kit.edu



This document – excluding parts marked otherwise, the cover, pictures and graphs – is licensed under a Creative Commons Attribution-Share Alike 4.0 International License (CC BY-SA 4.0): <https://creativecommons.org/licenses/by-sa/4.0/deed.en>



The cover page is licensed under a Creative Commons Attribution-No Derivatives 4.0 International License (CC BY-ND 4.0): <https://creativecommons.org/licenses/by-nd/4.0/deed.en>

2025

DOI 10.5445/IR/1000186052

Inhaltsverzeichnis

S. Baeuerle, P. Khanna, N. Friederich, A. J. Yamachui Sitcheu, D. Shakirov, A. Steimer, R. Mikut	1
(Karlsruhe Institute of Technology, Robert Bosch GmbH)	
Are Foundation Models Ready for Industrial Defect Recognition? A Reality Check on Real-World Data	
A. Subbotin, T. Bartz-Beielstein	23
(Technische Hochschule Köln)	
Physics-Informed Neural Networks for State Estimation Problem	
F. Wittich, F. Rezazadeh, A. Kroll	47
(University of Kassel)	
Four Benchmark Datasets for Nonlinear Regression in Engineering Sciences	
D. Hutter, T. Steinberger, M. Hellwig	65
(Fachhochschule Vorarlberg)	
Investigation of convergence behavior and solution reliability of a Genetic Algorithm for Flexible Job-shop Scheduling	
V. Katter, M. Kohlhasse	89
(Hochschule Bielefeld)	
Efficient Gait Analysis using Knowledge Distillation from Sparse Sensors	

M. Dockhorn, M. Kohlhase	97
(Hochschule Bielefeld)	
Discrepancy Modeling for Dynamical Systems	
O. Kolomenko, R. Knauer, E. Rodner	103
(University of Applied Sciences Berlin)	
Embedding World Knowledge into Tabular Models: Towards Best Practices for Embedding Pipeline Design	
A. Hinterleitner, C. Leitenmeier, S. Spengler, T. Bartz-Beielstein .	121
(TH Köln, Everllence SE)	
Tuning for Explainability: Incorporating XAI Consistency into Multi-Objective Hyperparameter Optimization	
Th. Lautenschlager, N. Friederich, A. J. Y. Sitcheu, K. Nau, G. Hayot, Th. Dickmeis, R. Mikut	131
(Karlsruhe Institute of Technology)	
Self-Supervised Learning Strategies for a Platform to Test the Toxicity of New Chemicals and Materials	
T. Walter, S. Oster, S. Altenburg	159
(Bundesanstalt für Materialforschung und -prüfung)	
Multispektrale Optische Tomografie trifft Machine Learning: Ein neuer Ansatz zur Defektvorhersage im PBF-LB/M-Prozess	
T. Niendorf, A. Engelhardt, M. Kahl, J. Richter, P. Krooß, T. Wegener, A. Kroll	169
(Universität Kassel)	
Modellgestützte Identifikation von Prozessparametern für die Additive Fertigung von Metallen	
M. Zehner, A. Cavaterra, S. Lambeck	175
(Hochschule Fulda)	
Direkte datenbasierte prädiktive Regelung nichtlinearer Prozesse unter Verwendung von Takagi-Sugeno-Fuzzy-Systemen	

F. Dietrich, H. Schulte	195
(University of Applied Sciences HTW Berlin)	
Large-Signal Takagi-Sugeno Sliding Mode Observer for Reconstruction of Parameter Changes in DC-DC Converters	
H. Schulte	223
(HTW Berlin)	
Adaptive Model Reference Control for Multivariable Takagi-Sugeno Fuzzy Systems	
J. Powell, H.Klein, O.Nelles	229
(Universität Siegen)	
Space-Filling Regularization for Modeling Hydraulic Excavator with Nonlinear State Space Models	
G. Andonovski, I. Škrjanc	237
(University of Ljubljana)	
Direct Model Reference Adaptive Control – Practical Examples with Evolving Controller	
K. Lange, A. Kroll	259
(Universität Kassel)	
Zur Multi-Start-Optimierung mit vorzeitiger Terminierung zur effizienten Lösung nichtlinearer Regressionsprobleme mittels MLP-Netzen	
Ž. Stržinar, B. Pregelj, I. Škrjanc	281
(University of Ljubljana)	
Evolving Clustering of Time Series for Unsupervised Analysis of Industrial Data Streams	
A. Kreusel, M. Hertel, M. Noskiewicz, H. Maaß, R. Mikut, V. Hagenmeyer	301
(Karlsruhe Institute of Technology)	
Evaluating Time-Series Foundation Models for Cooling Demand Forecasting with Little Data	

G. Ganesh Prasad, D. Hosurkar, H. Renz, F. Hoffmann, T. Bertram	323
(Technische Universität Dortmund)	
Vision-Language Model Embodiment for Robot Task Decomposition and Zero-Shot Control via Prompt Engineering	
D. Hosurkar, G. Ganesh Prasad, H. Renz, F. Hoffmann, T. Bertram	343
(Technische Universität Dortmund)	
An Investigative Study of Transformer-Based Generalist Robot Poli- cies on Physical and Simulated Robotic Embodiments	

Are Foundation Models Ready for Industrial Defect Recognition? A Reality Check on Real-World Data

Simon Baeuerle^{1,3,*}, Pratik Khanna^{1,3,*}, Nils Friederich^{1,2},
Angelo Jovin Yamachui Sitchou¹, Damir Shakirov⁴,
Andreas Steimer⁴, Ralf Mikut¹

¹ Institute for Automation and Applied Informatics (IAI)
Karlsruhe Institute of Technology

² Institute of Biological and Chemical Systems (IBCS)
Karlsruhe Institute of Technology

³ Mobility Electronics, Robert Bosch GmbH

⁴ Bosch Center for Artificial Intelligence, Robert Bosch GmbH

*shared first, e-mail: simon.baeuerle@kit.edu

Abstract

Foundation Models (FMs) have shown impressive performance on various text and image processing tasks. They can generalize across domains and datasets in a zero-shot setting. This could make them suitable for automated quality inspection during series manufacturing, where various types of images are being evaluated for many different products. Replacing tedious labeling tasks with a simple text prompt to describe anomalies and utilizing the same models across many products would save significant efforts during model setup and implementation. This is a strong advantage over supervised Artificial Intelligence (AI) models, which are trained for individual applications and require labeled training data. We test multiple recent FMs on both custom real-world industrial image data and public image data. We show that all of those models fail on our real-world data, while the very same models perform well on public benchmark

datasets.

1 Introduction

Defect pattern recognition is carried out during quality inspection in automotive series manufacturing. During image-based quality inspection, images are recorded for a very wide range of manufacturing processes. Beyond color images, advanced inspection technologies yield, e.g., depth images, x-ray images or Scanning Acoustic Tomography (SAT) images. The recording procedure is set up to capture defects that are specific to the respective manufacturing process. This ensures high product quality and prevents defective products from being delivered to customers. Sometimes, further expensive processing steps of a defective product can also be saved. Due to the high volume of manufactured parts, an automated defect recognition procedure can significantly reduce manual inspection efforts. Here, Artificial Neural Networks (ANNs) are increasingly used besides classic image processing techniques [1, 2]. However, a significant drawback of ANN-based classifiers is their reliance on large amounts of manually labeled training data. While this effort can be mitigated by unsupervised AI approaches [3], it remains a considerable hurdle. At the same time, simpler methods, such as a direct comparison to a reference image, are often insufficient. Real-world challenges like manufacturing tolerances, image registration errors, or varying brightness prevent such an approach from reliably detecting defects.

These limitations of both supervised models and simple heuristics motivate the exploration of a new class of powerful, pre-trained models. On various text and image processing tasks in other domains, FMs have recently shown impressive performance [4–6]. For example, the Segment Anything Model (SAM) and Contrastive Language-Image Pre-training (CLIP) generalize very well across many domains and datasets in a zero-shot setting, significantly reducing labeling efforts. While CLIP accepts small text inputs, Large Vision and Language Models (LVLMS) like Gemini [11] are more powerful at the simultaneous processing of image and text input. Domain experts can formulate text prompts without

significant effort. Promising ideas for prompting during quality inspection are, e.g., a description of the normal state of the product or a description of the visual or physical properties of the defects. These reduced labeling efforts, combined with the additional opportunity to integrate domain expert knowledge via text input, would enable easier scaling across several products, i.e., with significantly lower efforts for each product. The seemingly clear advantages over State Of The Art (SOTA) approaches motivate the question of how suitable FMs are for image-based quality inspection tasks. To close these gaps for industrial use cases, we analyze the applicability of various recent FMs on real-world industrial data in this work. The achieved performance is compared to the performance of the same models on a public dataset.

The main objective during quality inspection is the distinction of defective and defect-free products. A classification model can perform such an inspection task with minimal setup. However, it might not cover all desired functionalities fully: In some cases, the classification of an AI model is re-checked by a human operator. Furthermore, a high level of explainability is preferred during model monitoring. A model that outputs a full segmentation mask as opposed to only a single class makes both manual re-checking and model monitoring easier. As such, we include both classification and segmentation models in our study.

The remainder of this work is structured as follows: The following section contains an overview of related work regarding e.g. FMs and SOTA pipelines for image segmentation. The datasets that we use for benchmarking are introduced in Section 3. The setup of our experiments is described in Section 4 and respective results are shown in Section 5. Section 6 contains our interpretation of results and potential implications.

2 Related Work

Several studies have investigated FMs for computer vision. This section contains an overview of the most relevant models and studies in related applications.

Radford et al. [6] propose the CLIP approach, which jointly trains an image encoder and a text encoder. It is trained on a dataset, which consists of 400 million pairs of images and corresponding text descriptions. CLIP can perform zero-shot inference on unseen objects, i.e., it can be applied to new datasets without any finetuning. It outperforms a SOTA supervised approach on multiple datasets [6].

The SAM [5] consists of an image encoder, a prompt encoder and a mask decoder. The image encoder uses a Vision Transformer (ViT) [7], which is pre-trained using the Masked Autoencoder approach [8]. The prompt encoder encodes prompts like points, boxes, text or masks, whereas the prompt encoder for text uses the text encoder from CLIP. The mask decoder creates masks based on the image embeddings and the prompt embeddings. The SAM model is trained on the SA-1B dataset, which contains over one billion annotated masks on 11 million images. This covers a wide range of different objects, locations and scenarios. Images of people, buildings, vehicles, animals, and other elements from everyday life around the world are well represented.

Li et al. [9] propose CLIPSurgery, which introduces small adaptations to the CLIP architecture. They remove redundant features and modify the attention mechanism to link semantically similar regions better. This significantly improves the model's explainability.

Liu et al. [10] introduce GroundingDINO, which can detect objects based on text input. The feature enhancer includes cross-attention between text and image information. The language-guided query selection selects features that match the input text. The cross-modality decoder fuses the text modality with the image modality for the generation of the output regions. It is pretrained on large datasets. The prediction head outputs multiple object bounding boxes along with a text and a similarity score. A box threshold can be set to only include bounding boxes with a minimum score. A text threshold can be set to additionally filter for bounding boxes that match the given input text prompt well.

LVLMS are models that accept multiple modalities such as text, images, audio and/or video as input. They can be utilized in a wide range of applications. Gemini 2.5 Pro [11] can be directly applied to defect classification tasks, since

it can process a text prompt along with an image. It has furthermore shown strong reasoning capabilities.

Zhang et al. [12] propose different pipelines of multiple foundation models, e.g., GroundingDINO + SAM, SAM + CLIP or CLIPSurgery + SAM. GroundingDINO and CLIP can effectively capture semantic features and provide visual prompts for subsequent instance segmentation by SAM. Zhang et al. analyze the different pipelines on aerial images as taken by, e.g., an Unmanned Aerial Vehicle.

Cao et al. [13] propose the framework Segment Any Anomaly + (SAA+) for zero-shot anomaly segmentation. This includes GroundingDINO to propose abnormal regions which are then fed into a second model, such as SAM to refine the abnormal regions. They specifically study defect detection and analyze performance on the public industrial dataset *MVTec AD*.

Xu et al. [14] combine human expert knowledge with the capabilities of Visual-Language-Foundation Models. They study different prompts for the task of anomaly detection in images. This includes simple prompts that only query for any defect or anomaly and more advanced prompts that provide, e.g., more detailed information about the shown object or the expected defects.

The Text2Seg approach contains multiple of the most relevant FM models. They have been successfully tested on an image segmentation task. SAA+ follows a similar approach to Text2Seg and its authors even include the task of defect recognition into their study. As outlined above, the Gemini model is widely applicable and has shown promising capabilities. This raises high expectations towards the application of those models during industrial defect recognition. While there is a wide range of further models that would also be interesting in this context, we limit the scope of our study to these models.

3 Datasets

During this research, we utilized three different image datasets: our custom real-world industrial dataset *IndustrialSAT*, the public industrial dataset *MVTec*

AD and the general public image dataset *Oxford-IIIT-Pet*. The main focus of our study is the real-world data. The public datasets serve as a reference.

3.1 Oxford-IIIT-Pet



Figure 1: Exemplary images from the dataset *Oxford-IIIT-Pet* [15]

The dataset *Oxford-IIIT-Pet* [15] contains images of various pet animals in different everyday scenarios. The animals are recorded both outdoors and indoors in different scales, pose and lighting. Such animal images are found across the internet and are well-represented in the SA-1B dataset. During this work, only cats and dogs are utilized. The dog images are treated as defect-free images ("OK"), whereas cat images are treated as defective ("NOK"). During testing, we utilize 140 dog images and 60 cat images. Examples are shown in Figure 1.

3.2 MVTec AD

The *MVTec AD* dataset [16] is utilized during the benchmarking of various models for industrial anomaly detection [17, 18]. In this work, we analyze only object categories that are visually related to our domain of electronic packaging. This includes the categories carpet, grid, leather, tile and wood with a resulting test dataset of 253 defect-free images ("OK") and 76 defective images ("NOK").

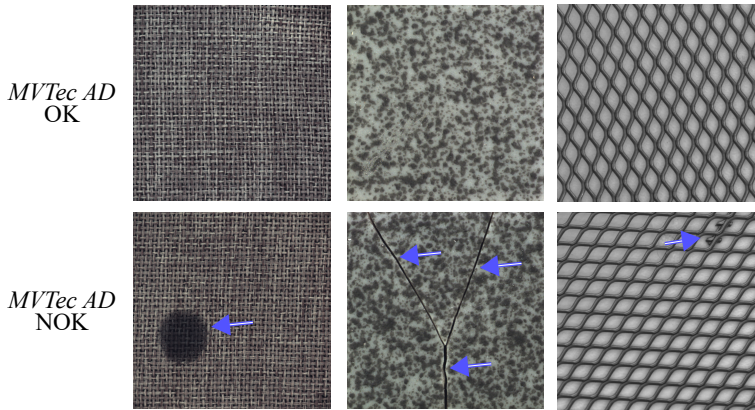


Figure 2: Exemplary images from the dataset *MVtec AD* [16] from the classes carpet (left), tile (center) and grid (right). Top row: defect-free images. Bottom row: defective images.

No distinction is made between different defect types. Examples from the categories carpet, tile and grid are shown in Figure 2.

3.3 IndustrialSAT

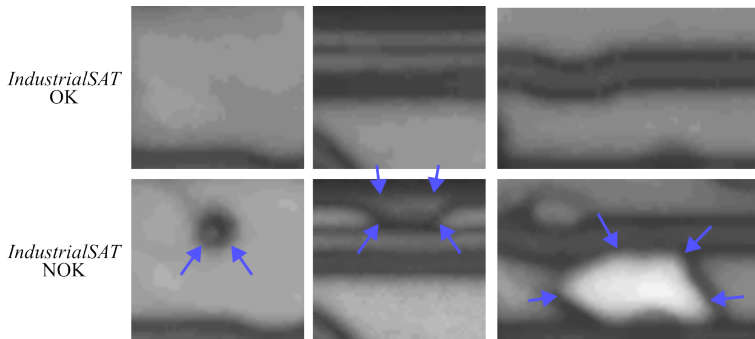


Figure 3: Exemplary images from the dataset *IndustrialSAT*. Crop of defective images (bottom row) and corresponding regions without defect (top row).

Our custom real-world industrial dataset *IndustrialSAT* is a dataset of greyscale images. They are recorded with SAT [19,20] as follows. The electronic package

is submerged in water. A toolhead with an ultrasonic sender and receiver is moved to a position above the package. An ultrasonic wave is sent out and the signal reflection is recorded. This yields a time-series of data points for this position. From the time-series, the signal value at a certain time value is extracted with special postprocessing methods. This outputs a single value for this position of the toolhead. The procedure is repeated for multiple toolhead positions. The toolhead positions correspond to the pixels of the resulting image. The extracted signal values correspond to the grey values of the resulting image. SAT is used for quality inspection during electronic packaging of products in the field of, e.g., sensors, electronic control units or power electronics. These products are highly relevant for electric and autonomous vehicles. Examples of defect types that could occur during electronic packaging are cracks, voids or delaminations between or within the different layers of an electronic package. Our test dataset, *IndustrialSAT*, contains 231 defect-free images and 32 images with one of the defect types: void, crack or delamination. Exemplary defects on SAT images are shown in the bottom row of Figure 3. The shown images are cropped down to the immediate area around the defects. The top row of Figure 3 shows the same crop position of a non-defective product. To evaluate our model, we assigned the defective label to images with defects of any size. No distinction is made with respect to different defect types during model evaluation.

During a preliminary study, we trained a classification model on the dataset *IndustrialSAT*. This was built by concatenating a pre-trained feature extractor based on a ResNet model [21] with a multi-layer perceptron. This classifier reached an F2-score of 0.82 on a stratified five-fold cross-validation dataset split. This proves that the defects on this dataset (such as visualized in Figure 3) can be recognized by a Machine Learning (ML) model.

The SAT images during manufacturing look similar to each other at first sight. During another preliminary study, we tested a simple approach: we aligned all images with respect to a predefined reference image. Then, we calculated the difference on the pixel level with respect to the reference image. However, this was not sufficient to detect defects such as voids or delaminations. The variation during the SAT recording, along with the variation in mechanical tolerances, was too large.

4 Methodology

We test multiple FMS both on the task of segmentation and the task of classification on the above datasets. Our experiments are set up as follows.

4.1 Segmentation

For the task of segmentation, we use the Intersection over Union (IoU) as a metric to assess the segmentation output:

$$IoU = \frac{|A \cap B|}{|A \cup B|}, \quad (1)$$

with A being the set of pixels predicted positive by the model and B being the set of pixels marked positive in the ground-truth. The IoU metric yields values between zero and one, whereas one corresponds to an optimum result. This metric penalizes especially the slip of small defects (=false negatives): The correctly detected defective area (intersection of defective prediction and ground truth) is divided by the union of the defective prediction and the ground truth. A miss of a small defect, such as a void, will thus return low metric values. Such false negatives are especially critical since defective products would be delivered to customers.

Three pipelines are chosen from Text2Seg for evaluation: GroundingDINO + SAM, SAM + CLIP and CLIPSurgery + SAM. The Git repository of the CLIPSurgery paper is used both for the CLIP Surgery model and the CLIP model. The original Git repository by IDEA research [24] is used for the GroundingDINO model. All of our Text2Seg pipelines use the huge backbone of SAM. The base ViT is used for both CLIP and CLIP Surgery with weights *CS-ViT-B/16* [22] for CLIP Surgery and weights *ViT-B/16* [23] for CLIP. The checkpoint *groundingdino swint ogc* [24] is used for the GroundingDINO model, along with a box threshold of 0.35 and a text threshold of 0.25. Minor issues such as the handling of empty segmentation masks, are resolved to enable an automated end-to-end evaluation in all cases. The utilized text prompts are short and simple, e.g., “defect” or “cat”.

For SAA+, we use GroundingDINO with weights *groundingdino swint ogc* as the region proposal network and a SAM model with the huge backbone as region refiner. The box threshold is set to 0.1 and the text threshold to 0.1. While the authors of Text2Seg evaluate their approach on the task of semantic segmentation in remote sensing images as taken, e.g., by an Unmanned Aerial Vehicle (UAV), the authors of SAA+ focus specifically on the segmentation of anomalies and also utilize the *MVTec AD* dataset during evaluation.

Since initial tests for the segmentation models have shown insufficient performance on both the real-world *IndustrialSAT* dataset and the public industrial *MVTec AD* dataset, we extend the evaluation for those models to include the more general public dataset *Oxford-IIIT-Pet*. This is done to validate the correct setup of our software pipelines and to gain additional insights.

The chosen model pipelines represent a diverse setup of multiple prominent SOTA FMs. The models are all executed in their inference mode and are tested on the defective images of all three datasets that were introduced in the previous section. The reported IOU metric values are averaged over all defective images. This yields an insight into how well defects can be recognized in the different datasets.

4.2 Classification

For the task of classification, we analyze the LVLM model Gemini 2.5 Pro. It features a high token limit, which enables long prompts. Furthermore, it can process multi-modal input, i.e., it can receive prompts consisting of both written text and images. The model's output includes a single class label and a corresponding reasoning string. The reasoning string is more verbose and gives a deeper insight into how the model has made a decision, whereas the class label can be used during automated postprocessing. For each dataset, a reference image is defined. This reference is used to automatically align all images, ensuring the same position and orientation of the product in all images.

The testing is carried out using the Google Cloud Platform, which offers a convenient access to the model via an Application Programming Interface

(API). It is carried out using both defect-free and defective images from the industrial datasets *IndustrialSAT* and *MVTec AD*. Initial experiments have shown promising performance on industrial data. Thus, an assessment on the general public dataset *Oxford-IIIT-Pet* is omitted, since the aim of this paper is to assess model performance on real-world industrial data.

The F2-score is used as a metric during evaluation:

$$F_2 = \frac{5 \cdot \text{precision} \cdot \text{recall}}{4 \cdot \text{precision} + \text{recall}} \quad (2)$$

Precision is the ratio of true positives (defects detected as defects) divided by the sum of true and false positives. Recall is the ratio of true positives divided by the sum of true positives and false negatives. As compared to F1-score, the F2-score puts a higher weight on recall than on precision. This penalizes false negatives (slips of defective products) more heavily compared to false positives (wrong alerts for defect-free products) [26]. This is desired, since it is better to re-check a suspicious product than to deliver a defective product to the customer.

When querying the Gemini model, a test image with an unknown state is sent to the model along with a prompt as described below. The prompt may include a predefined reference image. Two types of prompts were defined for the Gemini model. The basic prompt consists of a) a simple text prompt that queries for any anomalies or defects and b) an exemplary defect-free image. The refined prompt additionally includes specific information about the physical structure of e.g. the electronic package in *IndustrialSAT* or the object properties in *MVTec AD* and information about the visual appearance of the respective defects. This follows the conceptual approach of varying information depth in prompts by Xu et al. [14]. On *MVTec AD*, the prompts are as follows. The basic prompt for *MVTec AD* is “Please determine whether the image contains anomalies or defects. If yes, give a specific reason” - similar to the naive prompt studied by Xu et al. [14]. The refined prompt for *MVTec AD* is “Please determine whether the last image given about object contains anomalies or defects. If so, please provide a specific reason. Normally, the image given should depict a clear and identifiable object. It may have defects such as broken/bented parts, contaminations, threads, color stains, cuts, holes, scratches, liquids, glue, folds,

pokes, oil or glue strips.”. The formulation of the prompts for *IndustrialSAT* is made in a close alignment with process engineers, who are responsible for the respective packaging processes. These prompts include specifics of the respective product and thus may not be published in full detail.

5 Results

This section contains the benchmarking results of both the segmentation and classification models on the different datasets as introduced before.

5.1 Segmentation

The IoU metric values for the segmentation models are reported in Table 1. Essentially, the defects in the SAT images cannot be detected in any of the tested pipelines. Visual examples are shown in Figure 4 for each model to give deeper insights. The pipeline based on GroundingDINO + SAM seems to be very sensitive and segments large defect-free regions of the image. For the other pipelines, the defects are missed entirely. In some cases, the pipeline CLIPSurgery + SAM segments geometric features of sub-components instead of defective regions. For the *MVTec AD* dataset, metric values are rather low. A visual depiction of results in Figure 5 shows that some defects can be detected accurately. For the public dataset *Oxford-IIIT-Pet*, the IoU metric results are promising, with IoU scores ranging from 0.65 to 0.80. A visual depiction of results in Figure 6 shows that the cat can be segmented very well by all models. We conducted a thorough inspection of various images and report that the model’s segmentation results align with the depicted cat even more closely than the ground truth masks. Since the cat is configured to represent defective regions, this validates the correct setup of all pipelines for the segmentation task. Furthermore, we report a correct segmentation of other animals (e.g. dogs). Only IoU scores larger than 0.60 are considered for a comparison between models (indicated in Table 1 by bold print), since lower values are too low to offer practical value.

Table 1: IoU scores on all datasets for various FMS pipelines. None of the methods can detect defects on our internal dataset *IndustrialSAT*. Only SAA+ works somehow reasonable on the public *MVTec AD* dataset. All segmentation models work well on the public dataset *Oxford-IIIT-Pet*.

Classes and datasets	GroundingDINO + SAM	SAM + CLIP	CLIPSurgery + SAM	SAA+
Defect types void, crack and delamination in dataset <i>IndustrialSAT</i>	0.00	0.00	0.00	0.00
Various defects such as cracks, holes, scratches in dataset <i>MVTec AD</i>	0.13	0.05	0.19	0.52
Animal cat in dataset <i>Oxford-IIIT-Pet</i>	0.80	0.65	0.78	0.77

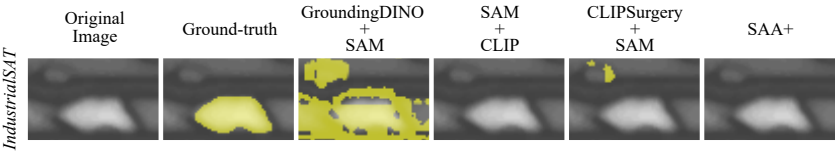


Figure 4: Exemplary segmentation results on dataset *IndustrialSAT*. A yellow color overlay is used to indicate defective regions as defined in ground-truth data or as predicted by the respective model. All models fail to detect the defects in most images.

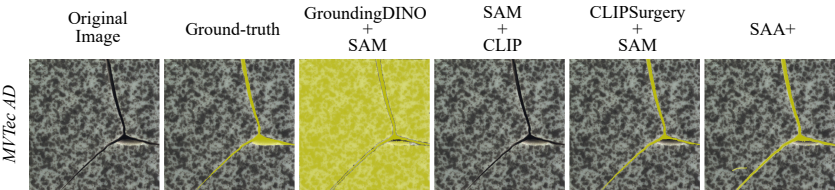


Figure 5: Exemplary segmentation results on dataset *MVTec AD* [16]. A yellow color overlay is used to indicate defective regions as defined in ground-truth data or as predicted by the respective model. In some cases, defects can be accurately recognized. In other cases, the models fail to recognize the defects.

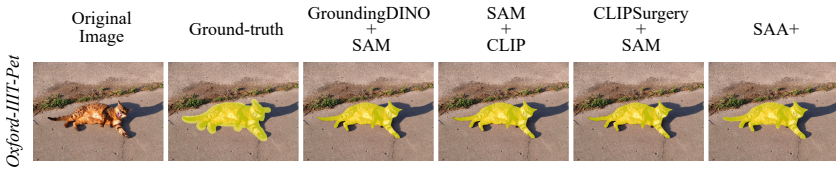


Figure 6: Exemplary segmentation results on dataset *Oxford-IIIT-Pet* [15]. A yellow color overlay is used to indicate defective regions as defined in ground-truth data or as predicted by the respective model. Animals such as the cat shown in this image can be segmented very accurately by all tested models.

5.2 Classification

Results for the classification use case are shown in Table 2. The F2-score values are far too low to be suitable for industrial applications during quality inspection. However, the model can recognize some defects accurately, both with the basic and refined prompt strategy. Also, we report that the reasoning string output looks promising and shows that the LVLm model was able to recognize the package build-up to a certain extent.

The Gemini model performs significantly better on the public industrial dataset *MVTec AD* as compared to our internal industrial dataset *IndustrialSAT*. Interestingly, the highest metric results were achieved with simple prompts, which consist of a naive text prompt for defects along with a defect-free reference image. The maximum F2-score of 0.37 achieved by the Gemini model on *IndustrialSAT* is far below the F2-score of 0.82 reached with a supervised AI model in a preliminary study.

In analogy to the *Oxford-IIIT-Pet* dataset on the segmentation task, the good performance on the *MVTec AD* dataset validates the correct setup of our evaluation pipeline for the classification task.

6 Discussion

The three tested datasets can be ordered according to their visual similarity to datasets like SA-1B that are used to train the FMs: *Oxford-IIIT-Pet* is in a very

Table 2: Classification results (F2-score) on both industrial datasets for the Gemini 2.5 Pro LVLm model. While the model performs well on the *MVTec AD* dataset, its performance on SAT imaging data is insufficient for industrial-quality inspection.

Dataset	Gemini (basic prompt)	Gemini (refined prompt)
<i>IndustrialSAT</i>	0.37	0.31
<i>MVTec AD</i>	0.99	0.92

similar visual domain as SA-1B. *MVTec AD* is more focused on defects and includes industrially relevant parts, but some of its object categories, such as carpets and wood, are likely to be found in, e.g., SA-1B. However, our real-world dataset, *IndustrialSAT*, differs significantly from datasets like SA-1B. It is not a color image, but rather a greyscale image. The geometric features that are shown resemble mostly geometric primitives, such as rectangular areas and line features. The performance of all the FMs is decreasing along with the visual similarity of the test dataset to training datasets such as SA-1B. The domain gap seems to be significant, especially for the real-world dataset *IndustrialSAT*. This is likely to be the cause of the insufficient performance on our real-world industrial data.

Furthermore, the performance difference on the datasets *MVTec AD* and *IndustrialSAT* motivates the question of how well *MVTec AD* can represent advanced imaging procedures such as SAT. The *MVTec AD* dataset contains industrially relevant defects, but it is in a similar visual domain as public datasets. For example, screws and fabric-like materials can be found in everyday items, which are likely to be included in the training datasets of FMs to a certain extent. Defect patterns such as voids and cracks in SAT imaging are not well represented.

The reasoning strings of the Gemini model show a deeper understanding of the product build-up. This may not be fully correct, but it gives an interesting starting point for further studies. If this knowledge is available at least partially for the model, more advanced prompt strategies could leverage this knowledge better. The Gemini model was the only model that could detect some of the defects and a more refined prompting strategy seems to be a promising way to

improve performance.

7 Conclusion and outlook

Industrial quality inspection regularly involves advanced imaging procedures such SAT. In this work, we investigate the use of FMS to improve industrial defect recognition. None of the tested FMS were able to detect defects in our real-world SAT images. The results on other public datasets validate the correct setup of our models. While an application of FMS to such data could be scaled across many manufacturing stations, currently available FMS are not ready for practical application in series manufacturing for advanced inspection technologies such as SAT imaging.

The low performance in *IndustrialSAT* may be due to a large domain gap between the training data of FMS. Future work could thus involve fine-tuning FMS on image data that better matches the real-world industrial domain. Furthermore, more advanced prompt strategies for LVLMS seem to be a promising direction for an in-depth follow-up study.

8 Acknowledgments

The Helmholtz Association funds this project under the research school "Helmholtz Information and Data Science School for Health (HIDSS4Health)", the program "Natural, Artificial and Cognitive Information Processing (NACIP)" and the Initiative and Networking Fund on the HAICORE@KIT partition. This work was funded by the European Union (NextGenerationEU) and the German Federal Ministry for Economic Affairs and Energy (BMWE) based on a decision by the German Bundestag. The authors acknowledge financial support within the SmartMan project (reference number: 13IK033) in the Kopa35c program.

We describe the contributions of Simon Baeuerle (SB), Pratik Khanna (PK), Nils Friederich (NF), Angelo Jovin Yamachui Sitcheu (AJYS), Damir Shakirov

(DS), Andreas Steimer (AS) and Ralf Mikut (RM) according to CRediT [25]: Writing-Original Draft: SB; Writing-Review & Editing: PK, NF, AJYS, DS, AS, RM; Conceptualization: SB, RM; Investigation: PK (within masters thesis [27]); Methodology: SB, PK, NF, AJYS, DS, AS, RM; Software: PK; Supervision: SB, RM; Project administration: SB, RM; Funding Acquisition: SB, RM.

References

- [1] Zihan He, Yudong Lian, Yulei Wang and Zhiwei Lu. A comprehensive review of research on surface defect detection of PCBs based on machine vision. *Results in Engineering*, 27:106437, 2025.
- [2] Prahar M. Bhatt, Rishi K. Malhan, Pradeep Rajendran, Brual C. Shah, Shantanu Thakar, Yeo Jung Yoon and Satyandra K. Gupta. Image-based surface defect detection using deep learning: a review. *Journal of Computing and Information Science in Engineering*, 21(4):040801, 2021.
- [3] Alina Pleli, Simon Baeuerle, Michel Janus, Jonas Barth, Ralf Mikut and Hendrik P. A. Lensch. Iterative Cluster Harvesting for wafer map defect patterns Technical report, arXiv:2404.15436, 2024. URL <http://arxiv.org/abs/2404.15436>.
- [4] Rishi Bommasani et al. On the opportunities and risks of foundation models. Technical report, arXiv:2108.07258, August 2021. URL <https://arxiv.org/pdf/2108.07258>.
- [5] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything. Technical report, arXiv:2304.02643, April 2023. URL <http://arxiv.org/abs/2304.02643>.
- [6] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning

- transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, July 2021. URL <https://proceedings.mlr.press/v139/radford21a.html>.
- [7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
 - [8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollar, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, New Orleans, LA, USA, June 2022. IEEE. doi:10.1109/CVPR52688.2022.01553. URL <https://ieeexplore.ieee.org/document/9879206/>.
 - [9] Yi Li, Hualiang Wang, Yiqun Duan, Jiheng Zhang, and Xiaomeng Li. A closer look at the explainability of Contrastive language-image pre-training. *Pattern Recognition*, 162:111409, June 2025. ISSN 00313203. doi:10.1016/j.patcog.2025.111409. URL <https://linkinghub.elsevier.com/retrieve/pii/S003132032500069X>.
 - [10] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, volume 15105, pages 38–55. Springer Nature Switzerland, Cham, Switzerland, 2025. Series Title: Lecture Notes in Computer Science.
 - [11] Google Gemini Team. Gemini 2.5: pushing the frontier with advanced reasoning, multimodality, long context, and

- next generation agentic capabilities. Technical report, Google, 2025. URL https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf.
- [12] Jieli Zhang, Zhongliang Zhou, Gengchen Mai, Mengxuan Hu, Zihan Guan, Sheng Li, and Lan Mu. Text2Seg: Zero-shot Remote Sensing Image Semantic Segmentation via Text-Guided Visual Foundation Models. In *Proceedings of the 7th ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery*, pages 63–66, Atlanta, GA, USA, October 2024. ACM. doi:10.1145/3687123.3698287. URL <https://dl.acm.org/doi/10.1145/3687123.3698287>.
 - [13] Yunkang Cao, Xiaohao Xu, Yuqi Cheng, Chen Sun, Zongwei Du, Liang Gao, and Weiming Shen. Segment Any Anomaly without Training via Hybrid Prompt Regularization. Technical report, arXiv:2305.10724, 2023. URL <https://arxiv.org/abs/2305.10724>.
 - [14] Xiaohao Xu, Yunkang Cao, Huaxin Zhang, Nong Sang, and Xiaonan Huang. Customizing visual-language foundation models for multi-modal anomaly detection and reasoning. Technical report, arXiv:2403.11083, 2025. URL <https://arxiv.org/abs/2403.11083>.
 - [15] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012. (Images under license CC BY-SA 4.0, see <https://creativecommons.org/licenses/by-sa/4.0/>).
 - [16] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD - A comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9584–9592, Long Beach, CA, USA, June 2019. doi:10.1109/CVPR.2019.00982. (Images under license CC BY-NC-SA 4.0, see <https://creativecommons.org/licenses/by-nc-sa/4.0/>).
 - [17] Kilian Batzner, Lars Heckler, and Rebecca König. EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies. In *2024*

- IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 127–137, Waikoloa, HI, USA, January 2024. IEEE. doi:10.1109/WACV57701.2024.00020. URL <https://ieeexplore.ieee.org/document/10484326/>.
- [18] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. PaDiM: a patch distribution modeling framework for anomaly detection and localization. In *Pattern Recognition. ICPR International Workshops and Challenges*, volume 12664, pages 475–489. Springer International Publishing, Cham, Switzerland, 2021. doi:10.1007/978-3-030-68799-1_35. Series Title: Lecture Notes in Computer Science.
 - [19] Fan Liu, Lei Su, Mengying Fan, Jian Yin, Zhenzhi He, and Xiangning Lu. Using scanning acoustic microscopy and LM-BP algorithm for defect inspection of micro solder bumps. *Microelectronics Reliability*, 79:166–174, December 2017. ISSN 00262714. doi:10.1016/j.microrel.2017.10.029. URL <https://linkinghub.elsevier.com/retrieve/pii/S002627141730505X>.
 - [20] Hyunung Yu. Scanning acoustic microscopy for material evaluation. *Applied Microscopy*, 50(1):25, December 2020. ISSN 2287-4445. doi:10.1186/s42649-020-00045-4. URL <https://appmicro.springeropen.com/articles/10.1186/s42649-020-00045-4>.
 - [21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. Technical report, arXiv:1512.03385, 2015. URL <https://arxiv.org/abs/1512.03385>.
 - [22] xmed-lab. GitHub - xmed-lab - CLIPSurgery. Web-page, 2025. URL https://github.com/xmed-lab/CLIP_Surgery.
 - [23] OpenAI. GitHub - openai - CLIP. Web-page, 2025. URL <https://github.com/openai/CLIP>.
 - [24] IDEA-Research. GitHub - idea-research - GroundingDINO. Web-page, 2025. URL <https://github.com/IDEA-Research/GroundingDINO>.

- [25] Amy Brand, Liz Allen, Micah Altman, Marjorie Hlava and Jo Scott. Beyond authorship: attribution, contribution, collaboration, and credit. *Learned Publishing*, 28(2):151–155, 2015.
- [26] Seyed Raein Hashemi, Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, Sanjay P. Prabhu, Simon K. Warfield, and Ali Gholipour. Asymmetric loss functions and deep densely-connected networks for highly-imbalanced medical image segmentation: application to multiple sclerosis lesion detection. *IEEE Access*, 7:1721–1735, 2019.
- [27] Pratik Khanna. Foundation models for semantic segmentation of industrial defect patterns. M.Sc. thesis, Karlsruher Institut für Technologie (KIT), Karlsruhe, Germany (2025).

Physics-Informed Neural Networks for State Estimation Problem

Aleksandr Subbotin, Thomas Bartz-Beielstein

Technische Hochschule Köln, Institute for Data Science, Engineering, and
Analytics

E-Mail: {aleksandr.subbotin,thomas.bartz-beielstein}@th-koeln.de

Abstract

Accurate state estimation in nonlinear dynamical systems is essential for control, monitoring, and decision-making in engineering applications. Traditional approaches rely on either knowledge-based models, grounded in first-principles physics, or data-driven models, which may lack generalization and interpretability. This work proposes a combined method that integrates Physics-Informed Neural Networks (PINNs) with the Unscented Kalman Filter (UKF) to address the state estimation problem. The PINN serves as a surrogate model for system dynamics within the UKF framework. The approach is evaluated through a case study on vehicle sideslip angle estimation, a parameter critical for stability control but difficult to measure directly. Experimental results using real-world data demonstrate that the PINN-enhanced UKF achieves improved estimation accuracy and generalization compared to the UKF with physics-based and purely data-driven models. These findings highlight the potential of PINNs for robust state estimation in practical engineering systems.

1 Introduction

State estimation is a fundamental problem in control theory and process engineering. It refers to the process of determining the internal state variables

of a dynamic system from incomplete and noisy measurements, typically using mathematical models and estimation algorithms. Applications of state estimation span diverse domains, including power systems, battery management, robotics, and process control [1]. Over the past decades, numerous formulations of the state estimation problem have been studied, and various solutions have been proposed [2]. Among these, Kalman filtering (KF) techniques remain the most widely adopted. KF requires system and observation models that describe how the system evolves over time by defining the relationships between the inputs, outputs, and internal state variables [1]. The accuracy of the estimates critically depends on the fidelity of these models to real system behavior. For linear system dynamics and observation models, state estimation can be performed using the standard linear KF [3]. However, most real-world systems exhibit nonlinear behavior [7]. To address this challenge and extend applicability across diverse settings, several nonlinear variants of the KF have been developed, including the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), Indirect Kalman Filter (IKF), Cubature Kalman Filter (CKF), and adaptive formulations [6].

There are three primary paradigms for modeling system dynamics in state estimation: knowledge-based, data-driven, and hybrid approaches. Knowledge-based methods leverage first-principles physics and expert knowledge to construct mathematical models that describe system behavior. These models are interpretable and can generalize well to unseen scenarios, but they may be limited by simplifications and assumptions that do not fully capture complex real-world dynamics. In contrast, data-driven methods utilize machine learning techniques to learn system dynamics directly from empirical data. While these approaches can model complex nonlinear relationships, they often require large datasets for training and may lack interpretability and robustness when extrapolating beyond the training distribution. Hybrid modeling combines both paradigms, integrating the strengths of knowledge-based and data-driven approaches into a cohesive framework [5].

In recent years, Physics-Informed Neural Networks (PINNs) have emerged as a promising framework for hybrid modeling. By embedding physical laws into the training of neural networks, PINNs leverage both empirical data and prior knowledge of system dynamics. Furthermore, PINNs inherently enable

parameter identification during training, which makes them particularly suitable for state estimation tasks.

This work investigates the use of PINNs for state estimation in nonlinear dynamical systems. We propose a combined method that integrates a hybrid model—specifically, a Physics-Informed Neural Network (PINN)—with the UKF [7], a widely used nonlinear estimator. The PINN is employed as a surrogate model of system dynamics, incorporated within the UKF framework.

We evaluate our approach through a case study involving the estimation of a vehicle’s sideslip angle, a critical parameter for vehicle stability control that is difficult to measure directly. The sideslip angle is an important parameter for vehicle stability control systems, such as electronic stability control (ESC) and advanced driver-assistance systems (ADAS). Accurate estimation of the sideslip angle is essential for ensuring vehicle safety and performance, particularly during aggressive maneuvers or adverse road conditions. However, direct measurement of the sideslip angle typically requires expensive sensors, such as optical or inertial measurement units, which are not commonly available in standard vehicles. Therefore, developing reliable estimation methods using readily available measurements is of significant practical importance [4].

The remainder of this paper is structured as follows. Section 2 presents the necessary background on PINNs and the UKF. Section 3 details the proposed methodology. Section 4 reviews related literature. Section 5 describes the experimental setup, including the vehicle model, data acquisition, and preprocessing procedures. Section 6 reports the experimental results, while Section 7 discusses the findings. Section 8 concludes the paper and suggests directions for future research.

2 Background

2.1 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) [7] is a popular algorithm for estimating the state of nonlinear systems. Unlike the Extended Kalman Filter (EKF),

which relies on linearizing the system using Jacobians, the UKF uses the Unscented Transformation (UT) to more accurately estimate the mean and covariance of a random variable that passes through a nonlinear function. The UT works by deterministically selecting a set of so-called sigma points around the current mean. These sigma points are propagated through the nonlinear system equations, and the resulting transformed points are used to reconstruct the predicted mean and covariance. This approach allows the UKF to better capture the effects of nonlinearity without requiring explicit calculation of derivatives, making it more robust and easier to implement for highly nonlinear systems.

In state estimation problems, the system is typically modeled in state-space form, which predicts how the state evolves and how measurements are generated from it. For nonlinear discrete-time systems, the general state-space model is given by:

$$x_{k+1} = F(x_k, u_k) + w_k \quad (1)$$

$$z_k = H(x_k) + n_k \quad (2)$$

where x_k is the state vector of the system at timestep k ; u_k is the input (control) vector; $w_k \sim N(0, Q_k)$ is the process (state) noise vector with covariance matrix Q_k ; z_k is the measurement (observation) vector; $n_k \sim N(0, R_k)$ is the measurement (observation) noise vector with covariance matrix R_k ; and $F[\cdot]$, $H[\cdot]$ are nonlinear state transition and measurement functions, respectively.

Similar to other Kalman filter variants, the UKF operates in two main steps: prediction and update. The UKF algorithm can be summarized as follows:

- Sigma Point Generation:
 - Given the current state estimate $\hat{x}_k$ and covariance P_k , generate $2L + 1$ sigma points $x_k^{(i)}$, where L is the dimension of the state vector.
- Prediction Step:

- Propagate each sigma point through the nonlinear process model $F[\cdot]$ (Eq. (1)):

$$x_{k+1|k}^{(i)} = F(x_k^{(i)}, u_k) \quad (3)$$

- Compute the predicted state mean $\hat{x}_{k+1|k}$ and covariance matrix $P_{k+1|k}$.

- Update Step:

- Propagate the predicted sigma points through the nonlinear measurement model $H[\cdot]$ (Eq. (2)):

$$z_{k+1|k}^{(i)} = H(x_{k+1|k}^{(i)}) \quad (4)$$

- Compute the predicted measurement mean $\hat{z}_{k+1|k}$ and covariance S_{k+1} .
- Compute the cross-covariance C_{k+1} between the state and measurement.
- Compute the Kalman gain K_{k+1} and update the state estimate $\hat{x}_{k+1}$ and covariance P_{k+1} :

$$K_{k+1} = C_{k+1} S_{k+1}^{-1} \quad (5)$$

$$\hat{x}_{k+1} = \hat{x}_{k+1|k} + K_{k+1} (z_{k+1} - \hat{z}_{k+1|k}) \quad (6)$$

$$P_{k+1} = P_{k+1|k} - K_{k+1} S_{k+1} K_{k+1}^T \quad (7)$$

The UKF iteratively performs these steps as new measurements become available, providing real-time state estimates for nonlinear systems without requiring Jacobian matrices or explicit linearization.

2.2 Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINNs) were introduced by Raissi *et al.* as a method of solving nonlinear partial differential equations (PDEs) in a series

of publications [8–10]. The original approach has been extensively utilized and refined in numerous research and review articles [11–14].

Raissi *et al.* addressed two principal problem types: the *forward problem*, which involves solving PDEs with known parameters, and the *inverse problem*, which focuses on data-driven identification of unknown PDE parameters [10]. Both problems are formulated in continuous and discrete time domains. The PINN framework can be divided on three main components: a neural network (NN), physics equations that describe the dynamics of a system, and a composite loss function that combine the NN and physics equations in one joined solution. The original vanilla PINN used a feed-forward neural network with hyperbolic tangent activation function. Various PINN extensions based on different types and architectures of neural networks have been investigated, but the feed-forward NN remains the most commonly used [13]. The NN is parametrized by a set of parameters θ that includes all NN parameters(weights and biases) for all layers. The second component is physics equations that describe the dynamics of a system. In a wide range of problems, system dynamics can be described by parameterized nonlinear partial differential equations in general form

$$\frac{\partial u}{\partial t} + N[u; \lambda] = 0, \quad x \in \Omega, \quad t \in [0, T], \quad (8)$$

where, $u(t, x)$ - the latent solution, represents the state of the dynamic system and depends on time t and a spatial variable x ; $\Omega \subset \mathbb{R}^n$ is n -dimensional space domain; $N[u; \lambda]$ represents a non-linear differential operator parametrized by λ . If the parameters λ are known, the PINNs can solve a forward problem or data-driven solutions of PDEs. In the other case the unknown parameters λ can be added to the set of NN's parameters θ and the PINN can solves an inverse problem or data-driven discovery of PDEs. PINNs supports a wide range of PDEs: integer-order PDEs, integro-differential equations, fractional PDEs, and stochastic PDEs [12]. To combine the NN and physics equations PINNs utilize a composite loss function shown in Eq. (9) that includes the governing physics equations as a regularization term [13]. A composite loss function is generally formed as follows:

$$Loss(\theta) = w_{data} \cdot L_{data}(\theta) + w_F \cdot L_F(\theta) + w_b \cdot L_b(\theta) \quad (9)$$

where L_{data} - data loss term, represents the loss produced by a mismatch of model's predictions with the available data (measurements); L_F - represents the loss produced by a mismatch with the governing physics equations; L_b - loss due to mismatch with the boundary and/or initial conditions; coefficients w_{data}, w_F, w_b are weighting factors for corresponding loss terms. These coefficients can be user-defined or automatically tuned as a hyper-parameters. This general formulation of the loss function is very convenient for a theoretical study and makes it easy to switch between unsupervised mode (forward problem) and supervised mode (inverse problem) [13].

An important distinction within the PINN framework lies in whether the underlying dynamics are represented in a continuous or discrete time domain. In the continuous-time formulation, the neural network is trained to approximate the solution $u(t, x)$ of the governing PDE from Eq. (8) directly. Here, the time variable t is treated as a continuous input to the network, and the residual of the PDE is enforced across collocation points sampled in the spatio-temporal domain. Crucially, in this formulation the nonlinear differential operator $N[u; \lambda]$ and the time derivative $\frac{\partial u}{\partial t}$ are evaluated using automatic differentiation (autograd), which allows for exact computation of derivatives with respect to the network inputs.

By contrast, the discrete-time formulation of PINNs is constructed by applying numerical time-stepping schemes (e.g., Runge–Kutta method) to the PDE before incorporating it into the loss function. In this setting, the network approximates the solution only at discrete time levels, and the governing equations are enforced through consistency with the chosen discretization scheme. Discrete-time PINNs thus embed a numerical integrator within the learning framework, instead of relying solely on autograd.

3 Proposed Approach

In our approach, following the methodology outlined in [23], the PINN is employed as a surrogate model for the state transition function:

$$x_{k+1} = F_{\text{PINN}}(x_k, u_k) + w_k \quad (10)$$

where F_{PINN} denotes the state transition function learned by the trained PINN. The observation model remains in the standard form (Eq. (2)). In contrast to [23], this work employs a simplified discrete-time PINN and real-world measurements instead of simulated data. Moreover, the underlying physical process model is deliberately simplified and does not completely represent the system dynamics, thereby increasing the challenge of integrating it with real measurement data.

The original PINNs framework is not directly suitable for state-space modeling due to two primary limitations. First, in the standard PINN architecture, the initial conditions are embedded within the network parameters during training. As a result, the trained model is valid only for the specific initial conditions used in the training process. Second, the original PINN formulation lacks support for control inputs, limiting its applicability to systems governed by control signals.

A solution to these limitations was proposed by Antonelo *et al.* [15]. Following the proposed methodology, we construct PINN that accepts the current state x_k and control input u_k as inputs and predicts next state x_{k+1} . In our approach we use a discrete-time formulation of PINNs; therefore the time variable t is not included as an input and the model operates in discrete time steps.

The network is trained using a composite loss function shown in Eq. (11) that combines data-driven loss from available measurements and physics-informed term. In our case, the physics-informed term enforces consistency between the predicted states and the states obtained from the governing equations solved by one step of the Runge–Kutta method at a set of collocation points. Collocation points are the discrete set of points in the input domain (current state and control input) where the physics-informed loss term is evaluated.

The composite loss function is defined as:

$$Loss(\theta) = w_{data} \cdot L_{data}(\theta) + w_F \cdot L_F(\theta) \quad (11)$$

where L_{data} is the data loss term, representing the mean squared error (MSE) between the predicted and measured system states, and L_F is the physics-informed

term, representing the MSE between the predicted and calculated states. The weighting factors are $w_{data} = 1$, and w_F is treated as a hyperparameter.

4 Overview of Related Works

The application of PINNs for control and state estimation has gained significant attention in recent years. Arnold *et al.* [17] pioneered this field by extending PINNs to develop state-space models suitable for Model Predictive Control (MPC) and state estimation. The resulting PINN-based state-space model was evaluated within the contexts of MPC and the EKF. In both scenarios, the proposed approach demonstrated adequate performance while offering a significant reduction in computational cost for model prediction compared to conventional numerical solvers.

Gokhale *et al.* [18] applied a PINN-based approach to develop a control-oriented thermal model of a building. The authors introduced two PINN variants, both designed to predict the system's next state using a high-dimensional input comprising the current state, control action, and exogenous variables. Experimental results demonstrated that both architectures achieved low prediction errors and performed effectively for the given task. Furthermore, the study confirmed that PINNs exhibit superior performance in scenarios with limited training data, demonstrating strong generalization capabilities even when trained on smaller datasets.

A major advancement came from Antonelo *et al.* [15, 16], who introduced the Physics-Informed Neural Networks for Control (PINC) framework. Their approach extends continuous-time PINNs by incorporating initial system states and control inputs as additional network inputs, enabling their use as predictive models in MPC frameworks and significantly broadening their applicability to control problems.

Building on the PINC framework, several researchers have demonstrated successful applications across diverse domains. Nicodemus *et al.* [19] achieved improved computational efficiency over traditional time integration methods in multi-link manipulator control. Kittelsen *et al.* [20] enhanced the framework

for highly nonlinear systems like gas-lifted oil wells through architectural improvements including skip connections and ODE reformulations. The study demonstrated that the improved PINC framework enables accurate long-range predictions in self-loop operation and is effective for controlling complex systems within a MPC setting. Sanyal *et al.* [21] proposed RAMP-Net, a robust adaptive MPC framework using PINNs that combines physics-based and data-driven loss components to handle uncertainties in nonlinear systems. Dave *et al.* [22] introduced the Physics-informed State-space neural network Model (PSM) for transport phenomena, integrating sensor data with PDE constraints to model pressure, velocity, and temperature fields.

Most relevant to our work, De Curtò *et al.* [23] recently proposed combining PINNs with adaptive UKF for state estimation. This methodology was validated on double pendulum and quadcopter drone simulations, showing notable improvements in estimation accuracy and robustness over conventional approaches.

The reviewed literature demonstrates a rapidly growing interest in applying PINNs to control and state estimation tasks, with significant progress made in recent years. While PINNs have been successfully integrated into MPC frameworks, their application to state estimation remains relatively limited.

5 Experimental Setup

To evaluate the proposed approach, we conduct a case study on the estimation of a vehicle's sideslip angle. Two experiments were performed. In the first experiment, we compared the performance of a conventional neural network (NN) and a PINN as state transition models. In the second experiment, we evaluated three different UKF configurations: (a) a standard UKF using a kinematic vehicle model, (b) a UKF utilizing a NN as the state transition model, and (c) a UKF incorporating a PINN as the state transition model. This section describes the vehicle model, data collection and preprocessing steps, and the implementation details of the NN and PINN models.

5.1 Vehicle Model

The vehicle model used in this study is a kinematic model vehicle model shown in Eq. (12). This model assumes the vehicle to be a point mass located at the center of gravity, moving in a two-dimensional plane, and does not incorporate any physical parameters of the vehicle. It makes the model simple, not sensitive to vehicle parameter changes and straightforward to implement. The state observer for estimating the vehicle's lateral velocity v_y and sideslip angle β based on the kinematic model introduced by Farrelly *et al.* [24]. In this approach, the longitudinal and lateral velocities v_x and v_y are state variables, longitudinal and lateral accelerations a_x and a_y serve as system inputs. The yaw rate r is considered as a time-varying parameter of the system. The kinematic model is described by the following equations:

$$\begin{aligned}\dot{v}_x &= v_y \cdot r + a_x \\ \dot{v}_y &= -v_x \cdot r + a_y\end{aligned}\tag{12}$$

The observer estimates the vehicle's lateral velocity v_y using measurements of the longitudinal velocity v_x , yaw rate r , and longitudinal and lateral accelerations a_x and a_y . The sideslip angle, β , is then obtained as follows:

$$\beta = \arctan\left(\frac{v_y}{v_x}\right)\tag{13}$$

Farrelly *et al.* [24] demonstrated that the kinematic model-based state observer provides reliable estimates and exhibits stability. However, a well-known limitation of this observer is that the kinematic model becomes unobservable when the yaw rate $r = 0$, which typically occurs during straight-line driving. As a result, the observer's estimates may drift under such conditions [25]. Despite this drawback, the kinematic model remains widely used, often with certain modifications [26, 28] or in combination with dynamic models [25].

5.2 Data Collection

For the experimental evaluation, we employ the publicly available Revs Vehicle Dynamics Database, provided by the REVs Program at Stanford University [27]. This dataset contains high-fidelity measurements collected from vintage race cars, specifically a 1965 Ferrari 250 LM Berlinetta GT and a 1963 Corvette Grand Sport. Data were recorded during race sessions conducted by professional drivers at two distinct tracks: Palm Beach International (PBI) Raceway (Jupiter, Florida) and Laguna Seca (LC) Raceway (California), thereby covering different track geometries and driving conditions. The dataset comprises time-synchronized measurements from 56 channels, available in both comma-separated values (.csv) format at a 1000 Hz sampling frequency and MATLAB MAT-file (.mat) format at the original sampling frequency.

5.3 Data Preprocessing

From the 56 available parameters, we selected a subset that is relevant for state estimation using the kinematic vehicle model. Table 1 gives an overview. All selected parameters were recorded at same frequency of 100 Hz. So there is no resampling needed.

Table 1: Selected parameters from the Revs Vehicle Dynamics Database

Parameter	Description	Units	Notation
<i>vxCG</i>	Longitudinal velocity	m/s	v_x
<i>vyCG</i>	Lateral velocity	m/s	v_y
<i>axCG</i>	Longitudinal acceleration	m/s^2	a_x
<i>ayCG</i>	Lateral acceleration	m/s^2	a_y
<i>yawRate</i>	Yaw rate	rad/s	r
<i>sideSlip</i>	Sideslip angle	deg	β

To emulate limited data scenarios, only a single file (one session) from the 2013 Monterey Motorsports Reunion at LC Raceway was utilized for training and validation purposes. Seventy percent of this data was allocated for training,

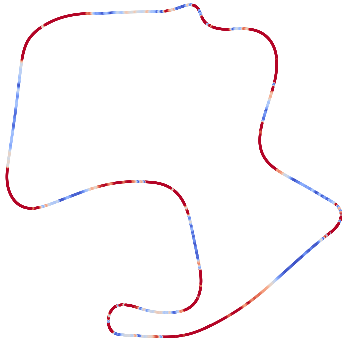
while the remaining thirty percent was reserved for validation. Two separate test datasets were created from distinct sessions. The first test dataset is derived from a different session from the same event with a different driver but with the same vehicle, a 1963 Corvette Grand Sport. The second test dataset comes from the 2014 Targa 66 event at PBI Raceway, offering data from an alternate track, vehicle, and driving conditions. We manually excluded the beginning and end of each session where the vehicle was stationary or moving at low speed, as a car typically does not experience slippage at low speeds. For the performance evaluation of the UKF, only one complete lap from each test dataset was utilized; thus, Test 3 includes one lap from Test 1, and Test 3 comprises one lap from Test 2. Key details of the processed dataset are summarized in Table 2. The track layouts for Test 3 and Test 4 are shown in Fig. 1, where the highlighted sections correspond to regions with high absolute sideslip angle values.

Table 2: Processed datasets

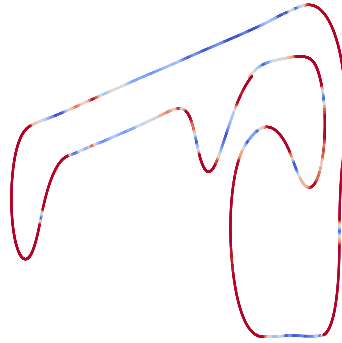
Dataset Name	Raceway	Duration	Purpose
Train	LC	0:21:30	Train and Validation
Test 1	LC	0:17:00	Models Evaluation
Test 2	PBI	0:11:30	Models Evaluation
Test 3	LC	0:01:44	UKF Evaluation
Test 4	PBI	0:01:43	UKF Evaluation

5.4 Neural Networks Implementation

Both the NN and the PINN were implemented using the PyTorch framework. Both networks consist of feed-forward layers with Gaussian Error Linear Units (GELU) activation functions. In both cases, networks were trained using the Adam optimizer and a mean squared error (MSE) loss function. The hyperparameter tuning was performed on the training and validation parts of the Train dataset. The tuned hyperparameters were selected based on the lowest validation loss. For the NN the hyperparameters are: number of layers, number of neurons, learning rate, and batch size. For the PINN, in addition to these hyperparameters, the physics-informed loss weight w_F and the number of collocation points were also optimized.



(a) Laguna Seca Raceway (Test 3)



(b) Palm Beach International Raceway (Test 4)

Figure 1: Track layouts of the test datasets: (a) Laguna Seca Raceway (Test 3) and (b) Palm Beach International Raceway (Test 4).

Collocation points can be generated using various strategies, such as random sampling, Latin Hypercube Sampling (LHS), or grid-based sampling. In our experiments, we employed LHS to generate collocation points dynamically within each PINN training iteration. Specifically, for each training loop, we generate N_c collocation points, where $N_c = k_c \cdot \text{BatchSize}$. Here, BatchSize is the training batch size, and k_c is a tunable hyperparameter controlling the number of collocation points per batch.

5.5 UKF Implementation

The UKF was implemented using the FilterPy library [29], which provides a flexible framework for Kalman filtering in Python. Although the kinematic vehicle model (KM) is linear, allowing us to use the standard KF, we opted to use the UKF for all three configurations for consistency and a fair comparison. The UKF was configured with the same state vector and measurement vector across all configurations:

- State vector: $x = [v_x, v_y]^T$ where v_x is the longitudinal velocity and v_y is the lateral velocity.

- Measurement vector: $z = [v_x]^T$ where only the longitudinal velocity is measured.
- Control input vector: $u = [a_x, a_y, r]^T$ where a_x is the longitudinal acceleration, a_y is the lateral acceleration, and r is the yaw rate.

The state transition function $F[\cdot]$ varies depending on the configuration:

- For the UKF with KM, $F[\cdot]$ is defined by the kinematic vehicle model from Eq. (12).
- For the UKF with NN, $F[\cdot]$ is defined by the trained NN model.
- For the UKF with PINN, $F[\cdot]$ is defined by the trained PINN model.
- The measurement function $H[\cdot]$ is defined as $H(x) = [v_x]$ for all configurations.

For the UKF with KM the estimated lateral velocity v_y is set to zero when $|r| < 0.01 \text{ rad/s}$, to reduce possible drift during straight-line motion.

For all three configurations, the UKF was initialized with the same following parameters. Initial state: $[v_x, 0]$ where v_x is the initial longitudinal velocity from the dataset and the initial lateral velocity is set to zero.

Initial covariance: $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, process noise covariance matrix $Q = \begin{bmatrix} 0.005 & 0 \\ 0 & 0.003 \end{bmatrix}$,
measurement noise covariance matrix $R = 0.002$

6 Experimental Results

6.1 Models Performance Evaluation

The hyperparameters of the neural network (NN) were optimized using the Optuna framework [30]. For the PINN, the same hyperparameters as the NN were adopted, while additional parameters were determined through manual tuning. The tuned hyperparameter configurations for both models are summarized in Table 3.

Table 3: Optimal hyperparameter settings for NN and PINN

Hyperparameter	Neural Network	PINN
Number of layers	2	2
Neurons per layer	64	64
Learning rate	0.00068	0.0007
Number of epochs	45	45
Batch size	128	128
Physics loss weight w_F	N/A	0.1
Collocation points factor k_c	N/A	64

After hyperparameter tuning, both models were trained on the complete Train dataset using the optimized hyperparameter set. Model performance was evaluated on the Train dataset and two test datasets, Test 1 and Test 2. To account for variability, the training and evaluation procedures were repeated ten times. The root mean square error (RMSE) was computed separately for the longitudinal velocity and the lateral velocity. Table 4 presents the mean and standard deviation of RMSE values across the ten runs, while Fig. 2 illustrates the results using a box plot.

Table 4: RMSE performance comparison (mean $\pm$ standard deviation, 10 runs)

Model	Dataset	RMSE v_x (m/s)	RMSE v_y (m/s)
Neural Network	Train	0.0414 ± 0.0055	0.0325 ± 0.0002
Neural Network	Test 1	0.0624 ± 0.0085	0.0449 ± 0.0005
Neural Network	Test 2	0.0528 ± 0.0057	0.0279 ± 0.0018
PINN	Train	0.0312 ± 0.0030	0.0333 ± 0.0003
PINN	Test 1	0.0391 ± 0.0028	0.0466 ± 0.0004
PINN	Test 2	0.0391 ± 0.0047	0.0271 ± 0.0015

The results indicate that the PINN model provides superior accuracy in predicting v_x in all three cases. The RMSE for v_y is comparable in the Train dataset, slightly higher in Test 1 and smaller in Test 2. In addition, the PINN demonstrates a notably lower standard deviation, highlighting improved robustness and reproducibility compared to the NN. The overall improvement in both models during Test 2 can be attributed to the raceway geometry, as the

PBI raceway contains a long straight section where the models achieve higher prediction accuracy.

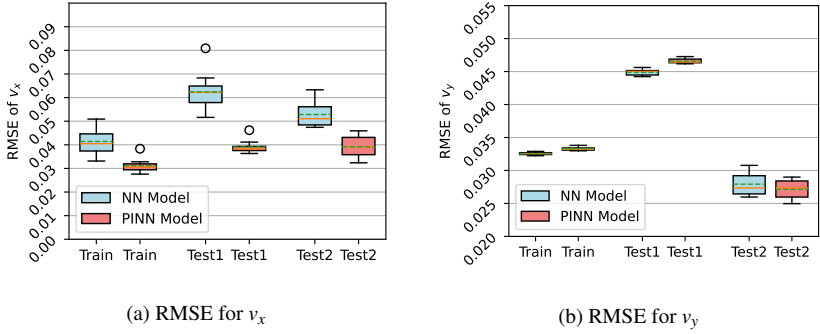


Figure 2: RMSE comparison of neural network and PINN on train and test datasets for (a) v_x and (b) v_y

6.2 UKF Performance Evaluation

Three UKF configurations were evaluated: UKF with a kinematic vehicle model (UKF+KM), UKF with the NN (UKF+NN), and UKF with the PINN (UKF+PINN) on the Test 3 and Test 4 datasets. For the combinations UKF with NN and UKF with PINN, the best-performing models from the previous experiments were used. The estimated lateral velocity v_y was used to compute the sideslip angle β according to Eq. (13). In this experiment, the improved accuracy of the PINN in predicting v_x provided no advantage, since v_x is a measured state. The RMSE between estimated and measured β values is reported in Table 5, and the corresponding comparisons are shown in Fig. 3.

Table 5: RMSE of sideslip angle β estimation

Model	Test 3 RMSE β (deg)	Test 4 RMSE β (deg)
UKF + PINN	1.03	0.68
UKF + NN	0.97	0.73
UKF + KM	1.22	1.19

The UKF combined with the kinematic model (UKF+KM) shows higher RMSE values for both tests, as the KM is a simple vehicle model and provides baseline results. On the Test 3 dataset, the combination of UKF with NN achieves better performance. However, on the Test 4 dataset, which features a different raceway geometry, the combination of UKF with PINN shows the lowest RMSE. These results are consistent with the previous experiments, as the accuracy of β estimation primarily depends on the accuracy of v_y prediction. In both experiments, the PINN and the combination of UKF with PINN demonstrate superior performance on datasets with raceway geometry different from the Train dataset, highlighting the improved generalizability of the PINN compared to the NN.

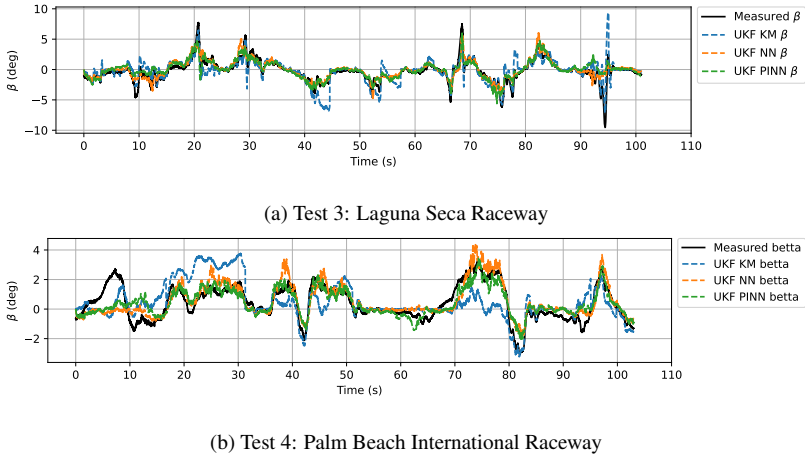


Figure 3: Sideslip angle estimation results for (a) Test 3 and (b) Test 4 datasets. The plots show the true sideslip angle (ground truth) and the estimates from UKF with KM, NN, and PINN models.

7 Discussion

The experimental results provide several important insights into the application of PINNs for state estimation problems, highlighting both potential benefits and key challenges encountered during development and evaluation.

The primary objective of hybrid modeling using PINNs is to integrate the complementary strengths of knowledge-based and data-driven approaches, combining the generalizability of physical models with the predictive accuracy of neural networks. In the present study, however, coupling with a simple physical model may reduce the accuracy of the final solution. Therefore, careful tuning of the balance between the data-driven and physics-informed loss terms, together with an effective collocation point selection strategy, is essential for achieving optimal model performance and emerged as a critical success factor.

In this study, the physics loss weight w_F was determined manually, a process that is both computationally intensive and time-consuming. Additionally, the superior generalization capability expected from PINNs is not evident from standard validation metrics based on conventional training/validation splits.

8 Conclusion and Future Work

This paper presented a combined approach for nonlinear state estimation by integrating Physics-Informed Neural Networks (PINNs) with the Unscented Kalman Filter (UKF). The experimental results on vehicle sideslip angle estimation using real-world data demonstrate that the proposed PINN-enhanced UKF achieves improved estimation accuracy and generalization compared to the UKF with physics-based and purely data-driven models. These findings highlight the potential of PINNs for robust state estimation in real-world engineering systems with limited and noisy measurements. Due to the ongoing development of PINNs for control and state estimation tasks, numerous research questions in this area remain unanswered. Several promising avenues for future research emerge from this work:

- **Advanced Collocation Strategies:** Development of adaptive collocation point selection algorithms that optimize sampling density could significantly improve training efficiency and model accuracy.
- **Multi-Physics Integration:** Extension of the framework to incorporate multiple physical models or combine different levels of model fidelity

could enhance the approach's applicability to complex engineering systems.

- **Simultaneous State and Parameter Estimation:** Extension of the framework to jointly estimate unknown system states and physical parameters could broaden applicability to systems with time-varying or uncertain model parameters.

References

- [1] Y. Bai, B. Yan, C. Zhou, T. Su, and X. Jin, “State of art on state estimation: Kalman filter driven by machine learning”. In: *Annual Reviews in Control*, vol. 56, p. 100909. 2023.
- [2] T. D. Barfoot, “State Estimation for Robotics”, 1st ed. Cambridge University Press 2017.
- [3] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems”. In: *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45. 1960.
- [4] J. Liu, Z. Wang, L. Zhang, and P. Walker, “Sideslip angle estimation of ground vehicles: a comparative study”. In: *IET Control Theory & Applications*, vol. 14, no. 20, pp. 3490–3505. 2020.
- [5] M. Rudolph, S. Kurz, B. Rakitsch, “Hybrid modeling design patterns”. In: *Journal of Mathematics in Industry*, vol. 14, no. 1, p. 3. 2024.
- [6] S. Kim, I. Petrunin, and H.-S. Shin, “A Review of Kalman Filter with Artificial Intelligence Techniques”. In: *2022 Integrated Communication, Navigation and Surveillance Conference (ICNS)*, Dulles, VA, USA, pp. 1–12. 2022.
- [7] S. J. Julier, J. K. Uhlmann, “New extension of the Kalman filter to nonlinear systems”. In: *AeroSense '97*, Orlando, FL, USA, p. 182. 1997.
- [8] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations” 2017.
- [9] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations” 2017.
- [10] M. Raissi, P. Perdikaris, and G. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse

- problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics*, vol. 378, pp. 686–707. 2019.
- [11] J. Blechschmidt, O. G. Ernst, “Three ways to solve partial differential equations with neural networks — A review”. In: *GAMM-Mitteilungen*, vol. 44, no. 2, p. e202100006. 2021.
 - [12] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, “Physics-informed machine learning”. In: *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440. 2021.
 - [13] S. Cuomo, V. S. di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, “Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What’s next”. 2022.
 - [14] A. Farea, O. Yli-Harja, and F. Emmert-Streib, “Understanding Physics-Informed Neural Networks: Techniques, Applications, Trends, and Challenges”. In: *AI*, vol. 5, no. 3, pp. 1534–1557. 2024.
 - [15] E. A. Antonelo, E. Camponogara, L. O. Seman, E. R. de Souza, J. P. Jordanou, and J. F. Hubner, “Physics-Informed Neural Nets for Control of Dynamical Systems”. 2021.
 - [16] E. A. Antonelo, E. Camponogara, L. O. Seman, J. P. Jordanou, E. R. de Souza, and J. F. Hübner. “Physics-informed neural nets for control of dynamical systems”. In: *Neurocomputing* vol. 579, p. 127419. 2024.
 - [17] F. Arnold, R. King. “State–space modeling for control based on physics-informed neural networks”. In: *Engineering Applications of Artificial Intelligence*, vol. 101, p. 104195. 2021.
 - [18] G. Gokhale, B. Claessens, and C. Develder. “Physics informed neural networks for control oriented thermal modeling of buildings”. In: *Applied Energy*, vol. 314, p. 118852. 2022.
 - [19] J. Nicodemus, J. Kneifl, J. Fehr, and B. Unger. “Physics-informed Neural Networks-based Model Predictive Control for Multi-link Manipulators”. In: *IFAC-PapersOnLine*, vol. 55, no. 20, pp. 331–336. 2022.

- [20] J. E. Kittelsen, E. A. Antonelo, E. Camponogara, and L. S. Imsland. “Physics-Informed Neural Networks with skip connections for modeling and control of gas-lifted oil wells”. In: *Applied Soft Computing*, vol. 158, p. 111603. 2024.
- [21] S. Sanyal, K. Roy. “RAMP-Net: A Robust Adaptive MPC for Quadrotors via Physics-informed Neural Network”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*, London, United Kingdom: IEEE, pp. 1019–1025. 2023.
- [22] A. J. Dave, R. B. Vilim. “Physics-informed State-space Neural Networks for transport phenomena”. In: *Engineering Applications of Artificial Intelligence*, vol. 134, p. 108245. 2024.
- [23] J. de Curtò, I. de Zarzà. “Hybrid State Estimation: Integrating Physics-Informed Neural Networks with Adaptive UKF for Dynamic Systems”. In: *Electronics*, vol. 13, no. 11, 2024.
- [24] J. Farrelly, P. Wellstead. “Estimation of vehicle lateral velocity”. In: *Proceedings of the 1996 IEEE International Conference on Control Applications*, Dearborn, MI, USA: IEEE, pp. 552–557. 1996.
- [25] G. Righetti, B. Lenzo. “A Combined Dynamic–Kinematic Extended Kalman Filter for Estimating Vehicle Sideslip Angle”. In: *Applied Sciences*, vol. 15, no. 3, p. 1365 2025.
- [26] D. Selmanaj, M. Corno, G. Panzani, and S. M. Savaresi. “Vehicle sideslip estimation: A kinematic based approach”. In: *Control Engineering Practice*, vol. 67, pp. 1–12. 2017.
- [27] Kegelmann, John C. “Revs Vehicle Dynamics Database”. URL: <https://searchworks.stanford.edu/view/cv083fq4881> Accessed: 2024-06-20.
- [28] B.-C. Chen and F.-C. Hsieh. “Sideslip angle estimation using extended Kalman filter”. In: *Vehicle System Dynamics*, vol. 46, no. sup1, pp. 353–364. 2008.

- [29] “FilterPy: Kalman filtering and optimal estimation library”. URL: <https://github.com/rlabbe/filterpy> Accessed: 2024-06-20.
- [30] A. Takuya and S. Shotaro “Optuna: A Next-generation Hyperparameter Optimization Framework”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 2019.

Four Benchmark Datasets for Nonlinear Regression in Engineering Sciences

Felix Wittich¹, Farzad Rezazadeh¹, Andreas Kroll¹

¹Department of Measurement and Control, University of Kassel

Mönchebergstr. 7, 34125 Kassel, Germany

E-Mail: felix.wittich, farzad.rezazadeh, andreas.kroll@mrt.uni-kassel.de

1 Introduction

This paper presents four datasets designed as benchmarks for nonlinear regression tasks in engineering applications. Benchmark datasets provide a consistent reference point and help align researchers on shared methodological challenges [1–3]. Four datasets related to manufacturing and material science are presented, covering building materials (concrete), subtractive and additive metallic manufacturing as well as a complex calibration problem. Each dataset originates from laboratory measurements and is briefly described below. All datasets are publicly accessible.

2 Dataset 1: Ultra-High-Performance Concrete

Ultra-high-performance concrete (UHPC) production is a costly process because of its 28-day curing time, the required destructive testing, and expensive raw materials limit the number of batches that can be produced. The production process also involves numerous controllable and uncontrollable factors.

This section introduces a publicly available dataset of UHPC production experiments [4] designed for regression tasks in a high-dimensional and sparse input space. The data capture potential variations in raw material quality, dosing errors, mixing regimes, fresh concrete properties, and curing conditions that

may cause non-reproducibility of the UHPC product at the desired quality, even when the same UHPC recipe is used to produce batches.

2.1 Experimental Design and Data Collection

To ensure systematic coverage of the high-dimensional input space, a three-phase design of experiments framework is implemented [5]. Fifteen controllable factors affecting UHPC quality are identified, including material quality (cement reactivity, graphite content, ingredient moisture and temperature), particle-size distributions, measurement errors (sand and filler contents, superplasticizer dosage), mixing conditions (speed and duration), and curing conditions.

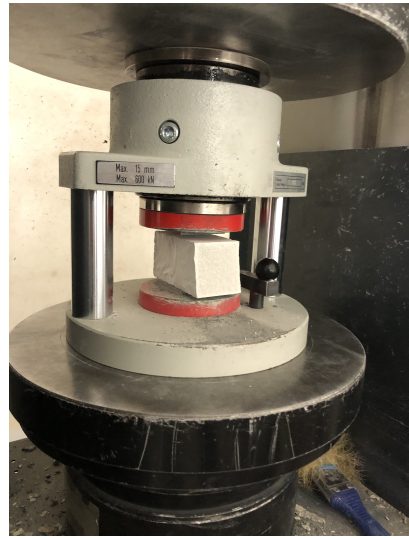
The screening phase used a Taguchi L50 orthogonal array to allocate features across 50 experiments. This design ensures balanced level distributions and is well suited for high-dimensional, small-sample-size experimental designs [6]. Analysis of the screening data identified three factors – cement reactivity, mixing speed, and mixing duration – as negligible, while curing conditions emerged as the most influential determinants of mechanical performance [7]. Therefore, cement reactivity, mixing speed, and mixing duration were removed from further analysis. In the modeling phase, the design matrix is augmented using Latin hypercube sampling (LHS). This augmentation identifies optimal locations for 50 experiments in addition to the 50 runs from the screening phase, resulting in a more uniform distribution of the input variables [8,9]. Finally, the domain-optimization phase replicated each experiment from the modeling phase under two curing condition-sets – one chosen from the LHS design and another based on expert knowledge – to emphasize the critical role of curing in UHPC quality.

Fresh UHPC from each experiment was cast into three specimens and cured under the specified conditions. Flexural strength was measured first; each specimen was then halved, and the halves were used for compressive strength testing (Fig. 1). Accordingly, for each experiment, flexural strength is reported as the average of the three specimens, and compressive strength as the average of the six halves. Additional measurements include the average power consumption

during mixing and five fresh concrete properties – temperature, air content, slump-flow velocity, electrical conductivity, and funnel runtime.



(a) Flexural strength (FS) testing of UHPC specimens prior to halving.



(b) Compressive strength (CS) testing on the six halves obtained from the flexural specimens.

Figure 1: Mechanical testing sequence per experiment: FS is measured first on three specimens; each specimen is then halved and the six halves are used for CS.

2.2 Dataset Characteristics

The dataset comprises 18 variables, which can be grouped into five categories (see Tab. 1 and Fig. 2):

Raw material quality and storage conditions: variables describing the state of raw materials, such as graphite impurity content, ingredient moisture and temperature. These factors capture how storage conditions and impurities influence UHPC workability and strength.

Particle-size distributions and measurement errors: four continuous features quantify deviations from nominal filler and sand contents and superplasticizer

dosage. These values simulate potential dosing errors in raw materials and variations in particle packing density of UHPC.

Mixing condition: the mixing condition is captured through the average mixer power consumption recorded during each batch, which characterizes mixing energy.

Fresh concrete properties: five variables – temperature, air content, slump flow velocity, electrical conductivity, and funnel runtime – are measured for each experiment directly after the mixing process. These properties provide early indicators of workability and rheology that may correlate with final material strengths.

Curing conditions: four categorical features represent the temperature and relative humidity during the first day (*CT1*, *CC1*) and days 2–28 (*CT28*, *CC28*) of curing. By varying the curing environment and temperature, the dataset captures real-world variabilities that influence hydration and strength development.

The outputs include early-age (28-hour) and 28-day compressive and flexural strengths.

Missing values are represented as empty cells; outlier experiments flagged by domain experts are labelled for exclusion [4].

3 Dataset 2: Surface Integrity

Surface finishing machining processes are well-known to strongly influence the near-surface properties of a component, e.g. the residual stress state and the surface hardness. As the surface usually represents the area being subject to highest stresses, especially in terms of highly-loaded components, the condition of the near-surface layer has a significant influence on the mechanical properties of the component (this is also referred to as surface integrity).

This dataset [10] shows the relationship between cutting parameters in the subtractive manufacturing process of hard turning of 51CrV4 steel and the resulting near surface properties residual stress, hardness and roughness.

Table 1: Summary of inputs and outputs in the UHPC dataset. Variables are grouped by material quality, particle-size distributions and measurement errors, mixing condition, fresh concrete properties, curing conditions, and mechanical properties. Minimum and maximum values are shown for each factor.

Group	Variable	Symb.	Unit	Min.	Max.
Material quality	Ingredient moisture	<i>IM</i>	kg	2.92	3.36
	Ingredient/Water temperature	<i>IT</i>	°C	10.00	40.00
	Graphite	<i>GRP</i>	kg	0.00	0.22
Particle size distributions & measurement errors	Sand I	<i>SAI</i>	kg	5.10	6.90
	Sand II	<i>SAII</i>	kg	8.92	12.07
	Filler I	<i>FLI</i>	kg	5.10	6.90
	Filler II	<i>FLII</i>	kg	0.63	0.86
	Superplasticizer	<i>SPP</i>	kg	0.29	0.35
Mixing condition	Average power consumption	<i>APW</i>	kW	0.36	1.40
Fresh concrete properties	Fresh concrete temperature	<i>FCT</i>	°C	17.60	33.30
	Electrical conductivity	<i>EC</i>	V	4.541	4.745
	Air content	<i>AC</i>	%	0.40	7.00
	Slump flow	<i>SF</i>	mm	215.00	395.00
	Funnel runtime	<i>FR</i>	s	4.00	24.10
Curing conditions	Curing temperature, day 1	<i>CT1</i>	°C	10.00	40.00
	Curing class, day 1	<i>CCI</i>	Class	1	2
	Curing temperature, days 2–28	<i>CT28</i>	°C	10.00	40.00
	Curing class, days 2–28	<i>CC28</i>	Class	1	2
Outputs	Compressive strength at day 1	<i>CS1</i>	MPa	11.36	98.23
	Flexural strength at day 1	<i>FS1</i>	MPa	1.98	12.23
	Compressive strength at day 28	<i>CS28</i>	MPa	85.06	135.71
	Flexural strength at day 28	<i>FS28</i>	MPa	8.15	24.57

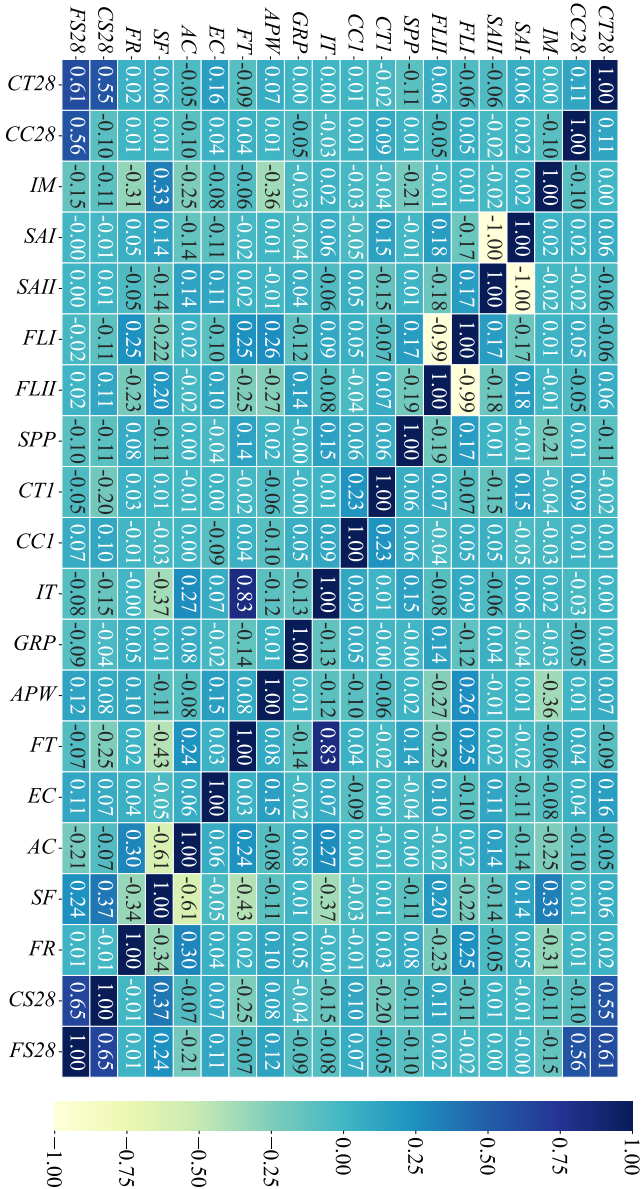


Figure 2: Correlation matrix plot of the input variables for the UHPC dataset (Pearson correlation coefficient). Symbols are defined in Table 1.

3.1 Experimental Design

A full factorial experiment design was used with three levels for the initial hardness levels HV_{init} and the cutting parameters. Hard-turning experiments were conducted on cylindrical specimens made of a quenched and tempered (Q&T) steel 51CrV4 in different initial surface hardness levels, i. e. 400 HV30, 500 HV30 and 600 HV30. Prior to heat treatment, all specimens were manufactured with the same roughness requirements in order to ensure an almost identical surface finish before hard-turning. Three different sections of nine specimen of each hardness level (i.e. in total 27 sections per hardness level) were machined with varied cutting parameters feed rate (f), depth of cut (a_p) and cutting speed (v_c). Hard-turning of the specimens was carried out on a servo-conventional lathe of type Weiler C30 using polycrystalline boron nitride (PCBN) inserts with a corner radius of 0.8 mm. After hard-turning, residual stress depth profiles were determined using X-ray diffraction (XRD), respectively for each specific area of a specimen. Residual stress measurements were conducted using a Pulstec μ -X360 diffractometer equipped with a 0.3 mm collimator and $CrK\alpha$ -radiation with an exposure time of 120 sec. Depth profiles were determined by successive removal of the material surface layer using electro-chemical polishing. The obtained data have been evaluated applying the $\cos\alpha$ -method without consideration of any mathematical stress correction.

Post process surface roughness in axial direction was determined using a Mitutoyo SJ-210 tactile roughness measuring device. Post process Vickers hardness testing was carried out using a Struers DuraScan-70 system employing a load of 294.2 N (HV30). For more detailed information on the material and the experimental setup, i.e., chemical composition, specimen geometry, machines and parameters used for hard-turning operations and post process measurements, the reader is referred to [11].

3.2 Dataset Characteristics

The dataset contains the three machining parameters and the initial hardness as input variables. The full factorial experiment design results in 81 different combinations. Three measured parameters are available as output

Table 2: Summary of inputs and outputs in the surface integrity dataset. Input variables are grouped by the initial state and machining parameters. For the input variables, the three factorial levels are given. For the output variables, the resulting minimum and maximum values are shown.

Group	Variable	Symb.	Unit	Levels	Min.	Max.
Initial state	Initial Hardness	HV_{init}	HV	400, 500, 600	–	–
Machining parameters	Feed	f	mm	0.05, 0.25, 0.5	–	–
	Cutting Speed	v_c	m/min	100, 175, 250	–	–
	Depth of Cut	a_p	mm	0.05, 0.25, 0.5	–	–
Outputs	Surface Hardness	HV	HV	–	480	638
	Surface Roughness	R_z	μm	–	0.951	14.393
	Surface Roughness	R_a	μm	–	0.141	3.614
	Residual Stress	σ	MPa	–	–891	720

variables: surface roughness, surface hardness and residual stress. Roughness and Hardness are measured at the surface, resulting in 81 data points each. For the residual stress depth profile, measurements for 12 different depths are available: $\{0, 10, 20, 30, 40, 50, 60, 80, 100, 120, 150, 200\}$ μm , providing for $N = 972$ samples in total. The summary of the dataset is given in Tab. 2.

4 Dataset 3: Micromagnetic Sensor

Key properties of workpieces such as residual stress are currently measured using time-consuming and/or destructive methods like X-ray diffraction (as in case of dataset 2) or the hole-drilling method. Micromagnetic (MM) material characterization allows for non-destructive, in-process evaluation of the mechanical properties of ferromagnetic components.

This dataset [12] is used for calibrating a MM sensor. It contains 82 features that were measured using an MM sensor as input variables and corresponding reference measurements of the residual stress and hardness as outputs.

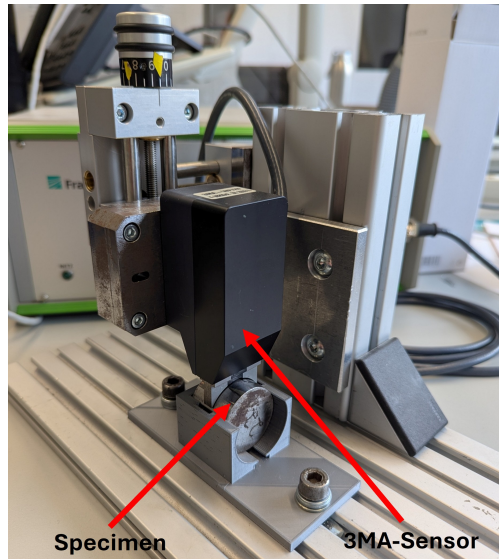


Figure 3: Ex situ micromagnetic measurements on the machined sections of the cylindrical specimens.

4.1 Experimental Design

The same specimen from the hard turning experiment in Section 3 were reused to conduct the MM measurements. Therefore, no special experimental design was used. The MM measurements were carried out using a 3MA-II device from Fraunhofer IZFP. The sensor utilizes four measurement methods: Barkhausen noise, incremental permeability, multi-frequency Eddy current analysis and harmonic analysis of the tangential magnetic field strength, resulting in 41 features that are extracted from the measurement curves. By using two different magnetization amplitudes, 82 input variables are available in total. The measurements were performed ex situ on the machined sections of the cylindrical specimens, see Fig. 3. For more details on the sensor and the experimental setup, please refer to [13, 14].

4.2 Dataset Characteristics

The dataset includes 81 data points each with 82 input variables and 3 output variables. The pairwise correlation of the 82 input variables is shown in Fig. 4. A description of the input variables is given in Tab. 3. As output variables the surface hardness is available and two characteristic values for the near surface residual stress state: the residual stress on the surface, i.e. at depth 0 μm and the maximum compressive residual stress. These values were extracted from the residual stress depth profile, see Section 3.2.

5 Dataset 4: Additive Manufacturing

Additive manufacturing (AM) is an important process for producing complex and customized geometries. One of the most common processes of AM is laser powder bed fusion (PBF-LB/M), where metal powder is deposited in thin layers and selectively fused by a laser. One way to increase the manufacturing speed is to increase the layer thickness. However, this can negatively affect the porosity of the components. This dataset [15] concerns the relationship between process parameters and the resulting porosity of the workpiece. A standard industrial PBF-LB/M system equipped with a 400 W laser was employed to produce parts of AlSi10Mg using different parameter combinations.

5.1 Experimental Design

In order to explore a wide value range of process parameters and to allow for investigating higher order nonlinearities in the model, a space filling experiment design following a Latin hypercube sampling (LHS) design was used. In LHS, each factor has as many levels as there are runs in the design. Thus, for n process parameters each dimension of the n -dimensional design space is divided into p levels such that there is only one sample at each level. In order to determine the location of each sample, a random procedure was used, where in each iteration the sample locations are randomly distributed and permuted. Afterwards, the minimum distance between each sample is maximized and the

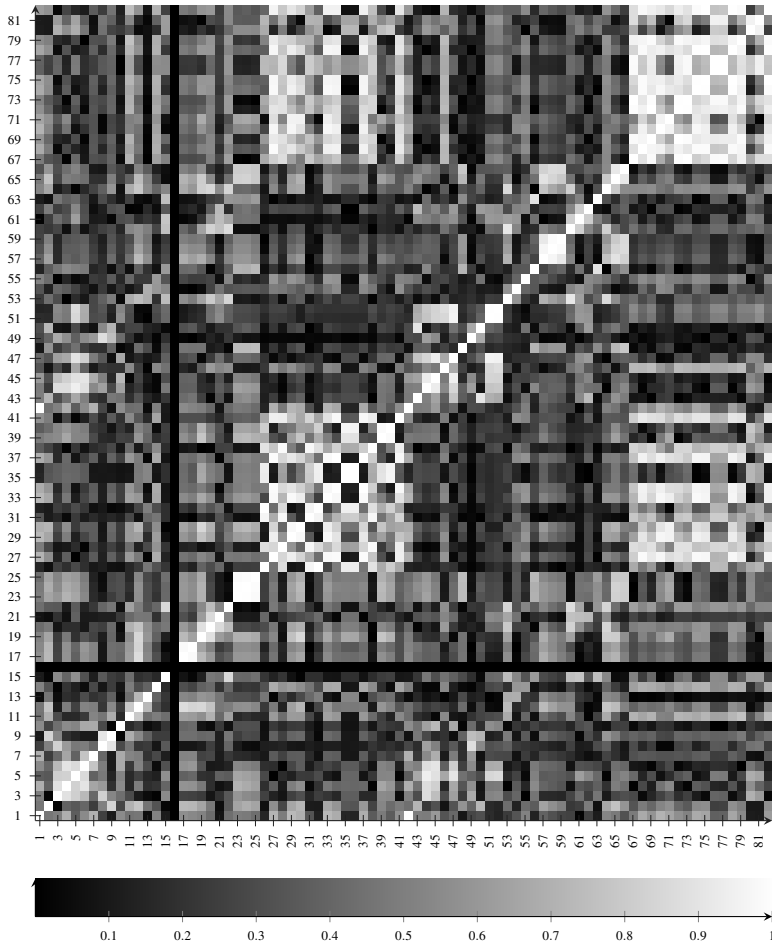


Figure 4: Correlation matrix plot of the input variables for the micromagnetic sensor dataset (Pearson correlation coefficient). Numbering is defined in Tab. 3 .

Table 3: Overview of the input variables (identifiers) and their explanations. Variables are grouped by measurement principle.

Group	Variable (Identifier)	Unit	Explanation	Nr.
Harmonic analysis of tangential magnetic field	A3, A5, A7	A/cm	Amplitudes of the 3rd, 5th, and 7th harmonics	1–3
	P3, P5, P7	rad	Phase shifts of the 3rd, 5th, and 7th harmonics relative to the fundamental wave	4–6
	Vmag	V	Final stage voltage of the electromagnet	7
	UHS	A/cm	Sum of all harmonic amplitudes (including 3rd, 5th, and 7th)	8
	K	%	Total harmonic distortion factor (measure of distortion)	9
	Hco	A/cm	Coercive field strength from harmonic analysis	10
	Hro	A/cm	Harmonic content of the magnetic field strength at zero crossing	11
Barkhausen noise	Mmax	V	Maximum noise amplitude	12
	Mmean	V	Noise amplitude averaged over one signal period	13
	Mr	V	Noise amplitude at the remanence point	14
	Hcm	A/cm	Coercive field strength from Barkhausen noise (field strength at the noise maximum)	15
	DH25m	A/cm	Curve broadening at 25 % of curve height	16
	DH50m	A/cm	Curve broadening at 50 % of curve height	17
	DH75m	A/cm	Curve broadening at 75 % of curve height	18
Incremental permeability	umax	V	Maximum superposition permeability	19
	umean	V	Superposition permeability averaged over one signal period	20
	ur	V	Permeability at the remanence point	21
	Hcu	A/cm	Coercive field strength (field strength at the noise maximum)	22
	DH25u	A/cm	Superposition permeability at 25 % of curve height	23
	DH50u	A/cm	Superposition permeability at 50 % of curve height	24
	DH75u	A/cm	Superposition permeability at 75 % of curve height	25
Multi-frequency eddy current	Re1–Re4	V	Real parts of the operating frequencies	26–29
	Im1–Im4	V	Imaginary parts of the operating frequencies	30–33
	Mag1–Mag4	V	Magnitudes of the operating frequencies	34–37
	Ph1–Ph4	rad	Phases of the operating frequencies	38–41

levels are adjusted to the lower and upper bound of each factor. The samples were divided into six groups of 24 samples each. Each group was built as one build job using a SLM 280HL selective laser melting system from SLM Solutions Group AG. After the process all samples were examined using an optical microscope and the porosity was determined using image processing software. Details on the experimental procedure can be found in [16].

5.2 Dataset Characteristics

The dataset includes 112 different parameter combinations and 144 samples. Due to limitations regarding the choice of the layer thickness s , this quantity was chosen as $s \in \{30, 45, 60, 90\} \mu\text{m}$. For each layer thickness, the influence of the laser power P , the scan speed v_s , and the hatching distance h was investigated in wide ranges. The variables are described in Tab. 4. In addition to the process parameters as input variables and porosity in % as the target variable, the dataset contains further parameters, derived from the process parameters:

Energy density:

$$E = \frac{P}{v_s \cdot h \cdot s}$$

Line Energy:

$$L = \frac{P}{v_s}$$

Build Rate:

$$B = v_s \cdot h \cdot s$$

In addition, a discrete variable is included that describes the defect mode causing the porosity. The following defects are possible: stripe pores, keyholes, lack of fusion.

Table 4: Summary of inputs and outputs in the additive manufacturing dataset. Input variables are grouped as process parameters. For inputs and outputs, the minimum and maximum values across the dataset are shown.

Group	Variable	Symb.	Unit	Min.	Max.
Process parameters	Layer thickness	s	μm	30	90
	Laser power	P	W	200	400
	Scan speed	v_s	mm/s	370	1570
	Hatching distance	h	mm	0.1	0.4
Outputs	Porosity	–	%	0	37.41

6 Conclusions

In this contribution four publicly available datasets in the area of manufacturing and material science were presented. The datasets are particularly suitable for evaluating regression methods under sparse conditions. The datasets can, for example, be used for the following purposes, among other applications:

- **Sparse regression modeling (Dataset 1–4):** Testing the efficacy of various regression algorithms (e.g., Gradient Boosting, Support Vector Regression, Elastic Net [18], and ensemble methods [19]) in predicting continuous mechanical properties in a complex, sparse input space.
- **Benchmarking of feature-selection algorithms (Dataset 1 and 3):** The small number of observations relative to the number of variables provides a test bed for determining feature importance as well as for single- and multi-objective feature selection approaches [5, 17].

Acknowledgment

Funded in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project number 532921704. We thank G.tecz Engineering GmbH for carrying out the experiments for dataset 1 and the Institute of

Materials Engineering, Metallic Materials, University of Kassel for carrying out the experiments for dataset 2,3 and 4.

References

- [1] F. Hoffmann, T. Bertram, R. Mikut, M. Reischl, und O. Nelles. “Benchmarking in classification and regression.” In: *WIREs Data Mining and Knowledge Discovery*, 9(5), e1318 2019.
- [2] A. Kroll, H. Schulte. “Benchmark problems for nonlinear system identification and control using Soft Computing methods: need and overview.” In: *Applied Soft Computing*, 25, S. 496–513. 2014.
- [3] F. Hoffmann, R. Mikut, A. Kroll, M. Reischl, O. Nelles, H. Schulte, T. Bertram. “Computational Intelligence: State-of-the-Art Methoden und Benchmarkprobleme.” In: *Proceedings – 22. Workshop Computational Intelligence: Dortmund, 6.–7. Dezember 2012*, S. 1–42. KIT Scientific Publishing. 2012.
- [4] F. Rezazadeh, A. Dürrbaum, A. Abrishambaf, G. Zimmermann und A. Kroll. “Mechanical Properties of Ultra-High Performance Concrete (UHPC).” Datensatz. Kassel: DaKS — Research Data Repository, University of Kassel. 2025.
- [5] F. Rezazadeh P., A. Abrishambaf, A. Dürrbaum, G. Zimmermann und A. Kroll. “Investigating Reproducibility of Ultra-High Performance Concrete with Consistent Mechanical Properties: A Modeling Pipeline for Complex Manufacturing.” In: *Proceedings – 34. Workshop Computational Intelligence: Berlin, 21.–22. November 2024* (Schulte, H.; Hoffmann, F.; Mikut, R., Hg.), S. 143–148. KIT Scientific Publishing. 2024.
- [6] G. Taguchi. “System of Experimental Design: Engineering Methods to Optimize Quality and Minimize Costs.” White Plains, NY; Dearborn, MI: UNIPUB/Kraus International Publications; American Supplier Institute. 1987.

- [7] F. Rezazadeh P., A. Dürrbaum, G. Zimmermann und A. Kroll. “Leveraging Ensemble Structures to Elucidate the Impact of Factors That Influence the Quality of Ultra-High Performance Concrete.” In: *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, S. 180–187. IEEE. 2023.
- [8] M. Stein. “Large Sample Properties of Simulations Using Latin Hypercube Sampling.” In: *Technometrics* 29.2, S. 143–151. 1987.
- [9] R. Carnell. “optAugmentLHS.R [R script].” GitHub: bertcarnell/lhs. Zugriff: 12.02.2025. 2019.
- [10] F. Wittich, T. Wegener, A. Liehr, W. Zinn, T. Niendorf und A. Kroll. “Surface layer state after hard turning of 51CrV4.” Datensatz. Kassel: DaKS — Research Data Repository, University of Kassel. 2025.
- [11] T. Wegener, A. Liehr, A. Bolender, S. Degener, F. Wittich, A. Kroll und T. Niendorf. “Calibration and validation of micromagnetic data for non-destructive analysis of near-surface properties after hard turning.” In: *Production Engineering*, 18, S. 381–392. 2024.
- [12] F. Wittich, T. Wegener, A. Liehr, W. Zinn, T. Niendorf und A. Kroll. “Micromagnetic measurement of surface layer state of hard turned 51CrV4 steel.” Datensatz. Kassel: DaKS — Research Data Repository, University of Kassel. 2025.
- [13] F. Wittich, T. Wegener, A. Liehr, W. Zinn, T. Niendorf, A. Kroll. “Data-driven prediction of the surface layer condition in hard-turning for controlling component quality.” In: *HTM Journal of Heat Treatment and Materials*, 55(2), S. 152–172. 2022.
- [14] F. Wittich und A. Kroll. “Datengetriebene Modellierung und Merkmalsselektion für die automatische Kalibrierung eines Sensorsystems für die mikromagnetische Materialcharakterisierung.” In: *Proceedings – 34. Workshop Computational Intelligence: Berlin, 21.–22. November 2024*, S. 21–26. KIT Scientific Publishing. 2024.
- [15] A. Engelhardt, M. Kahl, J. Richter, P. Krooß, A. Kroll, and T. Niendorf. “Investigation of processing windows in additive manufacturing

- of AlSi10Mg for faster production utilizing data-driven modeling.” Datensatz. Kassel: DaKS — Research Data Repository, University of Kassel. 2022.
- [16] A. Engelhardt, M. Kahl, J. Richter, P. Krooß, A. Kroll, and T. Niendorf. “Investigation of processing windows in additive manufacturing of AlSi10Mg for faster production utilizing data-driven modeling.” In: *Additive Manufacturing*, 55, S. 102858. IEEE. 2022.
 - [17] F. Rezazadeh P., A. Dürrbaum, G. Zimmermann und A. Kroll. “Holistic Modeling of Ultra-High Performance Concrete Production Process: Synergizing Mix Design, Fresh Concrete Properties, and Curing Conditions.” In: *Proceedings – 33. Workshop Computational Intelligence: Berlin, 23.–24. November 2023*, S. 215–237. KIT Scientific Publishing. 2023.
 - [18] F. Rezazadeh und A. Kroll. “Predicting the Compressive Strength of Concrete up to 28 Days-Ahead: Comparison of 16 Machine Learning Algorithms on Benchmark Datasets.” In: *Proceedings – 32. Workshop Computational Intelligence: Berlin, 1.–2. Dezember 2022*, S. 53–75. KIT Scientific Publishing. 2022.
 - [19] F. Rezazadeh, E. Olfatbakhsh und A. Kroll. “Sign Diversity: A Method for Measuring Diversity in Base Learner Selection for Ensemble Regression.” In: *2025 IEEE Symposium on Computational Intelligence on Engineering/Cyber Physical Systems (CIES)*, S. 1–9. IEEE. 2025.

Investigation of convergence behavior and solution reliability of a Genetic Algorithm for Flexible Job-shop Scheduling

David Hutter, Thomas Steinberger, and Michael Hellwig

JR Centre for Robust Decision Making, Vorarlberg University of Applied Science
Hochschulstrasse 1, 6850 Dornbirn
E-Mail: {david.hutter,thomas.steinberger,michael.hellwig}@fhv.at

Abstract

The flexible job-shop scheduling problem (FJSSP) is frequently used in production planning to optimize manufacturing in given production environments. There are several methods to solve the standard FJSSP and many of its extensions. However, in reality production environments suffer from uncertainties in various forms that can not be appropriately addressed by the majority of the used solving methods. Meta-heuristics, like Genetic Algorithms (GA), can be used to address the FJSSP, for example with simulation-based optimization. Given the stochastic nature of GAs however, it is necessary to examine the stability of its optimization process. This paper analyzes the convergence behavior, the result diversity and the influences of problem characteristics on the performance of a GA used to solve the FJSSP and the FJSSP with worker flexibility (FJSSP-W). It further examines its solutions for potential robustness indicators by comparing different alternative evaluation criteria.

1 Introduction

A frequently considered use case in the manufacturing industry is production scheduling, where a production environment (PE) needs to process different

orders in different ways. The assignment of specific operations and their sequence have an impact on the overall planning efficiency. The type of scheduling problem depends on the capabilities of the PE and the resource requirements for the individual work steps [3, 5, 12]. Generally in a Job-shop Scheduling Problem (JSSP), a number of jobs, typically consisting of ≥ 1 operations with varying processing times, have to be scheduled on the machines (or workstations) available in the PE, while attempting to optimize a certain objective. The most commonly used objective is the total processing time until all jobs are fully processed, i.e. the makespan [3]. In the standard JSSP, the operations of a job need to be scheduled in a predefined sequence with no overlap. Each operation needs to be assigned to a specific machine. The Flexible Job-shop Scheduling Problem (FJSSP) is a common relaxation of the JSSP [3, 10]. For the FJSSP, each operation can be processed on ≥ 1 machine. The machines do not necessarily offer the same processing time for an operation, which means the additional machine assignment problem impacts the processing time of each operation. While this allows for greater planning flexibility, the problem complexity is increased significantly. To bring the problem closer to a real-world use case, there are several extensions to the FJSSP. One such extension is the added consideration of the work force as re-usable resources [3, 10, 11]. The Flexible Job Shop Scheduling Problem with Worker Flexibility (FJSSP-W) considers the assignment of workers to the different operations, in addition to the assignment of machines. The assigned workers and machines in combination affect the processing time of an operation, in order to reflect different skill and experience levels of the assigned workers. This adds even more complexity to the base problem in order to find the ideal operation - machine - worker combinations on top of the best order of operations for the sequencing of all required operations [6].

In prior work [7, 9] a genetic algorithm GA was compared to several other commonly used solvers for the FJSSP and FJSSP-W. While the GA demonstrated the highest resilience towards the increased complexity, the best performing solver was a constraint programming (CP) solver, IBM CPLEX ¹ Since the CP solver is not suitable for the additional complexity of added uncertainties,

¹ Mathematical program solvers - IBM CPLEX - <https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer> Accessed: 19.09.2025

the GA is a better fit for the more realistic problem. However, to establish the reliability and effectiveness and potentially identify improvements that can be made to the GA, a deeper analysis of its stochastic optimization behaviour is necessary. To that end, the GA is evaluated concerning its stability. Additionally, the GA has the capability to collect multiple candidate solutions with the same quality. The diversity of these collections is also analysed. The overall goal is to identify measures to increase the stability of the GA to improve its convergence behavior for a performance closer to the CP solver, as well as gaining a closer understanding on the influence of the rise of problem complexity between the FJSSP and FJSSP-W on the dynamics of the GA. In addition, the GA enables the optimization to account for uncertainties. The uncertainties examined in this case concern stochastic processing times to reflect real life conditions in production environments.

2 Problem Formulation

This section introduces the problem formulation for both the FJSSP and the necessary additions for the FJSSP-W. Both problems use the same set of problem instances, which are given as FJSSP instances and translated to FJSSP-W instances using a repeatable procedure [9].

2.1 FJSSP

The FJSSP is a combinatorial NP-complete optimization problem [4]. The problem requires to schedule a set of jobs consisting of 1 or more operations each onto the available machines of a production environment. For each operation, 1 or more of the available machines are eligible for use. On each of the assigned machines, the operations can have differing processing times. The goal is to determine the order of operations and to assign each operation to a machine such that a certain cost function C is minimized.

The general representation of the FJSSP, using the same notation as [7], introduces a set of n jobs $\mathcal{J} = \{J_1, \dots, J_n\}$. Each of these jobs needs to be processed on a set of m machines $\mathcal{M} = \{M_1, \dots, M_m\}$. Each job J_i ($i = 1, \dots, n$) consists of

a sequence of n_i operations. The operations of each job need to be processed in sequence, and the following operation can only start once the previous operation is fully completed. The notation $O_{i,j}$ refers to the j th operation of job i . For each operation $O_{i,j}$ a machine needs to be assigned from the admissible subset of machines $\mathcal{M}_{i,j} \subset \mathcal{M}$ which contains $m_{i,j}$ machines, which are eligible to process operation $O_{i,j}$. The specific machine assigned to $O_{i,j}$ is denoted by $M_{i,j}$. The total number of operations to be scheduled for the FJSSP can be calculated as $N = \sum_{i=1}^n n_i$. The processing time for operation $O_{i,j}$ on one of its eligible machines $M_k \in \mathcal{M}_{i,j}$ ($k = 1, \dots, m_{i,j}$) is denoted as $T_{i,j,k}$. For the most common scenarios, the FJSSP assumes that all required jobs are known and all machines are available at time 0. Several different objective functions are used in the context of the FJSSP [1, 3]. The most common criteria include minimizing the makespan, minimizing the tardiness, or maximizing the resource utilization. The makespan, i.e. the total time required for a schedule to be completely executed from start to finish, is by far the most commonly used objective to measure a schedule's performance. The objective of the optimization is to find the operation-machine combinations that minimize the *makespan*. Therefore, the cost function becomes a mapping of jobs and associated machines to the total processing time of all operations $C : \mathcal{Y} \rightarrow (0, +\infty)$. Hence, one is interested in the search space vector $\mathbf{y}$ that realizes a minimal $C(\mathbf{y})$ value. The concrete examples of a parameter vector encoding for $\mathbf{y}$ is depending on the modelling approach and will be provided below. Typically for the field of meta-heuristic optimization, the term fitness will be used synonymously for the quality of a potential (feasible) solution $\mathbf{y}$. A feasible solution represents a complete schedule, containing all operations of all jobs that need to be scheduled for the problem instance, with no operation overlap on the same machine and no violation of the sequence dependence of operations within the same job. The quality of two feasible solutions can be compared on the basis of their objective function values. Lower cost function values $C(\mathbf{y})$ correspond to a better schedule $\mathbf{y} \in \mathcal{Y}$. For the comparison of two feasible schedules $\mathbf{y}$ and $\mathbf{z}$, an order relation $\prec_c$ is defined by

$$\mathbf{y}, \mathbf{z} \in \mathcal{Y} : \quad \mathbf{y} \prec_c \mathbf{z} \Leftrightarrow C(\mathbf{y}) < C(\mathbf{z}). \quad (1)$$

That is, in this case the fitness of $\mathbf{y}$ is rated higher than that of $\mathbf{z}$ with respect to the cost function C , which in this case is the makespan of the schedule. The corresponding smaller or equal relation $\preceq_c$ is defined analogously.

For the encoding used by the GA, each schedule $\mathbf{y} = (\mathbf{s}|\mathbf{a}) \in \mathcal{Y} \subset \mathbb{N}^N + \mathbb{N}$ is encoded by two distinct vectors. The sequence vector $\mathbf{s} \in \mathbb{N}^N$, and the assignment vector $\mathbf{a} \in \mathbb{N}^N$, leading to $2N$ optimization variables. The sequence vector $\mathbf{s}$ refers to the order of operations in which each operation is inserted into the schedule on their respective assigned machines, as defined by $\mathbf{a}$. A feasible $\mathbf{s}$ contains each job index i exactly n_i times. The j th appearance of each job index i corresponds to operation $O_{i,j}$. This way, the job dependent sequence requirements of the FJSSP are always fulfilled. The assignment vector $\mathbf{a}$ contains the machine assignments for all operations. The order of the operations in $\mathbf{a}$ is fixed and independent of the sorting in $\mathbf{s}$. For every operation $O_{i,j}$, $\mathbf{a}$ contains a machine index k for machine $M_k \in \mathcal{M}_{i,j}$ to ensure only eligible machine are assigned to each machine.

2.2 FJSSP-W

As an extension to the FJSSP, the FJSSP-W additionally introduces a set of w workers $\mathcal{W} = \{W_1, \dots, W_w\}$. For every operation $O_{i,j}$, there is a subset $\mathcal{W}_{i,j,k} \subset \mathcal{W}$ of $w_{i,j,k}$ workers available for each possible machine $M_k \in \mathcal{M}_{i,j}$. The processing time for the FJSSP-W depends on the specific machine-worker combination for each operation. If operation $O_{i,j}$ is assigned to machine $M_k \in \mathcal{M}_{i,j}$ and worker $W_l \in \mathcal{W}_{i,j,k}$ the processing time is given as $T_{i,j,k,l}$. In addition to the feasibility requirements of the FJSSP, the FJSSP-W only allows a worker to work on one operation at a time, and only on operation-machine combinations they are eligible for. The encoding for the FJSSP-W extends the encoding for the FJSSP ($\mathbf{y} = (\mathbf{s}|\mathbf{a})$) with an additional vector $\mathbf{w} \in \mathbb{N}^N$ to determine the worker assignment for an operation, turning the encoding for a schedule into $\mathbf{y} = (\mathbf{s}|\mathbf{a}|\mathbf{w}) \in \mathcal{Y} \subset \mathbb{N}^N + \mathbb{N} + \mathbb{N}$, increasing the amount of optimization variables from $2N$ to $3N$. The sequence vector $\mathbf{s}$ and assignment vector $\mathbf{a}$ remain the same in comparison to the FJSSP. Like $\mathbf{a}$, $\mathbf{w}$ has a fixed order, corresponding to the operations in the same order as $\mathbf{a}$, independent of the current sorting of operations in $\mathbf{s}$. It contains the workers assigned to the operation-machine

combinations. For every operation $O_{i,j}$, $\mathbf{w}$ contains a worker index l for worker $M_l \in \mathcal{W}_{i,j,k}$. This ensures, like for the machine assignments, that only eligible workers for the operation-machine combination are assigned to process the operation.

3 Algorithm

This section introduces an overview over the algorithm design of the used GA. The GA has a version for the FJSSP as well as the FJSSP-W. While the base routine stays the same, there are some significant differences based on the increased problem complexity of the FJSSP-W.

3.1 FJSSP

For the FJSSP, the algorithm design of [7] is used. The described GA utilizes an adaptive mutation rate and a restarting scheme with preceding local search to prevent premature convergence of the optimization, as well as a dissimilarity scheme for population initialization in the beginning and after each restart. The used encoding guarantees feasible solutions. The genetic operators are designed to ensure that every candidate solution stays feasible as well. The GA utilizes a limited elitist selection for each new generation. The limit for the individuals from the previous generation is calculated based on problem characteristics. The same is true for the parent selection parameters (tournament selection). This is done to account for the randomness necessary to effectively solve problems with a low duration variety dv . This automatic parametrization is what allows the GA to outperform the other solvers on especially very large and complex problem instances. Additionally, the GA adapts the found schedules to fill in potential empty space in the timeline if there are later operations that are not prevented by any of the problem constraints to start earlier, assuming their processing times fit inside the gap. This remapping of solutions reduced the amount of function evaluations necessary to find solutions of the same quality. The runtime, however, was not affected by the re-adjustment of the solutions

since the deterministic closing of the gaps itself had a negative impact on the runtime.

3.2 FJSSP-W

The GA for the FJSSP-W is described in [9]. The GA is based on the FJSSP version in [7], but introduces some modifications to account for the higher problem complexity. Preliminary comparisons have shown that the employed local search in the FJSSP version does not contribute to the search for the FJSSP-W positively, and is therefore omitted. While the remapping of candidate solutions still presents the same benefits in terms of function evaluation reduction, the increased problem complexity impacted the runtime of the subroutine significantly, and given the time constraints used for the experiments, the step was therefore omitted. The dissimilarity scheme for the population initialization is also removed, since the search space for the FJSSP-W is so large, that a random initialization is sufficient. Since all of these steps had either a negative impact on the GAs performance or a negligible impact at best, the steps were removed for algorithmic simplicity.

4 Experiments

This section describes the problem selection and experimental setup for the performance comparisons between the FJSSP and the FJSSP-W.

4.1 Problem Selection

The most used benchmark instances for the FJSSP consists of 402 instances from 7 different sources, in 10 different collections [2, 8]. However, since repeated experiments for all instances are computationally expensive, a representative set of 30 instances was selected based on their problem characteristics (cf. Fig. 1). The selected instances are translated into FJSSP-W instances using a repeatable algorithm [9].

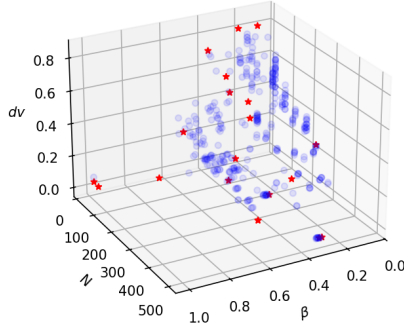


Figure 1: Selection of instances to cover a wide range of scenarios. The red stars indicate the selected problem instances.

4.1.1 Problem Characteristics

The behavior of an optimization algorithm may be different depending on some characteristics inherent to the problem instance that the algorithm is trying to solve. Problem characteristics can be used to derive parameter settings in order to increase the efficiency of the optimization algorithm based on pre-existing knowledge about the problem.

Amount of Operations The amount of operations N of a problem has a high impact on the optimization process, since the size of the search space directly depends on N . For the FJSSP, there are $2N$ decision variables, while for the FJSSP-W, the number of decision variables are $3N$.

Flexibility The flexibility β of a problem instance is determined by the ratio of average available assignment options ω_{avg} per operation and the overall available combinations ω_{unique} , i.e.

$$\beta = \frac{\omega_{avg}}{\omega_{unique}} \quad (2)$$

If all options are available to every operation in an instance, $\beta = 1$, which would be considered a fully flexible problem instance. Otherwise, for $\beta \in [0; 1]$, the problem instance is considered partially flexible. If $\beta = 0$, no assignment options would be available for any operation and the problem instance would be invalid. For the FJSSP ω_{avg} is defined as

$$\omega_{unique} = m \quad \text{and} \quad \omega_{avg} = \frac{1}{N} \sum_{i=1}^n \sum_{j=1}^{n_i} m_{i,j} \quad (3)$$

For the FJSSP-W, ω_{avg} is determined by

$$\omega_{avg} = \frac{1}{N} \sum_{i=1}^n \sum_{j=1}^{n_i} \sum_{k=1}^{m_{i,j}} w_{i,j,k} \quad (4)$$

while ω_{unique} reflects the size of the set of unique machine-worker combinations occurring in the problem instance across all operations.

Duration Variety The duration variety dv is defined as the quotient of the number of distinct values of the processing times, $T_{i,j,k}$ for the FJSSP and $T_{i,j,k,l}$ for the FJSSP-W of the instance d_{unique} and the total amount of assignment options ω . For the FJSSP,

$$\omega = \sum_{i=1}^n \sum_{j=1}^{n_i} m_{i,j} \quad (5)$$

and for the FJSSP-W,

$$\omega = \sum_{i=1}^n \sum_{j=1}^{n_i} \sum_{k=1}^{m_{i,j}} w_{i,j,k} \quad (6)$$

Subsequently, dv is defined as

$$dv = \frac{d_{unique}}{\omega_{avg} \cdot N}. \quad (7)$$

Low dv values indicate very similar processing times across all possible combinations. This corresponds to many distinct candidate solutions that have the same level of fitness, leading to difficulties for optimization algorithms.

Other Other characteristics have been investigated but did not turn out as significant as the previously mentioned ones.

4.2 Setup

The hardware of the test computer is specified as a Intel Core i7-6700 (3.40GHz) CPU with 16 GB RAM, running Windows 10 as operating system. The experiments of the GA are repeated 30 times for each problem setup, with a runtime limit of 300 seconds and compared to the results of a CP solver, after 300 (5 minutes) and 1200 seconds (20 minutes). For the analysis of the result diversity, convergence behavior and uncertainty, a smaller selection of instances was tested for 500000, 1000000, 2000000, 5000000 and 10000000 function evaluations while keeping track of the self-adaptive parameters of the experiments and the solution collection. To determine robustness indicators in the FJSSP-W instances, each solution in the solution collections of all 30 runs of each instance are subjected to simulations with stochastic processing times.

4.3 Simulation

The employed simulation is used to gauge the robustness of the schedules based on stochastic processing times. To this end, each received schedule is run through the simulation 100 times. For each simulation, the processing times are recalculated, following a beta-distribution depending on the assigned worker. After every respective simulation, the actual makespan, among other measures, is determined. Using the results of all 100 simulations, the distribution of results is analyzed and compared to the originally expected makespan.

5 Results

The results of the experiments is analyzed considering different aspects. One is how well the GA handles the increased complexity of the FJSSP-W compared

to the FJSSP in terms of the stability of the optimization behavior. Other areas of interest include the influence of the problem characteristics of the different benchmark instances on the stability of the GAs performance. Lastly, the diversity within the found solutions is compared and analyzed towards their robustness against stochastic processing times.

5.1 FJSSP vs. FJSSP-W

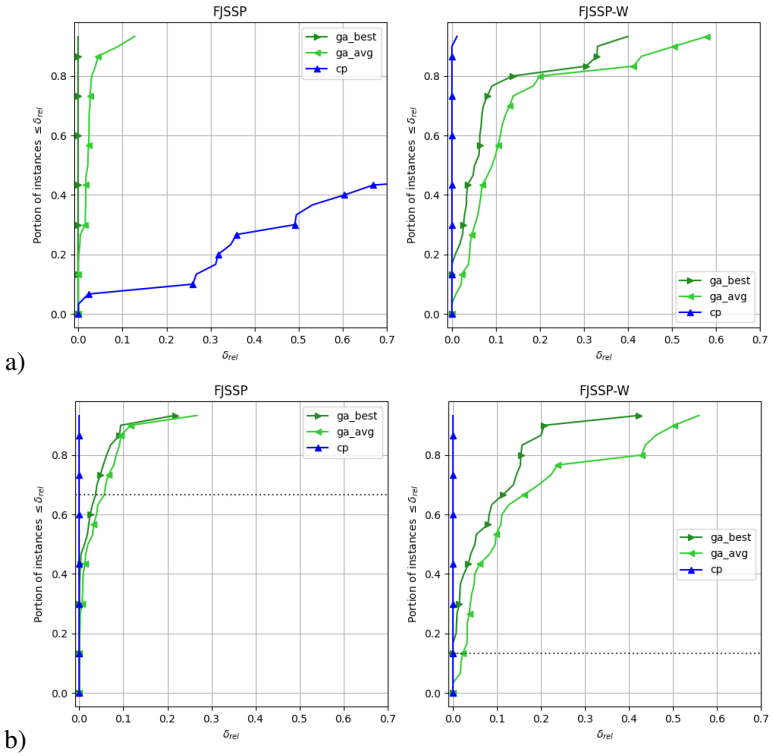


Figure 2: Comparison with the results of the CP solver included in CPLEX. a) shows the comparison after 5 minutes, while b) shows the comparison after 20 minutes. The x-axis shows the relative deviation δ_{rel} (cf. Eq. (8)) to the best known result. The y-axis shows the the amount of instances from the range 0.0 to 1.0, where 1.0 corresponds to 100% of the used instances.

The change from the FJSSP to the FJSSP-W introduces several new decision variables into the combinatorial optimization problem for the GA to solve, raising the amount of variables from $2N$ to $3N$. Additionally, the FJSSP-W introduces new dependencies between the machine assignments and the available workers. As expected, the increased complexity decreases the performance of all solvers across the board [9], including the GA. For comparison, Fig. 2 compare the GA with a constraint programming optimizer, which are inherently well suited for scheduling problems, after 5 minutes and after 20 minutes of optimization time respectively. The measure δ_{rel} measures the relative gap of the fitness result C_{fb} of the optimization with the best known result C_{best} for the same problem instance.

$$\delta_{rel} = \frac{C_{fb} - C_{best}}{C_{best}}. \quad (8)$$

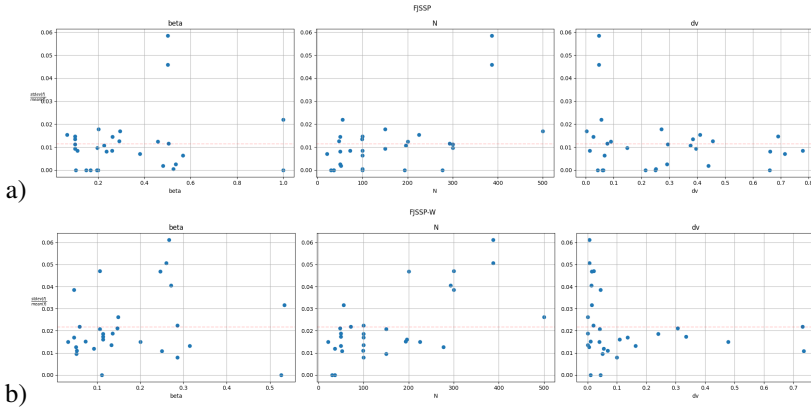


Figure 3: The relative deviation (cf. Eq. (8)) of the makespan of the different problem instances in relation to the flexibility β , the amount of operations N and the duration variety dv . a) Shows the FJSSP, while b) shows the FJSSP-W after the same amount of function evaluations. The red line indicates the average stability of the results.

Fig. 3 show the variation in results for the FJSSP and FJSSP-W respectively, in relation to the different problem characteristics. In both cases, the plots show that the standard deviation does not exceed approximately 6% of the makespan of a solution. The average standard deviation for the FJSSP is a little over 1% for the FJSSP and a little over 2% for the FJSSP-W.

5.2 Convergence Behavior

The convergence behavior is observed to determine whether or not there are any pitfalls present for the optimization problems, e.g. a dependence on the initial population or local minima that the optimization method can not easily escape, in order to reach a good or the overall best result.

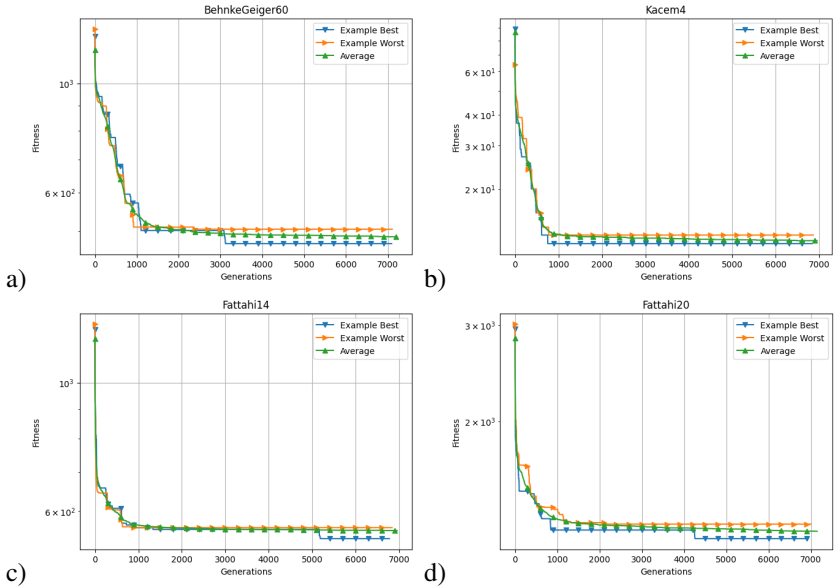


Figure 4: The progress of the optimization process over generations. The logarithmic y-axis shows the fitness, while the x-axis shows the generations. The average line refers to the average progress of all 30 experiment repetitions. The "Example Best" refers to one example run that achieved the best result, while the "Example Worst" shows a run that ended in the worst found result. The function evaluation budget for each shown experiment was 10,000,000.

Fig. 4 shows the average progress of the GA across 30 repetitions on the same problem instance for 4 different example instances. Additionally, there is a single run that reaches the best result (Example Best) and one single run that ends up at the worst result of all 30 runs (Example Worst) to illustrate the overall trajectory of the optimization. The amount of generations can be

different between the experiment repetitions due to the restarting and population growth schemes present in the employed GA.

5.3 Problem Characteristics Influence on Stability

Given that the problem characteristics of each problem instance differ a lot in the used benchmark set, the influence of these characteristics on the performance stability of the GA can be analyzed. One interesting aspect of course is the objective function, but considering that the GA is collecting multiple solutions, the amount of equal quality solutions that can be found from different problem instances is also of interest.

Fig. 5 shows the results related to the problem characteristics of the respective instances. The data contains results from all experiments with the different function evaluation budgets. Fig. 5a) shows the variation in the makespan. The y-axis shows the standard deviation in relation to the average makespan achieved by the 30 individual runs of each experiment. Fig. 5b) shows the variation in the amount of found equal solutions. The y-axis shows the average μ amount of equal quality solutions found at the end of the optimization.

5.4 Result Diversity

Depending on the problem instance, the GA can find a huge variety of solutions that share the same fitness, while for other instances, only few or one solutions can be found. The diversity in the solution collection can show insights into the different aspects of the problem solving behavior and where the difficulties emerge.

The first half of the plots in Fig. 6 (left of the green center line) show the diversity in the sequence vector, while the second half shows the diversity in the assignments to each operation. The red line of the sequencing problem is a straight line considering that every variable of the sequence could have the same amount (all) possible values present in the sequencing problem (the job IDs), while the assignment problem has a different amount of possible values available for each operation.

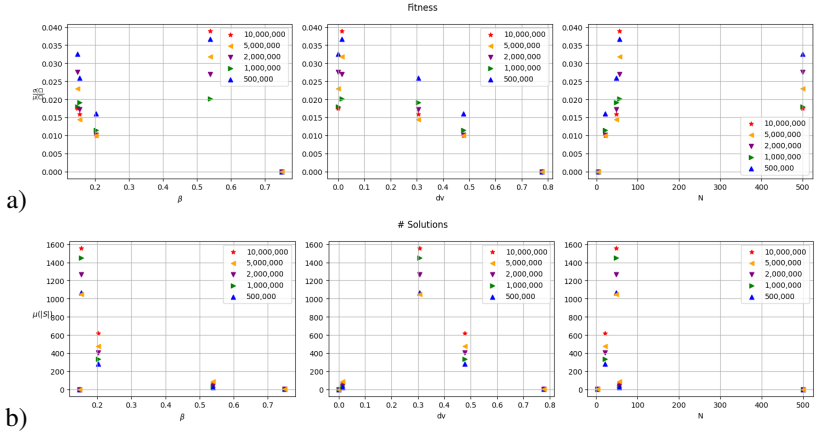


Figure 5: The dependence of the a) fitness and b) number of equal solutions found for the problem instances. The fitness in a) is given by their standard deviation across multiple runs in relation to the average makespan, while the number of solutions is displayed as the average of all runs. Each point represents a problem instance, for which 30 repetitions are performed. The different markers represent the size of the function evaluation budget of the respective experiment.

5.5 Uncertainty Indicators

Since the GA can gather multiple solutions sharing the same makespan C , some of the solutions may be more robust than others. To determine whether potential other objective function could help indicate the robustness of a solution, a selection of evaluation criteria is selected to evaluate all n_{sol} solutions. The robustness of each solution is determined through n_{sim} simulation runs in which the processing times are stochastic, resulting in the actual makespans S . The robust makespan C_r of each solution is determined as the average makespan of all simulation runs.

$$C_r = \frac{1}{n_{sim}} \sum_{i=1}^{n_{sol}} S_i \quad (9)$$

To keep the results comparable between different problem instances, the robustness measure R is defined as

$$R = \frac{C_r}{C} \quad (10)$$

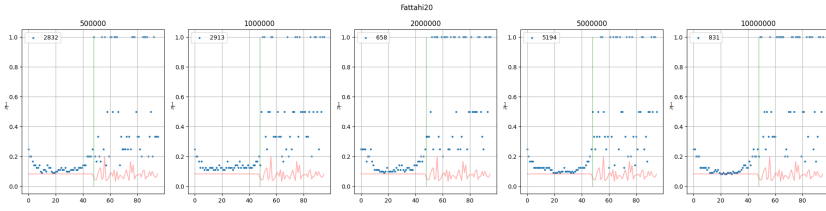


Figure 6: The distribution of the sequence and assignment vectors in the collection of equal solutions of the best result for the example instance Fattahi20 (FJSSP). The green line divides the sequence vector from the assignment vector. The red line indicates at which point each possible value for a variable (x-axis) would be present in the solution population with a uniformly distributed probability. A value of 1.0 for a specific variable means that all found solutions share the same value for the specific variable.

The robustness R in relation to the original makespan C is displayed in Fig. 7 for the specific instances: BehnkeGeiger60, Kacem4, Fattahi14, and Fattahi20. The different markers represent the different function evaluation budgets for the conducted experiments. Data from all 30 experiment repetitions are included.

Fig. 8 show different alternate evaluation criteria for the produced schedules, with the average of each respective criteria across the simulations for each respective equal solution on the x-axis and the robustness indicator R (cf. Eq. (10)) on the y-axis. The color indicates the standard deviation of the simulation results to display the stability of the solution. The shown data is of the instance Fattahi20. The shown solutions are filtered to only contain the solutions with $R \leq 1.5$.

The considered criteria are the idle times of both machines and workers, the balancing of the working times of the machines and workers based on the average working time of the respective resources, the average duration of all jobs, as well as the job duration in relation to the minimum time a job would take to process if all operations would be executed back to back without any delays.

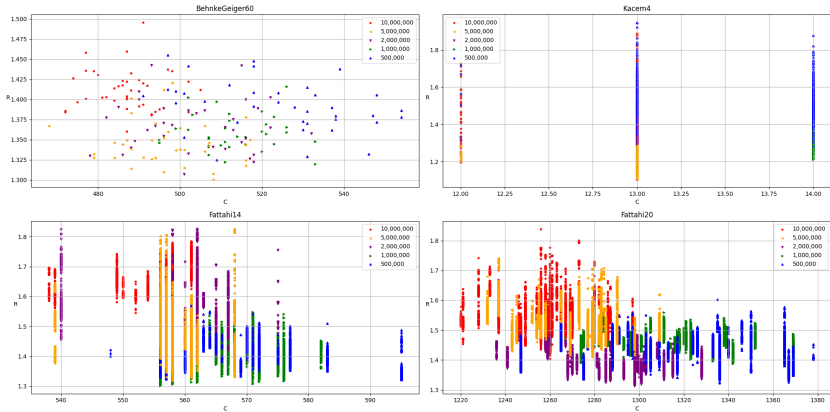


Figure 7: The original fitness in comparison to the robust fitness after 100 simulation runs for each resulting solution. A smaller R is better.

6 Discussion

The produced data shows interesting observations for the different considered aspects of the problem. While some results point to potential improvements that can be made to the GA, other results do not show quite as obvious patterns in the optimization process. The section is divided into the different aspects that were observed through the experiments. These aspects are the impact of the increased problem complexity from FJSSP to the FJSSP-W, the convergence behavior of the algorithm during the optimization process, the influence of the problem characteristics on the performance of the GA, the diversity of the found results that share the same solution quality, and the robustness of the found solutions towards uncertain processing times. Additionally, alternative evaluation metrics are used to determine if they can be used as indicator for the robustness of a solution.

Problem complexity For the comparison between the complexity increase from the FJSSP to the FJSSP-W, Fig. 2 shows, that the GA does not lose efficiency in comparison, especially considering the performance drop off of comparable methods, as shown by the low rate of optimal results found and

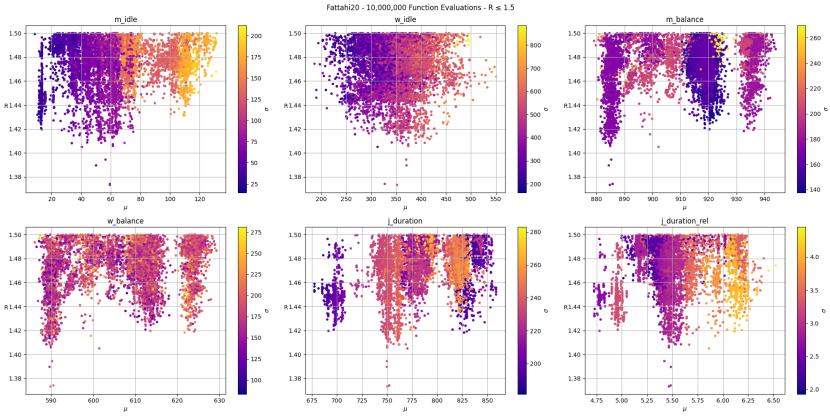


Figure 8: Alternative objective functions for solutions found over the 30 experiment repetitions for an example instance with 10,000,000 function evaluations. The y-axis shows the robustness indicator R . The solutions with $R > 1.5$ are omitted. The x-axis shows the average of the objective value across 100 simulation runs. The color indicates the standard deviation over the 100 simulation runs.

observations in previous work [9]. Fig. 3 shows in both cases, that the duration variety dv seems to be a good indicator of how stable the optimization process will be, with a low dv leading to more varied results. In the same way, a large amount of operations N , i.e. a large amount of decision variables, has a negative effect on the optimization stability, for the FJSSP-W even more-so than for the FJSSP. The flexibility β is harder to interpret. A low β , simplifying the problem towards a standard JSSP(-W) increases the stability in most cases, and high values don't conclusively show a negative or positive effect for the behavior. Partial flexibility (i.e. a mid-range β), while also showing mixed results, also shows the highest amount of variation in the result quality. It should be noted, however, that the 2 most extreme points also share a combination of a big N and very low dv .

Convergence behavior The progress history shows that the best results are not necessarily found based on the initial population, given that at times the worst result example has better intermediate solutions than the best example as well as the average progress. Considering the measures taken to help the

GA to escape local minima, the re-occurring improvements after stretches of no improvements point at the measures working as intended. However, overall the GA tends to find a good solution quickly, but struggles to find the best solution for a long while after. Looking at the progress of the best and worst examples of either of the instances shows that there tend to be longer stretches with no progress. Once an improvement has been found, multiple improvements follow in rapid succession. This points to the fact that there are many possible alternative solutions in every direction of the search and finding improvements becomes increasingly difficult. It could be interesting in future work to track whether the GA often re-explores areas once stuck in the neighborhood or if many distinct alternatives are examined. Previous testing and the amount of equal solutions found for some instances point to the latter. However, if the GA re-visits certain neighborhoods more often, it could be beneficial to implement a deterrent for the GA to re-explore already known solutions. This would increase the required memory for the GA significantly, but could lead to better results overall.

Influence of characteristics In previous work [7, 9] the examined problem characteristics were found to have an influence on the progress of the optimization process as well as the best solution that is found. However, the influence on the stability of the optimization process was not analyzed. Concerning the problem characteristics shown in Fig. 5, for a), the only clearly visible pattern pertains the duration variety dv . As the dv gets bigger, the stability of the found solutions increases. Note that the last point represents a problem instance for which only a single result was found, while it can be seen in b) that this is not the case for the earlier data points. Considering the optimization runs with a higher function evaluation budget, the effect is even more pronounced.

Diversity Fig. 6 shows that the sequencing problem has many possible solutions, while for the assignment problem, many variables have a more narrow field for the best option. While the amount of options available is captured by the flexibility β , the amount of different available values is captured in the duration variety dv . This means that for some instances, there might be a higher variance in the selected assignments present in the solution pool, which do

not make an actual difference for the fitness value directly, but can make a difference depending on the sorting. A higher diversity for each variable could still be favorable, since this would correspond to a solution that does not assign a resource to many operations while ignoring an equally well suited resource, which can lead to waiting times on the overused resource.

Uncertainty Based on Fig. 7, it seems apparent that the robustness of the most optimized solutions does not tend to be the highest, which suggests that a compromise between result quality and result robustness needs to be made depending on the requirements.

The different evaluation criteria shown in Fig. 8 do not show clear patterns for all of the indicators. For both the machine balance and worker balance indicators a lower average working time, i.e. a better distribution of the operations on all the available resources have a better potential to lead to a more robust solution. Note that the plots do not indicate the solution quality, but the relation between the original quality and the robustness of the solution. For the job duration, both used indicators show a better result in the mid-range. At the mid-range for most instances, there are buffer times in place in case of delay, but they are not unnecessarily spread apart even further. The idle times for both machines and workers do not show a clear pattern but seem to be slightly more robust in the mid-range as well, which also indicates inherent buffer times for the resources in between operations, without them being spread apart unnecessarily.

7 Conclusion

The examined results were used to analyze the performance in terms of the problem complexity, the convergence behavior, the influence of the problem characteristics, the result diversity, and the robustness towards uncertainties, as well as potential indicators for preferable solutions in terms of robustness.

In conclusion, the increased problem complexity does not have a big impact on the stability and performance of the GA. The convergence behavior shows the effectiveness of the employed measures to avoid premature convergence, but also

warrants closer analysis for potential wasted computation on already explored areas of the search space. The observations on the influence of the problem characteristics show that the duration variety dv is the most reliable indicator to estimate how difficult the given problem instance is to solve. For the diversity in the found solutions, while both the sequencing and the assignment problems contribute to the large pool of equal solutions, the sequencing problem shows a higher diversity. This leads to the assumption that there are many operations that can be swapped in their sorting in the encoding without a negative effect on the solution quality. Some of these similar encodings may even produce the exact same schedule. However, the combination of different sequence encodings and assignments lead to many distinct solutions with the same quality. Using these differing solutions to analyze potential robustness indicators, many of the used evaluation methods did not show clear patterns. The most helpful indicators are the balancing aspects of the employed resources, showing that a more equal distribution reduces the risks for each operation. In general, the solutions with the highest grade of optimization did not necessarily display the highest robustness towards uncertainties, meaning that the robust optimization of the FJSSP(-W) requires a balanced approach to solve the problem.

8 Outlook

In future work, the gathered observations can be applied to improve the used GA. Additionally, using the gained insights about possible uncertainty indicators, a multi-objective version of the GA can be used to find a well balanced solution to increase the robustness of the optimization. The indicators also suggest that purposefully inserting buffer times into the schedules to make them more reliable would improve the robustness of the results. This could either be achieved during the optimization or after the fact on the obtained result. Both possibilities are also explored in future work. In terms of the convergence behavior, some closer investigation may be necessary to determine whether or not the GA re-explores the same areas of the search space multiple times during the long stretches of no progress.

Acknowledgments

The financial support by the Austrian Federal Ministry of Labour and Economy, the National Foundation for Research, Technology and Development, the Christian Doppler Research Association is gratefully acknowledged. This article is based upon work from COST Action Randomised Optimisation Algorithms Research Network (ROAR-NET), CA22137, supported by COST (European Cooperation in Science and Technology).

References

- [1] N. Bagheri Rad and J. Behnamian. Recent trends in distributed production network scheduling problem. *Artificial Intelligence Review*, 55(4):2945–2995, Apr. 2022.
- [2] D. Behnke and M. J. Geiger. Test Instances for the Flexible Job Shop Scheduling Problem with Work Centers. Working Paper, 2012. Accepted: 2017-10-24T14:10:24Z.
- [3] S. Dauzère-Pérès, J. Ding, L. Shen, and K. Tamssaouet. The flexible job shop scheduling problem: A review. *European Journal of Operational Research*, 314(2):409–432, Apr. 2024.
- [4] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA, 1990.
- [5] A. Ghasemi, F. Farajzadeh, C. Heavey, J. Fowler, and C. T. Papadopoulos. Simulation optimization applied to production scheduling in the era of industry 4.0: A review and future roadmap. *Journal of Industrial Information Integration*, 39:100599, 2024.
- [6] A. Gnanavelbabu, R. H. Caldeira, and T. Vaidyanathan. A simulation-based modified backtracking search algorithm for multi-objective stochastic flexible job shop scheduling problem with worker flexibility. *Applied Soft Computing*, 113, 2021.

- [7] D. Hutter, M. Hellwig, and T. Steinberger. An Interior-point Genetic Algorithm with Restarts for Flexible Job Shop Scheduling Problems. In *2024 IEEE Congress on Evolutionary Computation (CEC)*, 2024.
- [8] D. Hutter, M. Hellwig, and T. Steinberger. Benchmarking and Benchmark Creation for FJSSP-W Experiments, 2025. <https://github.com/jrc-rodec/FJSSP-W-Benchmarking>.
- [9] D. Hutter, T. Steinberger, and M. Hellwig. A parameter adaptive genetic algorithm with restarts for flexible job shop scheduling problems with worker flexibility. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO*, page 523–526, New York, NY, USA, 2025. Association for Computing Machinery.
- [10] X. Li, X. Guo, H. Tang, R. Wu, L. Wang, S. Pang, Z. Liu, W. Xu, and X. Li. Survey of integrated flexible job shop scheduling problems. *Computers & Industrial Engineering*, 174:108786, Dec. 2022.
- [11] S. Usman and C. Lu. Job-shop scheduling with limited flexible workers considering ergonomic factors using an improved multi-objective discrete Jaya algorithm. *Computers & Operations Research*, 162:106456, Feb. 2024.
- [12] J. Xie, L. Gao, K. Peng, X. Li, and H. Li. Review on flexible job shop scheduling. *IET collaborative intelligent manufacturing*, 1(3):67–77, 2019.

Efficient Gait Analysis using Knowledge Distillation from Sparse Sensors

Vincent Katter¹, Martin Kohlhasse¹

¹Hochschule Bielefeld

Interaktion 1, 33619 Bielefeld

E-Mail: vincent.katter, martin.kohlhasse@hsbi.de

1 Introduction

Diabetic foot syndrome (DFS) combines neuropathy, vascular impairment, and altered biomechanics, leading to hazardous plantar loads and recurrent ulcers [1, 2]. Orthoses with sensorized insoles can provide early warnings and personalized feedback; clinical guidelines recommend verifying substantial pressure reduction to mitigate risk [3, 4], and feedback-oriented insoles have shown reduced recurrence [5].

Practical systems must be lightweight and energy-efficient, favoring sparse sensor layouts that extend wear time and reduce wiring but also lower spatial detail and SNR. Soft padding and daily variability further obscure peaks and transients [6, 7]. Deep neural networks (DNNs) can compensate for sensor variability and noise, extracting spatiotemporal structure from plantar-pressure sequences [8], yet state-of-the-art models trained on dense grids typically exceed the compute and power budgets of embedded orthoses.

Knowledge distillation (KD) addresses this gap by training a compact student on sparse inputs with guidance from a high-capacity teacher trained on dense grids [9, 10]. The key challenge is input asymmetry: teacher and student observe different sensor densities. Reconstructing dense targets from sparse signals is an ill-posed inverse problem that can conflict with classification. This work investigates bridging objectives that align teacher and student representations without inventing unobserved information.

2 Previous Work

The setup comprises a teacher operating on a dense insole and a student using a biomechanically selected sparse subset. Both share a spatiotemporal backbone with a 2D Convolutional Neural Network for per-frame processing, a 1D temporal convolution, and a classifier. Bridging between teacher and student is achieved by auxiliary distillation modules (Fig. 1).

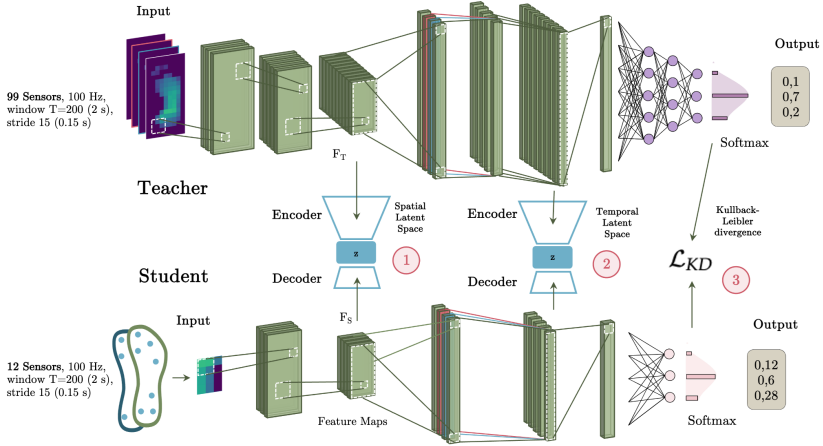


Figure 1: Teacher–student setup under input asymmetry. Teacher processes dense inputs and student a sparse subset. Bridging modules transfer knowledge from teacher to student: (1) spatial autoencoder, (2) temporal autoencoder, and (3) soft-logit distillation. These exemplify alternative mechanisms that can be substituted or extended in later methods.

Earlier experiments combined reconstructive auxiliary losses with soft-logit distillation (Fig. 1). Spatial and temporal autoencoders aimed to enrich student features by predicting dense structure from sparse inputs, while soft targets transferred class relations via Kullback–Leibler (KL) divergence with temperature τ . However, reconstruction from sparse inputs is ill-posed, encouraging hallucination and conflicting with classification, so the autoencoders acted only as weak regularisers. By contrast, soft-logit distillation provided modest but consistent gains without inventing unobserved content. These limitations motivate alignment-based objectives that transfer useful structure without reconstructing dense targets.

3 Distillation Strategies for Sparse Wearable Sensing

Three KD strategies were evaluated to address input asymmetry between dense teacher and sparse student sensing. These strategies can be understood as alternative bridging modules, replacing the autoencoders from Fig. 1.

Projection-based alignment (FitNets). A 1×1 projection head maps the student’s feature maps $\tilde{F}_s$ into the teacher’s channel space [10], producing $\hat{F}_s = \text{Head}(\tilde{F}_s)$, so that both networks can be compared at the same intermediate representation level. These are compared to the teacher’s mid-level features $\bar{F}_t$ with the alignment loss:

$$\mathcal{L}_{\text{proj}} = \|\hat{F}_s - \bar{F}_t\|_2^2$$

Projection aligns intermediate features without inventing missing regions. Stability depends strongly on the projection head’s capacity and on gradual ramping of the auxiliary weight. In practice, this makes projection a flexible yet capacity-sensitive mechanism for bridging teacher and student representations.

PCA-based subspace alignment. PCA-based alignment reduces the complexity of feature comparison by focusing on the principal directions of variation. Instead of matching full feature maps, teacher and student are compared in a compressed subspace that captures the most informative structures. Teacher features F_t and student features F_s are flattened, centered by mean μ , and projected onto the top k eigenvectors W_k with singular values Σ_k [11, 12], yielding

$$Z = (\text{vec}(F) - \mu) W_k \Sigma_k^{-1}.$$

The alignment loss

$$\mathcal{L}_{\text{pca}} = \|Z_s - Z_t\|_2^2,$$

compares student and teacher in this subspace, where whitening with Σ_k^{-1} balances principal directions. This approach trades fine-grained spatial detail for robustness against input asymmetry, acting as a regularizer whose stability depends on the choice of k .

Hybrid approach. Projection and PCA losses can be combined with independent scheduling to capture both local alignment and global structure.

Logit KD (soft targets). In addition to feature alignment, soft targets can be used to transfer class relations from teacher to student [9]. This route is also part of the baseline setup (Fig. 1) and serves as a lightweight complement to the alignment modules.

Training objective. The overall training loss is

$$\mathcal{L}(t) = \mathcal{L}_{\text{CE}} + \alpha(t) \mathcal{L}_{\text{logits}} + \lambda_{\text{proj}}(t) \mathcal{L}_{\text{proj}} + \lambda_{\text{pca}}(t) \mathcal{L}_{\text{pca}},$$

where $\alpha(t)$ and $\lambda_{*}(t)$ are scheduled during the first ten epochs. Different experimental conditions activate different terms (e.g., logits baseline with $\alpha^{\max} > 0$; projection with $\lambda_{\text{proj}}^{\max} > 0$; PCA with $\lambda_{\text{pca}}^{\max} > 0$). Projection width and PCA dimensionality were varied to probe sensitivity under input asymmetry.

4 Results

Soft targets (logit-based KD, $\tau = 2$, $\alpha = 0.3$) provided a small but consistent gain over the cross-entropy baseline (Table 1). Projection-based alignment showed strong dependence on capacity: the narrow head ($m = 64$) was sensitive to the ramp schedule, while the wider head ($m = 256$) achieved the best accuracy at 92.20%. PCA-based subspace alignment acted as a regularizer but did not exceed the baseline for $k = 64$ with whitening. The hybrid configuration reduced validation loss yet failed to improve accuracy on held-out subjects. Overall, the best setting outperformed the baseline by +1.6 percentage points.

5 Discussion and Conclusion

Input asymmetry between dense teacher and sparse student sensing imposes strict limits on distillation gains. A strong cross-entropy baseline, the low

Table 1: Results on sparse subset (same split; 10-epoch ramp). Bold: strongest run.^a

Run	Params	Val Loss	Acc (%)
Baseline (CE)	—	0.1717	90.57
Logit KD (soft targets)	$\tau = 2, \alpha = 0.3$	0.1689	91.12
FitNets (narrow)	$m = 64, \lambda = 0.5$	0.1887	89.12
FitNets (narrow)	$m = 64, \lambda = 1$	0.1769	89.80
FitNets (narrow)	$m = 64, \lambda = 2$	0.1696	90.88
FitNets (wide)	$m = 256, \lambda = 1$	0.2043	92.20
PCA-KD	$k = 64$, whiten	0.2178	89.25
Hybrid (proj+PCA)	$m = 64, k = 64^b$	0.2102	90.20

^a Single runs; best checkpoint by validation loss; no test-time augmentation.

^b $\lambda_{\text{pca}} = 1.0, \lambda_{\text{proj}} = 0.5$, whiten.

capacity of compact students, and practical constraints on compute and energy narrow the room for improvement. Nevertheless, alignment-based transfer proved effective: projection with a sufficiently wide head and gentle scheduling achieved the best gains, while PCA-guided subspace alignment acted as a stabilizing regularizer. Hybrid objectives improved fit but did not consistently surpass single strategies.

These results suggest that reconstruction is unsuitable under asymmetric sensing, whereas alignment-based objectives offer a more credible path forward. Future work should systematically explore projection capacity and adaptive loss schedules to mitigate hyperparameter sensitivity, investigate dynamic PCA dimensionality and whitening strategies to improve subspace transfer, and refine hybrid objectives to exploit complementary strengths of projection and subspace alignment. In addition, cross-subject evaluation and cohort-to-subject distillation remain critical for deployment, where compact students must generalize under strict compute and energy budgets in embedded orthoses.

References

- [1] Morbach, S.; Lobmann, R.; Kress, S.; Risse, A.: Das diabetische Fußsyndrom – eine komplexe Herausforderung. *Der Diabetologe* 16(Suppl 2):S362–S372, 2021. doi:10.1007/s11428-021-00759-7.
- [2] Zink, K.; Stumpf, J.; Türk, H.; Haak, T.: Rezidivprophylaxe beim diabetischen Fußsyndrom: Richtige Nachsorge ist gute Vorsorge. *Der Diabetologe* 16:35–44, 2020.
- [3] International Working Group on the Diabetic Foot (IWGDF): Guidelines on the prevention and management of diabetic foot disease. 2019. Available at: iwgdfguidelines.org.
- [4] International Working Group on the Diabetic Foot (IWGDF): Practical guidelines on the prevention and management of diabetes-related foot disease. 2023. Available at: iwgdfguidelines.org.
- [5] Abbott, C. A.; van Netten, J. J.; Witteveen, H.; et al.: Effectiveness of a smart insole system for reducing ulcer recurrence in people with diabetes: a multicentre randomized controlled trial. *The Lancet Digital Health* 1(5):e308–e318, 2019. doi:10.1016/S2589-7500(19)30128-1.
- [6] Castro-Martins, C.; et al.: In-shoe technologies for plantar pressure analysis: A review. *Heliyon* 10(2):e26967, 2024. doi:10.1016/j.heliyon.2024.e26967.
- [7] Ren, P.; et al.: Advancements in diabetic foot insoles with integrated sensors: design challenges and opportunities. *Frontiers in Bioengineering and Biotechnology* 12:1391836, 2024. doi:10.3389/fbioe.2024.1391836.
- [8] Mun, J. H.; Kim, S. H.; Park, S.; et al.: Predicting foot plantar pressure using an LSTM model with a reduced number of sensors. *Journal of NeuroEngineering and Rehabilitation* 19:67, 2022. doi:10.1186/s12984-022-00987-8.
- [9] Hinton, G.; Vinyals, O.; Dean, J.: Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*, 2015.

- [10] Romero, A.; Ballas, N.; Ebrahimi Kahou, S.; Chassang, A.; Gatta, C.; Bengio, Y.: FitNets: Hints for Thin Deep Nets. In: *International Conference on Learning Representations (ICLR)*, 2015. (arXiv:1412.6550)
- [11] Chiu, T.-Y.; Gurari, D.: PCA-Based Knowledge Distillation Towards Lightweight and Content-Style Balanced Photorealistic Style Transfer Models. In: *Proc. CVPR*, pp. 7844–7853, 2022.
- [12] Gavish, M.; Donoho, D. L.: The Optimal Hard Threshold for Singular Values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory* 60(8):5040–5053, 2014.

Discrepancy Modeling for Dynamical Systems

Mortimer Dockhorn, Martin Kohlhase

Hochschule Bielefeld

Interaktion 1, 33619 Bielefeld

E-Mail: {mortimer.dockhorn, martin.kohlhase}@hsbi.de

1 Motivation

The technological progress of recent decades has continuously provided new opportunities for the engineering sciences. Due to national and international competition, this trend has produced steadily increasing demands on research and development, which in turn has led to technical systems, such as automobiles, becoming increasingly complex in order to meet current market requirements. Model-based design has been successfully employed in research and industry to address many of the resulting challenges. However, traditional modeling approaches based on deriving mechanistic models from first principles are not always suited for the modeling of highly complex systems. On the one hand, this is because the derivation and parametrization of high-fidelity mechanistic models require a significant level of expertise and effort, resulting in a costly modeling process. On the other hand, traditional modeling is almost always infeasible without simplifying assumptions. This is either because a complete mathematical description of all system processes is not available or because the complexity of the model itself would become disproportionate if it were to capture all relevant system states. Such simplifications, however, result in modeling inaccuracies leading to deviations between the predicted and actual system behavior. Mechanistic models alone are therefore often unsuited for model-based tasks in demanding fields, such as robotics, biomechanics, and precision manufacturing [1].

Purely data-driven modeling presents an alternative to traditional approaches. It provides the option to model complex systems solely on the basis of measured

input and output data, without requiring additional knowledge of governing physical laws. When the data-driven modeling of a dynamical system is carried out, however, the training data must satisfy specific requirements. Particularly, they must contain information about the system's behavior at all excitation frequencies to ensure that its dynamics can be learned completely [2]. In theory, such excitation can be accomplished for many systems if specialized input signals, such as an APRBS or multisine [2], are applied for a sufficient amount of time. In practice, however, this process is often connected to high costs, as it requires extensive experiments. Furthermore, for certain systems this approach is infeasible, because actuator limitations or the risk of damaging the system itself does not allow for the required excitation. The purely data-driven modeling of dynamical systems is therefore limited in terms of its practical applicability.

Hybrid modeling combines traditional and data-driven approaches, making it possible to partially compensate for the mentioned limitations of each individual method. Discrepancy modeling is a hybrid technique that uses a data-driven model (i.e., the discrepancy model) to learn the error between predictions of a simplified mechanistic model and corresponding measurement data from the real system. The predictions of both models are then combined within the hybrid model resulting in improved accuracy. As mentioned above, this approach leverages the capabilities of traditional and data-driven modeling while reducing some of the requirements that apply when each method is employed on its own. In particular, the modeling effort for creating the mechanistic model can be kept reasonable as it only serves as a baseline predictor. Furthermore, the training of the data-driven model only requires data from specific operating points or states, reducing the number of necessary experiments.

2 Discrepancy Modeling Framework

The general discrepancy modeling framework presented in this work employs the methodologies suggested in [1, 3], only considering autonomous systems. It is assumed that a simplified mechanistic model of the system of interest is available which shows significant prediction inaccuracies at certain operating

conditions of the system. Furthermore, the mechanistic model is assumed to account for all relevant state variables, while the corresponding states of the true system are measurable. Therefore, the deviations between the predictions of the mechanistic model and these measurements are mainly caused by inaccuracies in the model equations, as the influence of measurement uncertainties and disturbances is assumed to be small relative to the influence of the model errors. Given these assumptions, the presented framework proposes an approach to accounting for such model errors with a discrepancy model.

Let the mechanistic model be of the general form

$$\dot{\hat{\mathbf{x}}}_m(t) = \mathbf{f}_m(\hat{\mathbf{x}}_m(t)), \quad \hat{\mathbf{x}}_m(0) = \hat{\mathbf{x}}_{m,0}, \quad (1)$$

where $t \in [0, T]$ is the time variable with $T \in \mathbb{R}_{\geq 0}$, and $\hat{\mathbf{x}}_m(t) \in \mathbb{R}^{n_x}$ is the vector of modeled states. The vector field $\mathbf{f}_m : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}$ is globally Lipschitz on $\mathbb{R}^{n_x}$, guaranteeing the existence of a unique solution on $[0, T]$. Furthermore, let the true dynamics of the system of interest be given by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)), \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (2)$$

where the vector field $\mathbf{f} : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}$ is unknown, and $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the vector of measurable states corresponding to $\hat{\mathbf{x}}_m(t)$. Given this scenario, the aim is to find a discrepancy model $\boldsymbol{\delta} : \Phi \rightarrow \mathbb{R}^{n_x}$, with the feature space $\Phi \subseteq \mathbb{R}^{n_\phi}$ and time-dependent feature vector $\boldsymbol{\phi} : [0, T] \rightarrow \Phi$, such that $\boldsymbol{\delta}(\boldsymbol{\phi}(t))$ compensates for the dynamical model equation error

$$\mathbf{e}(t) := \dot{\mathbf{x}}(t) - \mathbf{f}_m(\mathbf{x}(t)). \quad (3)$$

The combined hybrid model is then given by

$$\dot{\hat{\mathbf{x}}}_h(t) = \mathbf{f}_m(\hat{\mathbf{x}}_h(t)) + \boldsymbol{\delta}(\boldsymbol{\phi}(t)), \quad \hat{\mathbf{x}}_h(0) = \hat{\mathbf{x}}_{h,0}, \quad (4)$$

with a sufficiently accurate discrepancy model yielding $\dot{\hat{\mathbf{x}}}_h(t) \approx \dot{\mathbf{x}}(t)$ and consequently $\hat{\mathbf{x}}_h(t) \approx \mathbf{x}(t)$, if $\hat{\mathbf{x}}_{h,0} = \mathbf{x}_0$. It should be noted that (3) is defined as the difference of the true states' derivatives and the vector field of the mechanistic model (1), evaluated for corresponding values of the true states. Hence, $\mathbf{e}(t) \neq \dot{\mathbf{x}}(t) - \dot{\hat{\mathbf{x}}}_m(t)$. Furthermore, the presented approach may produce inher-

ently different results than learning a discrepancy model of the state space error $\mathbf{x}(t) - \hat{\mathbf{x}}_m(t)$, as discussed in detail in [1].

3 Practical Considerations and Future Work

Considering practical aspects, the generation of training data for the discrepancy model requires measurements of the true system containing sufficient information about the dynamical model equation error. Therefore, the data should be collected when the system operates in conditions where this error is pronounced. In the present case of an autonomous system, this equates to choosing initial conditions that will cause informative state trajectories. The optimal choices of n_ϕ and Φ (i.e., the number of features and their value ranges) may vary, depending on the selected type of data-driven model. Natural choices are the states of the hybrid model and possibly delay embeddings [4]. Any preprocessing methods applied to the training data must also be applicable to data streams, as the trajectories of the hybrid model are calculated iteratively by a numerical integrator, requiring the model output of each iteration for subsequent steps.

A promising future research topic is the extension of the presented framework to non-autonomous systems. In particular, when the system of interest is driven by controllable inputs, the question of optimal and efficient excitation with regard to the discrepancy arises. Approaches, such as the *OKID* algorithm [5], that recover the impulse response of a forced system from input-output data generated with feasible inputs, may provide useful tools within this context. Furthermore, the use of modern continuous-time system identification methods (e.g., [6, 7]) presents an opportunity for constructing interpretable mechanistic discrepancy models. This approach was used in [1], where promising results could be achieved in a comparison with neural networks and Gaussian process regression. A particular advantage of mechanistic models over black-box models is their interpretability, which may allow one to gain a better understanding of the nature of the discrepancy itself.

References

- [1] M. R. Ebers et al. “Discrepancy Modeling Framework: Learning Missing Physics, Modeling Systematic Residuals, and Disambiguating between Deterministic and Random Effects”. *SIAM Journal on Applied Dynamical Systems*, 2024. doi: 10.1137/22M148375X.
- [2] O. Nelles. “Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes”. Springer International Publishing, 2020, doi: 10.1007/978-3-030-47439-3.
- [3] M. Levine and A. M. Stuart. “A framework for machine learning of model error in dynamical systems”. *Communications of the American Mathematical Society*, 2022. doi: 10.1090/cams/10.
- [4] F. Takens. “Detecting Strange Attractors in Turbulence”. *Dynamical Systems and Turbulence*, 1981. doi: 10.1007/BFb0091924.
- [5] J.-N. Juang et al. “Identification of Observer/Kalman Filter Markov Parameters”. *AIAA GNC Conference*, 1991. url: <https://ntrs.nasa.gov/api/citations/19910016123/downloads/19910016123.pdf>.
- [6] S. L. Brunton et al. “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. *Proceedings of the National Academy of Sciences*, 2016. doi: 10.1073/pnas.1517384113.
- [7] J. N. Kutz et al. “Dynamic Mode Decomposition”. SIAM (Society for Industrial and Applied Mathematics), 2016. ISBN: 978-1-61197-449-2.

Embedding World Knowledge into Tabular Models: Towards Best Practices for Embedding Pipeline Design

Oksana Kolomenko^{*}, Ricardo Knauer^{*}, Erik Rodner

KI-Werkstatt, University of Applied Sciences Berlin

HTW Berlin, 10313 Berlin, Germany

E-Mail: ricardo.knauer@htw-berlin.de

^{}These authors contributed equally.*

Abstract

Embeddings are a powerful way to enrich data-driven machine learning models with the world knowledge of large language models (LLMs). Yet, there is limited evidence on how to design effective LLM-based embedding pipelines for tabular prediction. In this work, we systematically benchmark 256 pipeline configurations, covering 8 preprocessing strategies, 16 embedding models, and 2 downstream models. Our results show that it strongly depends on the specific pipeline design whether incorporating the prior knowledge of LLMs improves the predictive performance. In general, concatenating embeddings tends to outperform replacing the original columns with embeddings. Larger embedding models tend to yield better results, while public leaderboard rankings and model popularity are poor performance indicators. Finally, gradient boosting decision trees tend to be strong downstream models. Our findings provide researchers and practitioners with guidance for building more effective embedding pipelines for tabular prediction tasks.

1 Introduction

Large language models (LLMs), pretrained on massive web-crawled corpora, possess a rich world knowledge that can not only be leveraged for tasks in natural language processing, but also in tabular prediction [1, 2]. Tabular rows are commonly serialized into text and then converted into numerical representations (embeddings) suitable for downstream models [3, 4]. This allows for efficiently augmenting data-driven machine learning methods with LLMs’ compressed knowledge about the world.

Despite the growing interest in this paradigm [5–9], there is limited evidence on how to design effective embedding pipelines for tabular prediction, particularly regarding the preprocessing strategies, the selection of embedding models, and the choice of downstream models. This makes it difficult both to generalize insights from academic research and to develop effective embedding pipelines in practice.

Therefore, our paper focuses on systematically benchmarking different preprocessing options, embedding models, and downstream models for tabular prediction. Please refer to Figure 1 for an overview of our work. Our contributions are as follows:

1. Our empirical results demonstrate that the impact of enriching machine learning models with the LLMs’ world knowledge strongly depends on the specific embedding pipeline configuration. **We find that concatenating embeddings tends to be better than replacing the original columns with embeddings and that gradient boosting decision trees tend to be strong downstream models** (Sect. Predictive Performance).
2. We also evaluate the relationship between the predictive performance and embedding model characteristics. **We show that larger models tend to yield better results, whereas public leaderboard rankings and model popularity are poor performance indicators for tabular prediction tasks** (Sect. Linking Model Properties to Performance).

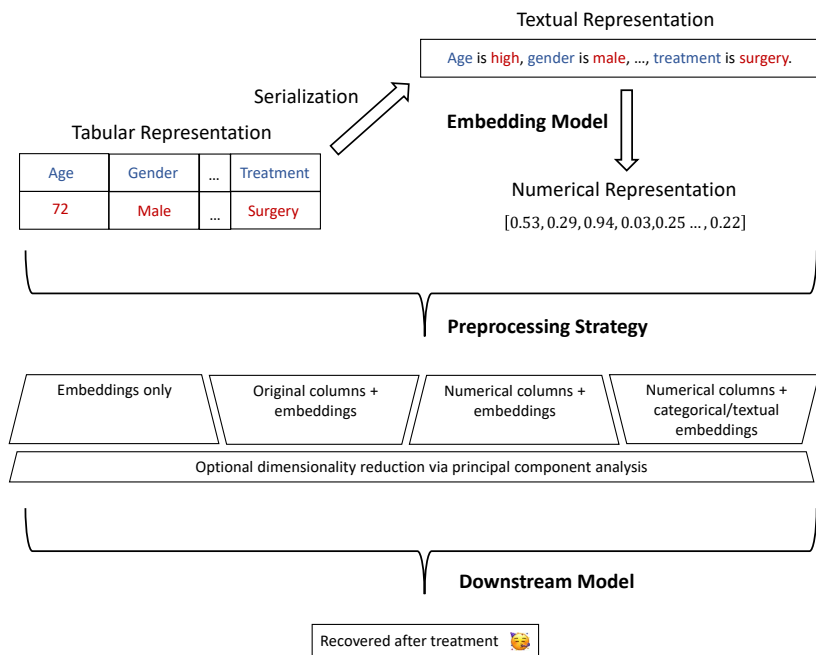


Figure 1: Overview of our work. We serialize tabular rows into text and use embedding models to convert them into numerical representations. These embeddings are preprocessed and serve as inputs for downstream models. We systematically evaluate 256 pipeline configurations, *i.e.*, 16 embedding models, 8 preprocessing strategies, and 2 downstream models.

2 Related Work

Carballo et al. (2023) [3] were the first to demonstrate that LLM-based embedding approaches can improve the predictive performance on tabular tasks. In their study, they included relevant meta-data, excluded missing values, converted numerical into categorical features, and then adopted a sentence-based serialization - one of the most common representations for tabular prediction [4].

Since then, several authors have explored alternative preprocessing strategies, embedding models, and downstream models. Grinsztajn et al. (2023) [5] replaced high-cardinality string features with embeddings and applied dimensionality reduction via principal component analysis (PCA) to them. They tested 26 embedding models, including `bge-large-en-v1.5` and `gte-large`, and used `scikit-learn`'s gradient boosting decision tree as a downstream model. Spinaci et al. (2024) [6] used a similar pipeline, but with `all-MiniLM-L6-v2` as the embedding model and `XGBoost` or `CatBoost` as the downstream model. Kasneci and Kasneci (2024) [7] preprocessed the embeddings with PCA and random forest-based feature selection, and assessed 2 preprocessing strategies (replacing versus concatenating the original columns with the embeddings), 3 embeddings configurations (RoBERTa, GPT-2, or both), and 3 downstream models (random forest, `XGBoost`, or `CatBoost`). Kim et al. (2024) [8] skipped PCA altogether and used `e5-small-v2` and `XGBoost` with or without bagging. Most recently, Koloski et al. (2025) [9] replaced all columns with embeddings, also without PCA. They employed 3 embedding models (`bge-base-en-v1.5`, unspecified MiniLM, or unspecified Llama 3) in combination with 3 downstream models (multi-layer perceptron, ResNet, or FT-Transformer).

Overall, several studies have explored LLM-based embeddings for tabular prediction, but typically under narrow design choices. As a result, best practices for developing effective embedding pipelines remain unclear. In the following, we systematically benchmark 256 embedding pipeline configurations, *i.e.*, 8 preprocessing strategies, 16 embedding models, and 2 downstream models - much more than any prior work. We then assess whether the predictive performance and embedding model characteristics are sufficiently correlated to guide embedding pipeline design.

3 Experimental Setup

3.1 Datasets

To avoid data leakage, we focused our empirical study on datasets that were published no earlier than 2025. Hence, the evaluated embedding models cannot have seen the datasets during their pretraining. We followed the selection criteria of TabArena [10] and chose datasets with a mixture of numerical and categorical/textual features from Kaggle:

- The lungs diseases dataset [11] is a binary classification dataset that aims to predict whether a patient recovered after treatment or not. It contains 5200 samples, 3 numerical features (age, lung capacity, hospital visits), 2 binary features (gender, smoking status), and 2 textual features (disease type, treatment type). The dataset is almost balanced (2408:2492) and there are 300 missing values per feature. We excluded the 300 missing examples in the prediction target.
- The cybersecurity intrusion detection dataset [12] is a binary classification dataset that aims to predict whether or not a cybersecurity attack was detected. It contains 9537 samples, 5 numerical features (network packet size, login attempts, session duration, IP trustworthiness score, failed login attempts), 1 binary feature (unusual time access), and 3 textual features (protocol type, encryption type, browser type). We excluded the session ID feature. The dataset is relatively balanced (4264:5273) and there are no missing values.

3.2 Embedding Pipelines

We followed Carballo et al. (2023) [3] to serialize our tabular rows into text. We then evaluated 256 pipeline configurations, *i.e.*, 16 embedding models across 2 downstream models and 8 preprocessing strategies. The predictive performance was assessed using a 80%/20% train/test split and hyperparameters were selected using a stratified, 5-fold cross-validation and a deviance validation score. All experiments were implemented using scikit-learn 1.5.2 and executed

Table 1: Embedding model characteristics.

Embedding model	Number of para- meters	Embed- ding di- mension	Down- loads August 2025	Release date
all-MiniLM-L6-v2	22M	384	89.63M	2021
bge-base-en-v1.5	109M	768	3.32M	2023
bge-large-en-v1.5	335M	1024	3.63M	2023
bge-small-en-v1.5	33M	384	4.35M	2023
e5-base-v2	109M	768	0.88M	2022
e5-large-v2	335M	1024	0.91M	2022
e5-small-v2	33M	384	0.29M	2022
ember-v1	335M	1024	0.04M	2023
GIST-Embedding-v0	335M	1024	0.36M	2024
GIST-large-Embedding-v0	109M	768	0.02M	2024
GIST-small-Embedding-v0	33M	384	0.46M	2024
gte-base	110M	768	0.67M	2023
gte-base-en-v1.5	137M	768	0.27M	2024
gte-large	330M	1024	1.20M	2023
gte-small	30M	384	0.37M	2023
stella_en_400M_v5	435M	1024	1.39M	2024

on a machine with an NVIDIA A100 GPU, 16 CPU cores, and 20GB of host memory.

3.2.1 Embedding Models

We selected a representative set of 16 LLM-based embedding models from the Massive Text Embedding Benchmark (MTEB) leaderboard [13] with ≤ 1 billion parameters, including `all-MiniLM-L6-v2` [6], `bge-large-en-v1.5` [5], `bge-base-en-v1.5` [9], `e5-small-v2` [8], and `gte-large` [5] from prior work. Our models are encoder-only (BERT-style) models with a masked language model pretraining objective. Please refer to Table 1 for further details. Embeddings were extracted from the `[CLS]` token or via mean pooling that either included or excluded the `[CLS]` and `[SEP]` tokens [14]. The extraction strategy was treated as a hyperparameter.

3.2.2 Downstream Models

We considered 2 downstream models: L2-regularized logistic regression (LR) and histogram-based gradient boosting decision trees (GBDTs), inspired by LightGBM [15]. We tuned the L2-regularization strength for LR on the grid [2, 10, 50] and the minimum number of samples per leaf for the GBDTs on the grid [5, 10, 15, 20].

3.2.3 Preprocessing Strategies

We adopted 8 preprocessing strategies based on prior work: replacing the original columns with the embeddings [7–9], concatenating the original columns with the embeddings (`conc 1`) [7], concatenating the numerical columns with the embeddings (`conc 2`) [8], and concatenating the numerical columns with the embeddings of the categorical/textual columns (`conc 3`) [5, 6]. We additionally applied dimensionality reduction via PCA to 50 components [6, 7] or no reduction [8, 9]. When PCA was used, we normalized our text embeddings prior

to reduction; when PCA was not used, we min-max scaled our embeddings for LR.

3.3 Baselines

As a non-LLM baseline, we included unsupervised random trees embeddings (RTE) [16]. Here, the maximum tree depth (2 or 5) and number of trees (10 or 100) were set as hyperparameters. Furthermore, both downstream models were run on the raw data as baselines, without any text or random trees embeddings. Binary and textual features were treated as nominal [8]. For RTE and LR, nominal values were imputed with the mode if missing and one-hot encoded, while missing numerical values were imputed with an iterative imputer (inspired by MICE [17]). Numerical values were also min-max scaled for LR. Our GBDTs natively support missing values and nominal features, so we only passed nominal feature indicators to them.

3.4 Evaluation Metrics

We evaluated the discriminative performance on the test sets using the area under the receiver operating characteristic curve (AUC). In addition, we report the macro-averaged F1-score (F1) and the balanced accuracy (BA) on the test sets.

To examine whether embedding model characteristics can provide guidance for embedding pipeline design, we analyzed the Spearman rank correlation between the predictive performance and several attributes: the MTEB leaderboard score [13], the information sufficiency score [18], the TransformerRanker score [19], the number of parameters, the embedding dimension, the number of downloads in August 2025 (as a proxy for popularity), and the release date (Table 1). The information sufficiency and TransformerRanker scores serve as unsupervised metrics to estimate the suitability of embedding models for downstream tasks [18, 19].

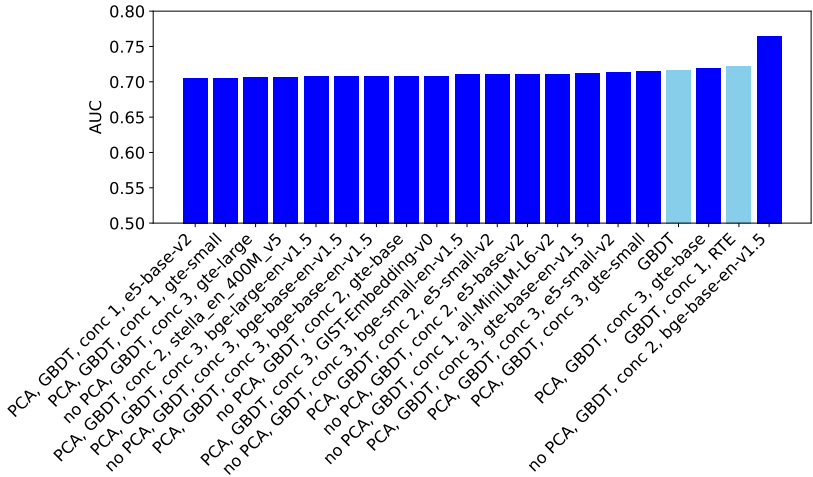


Figure 2: Top-20 mean embedding pipeline performance. Baselines are shown in light blue.

4 Experimental Results

4.1 Predictive Performance

Out of the 256 embedding pipeline configurations, only 1 pipeline achieved an appreciably better performance than vanilla GBDTs (Figure 2 and Table 2 in the Appendix). The pipeline employed `bge-base-en-v1.5` without PCA, concatenated the numerical columns with the embeddings, and used GBDTs as the downstream model. With a mean test AUC of 0.77, it outperformed vanilla GBDTs by 0.05.

In contrast to prior work [9], most pipeline configurations benefited from concatenating embeddings instead of replacing the original columns with embeddings (Figure 4 in the Appendix). Applying PCA showed mixed results (Figure 5 in the Appendix). Moreover, GBDTs generally achieved a better downstream performance than LR (Figure 6 to 9 in the Appendix). The best embedding model, on the other hand, could vary significantly depending on the preprocessing strategy and downstream model, highlighting that identifying

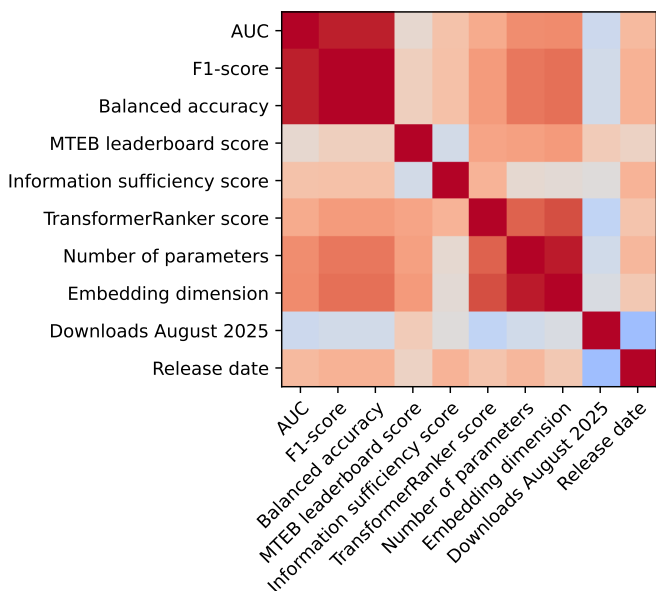


Figure 3: Correlation matrix heatmap between the predictive performance and embedding model attributes, based on the Spearman rank correlation.

consistently effective embedding models is challenging (Figure 6 to 9 in the Appendix).

4.2 Linking Model Properties to Performance

Embedding model characteristics can provide some guidance to embedding pipeline design (Figure 3). We observed a moderate correlation (0.56) between the discriminative performance and both the number of parameters and embedding dimension. Our best-performing pipeline configuration was based on `bge-base-en-v1.5`, though, which only has 109M parameters. The TransformerRanker score (0.40) and the information sufficiency score (0.26) showed weaker associations. Surprisingly, the MTEB leaderboard score was almost uncorrelated with the discriminative performance (0.08), while the number of downloads in August 2025 even showed a slight negative

association (-0.12). These findings indicate that larger embedding models are more promising for tabular prediction tasks, whereas public leaderboard rankings and model popularity provide little guidance in selecting effective models.

5 Conclusion

Embedding approaches allow researchers and practitioners to efficiently enrich data-driven machine learning methods with the world knowledge of LLMs. Based on a systematic benchmark of 256 pipeline configurations, we show that it strongly depends on the specific pipeline design whether incorporating the prior knowledge of LLMs improves the predictive performance. In general, concatenating embeddings tends to perform better than replacing the original columns with embeddings. While larger embedding models tend to deliver better results, their ranking on public leaderboards and their popularity are poor indicators of performance. Finally, gradient boosting decision trees tend to be strong downstream models. Together, our results offer guidance for building more effective embedding pipelines for tabular prediction and lay the groundwork for future research on compressible embedding models [20–22], embedding-specific downstream models [23], or alternative serialization options [4].

Acknowledgements

This research was funded by the Bundesministerium für Bildung und Forschung (BMBF, project number: 16DHBKI071) and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation, project number: 528483508).

References

- [1] R. Knauer, M. Koddenbrock, R. Wallsberger, N.M. Brisson, G.N. Duda, D. Falla, ... & E. Rodner. “Oh LLM, I’m Asking Thee, Please Give Me a

- Decision Tree: Zero-Shot Decision Tree Induction and Embedding with Large Language Models”. In: *KDD 2025*, pp. 1196-1206. 2025.
- [2] M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee & L. Zettlemoyer. “Deep Contextualized Word Representations”. In: *NAACL 2018*, pp. 2227-2237. 2018.
 - [3] K.V. Carballo, L. Na, Y. Ma, L. Boussioux, C. Zeng, L.R. Soenksen & D. Bertsimas. “TabText: A Flexible and Contextual Approach to Tabular Data Representation”. *arXiv preprint arXiv:2206.10381*. 2023.
 - [4] X. Fang, W. Xu, F.A. Tan, J. Zhang, Z. Hu, Y. Qi, ... & C. Faloutsos. “Large Language Models (LLMs) on Tabular Data: Prediction, Generation, and Understanding - A Survey”. *Transactions on Machine Learning Research*. 2024.
 - [5] L. Grinsztajn, E. Oyallon, M.J. Kim & G. Varoquaux. “Vectorizing string entries for data processing on tables: when are larger language models better?”. *arXiv preprint arXiv:2312.09634*. 2023.
 - [6] M. Spinaci, M. Polewczyk, J. Hoffart, M.C. Kohler, S. Thelin & T. Klein. “PORTAL: Scalable Tabular Foundation Models via Content-Specific Tokenization”. In: *NeurIPS 2024 Table Representation Learning Workshop*. 2024.
 - [7] G. Kasneci & E. Kasneci “Enriching Tabular Data with Contextual LLM Embeddings: A Comprehensive Ablation Study for Ensemble Classifiers”. *arXiv preprint arXiv:2411.01645*. 2024.
 - [8] M.J. Kim, L. Grinsztajn & G. Varoquaux. “CARTE: Pretraining and Transfer for Tabular Learning”. In: *ICML 2024*. 2024.
 - [9] B. Koloski, A. Margeloiu, X. Jiang, B. Škrlj, N. Simidjievski & M. Jamnik. “LLM Embeddings for Deep Learning on Tabular Data”. *arXiv preprint arXiv:2502.11596*. 2025.
 - [10] N. Erickson, L. Purucker, A. Tschalzev, D. Holzmüller, P.M. Desai, D. Salinas & F. Hutter. “TabArena: A Living Benchmark for Machine Learning on Tabular Data”. *arXiv preprint arXiv:2506.16791*. 2025.

- [11] S. Dalvi. “Lungs Diseases Dataset”. <https://www.kaggle.com/datasets/samikshadalvi/lungs-diseases-dataset>. 2025.
- [12] D.N.K. Samudrala. “Cybersecurity Intrusion Detection Dataset”. <https://www.kaggle.com/datasets/dnkumars/cybersecurity-intrusion-detection-dataset>. 2025.
- [13] N. Muennighoff, N. Tazi, L. Magne & N. Reimers. “MTEB: Massive Text Embedding Benchmark”. In: *EACL 2023*, pp. 2014-2037. 2023.
- [14] N. Reimers & I. Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *EMNLP 2019*, pp. 3982-3992. 2019.
- [15] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, ... & T.Y. Liu “LightGBM: A Highly Efficient Gradient Boosting Decision Tree” In: *NeurIPS 2017*. 2017.
- [16] F. Moosmann, B. Triggs & F. Jurie. “Fast Discriminative Visual Codebooks using Randomized Clustering Forests” In: *NeurIPS 2006*. 2006.
- [17] S. van Buuren & K. Groothuis-Oudshoorn. “mice: Multivariate imputation by chained equations in R”. *Journal of Statistical Software*, pp. 1-67. 2011.
- [18] M. Darrin, P. Formont, I. Ayed, J.C. Cheung & P. Piantanida. “When is an Embedding Model More Promising than Another?” In: *NeurIPS 2024*. 2024.
- [19] L. Garbas, M. Ploner & A. Akbik. “TransformerRanker: A Tool for Efficiently Finding the Best-Suited Language Models for Downstream Classification Tasks”. In: *NAACL 2025*, pp. 295-302. 2025.
- [20] M. Choi, S. Dua & A. Lisak. “Introducing EmbeddingGemma: The Best-in-Class Open Model for On-Device Embeddings”. <https://developers.googleblog.com/en/introducing-embeddinggemma/>. 2025.

- [21] Z. Nussbaum & B. Duderstadt. “Training Sparse Mixture Of Experts Text Embedding Models”. *arXiv preprint arXiv:2502.07972*. 2025.
- [22] S. Zhuang, S. Wang, F. Zheng, B. Koopman & G. Zuccon. “Starbucks-v2: Improved Training for 2D Matryoshka Embeddings”. *arXiv preprint arXiv:2410.13230*. 2025.
- [23] P. Janson, W. Zhang, R. Aljundi & M. Elhoseiny. “A Simple Baseline that Questions the Use of Pretrained-Models in Continual Learning”. In *NeurIPS 2022 Workshop on Distribution Shifts*. 2022.

Appendix

Table 2: Top-20 mean embedding pipeline performance.

Embedding pipeline	AUC	F1	BA
no PCA, GBDT, conc 2, bge-base-en-v1.5	0.77	0.76	0.75
GBDT, conc 1, RTE	0.72	0.71	0.71
PCA, GBDT, conc 3, gte-base	0.72	0.71	0.71
GBDT	0.72	0.70	0.70
PCA, GBDT, conc 3, gte-small	0.72	0.70	0.70
PCA, GBDT, conc 3, e5-small-v2	0.71	0.70	0.70
PCA, GBDT, conc 3, gte-base-en-v1.5	0.71	0.70	0.70
no PCA, GBDT, conc 1, all-MiniLM-L6-v2	0.71	0.70	0.70
no PCA, GBDT, conc 2, e5-base-v2	0.71	0.70	0.70
PCA, GBDT, conc 2, e5-small-v2	0.71	0.70	0.69
no PCA, GBDT, conc 3, bge-small-en-v1.5	0.71	0.71	0.71
PCA, GBDT, conc 3, GIST-Embedding-v0	0.71	0.71	0.70
no PCA, GBDT, conc 2, gte-base	0.71	0.70	0.69
PCA, GBDT, conc 3, bge-base-en-v1.5	0.71	0.70	0.70
no PCA, GBDT, conc 3, bge-base-en-v1.5	0.71	0.70	0.70
PCA, GBDT, conc 3, bge-large-en-v1.5	0.71	0.71	0.70
PCA, GBDT, conc 2, stella_en_400M_v5	0.71	0.70	0.70
no PCA, GBDT, conc 3, gte-large	0.71	0.69	0.68
PCA, GBDT, conc 1, gte-small	0.71	0.70	0.70
PCA, GBDT, conc 1, e5-base-v2	0.71	0.70	0.70

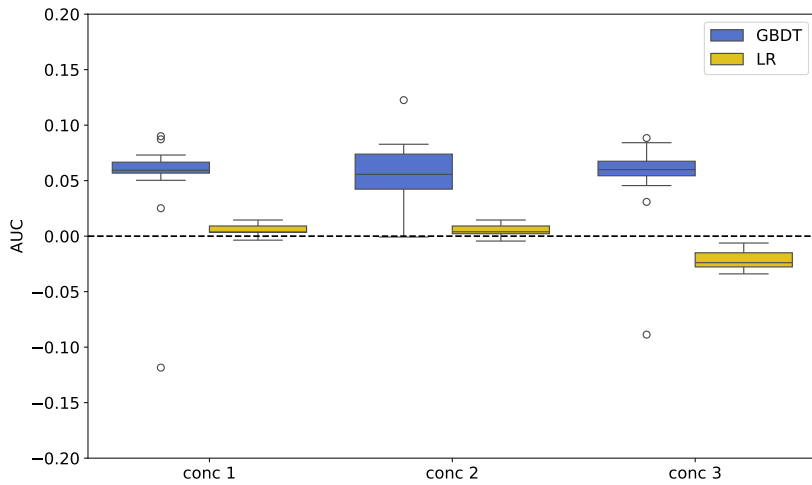


Figure 4: Mean AUC gains by concatenating text embeddings instead of replacing the original columns with text embeddings.

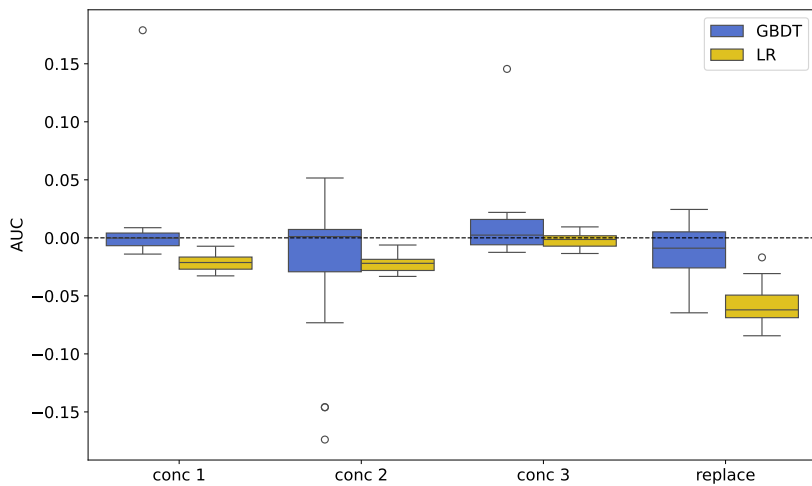


Figure 5: Mean AUC gains by applying dimensionality reduction via PCA to text embeddings.

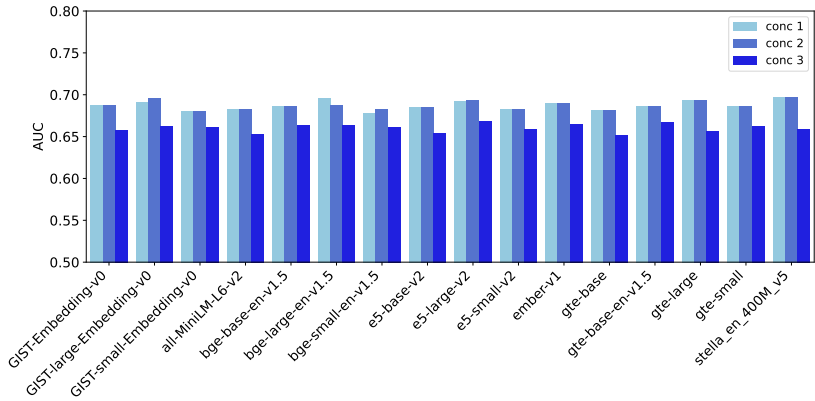


Figure 6: Mean embedding model performance for LR without PCA.

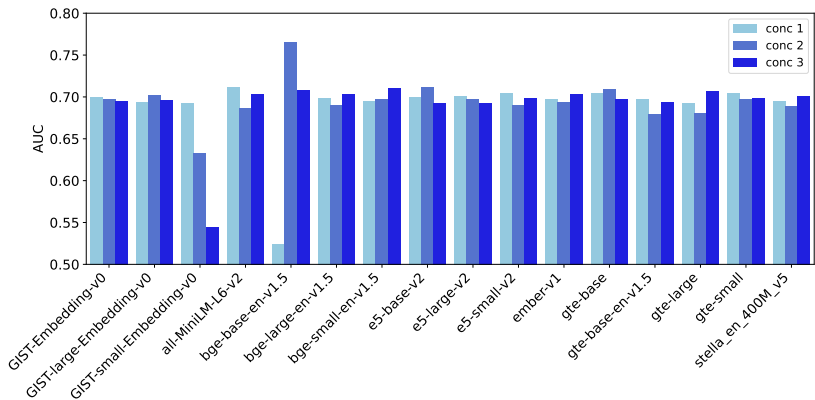


Figure 7: Mean embedding model performance for GBDTs without PCA.

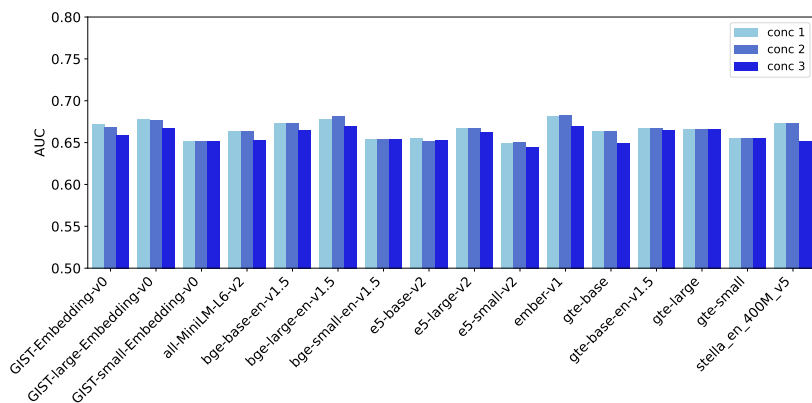


Figure 8: Mean embedding model performance for LR with PCA.

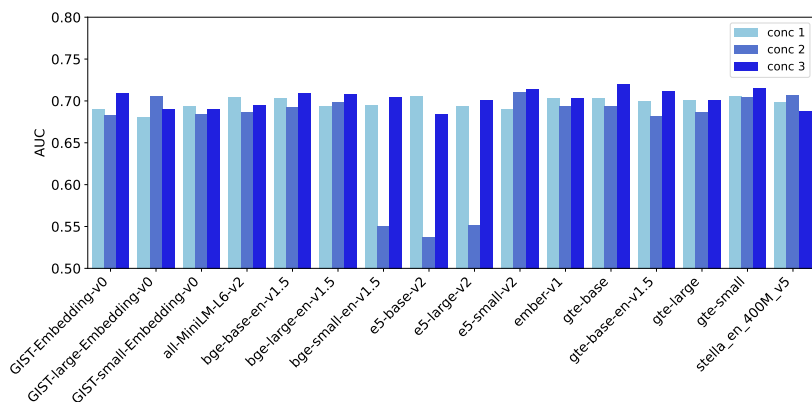


Figure 9: Mean embedding model performance for GBDTs with PCA.

Tuning for Explainability: Incorporating XAI Consistency into Multi-Objective Hyperparameter Optimization

Alexander Hinterleitner¹, Christoph Leitenmeier², Sebastian Spengler², Thomas Bartz-Beielstein¹

¹Institute IDE+a, TH Köln
Steinmüllerallee 1, 51643 Gummersbach
E-Mail: alexander.hinterleitner@th-koeln.de,

thomas.bartz-beielstein@th-koeln.de

²Everllence SE
Stadtbachstraße 1, 86224 Augsburg
E-Mail: christoph.leitenmeier@everllence.com,
sebastian.spengler@everllence.com

1 Introduction

Explainable artificial intelligence (XAI) has become essential for deploying machine learning in domains where decisions must be transparent and defensible. Yet a central obstacle remains the persistent disagreement between post-hoc feature attribution methods. When the same model and data are examined with different explainers such as SHAP [1], LIME [2], or Integrated Gradients [3], the resulting attributions often diverge [4–7]. Different evaluation criteria such as fidelity, robustness, or human alignment assess the different attributions from different perspectives, which might amplify these discrepancies [8–11]. The result is confusion about what the model has actually learned and diminished trust in its use.

We propose a shift from benchmarking explainers to tuning architectures for trustworthiness. Rather than asking which explainer is best, we evaluate the

model as the unit of analysis by quantifying agreement among diverse feature attribution methods, a perspective we call *cross-method XAI consistency*. A model is trustworthy when these methods yield coherent, convergent explanations of its decision logic. To operationalize this perspective, we formalize cross-method XAI consistency through an aggregation and normalization procedure that turns local attributions into comparable global profiles across explainers. We then quantify agreement with suitable measures and integrate the resulting consistency objective alongside predictive performance in a multiobjective hyperparameter-tuning framework, guiding model selection toward solutions that are both accurate and consistently interpretable.

We illustrate the need for explainable neural networks and the benefits of the perspective shift with a real world application from turbocharger development, where the target is to predict compressor blade High-Cycle-fatigue (HCF) vibrations from operating conditions and design variables.

2 Methods

We treat interpretability as a design objective and quantify it via a model-level, cross-method consistency score. For a trained model f_λ with hyperparameters λ , we compute global feature-attribution profiles from multiple post-hoc explainers and measure their agreement. This yields a scalar consistency objective that we optimize jointly with predictive loss in a surrogate-based, multi-objective hyperparameter-tuning framework.

2.1 Global attribution aggregation and consistency metric

Let $\mathcal{M} = \{M_1, \dots, M_m\}$ be m feature attribution methods and d data samples $\mathcal{D} = \{x^{(p)}\}_{p=1}^d$ with k features. Each method M_i produces local attribution vectors:

$$\vec{a}_i^{(p)} = (a_{i1}^{(p)}, \dots, a_{ik}^{(p)})^\top \in \mathbb{R}^k$$

We aggregate to a global vector per method by summing over $\mathcal{D}$:

$$\vec{\hat{m}}_i = \sum_{p=1}^d \vec{d}_i^{(p)} = (\tilde{m}_{i1}, \dots, \tilde{m}_{ik})^\top \in \mathbb{R}^k$$

For scale comparability, we L2-normalize vectors:

$$\hat{\vec{m}}_i = \frac{\vec{\hat{m}}_i}{\|\vec{\hat{m}}_i\|_2} \quad \text{with} \quad \|\vec{\hat{m}}_i\|_2 = \sqrt{\sum_{j=1}^k \tilde{m}_{ij}^2}, \quad \|\hat{\vec{m}}_i\|_2 = 1 \text{ if } \vec{\hat{m}}_i \neq \vec{0}$$

We retain attribution *signs* so that effect reversals are reflected in the agreement measure. Given normalized global vectors, we measure pairwise disagreement by Euclidean distance:

$$d_{i\ell} = \|\hat{\vec{m}}_i - \hat{\vec{m}}_\ell\|_2 = \sqrt{\sum_{j=1}^k (\hat{m}_{ij} - \hat{m}_{\ell j})^2} \in [0, 2]$$

Finally, the model-level consistency score c_λ is obtained by averaging over all unordered pairs:

$$c(\lambda) = \frac{2}{m(m-1)} \sum_{i < \ell} d_{i\ell}$$

Lower $c(\lambda)$ indicates higher cross-method agreement (identical vectors $\Rightarrow c(\lambda) = 0$). Alternative agreement measures (e.g., cosine similarity, Spearman rank correlation) are compatible with our framework *spotpython*¹. However in this work we use Euclidean distance as the primary consistency metric for experiments due to its metric properties and bounded, interpretable range on the unit sphere.

2.2 Desirability formulation

Let $\mathcal{L}(\lambda)$ denote validation loss (to be minimized) and $c(\lambda)$ the Euclidean distance as described in Section 2.1 (also to be minimized). We map each

¹ <https://github.com/sequential-parameter-optimization>

response to a unit scale via desirability functions $d_r(\lambda) \in [0, 1]$ and aggregate by the geometric mean with $R = 2$ [12–14]:

$$D(\lambda) = \left(\prod_r d_r(\lambda) \right)^{1/R}$$

For a response to be *minimized* (loss or distance), we use the standard Derringer–Suich form with user bounds $A < B$:

$$d^{\min}(y) = \begin{cases} 1, & y \leq A, \\ \left(\frac{B-y}{B-A} \right), & A < y \leq B, \\ 0, & y > B. \end{cases}$$

2.3 Surrogate model and optimization loop

We maximize $D(\lambda)$ using a Gaussian-process (Kriging) surrogate and Expected Improvement (EI) as follows: (i) initialize with a space-filling design; (ii) train f_λ , compute $\mathcal{L}(\lambda)$ and $c(\lambda)$, transform to $d_{\text{loss}}, d_{\text{cons}}$ and $D(\lambda)$; (iii) fit the surrogate to $\{(\lambda, D(\lambda))\}$; (iv) select the next configuration by maximizing EI; (v) iterate until the budget limit or convergence is reached.

3 Real World Use Case: HCF Vibration Prediction

Modern turbochargers are being pushed to higher pressure ratios and flow capacities across automotive, marine, and power applications, demanding blading that remains safe under HCF despite stronger excitations and tighter clearances [15–18]. While simulation and physical tests remain essential, their cost motivates data-driven surrogates to support early design screening, including direct prediction of vibration amplitudes. In this work, we predict vibration amplitudes at compressor blades from operating conditions and design variables. The input space comprises 45 features spanning geometry descriptors, mechanical quantities, and thermodynamic states (multiple temperatures and pressures). The target is the blade vibration amplitude measured in terms of

mechanical stress. A central challenge is generalizing vibrational behavior to new compressor-wheel geometries. To emulate this, we tune and train models on data from two specific geometries and evaluate on a third, previously unseen geometry to perform an out-of-distribution (OOD) split by design. This use case demonstrates a domain in which data-driven models are expected to be accurate, physically plausible, and consequently transparent and interpretable.

4 Experiment Setup

We tune a Multi-Layer Perceptron (MLP) using the `NNFunnelRegressor` class from *spotpython* [14] over 12 hyperparameters (number of layers, units, dropout, activation, optimizer, learning rate, batch size, weight decay, initialization, patience, etc.) under two tuning regimes: (i) MSE-only with objective $\mathcal{L}(\lambda)$, and (ii) a desirability formulation that combines $\mathcal{L}(\lambda)$ with cross-method XAI consistency $c(\lambda)$ as described in Sections 2.1 and 2.2. As feature attribution methods we use Integrated Gradients [3], KernelSHAP [1], and DeepLIFT [20]. The feature attribution baseline is constructed by averaging feature values within the lowest 2% quantile of amplitude values, which we assume to approximate a neutral state of the system. Each run starts with 20 space-filling initial points and continues with Bayesian optimization (Section 2.3) for 60 proposals. Per tuning protocol, we select the best model according to the respective objective and compare the two selected architectures on the OOD test set using paired evaluation on identical seeds. The number of seed pairs is $N = 25$, set by a paired sample-size determination on per-seed test differences (one-sided $\alpha = 0.05$, power 0.8) [19]. Primary endpoints are test MSE and Euclidean distance between the feature attribution profiles. We test the hypothesis that the multi-objective model improves over the loss-only baseline using a one-sided paired t -test on both metrics. We calculate $\text{mean} \pm \text{SD}$ over N seeds and 95% one-sided lower confidence bounds for the mean paired differences, together with the corresponding one-sided p -values.

5 Results and Conclusion

We compare the architectures determined by the two tuning approaches, as described in Section 4 (see Figure 1). Incorporating cross-method XAI consistency during hyperparameter tuning yields more consistent explanations on the test data, measured by the Euclidean distance between global attribution profiles. This is supported by the 95% one-sided lower confidence bound $[0.277, \infty)$ for the mean paired difference in Euclidean distances, defined as $\Delta(c(\lambda)) = c(\lambda)^{\text{loss-only}} - c(\lambda)^{\text{multi-objective}}$ (mean difference: 0.401, $p = 5.55 \times 10^{-6}$). A further, and counter-intuitive observation is that the performance of the architecture found by the multi-objective tuning approach on out-of-distribution data is significantly better, as indicated by the 95% one-sided lower confidence bound $[41.69, \infty)$ for the mean paired test-loss difference, defined as $\Delta(\mathcal{L}(\lambda)) = \mathcal{L}(\lambda)^{\text{loss-only}} - \mathcal{L}(\lambda)^{\text{multi-objective}}$ (mean difference: 45.55, $p = 6.92 \times 10^{-17}$). One possible reason is that adding an additional objective can act as an implicit regularizer that reduces attraction to poor local minima in the optimization landscape [21]. Another reason could be that more consistent explanations are associated with more robust and physically plausible decision processes in neural networks. This phenomenon should be investigated further in future studies.

References

- [1] S. M. Lundberg und S.-I. Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, S. 4765–4774. Red Hook, NY: Curran Associates. 2017.
- [2] M. T. Ribeiro, S. Singh und C. Guestrin. “Why Should I Trust You? Explaining the Predictions of Any Classifier”. In: *Proc., 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’16)*, S. 1135–1144. New York, NY: ACM. 2016.

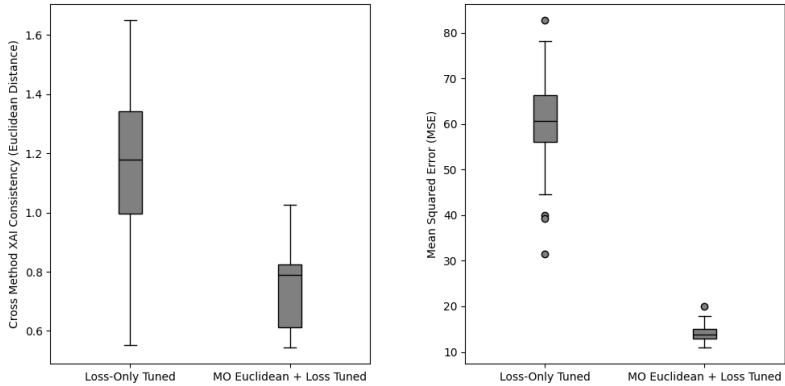


Figure 1: Box-plot comparison of architectures obtained via (i) loss-only tuning and (ii) multi-objective tuning that jointly optimizes training loss and the Euclidean distance between feature attributions. Each box summarizes results over 25 random seeds. Left: Euclidean distance between feature attributions (cross-method XAI consistency). Right: Mean Squared Error (MSE).

- [3] M. Sundararajan, A. Taly und Q. Yan. “Axiomatic Attribution for Deep Networks”. In: *Proc., 34th International Conference on Machine Learning (ICML 2017)*. Proceedings of Machine Learning Research, Bd. 70, S. 3319–3328. PMLR. 2017.
- [4] S. Roy, G. Laberge, B. Roy, F. Khomh, A. Nikanjam und S. Mondal. “Why don’t XAI techniques agree? Characterizing the disagreements between post-hoc explanations of defect predictions”. In: *Proc., 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, S. 444–448. IEEE. 2022.
- [5] S. Krishna, T. Han, A. Gu, S. Wu, S. Jabbari und H. Lakkaraju. “The disagreement problem in explainable machine learning: A practitioner’s perspective”. arXiv preprint arXiv:2202.01602. 2022.
- [6] D. Brughmans, L. Melis und D. Martens. “Disagreement amongst counterfactual explanations: how transparency can be misleading”. In: *TOP 32.3*, S. 429–462. 2024.

- [7] N. Bhardwaj und G. Parashar. “The disagreement dilemma in explainable AI: Can bias reduction bridge the gap”. In: *International Journal of System Assurance Engineering and Management*, S. 1–20. Springer. 2025.
- [8] F. Doshi-Velez und B. Kim. “Towards a rigorous science of interpretable machine learning”. arXiv preprint arXiv:1702.08608. 2017.
- [9] G. Vilone und L. Longo. “Notions of explainability and evaluation approaches for explainable artificial intelligence”. In: *Information Fusion* 76, S. 89–106. 2021.
- [10] M. Nauta, J. Trienes, S. Pathak, E. Nguyen, M. Peters, Y. Schmitt, J. Schlötterer, M. Van Keulen und C. Seifert. “From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable AI”. In: *ACM Computing Surveys* 55.13s, S. 1–42. 2023.
- [11] S. Stassin, A. Englebert, G. Nanack, J. Albert, N. Versbrægen, G. Peiffer, M. Doh, N. Riche, B. Benoît und C. De Vleeschouwer. “An experimental investigation into the evaluation of explainability methods for computer vision”. In: *Proc., Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2023)*, S. 91–106. Springer. 2023.
- [12] E. C. Harrington. “The desirability function”. In: *Industrial Quality Control* 21.10, S. 494–498. 1965.
- [13] G. Derringer und R. Suich. “Simultaneous optimization of several response variables”. In: *Journal of Quality Technology* 12.4, S. 214–219. 1980.
- [14] T. Bartz-Beielstein. “PyTorch Hyperparameter Tuning – A Tutorial for spotpython”. arXiv preprint arXiv:2305.11930. 2023.
- [15] T. Nicholas. “Critical issues in high cycle fatigue”. In: *International Journal of Fatigue* 21, S. S221–S231. 1999.

- [16] M. P. Castanier und C. Pierre. “Modeling and analysis of mistuned bladed disk vibration: current status and emerging directions”. In: *Journal of Propulsion and Power* 22.2, S. 384–396. 2006.
- [17] R. Weber und A. Kühborn. “Misstuning und Dämpfung: Misstuning und Dämpfung von Radialturbinen III”. Technischer Bericht Nr. 1179. Brandenburgische Technische Universität Cottbus. 2019.
- [18] S. Spengler, C. Leitenmeier, T. Willeke und L. Aurahs. “High Cycle Fatigue—Advanced development and design methods for increased robustness”. In: *Proc., 30th CIMAC World Congress*. 2023.
- [19] S. A. Julious. “Sample sizes for clinical trials with normal data”. In: *Statistics in Medicine*, Bd. 23, Nr. 12, S. 1921–1986. Wiley Online Library. 2004.
- [20] A. Shrikumar, P. Greenside und A. Kundaje. “Learning Important Features Through Propagating Activation Differences”. In: *arXiv preprint arXiv:1704.02685*. 2017.
- [21] J. D. Knowles, R. A. Watson und D. Corne. “Reducing Local Optima in Single-Objective Problems by Multi-Objectivization”. In: *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO '01)*. Springer-Verlag, Berlin/Heidelberg, 2001, S. 269–283.

Self-Supervised Learning Strategies for a Platform to Test the Toxicity of New Chemicals and Materials

Thomas Lautenschlager¹, Nils Friederich^{1,2}, Angelo Jovin Yamachui Sitcheu¹, Katja Nau¹, Gaëlle Hayot², Thomas Dickmeis², Ralf Mikut¹

¹Institute for Automation and Applied Informatics (IAI)

²Institute for Biological and Chemical Systems - Biological Information Processing (IBCS-BIP)

Karlsruhe Institute of Technology

E-Mail: thomas.lautenschlager@kit.edu

Abstract

High-throughput toxicity testing offers a fast and cost-effective way to test large amounts of compounds. A key component for such systems is the automated evaluation via machine learning models. In this paper, we address critical challenges in this domain and demonstrate how representations learned via self-supervised learning can effectively identify toxicant-induced changes. We provide a proof-of-concept that utilizes the publicly available EmbryoNet dataset, which contains ten zebrafish embryo phenotypes elicited by various chemical compounds targeting different processes in early embryonic development. Our analysis shows that the learned representations using self-supervised learning are suitable for effectively distinguishing between the modes-of-action of different compounds. Finally, we discuss the integration of machine learning models in a physical toxicity testing device in the context of the TOXBOX project.

1 Introduction

The REACH (Registration, Evaluation, Authorisation and Restriction of Chemicals) regulation, introduced in 2007, aims to better understand chemical compounds entering the EU (European Union) market [1]. According to REACH, companies that import or produce a certain compound in quantities exceeding one tonne, are obligated to test the compounds for toxicity and report the results to the European Chemicals Agency (ECHA) [1]. Over 23 000 different compounds were registered under REACH as of 2025 [2].

This illustrates the need for large amounts of toxicity tests to be conducted. Typically, these tests are done *in vivo* using rats or other mammals [3]. However, they are relatively costly due to housing and feeding needed, as well as the comparably low reproduction rate of these animals. Furthermore, animal testing requires lengthy legal procedures and poses ethical concerns. Russell and Burch formulated the 3R principles that aim to replace, reduce and refine tests conducted on animals, where possible [4]. Consequently, interest in alternative forms of toxicity testing is increasing [3].

For approaches deviating from traditional *in vivo* studies, the umbrella term of New Approach Methodologies (NAMs) was coined. In the context of NAMs, tests that are suitable for High-Throughput Screening (HTS) are often discussed. These include for example *in vivo* studies using zebrafish (*Danio rerio*) embryos [5]. They are cheaper to rear due to lower maintenance costs and a higher progeny number than animals traditionally used in toxicity testing. Furthermore, according to EU legislation, zebrafish embryos are not considered animals up until 5 days post fertilization (DPF), facilitating easier adoption for testing [6]. *Daphnia magna* is another species that is often considered for NAMs and HTS [7]. Apart from *in vivo* testing, *in vitro* tests using cell-based assays or organ models are also rising in popularity [3]. While cell-based assays are suitable for HTS experiments, the viability of organ models for HTS is being actively investigated [8]. More complex approaches, such as organs-on-a-chip or extensions using multiple connected organs for body-on-a-chip systems, are discussed as well [9].

The vast amount of data generated by such HTS approaches, however, necessitates the use of automated evaluation methods, which can be achieved using ML models. While most of the literature on ML in toxicology is focused

on *in silico* models, research on ML for toxicity test automation is fairly scarce [10]. However, since *in silico* ML models often show poor generalization to compounds with dissimilar properties to the ones they were trained on [10], there is a need for automatic evaluations of experimental HTS data, using ML. The data generated through HTS is often high-dimensional, encompassing microscopic images and time-series data such as electrochemical readouts. Since Deep Learning (DL) models generally perform better on high-dimensional data than traditional ML models [11], they are better suited for evaluating HTS data. Incorporating Self-Supervised Learning (SSL) into DL models can offer various advantages in the domain of toxicity testing. Since labeled data is often scarce, self-supervised pretraining is a valuable technique for building robust models using smaller datasets for downstream tasks such as classification. Furthermore, the continuous representations learned by SSL can model concentration-dependent gradients of toxicant-induced changes, while the inherent clustering of the learned representations can identify compounds with similar modes-of-action.

Recently, the use of Large Language Models (LLMs) has also been discussed in the context of toxicity predictions [12]. LLMs are mainly considered in the context of data extraction and data curation from different toxicological databases or from scientific literature [13]. They could also be utilized to directly make predictions based on a given literature database using Retrieval-Augmented Generation (RAG) or fine-tuned LLMs [14]. However, since LLMs are prone to hallucinations [15], their application and wider adoption should be done cautiously. Fusion approaches for ML models, combining different inputs, can also be explored. In this way, different experimental data as well as physicochemical properties of the tested compounds could be combined for a single toxicity prediction. A summary of the discussed approaches to computational models for toxicity predictions is given in Figure 1.

While the discussion of individual toxicological endpoints is beyond the scope of this paper, it is important to note that computational models often only make predictions on one or a few toxicological endpoints. ML models can only accurately predict toxicity for the endpoints for which the model was trained on. In this paper, we mostly focus on discussing image-based approaches because most of the published work on toxicity tests that are viable for HTS focuses on the automatic evaluation of images. The outline for the rest of this paper is

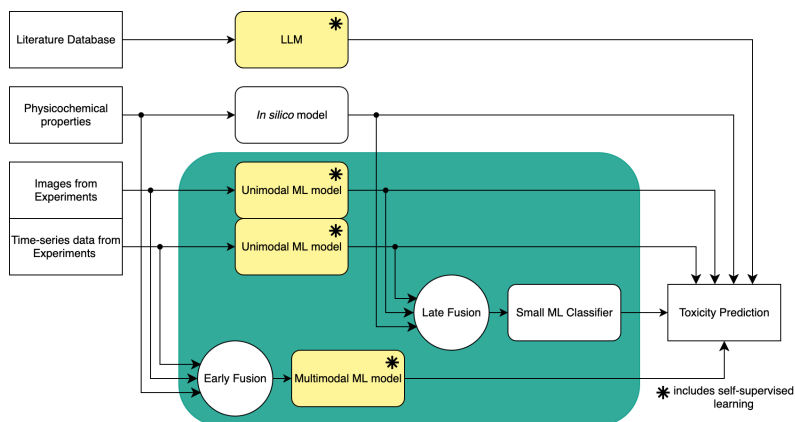


Figure 1: Diagram of different computational models for toxicity predictions. The green box denotes the approaches more closely discussed in this paper. Models that incorporate SSL are marked with an asterisk and highlighted in yellow.

as follows: In Section 2, we discuss existing DL approaches that are suitable for HTS scenarios. We provide a brief introduction to SSL in general and portray advantages of SSL specific to toxicity testing in Section 3. Section 4 contains a proof-of-concept demonstrating that representations learned by SSL can successfully identify toxicant-induced changes and relate the same modes-of-action to each other. Using various analyses, we investigate properties of the learned representations in detail and discuss them. Lastly, in Section 5, we address challenges arising from the integration of DL models into TOXBOX, a real-world toxicity testing device and outline strategies to tackle them.

2 Related Work

There are several works concerned with using DL models for evaluating toxicity tests, which we are going to discuss here. These works could be adapted for use in HTS scenarios. However, regarding species commonly used in toxicity testing, there is a notable lack of research that consistently focuses on a single aspect of toxicity testing for a specific species. Even less work focuses on the automation of established toxicity tests.

As already mentioned, zebrafish are a model organism often discussed in the context of toxicity testing [5]. Several test protocols have been established for zebrafish embryos and larvae <5 DPF. For example, the Fish Embryo Acute Toxicity (FET) Test is OECD-approved [16] and is often discussed as an alternative to the Fish Acute Toxicity Test [17], which uses adult fishes and is therefore not suitable for HTS. Various behavioral tests have also been established. These tests, though often lacking standardized protocols, enable the evaluation of neurotoxicity [18].

DL techniques for the classification of abnormally developing zebrafish embryos have been proposed in several studies [19–22]. However, the different investigations show low consistency. Most of these focus on the classification of hatched eleutheroembryos during different timepoints [19–21], only one publication focuses on earlier embryonic stages [22]. All the publications use different classes in their classification approaches. These inconsistencies in the existing research make comparisons difficult.

There are also no publications that tackle the automatic evaluation of the FET, the only OECD-approved toxicity test involving zebrafish <5 DPF that focuses on phenotypical changes [16]. However, the automatic identification of coagulated zebrafish embryos, one of the endpoints of the FET, has been addressed in several works [22, 23]. Existing approaches have also paid little attention to identifying modes-of-action of the compounds. Only Čapek et al. [22] define the classified phenotypes based on different developmental pathways that can be blocked by certain toxicants. These phenotypes, however, also do not cover all possible toxicant-induced changes.

Several papers focus on DL approaches for toxicity testing using *Daphnia magna*. They include models that determine and quantify morphological changes in *Daphnia magna* due to toxicant exposure [24], models to determine the size and growth rate [25] and approaches for tracking *Daphnia magna* [26] as well as identifying compounds based on locomotor tracks [27].

Few approaches exist for the use of DL models in cell assays or organ models for toxicity testing. One approach automatically detects the nuclei of the cells and classifies them as either 'healthy' or 'toxicity-affected' [28]. Another approach uses time-series data based on a cell impedance signal for the classification of different modes-of-action [29]. Cell tracking approaches such as [30, 31] are also suitable for toxicity testing, since features such as the number of cells or

size of cells can also be used to make predictions on the toxicity of a certain compound.

Hu et al. [32] use DL models for predicting the thickness of a skin model. They show that lower thickness of the epidermal layer can be used to predict skin toxicity.

To the best of our knowledge, only two studies are using SSL that can be considered for the automatic evaluation of toxicity tests. Toulany et al. [33] use a Twin Network trained with a triplet loss to investigate the embryonic development in zebrafish. The trained network can be used to determine the similarity between embryo images. This is used for identifying different developmental stages, comparisons regarding the development of zebrafish embryos under different temperatures and detecting deviations from normal development. The authors show that the model can also identify deviations from normal development that are toxicant-induced [33].

In the second paper on SSL in toxicity testing, Gendele et al. [34] use Twin Networks on the Motion Index, a measurement of movement based on pixel intensity changes between frames in videos, from different behavioral tests using 7 dpf zebrafish larvae. This approach can group similar modes-of-action together.

3 Potential of Self-Supervised Learning in Toxicology

3.1 Overview of Self-Supervised Learning

SSL encompasses methods that use a pretext task for learning useful lower-dimensional representations [35]. The pretext task focuses on optimizing for a target t_{SSL} that can be generated from the data itself [36]. Depending on the self-supervised algorithm, the pretext tasks in computer vision can range from predicting the correct order of shuffled image patches [37], mapping two differently augmented views of the same image together [38], or reconstructing image patches that were masked in the input [39].

Typically, SSL uses an encoder that maps the inputs x to a latent space h . Often,

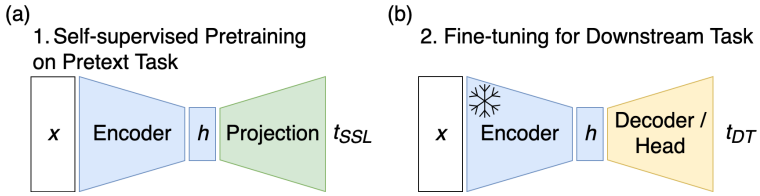


Figure 2: Simplified diagram of (a) pretraining via SSL and (b) fine-tuning for downstream task. The encoder is usually frozen during fine-tuning. x denotes the input of the model, h the learned latent space, and t_{SSL} and t_{DT} the targets of the self-supervised learning task and the downstream task, respectively.

some kind of projection is used on the latent space h , the output of which is used for the optimization regarding the target t_{SSL} . This can, for example, be a linear layer [40], a projection head [38] or a decoder [39].

The desired output of the SSL models are the lower-dimensional representations of the data in the learned latent space h . Generally, in the latent space h , representations from similar inputs are mapped closely together, while representations from dissimilar inputs are mapped further away from each other. In computer vision, these representations are often referred to as visual representations. For the sake of brevity and because SSL can also be used to attain lower-dimensional representations from non-image data, we use the term ‘representations’ for the rest of this paper.

In general computer vision tasks, SSL is often used for pretraining [35]. The learned representations are then used to fine-tune for a specific downstream task, such as classification or segmentation [35]. Depending on the kind of task, a decoder or head is used on top of the usually frozen encoder. The decoder or head is then trained regarding the target t_{DT} of the downstream task. This typical training procedure is pictured in Figure 2. Popular SSL methods for images include: SimCLR [38], MoCo and its extensions [40–42], BYOL [43], SwAV [44], DINO and its extensions [45–47], Masked Autoencoders [39] and SimSiam [48].

3.2 Label-Efficient Model Training

Using pretrained models for training on downstream tasks can also be applied to toxicity testing. This allows for the training of more label-efficient models. Further, the representations learned by SSL are often more generalizable, enabling transfer learning, where the representations from a different, but similar dataset can be used for fine-tuning to a task, where both data and labels are scarce.

Since few image datasets for toxicity testing are published and generating new datasets is expensive, leveraging similar datasets in SSL pretraining makes research on ML models for toxicity testing more feasible. A strategy that could be used to identify suitable datasets was outlined by Yamachui Sitcheu et al. [49].

3.3 Continuous Representations

The latent space h learned by SSL methods offers several additional advantages for toxicity testing. A problem when applying supervised DL models to toxicity tests is that the methods often used do not account for properties specific to toxicity testing.

For example, classification often falls short when evaluating toxicant-induced morphological changes. These changes are usually continuous and the cutoff point is often based on observer experience, with little to no standardization. Thresholds of toxicant-induced changes can therefore vary between studies. It is often unclear whether small changes are already labeled as 'toxic' or only if a clear abnormal phenotype can be observed. Additionally, there are studies that use several classes [22, 24] for different magnitudes of the same phenotypic changes, resulting in even more hazily defined cutoff points. By mapping the samples into a latent space that allows for continuous representations of the morphological changes, SSL offers an elegant solution for this problem. The resulting representation not only allows fine differentiation based on phenotypic changes but can be thought of as concentration-dependent gradients. The learned representations of a certain phenotypical change will be mapped into the same direction away from healthy phenotypes. Since toxicant-induced changes get

more severe with higher concentrations of the compound, the distance to the healthy phenotypes will increase with higher concentrations of the compound. Another downside of supervised classification is that biases can be introduced if small changes due to a toxicant are not represented in the labels used for training. For example, the EmbryoNet-Prime DL model, trained on data with labels shifted 4 hours into the past, can identify morphological changes earlier than the expert who labeled the data [22], indicating that small phenotypical changes are already present.

However, simply shifting the labels can result in false labels, since it is unclear when the phenotypical changes first occur. This can deteriorate model performance. SSL could potentially identify when small phenotypical changes occur without label-induced biases.

3.4 Identification of Similar Modes-of-Action

The clustering inherent to SSL can also be used for the identification of similar modes-of-action. Since similar images will be mapped together and dissimilar images mapped away from each other, similar phenotypes induced by compounds with a similar mode-of-action will be clustered. Gendele et al. [34] have shown that this is possible using time-series data of pixel intensity changes from behavioral zebrafish tests.

Additionally, a classifier without rejection class forces unknown toxicant-induced changes into one of the classes known from the training dataset. In the case of SSL, the representations of an unknown morphological change are mapped away from the representations of the known classes, making it apparent that the representations do not belong to any of the known classes.

4 Preliminary Experiment: Proof-of-Concept

4.1 Methods

We chose SimCLR [38] for our proof-of-concept investigating the properties of the latent space of a SSL model trained on image data showing toxicant-induced

morphological changes. SimCLR is an important baseline in the field of SSL and learns meaningful representations of the data. The self-supervised target t_{SSL} of SimCLR aims at minimizing the distance between the representations of two differently augmented views of the same image while maximizing the distance between views of different images. SimCLR uses cosine similarity as the distance measure between different views [38]. The resulting latent space is a hypersphere on which the representations are mapped.

We used ResNet50 [50] as the backbone, which we trained using SimCLR [38]. After the training, we evaluated the latent space using linear probing, where a linear classifier is trained on the representations that the frozen backbone outputs. This method is a standard procedure in SSL research to assess the quality of learned representations.

Since SimCLR training is unsupervised, the labels of the dataset are only used for training the linear classifier. This usually results in worse performance than training the network fully supervised. However, it can still be useful to evaluate the success of self-supervised training.

To better understand the latent space and the representations SimCLR learned, we visualize the latent space using UMAP for dimensionality reduction [52]. Furthermore, we investigate the representations of each class. Through a forward pass using the training dataset, we obtain the representations for the training dataset. We calculate the centers of each class by taking the mean of the respective class representations

$$\mathbf{c}_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{h}_i \quad (1)$$

where $\mathbf{h}_i$ denotes one learned representation, $\mathbf{c}_k$ represents the center $\mathbf{c}$ for class k , C_k denotes the set of representations belonging to class k and $|C_k|$ its cardinality. The dimensionality of $\mathbf{h}$ and $\mathbf{c}$ are dependent on the type of network used. Since we use ResNet50 as a backbone, the resulting dimensionality for $\mathbf{h}$ and $\mathbf{c}$ is 2048 in our analyses.

After the calculation, the centers are normalized

$$\tilde{\mathbf{c}}_k = \frac{\mathbf{c}_k}{\|\mathbf{c}_k\|} \quad (2)$$

where $\tilde{\mathbf{c}}_k$ refers to the normalized class center of class k and $\|\mathbf{c}_k\|$ to the Euclidean norm of class center $\mathbf{c}_k$.

Next, the class centers are used to calculate the mean cosine similarity of the representations of each class to each center

$$\overline{\text{sim}}_{\cos}(C_l; \tilde{\mathbf{c}}_k) = \frac{1}{|C_l|} \sum_{i \in C_l} \mathbf{h}_i^\top \cdot \tilde{\mathbf{c}}_k \quad (3)$$

where $\overline{\text{sim}}_{\cos}(C_l; \tilde{\mathbf{c}}_k)$ denotes the mean cosine similarity between the set C_l that includes the representations $\mathbf{h}_i$ of the test dataset. Note that the representations of $\mathbf{h}_i$ and $\tilde{\mathbf{c}}_k$ are both normalized in this calculation.

The distance of a point to a class center can be thought of as an anomaly score for that class. To achieve a deeper understanding of the constructed latent space, we also calculate the cosine similarities between the different class centers:

$$\text{sim}_{\cos}(\tilde{\mathbf{c}}_l; \tilde{\mathbf{c}}_k) = \tilde{\mathbf{c}}_l^\top \cdot \tilde{\mathbf{c}}_k \quad (4)$$

4.2 Dataset

For our analyses, we used the publicly available EmbryoNet dataset [22]. It features images of ten different zebrafish embryo phenotypes. There are seven phenotypes, where the used toxicant targeted a major signaling pathway in early embryonic development. The respective phenotypes are named after their affected pathway and whether a loss-of-function or gain-of-function is present: -BMP, +RA, -Wnt, -FGF, -Nodal, -Shh and -PCP. Other classes include the 'Normal' class, featuring normally developing embryos, the 'Dead' class, which includes embryos that have died and coagulated, and the 'Unknown' class, for embryos whose phenotype could not be identified.

The original dataset features embryos that are periodically imaged from 2 hours post fertilization (HPF) to 26 HPF, since developmental aspects are closely discussed in the EmbryoNet paper [22]. However, since we were most interested in classifying the different phenotypes that only become apparent as development progresses, we chose to use only images from the later timepoints. This not only reduced training time but also avoided learning visual features that are not necessary for phenotype classification.

The EmbryoNet dataset consists of images of wells containing multiple zebrafish embryos [22]. We adopted the predefined split between training and test dataset of the EmbryoNet dataset. Additionally, we defined a validation dataset using images from 10% of the wells that make up the training dataset. The chosen wells were randomly sampled. For extracting the individual embryo crops from the well images, we used the bounding boxes, which are provided with the dataset. For the training and validation dataset we used the crops of embryos ranging from 25 HPF to 26 HPF. The evaluations were done using only the last crop of each embryo in the test dataset, which was recorded at 26 HPF. The resulting training, validation and test datasets consist of 135 475, 15 670 and 772 embryo crops, respectively.

4.3 Implementation Details

We used MMPretrain [53] for training SimCLR and the linear classifier. The training was done using 8 NVIDIA A100-40s. Other evaluations were done using custom code and were run on an NVIDIA RTX 3090. The code is available at github.com/lautthom/self_supervised_learning_strategies_toxicity_testing. Details regarding the hyperparameters and augmentations used are available in the config files used for MMPretrain provided with the code.

Unless otherwise noted, we used the same hyperparameters as described in the original SimCLR paper [38]. We trained SimCLR for 200 epochs and reduced the batch size to 2048. The learning rate was adjusted accordingly with square root scaling [51].

For the linear classifier, we also adopted the training and testing procedure as well as the hyperparameters as defined in the SimCLR paper [38], unless otherwise noted. Since the classes in the EmbryoNet dataset are imbalanced, we used a weighted loss function for the training of the linear classifier

$$w_i = \frac{n}{n_i \cdot C} \quad (5)$$

where w_i specifies the weight of class i in the loss function, n the number of total samples, n_i the number of samples in class i and C the number of classes.

Furthermore, we used early stopping based on the accuracy the linear classifier achieved on the validation dataset.

The augmentations were adjusted to fit the domain-specific needs of zebrafish embryo images. The following augmentations were used for both the SimCLR training and the linear classifier training: random crop, horizontal flip, rotations of up to 360°, random brightness changes, random contrast changes, CLAHE, sharpen, motion blur, defocus, grid distortion, optical distortion, elastic transform, salt and pepper noise, Gaussian noise, Poisson noise and solarize.

4.4 Results

The linear classifier trained on top of the representations learned by SimCLR achieved an accuracy of 79.9% on the test dataset. This is about ten percentage points below the 89% accuracy reported in the original EmbryoNet paper [22]. The normalized confusion matrix of the linear classifier trained on the 10 classes is depicted in Figure 3. Dead embryos are classified most reliably with a recall of 100%. Other classes with a high recall are the -BMP, -FGF, -Nodal, +RA and -Wnt phenotypes. The 'Normal' class has a relatively low recall of 60%. The -PCP and -Shh phenotypes also have a low recall. The 'Unknown' phenotype has the lowest recall, however, only 4 images belong to the 'Unknown' class in our test dataset.

An UMAP visualization of the learned representations of the test dataset is given in Figure 4. The visualization reflects the recall values given in Figure 3. The 'Dead' class is mapped far away from the other classes. The representations of the other classes with a high recall are also mapped close to each other and are fairly easy to distinguish from the representations of other classes.

For each class, we calculated the mean cosine similarity between the representations and the centers of the respective class. The results are given in Figure 5. The lowest mean cosine similarity is 0.70. However, for all classes, the mean cosine similarities between their representations and their respective class centers are higher than the mean cosine similarities to all other class centers.

Figure 6 shows the cosine similarities between the different class centers. As in Figure 5 all cosine similarities are fairly high. The lowest cosine similarity between two class centers is 0.86.

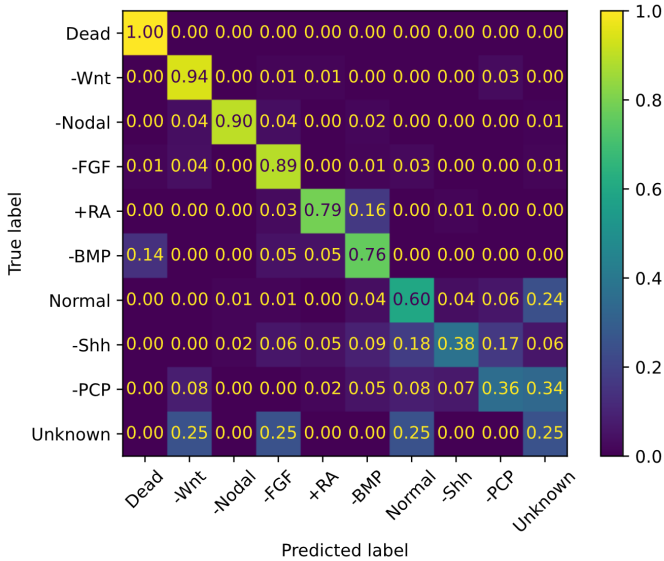


Figure 3: Confusion matrix of the linear classifier trained using SimCLR representations

Due to the high cosine similarities between the centers and representations in Figure 5 as well as between the centers themselves in Figure 6, we took a closer look at the cosine similarities of the individual representations. The minimal cosine similarity between two representations is 0.37, while the mean similarity is 0.64 and the highest cosine similarity is 1.0. This means that only a small part of the hypersphere that the images are mapped to during SSL training is populated by the representations.

4.5 Discussion, Limitations and Outlook

Our investigation of the learned latent space in Section 4.4 leads to different insights. The performance of the linear classifier shows that the learned representations have a fairly good quality. While the classification is markedly worse than the supervised baseline, it still reaches acceptable performance. Since most of the classes correspond to a certain mode-of-action, the clustering

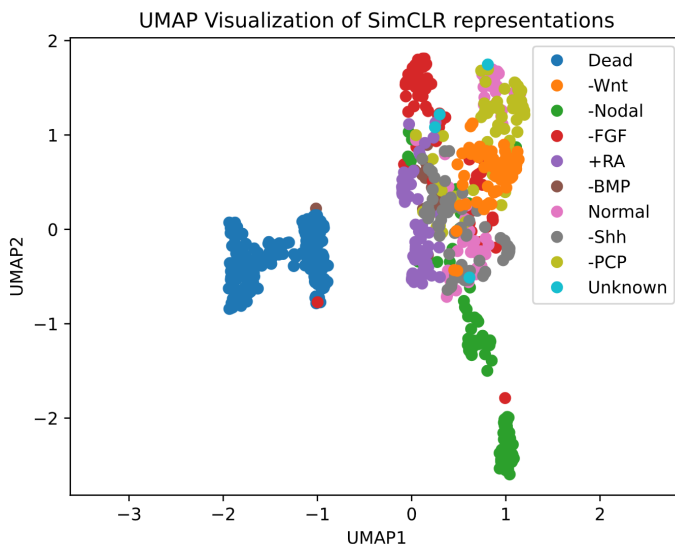


Figure 4: UMAP visualization of SimCLR representations

based on modes-of-action works fairly well. However, the phenotypes in the EmbryoNet dataset are elicited by the same compounds, which makes it hard to evaluate if the clustering was indeed based on the mode-of-action or on some other compound-specific properties. Certain phenotypes reach a particular low recall, for example the -Shh and -PCP phenotype. However, this is also true for the supervised model, as well as for the evaluations of experienced developmental biologists, reported in the EmbryoNet paper [22].

An interesting approach for future research could be to evaluate the representations learned via SSL on other downstream tasks, such as segmentation or transfer learning to another classification task. Presumably, the representations learned by SSL should outperform the ones learned during supervised training in these tasks. A possible application would be the automatic evaluation of the FET [16]. Future investigations could also explore fine-tuning the models with fewer available labels and compare the performance deterioration to fully supervised learning.

A problem with the present latent space is that the representations only populate

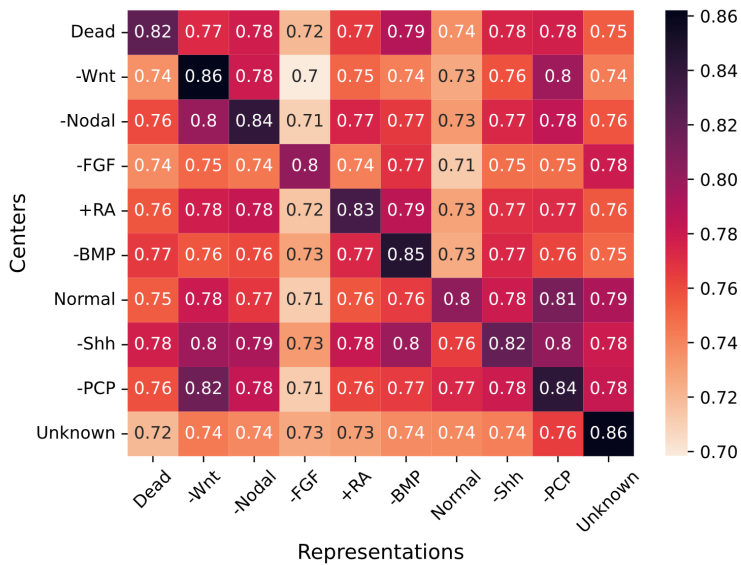


Figure 5: Mean cosine similarities between class centers and representations of the class

a small area of the hypersphere. Investigations show that more uniform distributions on the hypersphere generally improve performance [54]. It is unclear whether this is also true for domain-specific use cases, where the images are very similar to each other.

Further investigations are needed to see if the theoretical considerations presented in Section 3 can be empirically verified. Clustering based on similar modes-of-action was already shown to be feasible for one other application [34] and the results presented in this paper support this finding. Given this, it is likely that concentration-dependent gradients in the latent space also exist. Unfortunately, the EmbryoNet dataset is not best suited for this investigation, since the authors used the same concentrations for the elicitation of most of the phenotypes [22].

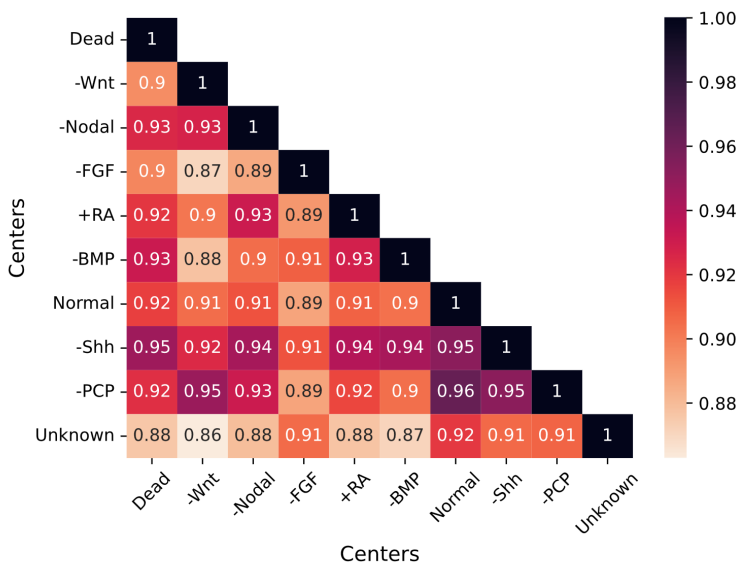


Figure 6: Cosine similarities between the class centers

5 Integration with TOXBOX device

Since the aim is to integrate ML models in high-throughput processes, the ML models also need to be integrated with a physical toxicity testing device. We discuss this in the context of the TOXBOX¹ project. The project aims to design an all-in-one platform for reliable toxicity testing [55]. A prototype of the TOXBOX device is pictured in Figure 7.

TOXBOX will feature different *in vitro* organ models as well as a zebrafish embryo module [55]. Due to the advantages illustrated in this paper, pretraining the models via SSL and then fine-tuning them to the specific toxicity prediction task seems to be the most viable option. This should be especially advantageous if the data and/or labels generated during the TOXBOX project are scarce and data from similar datasets can be leveraged using SSL. Fully supervised models should be trained as well and compared to SSL models, to ensure that the model

¹ <https://toxbox.eu>



Figure 7: Image of the TOXBOX prototype; screenshot from [56]

with the best performance will be used.

Explainable Artificial Intelligence (XAI) methods can help in the evaluation of the different ML models [57]. Since XAI makes the underlying factors that lead to a certain prediction of a ML model more transparent, experts can assess whether the factors used are indicative of toxicity.

Furthermore, the latent space of the SSL models could be used to gain more information about the tested compound. Based on the distance between the representations of known compounds and the tested compound, it can be determined whether the compound has a similar mode-of-action as known compounds or whether it has an unknown effect. Compounds with unknown effects merit more thorough investigation, both on the compounds themselves and also if something went wrong during testing.

A crucial topic to discuss when using ML models in toxicity testing is concept drift. Concept drift refers to gradual changes in the underlying data, which happen over time and can deteriorate the model's performance [58]. This can happen especially easily in toxicity testing if groups of compounds are tested that differ in important aspects from those used in acquiring the training data, particularly when compounds with different modes-of-action are tested. This means that the model's performance needs to be closely monitored to ensure reliable predictions.

If new modes-of-action are found or the model's performance drops due to concept drift, it may be necessary to retrain the model. This can be challenging, as the newly acquired data from the device may be highly imbalanced. Further, the data of the previously unknown mode-of-action can be scarce. Different strategies for retraining the models should be explored and closely evaluated in such scenarios.

6 Conclusion

In this paper, we have illustrated how aspects inherent to SSL are suitable to address different challenges specific to toxicity testing, specifically dealing with sparse labeled data, accounting for continuous changes due to toxicant exposure and identifying similar modes-of-action. We provided a proof-of-concept that demonstrates how representations learned via SSL can, in practice, be utilized for toxicity testing. Further, we discussed various challenges involved in adapting machine learning models to physical toxicity testing devices.

7 Acknowledgments

This work has received funding from the European Union's HORIZON-CL4-2023-RESILIENCE-01 Research and Innovation programme under Grant Agreement No 101138387 (TOXBOX). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union. The European Union cannot be held responsible for them.

This work is supported by the Helmholtz Association Initiative and Networking Fund on the HAICORE@KIT partition.

The present contribution is supported by the Helmholtz Association under the joint research school "HIDSS4Health - Helmholtz Information and Data Science School for Health." and the program "Natural, Artificial and Cognitive Information Processing (NACIP)".

7.1 Author contributions

The authors used the AI language models GPT-4.5 and GPT-5.0 to improve the language and style of this manuscript. The authors have accepted responsibility for the entire content of this manuscript and approved its submission. We describe the individual contributions of T. Lautenschlager (TL), N. Friederich (NF), A.J. Yamachui Sitcheu (AJYS), K. Nau (KN), G. Hayot (GH), T. Dickmeis (TD), R. Mikut (RM): Conceptualization: TL, NF, RM; Methodology: TL, NF, RM; Software: TL; Investigation: TL; Writing - Original Draft: TL; Writing - Review & Editing: TL, NF, AJYS, KN, GH, TD, RM; Supervision: NF, RM; Project administration: RM; Funding Acquisition: TD, RM.

References

- [1] "REACH Regulation," European Commission, [Online]. Available: https://environment.ec.europa.eu/topics/chemicals/reach-regulation_en. [Accessed: Jul. 18, 2025].
- [2] "REACH Registration statistics," European Chemicals Agency (ECHA), [Online]. Available: https://echa.europa.eu/documents/10162/2741157/registration_statistics_en.pdf/58c2d7bd-2173-4cb9-eb3b-a6bc14a6754b. [Accessed: Jul. 18, 2025].
- [3] "The use of alternatives to testing on animal for the REACH Regulation," European Chemicals Agency (ECHA), [Online]. Available: <https://op.europa.eu/en/publication-detail/-/publication/2203ad8b-1ec9-11ee-806b-01aa75ed71a1/>. [Accessed: Aug. 22, 2025]
- [4] W. M. S. Russell and R. L. Burch. "The Principles of Humane Experimental Technique". London: Methuen, 1959.
- [5] L. Yang, N. Y. Ho, R. Alshut, J. Legradi, C. Weiss, M. Reischl, R. Mikut, U. Liebel, F. Müller, and U. Strähle. Zebrafish embryos as models for embryotoxic and teratological effects of chemicals. *Reproductive Toxicology*, vol. 28, no. 2, pp. 245–253, Sep. 2009.

- [6] European Union. Directive 2010/63/EU of the European Parliament and of the Council of 22 September 2010 on the protection of animals used for scientific purposes. *Official Journal of the European Union*, vol. L 276, pp. 33–79, Sep. 2010.
- [7] A. Tkaczyk, A. Bownik, J. Dudka, K. Kowal, and B. Ślaska. *Daphnia magna* model in the toxicity assessment of pharmaceuticals: A review. *Science of The Total Environment*, vol. 763, Apr. 2021, Art no. 143038.
- [8] A. Astashkina, B. Mann, and D. W. Grainger. A critical evaluation of in vitro cell culture models for high-throughput drug screening and toxicity. *Pharmacology & Therapeutics*, vol. 134, no. 1, pp. 82–106, Apr. 2012.
- [9] S. Yang, Z. Chen, Y. Cheng, T. Liu, L. Yin, Y. Pu, and G. Liang. Environmental toxicology wars: Organ-on-a-chip for assessing the toxicity of environmental pollutants. *Environmental Pollution*, vol. 268, no. B, Jan. 2021, Art no. 115861.
- [10] A. H. Vo, T. R. Van Vleet, R. R. Gupta, M. J. Liguore, and M. S. Rao. An Overview of Machine Learning and Big Data from Drug Toxicity Evaluation. *Chemical Research in Toxicology*, vol. 33, no. 1, pp. 20–37, Oct. 2019.
- [11] I. Goodfellow, Y. Bengio, and A. Courville. "Deep Learning". Cambridge, MA: MIT Press, 2017.
- [12] M. Corradi, T. Luechtefeld, A. M. de Haan, R. Pieters, J. H. Freedman, T. Vanhaecke, M. Vinken, and M. Teunis. The application of natural language processing for the extraction of mechanistic information in toxicology. *Frontiers in Toxicology*, vol. 6, May 2024, Art no. 1393662.
- [13] A. Sonnenburg, B. van der Lugt, J. Rehn, P. Wittkowski, K. Bech, F. Padberg, D. Eleftheriadou, T. Dobrikov, H. Bouwmeester, C. Mereu, F. Graf, C. Kneuer, N. I. Kramer, and T. Blümmel. Artificial intelligence-based data extraction for next generation risk assessment: Is fine-tuning of a large language model worth the effort? *Toxicology*, vol. 508, Nov. 2024, Art no. 153933.

- [14] A. R. Kattamreddy, and H. Chinnam. The future of large language models in toxicological risk assessment: Opportunities and challenges. *Public Health and Toxicology*, vol. 5, pp. 1–3, Jan. 2025.
- [15] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, Mar. 2023.
- [16] OECD. "Test No. 236: Fish Embryo Acute Toxicity (FET) Test," in "OECD Guidelines for the Testing of Chemicals, Section 2," OECD Publishing, Paris, 2025.
- [17] OECD. "Test No. 203: Fish, Acute Toxicity Test," in "OECD Guidelines for the Testing of Chemicals, Section 2," OECD Publishing, Paris, 2025.
- [18] J. Legradi, N. el Abdellaoui, M. van Pomeran, and J. Legler. Comparability of behavioural assays using zebrafish larvae to assess neurotoxicity. *Environmental Science and Pollution Research*, vol. 22, pp. 16277–16289, 2015.
- [19] B. Wang, Q. Sun, Y. Liu, J. Zhang, G. Li, S. Wu, H. Zheng, J. Ye, M. Zhou, H. Zheng, Y. Yu, Y. Zhong, Y. Wu, D. Huang, B. Wang, and Z. Weng. Intelligent larval zebrafish phenotype recognition via attention mechanism for high-throughput screening. *Computers in Biology and Medicine*, vol. 188, Apr. 2025, Art no. 109892.
- [20] S. Shang, S. Lin, and F. Cong. Zebrafish Larvae Phenotype Classification from Bright-field Microscopic Images Using a Two-Tier Deep-Learning Pipeline. *Applied Sciences*, vol. 10, no. 4, Feb. 2020, Art no. 1247.
- [21] A. Tandon, B. E. Howard, A. J. Green, R. Elmore, R. Shah, A. Merrick, K. Shockley, K. Ryan, and J.-H. Hsieh. Artificial intelligence (AI)-driven morphological assessment of zebrafish larvae for developmental toxicity chemical screening. *Aquatic Toxicology*, vol. 285, Aug. 2025, Art no. 107415.
- [22] D. Čapek, M. Safroshkin, H. Morales-Navarrete, N. Toulany, G. Arutyunov, A. Kurzbach, J. Bihler, J. Hagauer, S. Kick, F. Jones, B.

- Jordan, and P. Müller. EmbryoNet: using deep learning to link embryonic phenotypes to signaling pathways. *Nature Methods*, vol. 20, no. 6, pp. 815–823, May 2023.
- [23] R. Alshut, J. Legradi, U. Liebel, L. Yang, J. van Wezel, U. Strähle, R. Mikut, and M. Reischl. "Methods for Automated High-Throughput Toxicity Testing Using Zebrafish Embryos," in *KI 2010: Advances in Artificial Intelligence*, 2010, pp. 219–226.
 - [24] P. Karatzas, G. Melagraki, L.-J. A. Ellis, I. Lynch, D.-D. Varsou, A. Afantis, A. Tsoumanis, P. Doganis, and H. Sarimveis. Development of Deep Learning Models for Predicting the Effects of Exposure to Engineered Nanomaterials on *Daphnia magna*. *Small*, vol. 16, no. 36, Jun. 2020, Art no. 202001080.
 - [25] S. Inagaki, Y. Kondo, P. Religia, N. Adhitama, Y. Kato, E. Watanabe, and H. Watanabe. Application of deep learning for evaluation of the growth rate of *Daphnia magna*. *Journal of Bioscience and Bioengineering*, vol. 139, no. 5, pp. 384–391, May 2025.
 - [26] C. Xie. "Development of an Artificial Intelligence-Based Video Monitoring System to Investigate Drug Effects on *Daphnia Magna*," in *2024 IEEE 3rd International Conference on Computing and Machine Intelligence (ICMI)*, Apr. 2024, pp. 1–4.
 - [27] S. Cheng, S. Yuan, X. Wu, T. Lei, J. Ji, Y. Yin, Y. Liu, C. Liu, Y. Zhang, and Y. Zhu. Identification of Chemicals Based on Locomotor Tracks of *Daphnia magna* Using Deep Learning. *Environmental Science & Technology Letters*, vol. 10, no. 11, pp. 998–1003, Mar. 2023.
 - [28] D. Jimenez-Carretero, V. Abrishami, L. Fernández-de-Manuel, I. Palacios, A. Quílez-Álvarez, A. Díez-Sánchez, M. A. del Pozo, and M. C. Montoya. Tox_(R)CNN: Deep learning-based nuclei profiling tool for drug toxicity screening. *PLoS Computational Biology*, vol. 14, no. 11, Nov. 2018, Art no. e1006238.
 - [29] Y. Zhang, Y. S. Wong, J. Deng, C. Anton, S. Gabos, W. Zhang, D. Y. Huang, and C. Jin. Machine learning algorithms for mode-of-action

classification in toxicity assessment. *BioData Mining*, vol. 9, no. 19, May 2016.

- [30] T. Scherr, K. Löffler, M. Böhland, and R. Mikut. Cell segmentation and tracking using CNN-based distance predictions and a graph-based matching strategy. *PLoS One*, vol. 15, no. 12, Dec. 2020, Art no. e0243219.
- [31] K. Löffler, and R. Mikut. EmbedTrack–Simultaneous Cell Segmentation and Tracking Through Learning Offsets and Clustering Bandwidths. *IEEE Access*, vol. 10, pp. 77147–77157, Jul. 2022.
- [32] F. Hu, S. F. Santagostino, D. M. Danilenko, M. Tseng, J. Brumm, P. Zehnder, and K. C. Wu. Assessment of Skin Toxicity in an *in Vitro* Reconstituted Human Epidermis Model Using Deep Learning. *The American Journal of Pathology*, vol. 192, no. 4, pp. 687–700, Apr. 2022.
- [33] N. Toulany, H. Morales-Navarrete, D. Čapek, J. Grathwohl, M. Ünalán, and P. Müller. Uncovering developmental time and tempo using deep learning. *Nature Methods*, vol. 20, no. 12, pp. 2000–2010, Nov. 2023.
- [34] L. Gendele, J. Taylor, D. Myers-Turnbull, S. Chen, M. N. McCarroll, M. R. Arkin, D. Kokel, and M. J. Keiser. Deep phenotypic profiling of neuroactive drugs in larval zebrafish. *Nature Communications*, vol. 15, Nov. 2024, Art no. 9955.
- [35] T. Uelwer, J. Robine, S. S. Wagner, M. Höftmann, E. Upschulte, S. Konietzny, M. Behrendt, and S. Harmeling. A survey on self-supervised methods for visual representation learning. *Machine Learning*, vol. 113, Mar. 2025, Art no. 111.
- [36] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, and H. Luo. A Survey on Self-Supervised Learning: Algorithms, Applications, and Future Trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 9052–9071, Jun. 2024.
- [37] M. Noroozi, and P. Favaro. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles," in *Computer Vision - ECCV 2016*, Sep. 2016, pp. 69–84.

- [38] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations," in *Proceedings of the 37th International Conference on Machine Learning (PMLR)*, 2020, pp. 1597–1607.
- [39] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. "Masked Autoencoders Are Scalable Vision Learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 16000–16009.
- [40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. "Momentum Contrast for Unsupervised Visual Representation Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.
- [41] X. Chen, H. Fan, R. Girshick, and K. He. Improved Baselines with Momentum Contrastive Learning. *arXiv preprint*, Mar. 2020, Art No. arXiv:2003.04297.
- [42] X. Chen, S. Xie, and K. He. "An Empirical Study of Training Self-Supervised Vision Transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9640–9649.
- [43] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. "Bootstrap Your Own Latent A New Approach to Self-Supervised Learning," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020, pp. 21271–21284.
- [44] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. "Unsupervised Learning of Visual Features by Contrasting Cluster Assignments," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020, pp. 9912–9924.
- [45] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. "Emerging Properties in Self-Supervised Vision Transformers,"

in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9650–9660.

- [46] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *arXiv preprint*, Apr. 2023, Art no. arXiv:2304.07193.
- [47] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski. DINOv3. *arXiv preprint*, Aug. 2025, Art no. arXiv:2508.10104.
- [48] X. Chen, and K. He. "Exploring Simple Siamese Representation Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 15750–15758.
- [49] A. Yamachui Sitcheu, N. Friederich, S. Baeuerle, O. Neumann, M. Reischl, and R. Mikut. "MLOps for Scarce Image Data: A Use Case in Microscopic Image Analysis," in *Proceedings 33. Workshop Computational Intelligence*, Nov. 2023, pp. 169–189.
- [50] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778.
- [51] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint*, Apr. 2014, Art no. arXiv:1404.5997.
- [52] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint*, Feb. 2018, Art no. arXiv:1802.03426.

- [53] MMPretrain Contributors. *OpenMMLab's Pre-training Toolbox and Benchmark*. (2023). <https://github.com/open-mmlab/mmpretrain>.
- [54] T. Wang, and P. Isola. "Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere," in *Proceedings of the 37th International Conference on Machine learning (PMLR)*, 2020, pp. 9929–9939.
- [55] "Toxicology-testing platform integrating immunocompetent in vitro/ex vivo modules with real-time sensing and machine learning based in silico models for life cycle assessment and SSbD (TOXBOX)," EU Funding & Tenders Portal, [Online]. Available: <https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/how-to-participate/org-details/890666223/project/101138387/program/43108390/details>. [Accessed: Sep. 2, 2025].
- [56] "TOXBOX: All-in-one toxicology platform for a comprehensive testing of chemicals," TOXBOX Project, [Online]. Available: https://www.youtube.com/watch?v=GXigQY_nZYU. [Accessed: Sep. 7, 2025].
- [57] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, and R. Ranjan. Explainable AI (XAI): Core Ideas, Techniques, and Solutions. *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–33, Jan. 2023.
- [58] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. "Learning under Concept Drift: A Review," in *IEEE Transactions on Knowledge and Data Engineering*, Oct. 2018, pp. 2346–2363.

Multispektrale Optische Tomografie trifft Machine Learning: Ein neuer Ansatz zur Defektvorhersage im PBF-LB/M-Prozess

Tina Walter, Simon Oster, Simon Altenburg

8.3 Thermografische Verfahren, Bundesanstalt für Materialforschung und
-prüfung

Unter den Eichen 87, 12205 Berlin

E-Mail: tina.walter@bam.de

1 Einführung

Das pulverbettbasierte Laserstrahlschmelzen von Metallen (PBF-LB/M) zählt zu den am weitesten verbreiteten additiven Fertigungsverfahren für metallische Bauteile. Trotz zunehmender industrieller Relevanz, etwa in Luft- und Raumfahrt, Medizintechnik und Energieanwendungen, bleibt eine umfassende Prozessüberwachung und Qualitätssicherung eine zentrale Herausforderung. Die hohe Komplexität des Verfahrens, bedingt durch eine Vielzahl interagierender Prozessparameter, sowie die resultierende Datenfülle erschweren die direkte Korrelation zwischen Prozessanomalien und resultierenden Bauteildefekten wie Poren oder Rissen.

Zahlreiche Monitoring Ansätze, insbesondere thermografische Verfahren, wurden bereits intensiv untersucht. Dennoch fehlt es bislang an praxistauglichen, wirtschaftlich skalierbaren Lösungen. Insbesondere die Kosten und begrenzte Sichtfelder vieler Systeme hemmen eine breite industrielle Anwendung.

Oster et al. [1] konnten zeigen, dass die Analyse kurzweiliger Infrarotstrahlung (SWIR-Thermografie) mittels auf künstlicher Intelligenz (KI) basierender Auswertung zur orts aufgelösten Detektion von Porosität im PBF-LB/M-Prozess

geeignet ist. Der Ansatz erlaubt erstmals die Identifikation nicht gezielt induzierter Poren, zeigt jedoch Einschränkungen hinsichtlich des erfassten Sichtfelds (ca. $9 \times 10 \text{ mm}^2$) und der Systemkosten (ca. 30k€).

Eine vielversprechende Alternative stellt die Multikanal Optische Tomografie (MK-OT) dar [2], die auf kostengünstigen visuellen Kameras basiert. Durch gleichzeitige Erfassung mehrerer Spektralbereiche lassen sich Schichtbilder mit hoher räumlicher Auflösung und verbesserter Robustheit, verglichen mit marktüblicher monospektralen OT, gegenüber prozessbedingten Schwankungen erzeugen [2]. Die auch hierbei anfallenden großen Datenmengen erfordern eine automatisierte Auswertung. Bereits Ero et al. [3] und Feng et al. [4] konnten zeigen, dass mittels KI Porositätsvorhersagen aus herkömmlichen monochromatischen OT Daten gewonnen werden können.

Ziel dieses Beitrags ist es, das Vorgehen zur Untersuchung der Übertragbarkeit des in [1] vorgestellten KI-basierten Auswerteverfahrens auf das MK-OT-System vorzustellen. Hierdurch könnte eine kosteneffiziente und skalierbare Prozessüberwachung mittels Porositätsbestimmung für den industriellen Einsatz ermöglicht werden. Der aktuelle Stand der Arbeit und das geplante Vorgehen wird dargelegt.

2 Vorgehen und Methoden

Im Folgenden werden die MK-OT-Messansätze und der verwendete Versuchsaufbau kurz beschrieben. Anschließend werden die Datenvorverarbeitung sowie die Extraktion relevanter Merkmale und das Labeling dargelegt.

2.1 Messansätze und Aufbau der MK-OT

Die MK-OT wird, wie in Abbildung 1a) und 1b) dargestellt, in zwei Varianten realisiert: der Zweikanal-OT (2K-OT) und der Dreifarb-OT (RGB-OT). Beide Ansätze sind in [2] ausführlich beschrieben und werden nachfolgend lediglich in ihren Grundzügen dargestellt.

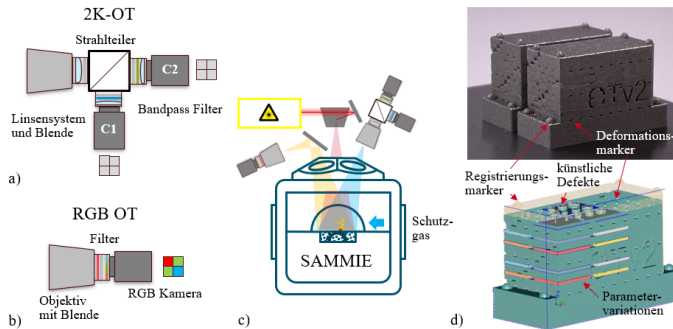


Bild 1: Schematischer Aufbau der a) 2K-OT, b) RGB-OT und des c) experimentellen Aufbaus an „SAMMIE“, d) betrachtete Probekörper mit Registriermarkern und Parametervariationen

Bei der 2K-OT erfolgt die Prozessbeobachtung mithilfe zweier industrieller Kameras im sichtbaren Spektralbereich (VIS), die über einen Strahlteiler gekoppelt sind. Beide Kameras sind mit auf die Messaufgabe abgestimmten, schmalbandigen Interferenzfiltern (um 700 nm und 905 nm) ausgestattet. Die RGB-OT nutzt eine industrielle RGB-Kamera, deren drei Farbkanäle (Rot, Grün und Blau) separat ausgewertet werden. In beiden Setups sind zusätzliche optische Filter erforderlich, um die Systeme an die Messaufgabe anzupassen.

Zur experimentellen Testung der Ansätze wurden beide Systeme an der Testanlage „SAMMIE“ wie in Abbildung 1c) schematisch dargestellt off-axis installiert und ein PBF-LB/M Baujob simultan aufgezeichnet. Hierbei wurden zwei identische Probekörper (10 mm x 20 mm x 10,4 mm) aus 316L Pulver aufgebaut. Wie auch in [1] angewendet, soll ein Probekörper dem Training und Validierung der Modelle, der andere zum Test dienen. Es wurde ein für die Maschine und eine Schichtdicke $t = 40 \mu\text{m}$ optimierten Parametersatz verwendet. Als Rotationswinkel wurde $\alpha = 90^\circ$ zwischen den Einzelbauteilschichten gewählt.

Zur späteren Registrierung der Datensätze sind die Probekörper mit halbkugelförmigen Registrierungs-markern versehen. Für eine detaillierte Registrierung der Bauteilhöhe sind zusätzlich interne Hohlräume und am Rand Stufen-Höhenmarker eingebracht. Zusätzlich wurden Bereiche mit nicht optimierten

Prozessparametern und größere Hohlräume eingebracht, um künstliche Fehlstellen zu forcieren bzw. simulieren. Die beschriebenen Marker und Bereiche sind in Abbildung 1d) dargestellt.

Nach dem Aufbauprozess wurden die Probekörper von der Bauplatzform getrennt und mittels hochauflösender Computerröntgentomographie (XCT) gescannt. Als Ground Truth wurde mit einer XCT-Anlage (Diondo sxMAX dual tube) unter Verwendung einer Beschleunigungsspannung von $U_b = 240\text{kV}$ ein XCT-Datensatz mit einer Voxelgröße von $11,9 \times 12,6 \times 12,0 \mu\text{m}^3$ erstellt. Diese Auflösung entspricht in etwa der räumlichen Auflösung der MK-OT Daten mit ca. $10 \times 10 \mu\text{m}^3$ für die einzelnen Schichten. Da die Monitoring Daten nur schichtweise vorliegen, weisen diese allerdings eine Auflösung von ca. $z_t = 40 \mu\text{m}$ in z-Richtung auf.

2.2 Datenvorverarbeitung - Datenfusion und räumliche Zuordnung

Der aktuell durchgeführte Arbeitsschritt der Datenvorverarbeitung zur präzisen Fusion der Kamera- und XCT-Daten wird an das Vorgehen nach Oster et al. [1], [5] angelehnt. In Abbildung 2 sind die einzelnen Schritte und der aktuelle Arbeitsstand dargestellt.

Zunächst erfolgte eine Bildkorrektur der Kameradaten, bestehend aus Bias-, Dark- und Flatfieldkorrektur [6]. Durch die bei der OT angewendeten Langzeitbelichtung ist für die räumliche Zuordnung lediglich eine Perspektiv- und Verzeichnungskorrektur mittels eines Kalibrierbildes notwendig. Eine zeitliche Zuordnung der Messdaten ist Prinzipbedingt nicht möglich.

Anschließend an die Korrektur wurden die Einzelkanäle gematcht: Bei der RGB-OT durch Aufsplittung in die drei Farbkanäle (R, G, B) und einfaches Übereinanderlegen. Hierbei wird vereinfachend angenommen, dass eine Bayer-Matrix-Einheit (RGBG) einen Raumpunkt erfasst. Bei der 2C-OT wurden die beiden separaten Kameraaufnahmen unter Verwendung des Messsignals der Halbkugelmarker registriert.

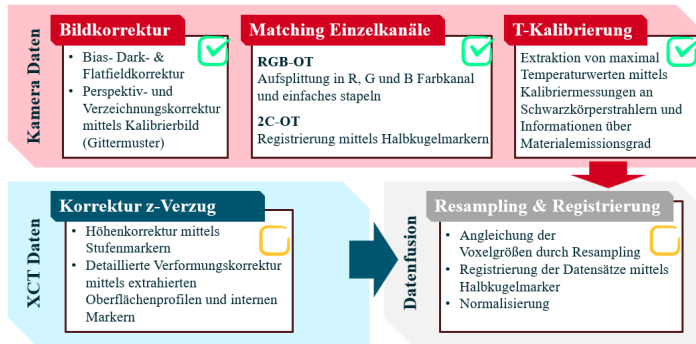


Bild 2: Schritte der Datenvorverarbeitung und aktueller Arbeitsstand

Ein Vorteil der MK-OT gegenüber der herkömmlichen OT ist, dass durch das multispektrale Messverfahren mittels Planckfitting näherungsweise eine tatsächliche physikalische Information, die maximale Oberflächentemperaturen T_{max} , zusätzlich zu einem einfachen Grauwertbild erfasst werden kann [2], [7]. Damit aus den Grauwertbildern näherungsweise maximale Oberflächentemperaturen T_{max} berechnet werden können, wurde eine Temperaturkalibrierung der Kamerasysteme mittels Messungen an Schwarzkörperstrahlern durchgeführt. Hierfür wurden auch die Materialemissionsgrade [8] berücksichtigt. In Abbildung 3 sind für eine Bauteilschicht beispielhaft Roh-, T_{max} - und XCT-Daten des RGB-Setups dargestellt.

Monospektrale OT-Daten weisen starke Schwankungen zwischen den Lagen auf, was deren Auswertung erheblich erschwert. Die Schwankungen entstehen durch die unterschiedlichen Scanwinkel α und dadurch unterschiedliche Abschattung durch die über dem Schmelzbad entstehenden Beiprodukte wie Rauch und Spritzer. Da die Abschattung die verschiedenen Farbkanäle sehr ähnlich beeinflusst wird dieser Effekt durch die Berechnung von T_{max} bei der MK-OT deutlich reduziert. In der klassischen OT können derartige Fluktuationen lediglich durch Verfahren wie eine Moving-Average-Glättung über mehrere Schichten [4] teilweise kompensiert werden.

Die gedruckten Bauteile weisen prozessbedingt im Vergleich zur geplanten Geometrie Abweichungen auf. Diese werden im XCT-Datensatz für die Reg-

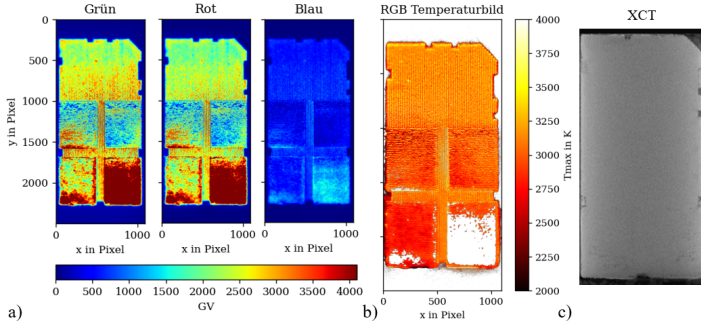


Bild 3: Beispielschichtdaten der RGB-OT, a) Rohdaten der drei Farbkanaäle b) Berechnetes $T_{\max}$ Schichtbild und c) XCT-Bild einer Bauteilschicht mit unterschiedlich gefertigten Bereichen

sitrierung korrigiert. Unter Verwendung der Stufenmarker wird aktuell eine Höhenanpassung durchgeführt. Eine detaillierte Verformungskorrektur erfolgt anhand der extrahierten Oberflächenprofile und der internen Marker. Das Vorgehen ist angelehnt an das von Oster et al. beschriebene Verfahren [5].

Abschließend werden beide korrigierten Datensätze durch Resampling und Registrierung zusammengeführt. Dabei erfolgt die Angleichung der Voxelgrößen sowie die räumliche Zuordnung beider Datensätze unter Zuhilfenahme der Halbkugelmarker. Als letzter Vorverarbeitungsschritt vor der Feature Extraktion werden alle Daten normalisiert.

2.3 Betrachtete Merkmale und Cluster

Obwohl nur ein Schichtbild pro Aufnahmekanal vorliegt, führt die hohe räumliche Auslösung der Bilddaten trotzdem zu nicht unerheblichen Datenmengen. Da auch die Auflösung der XCT-Daten und die Registrierungs Genauigkeit begrenzt ist, ist eine pixelgenaue Betrachtung nicht sinnvoll. Die Datensätze werden deshalb in Blöcke F_i mit Länge und Breite w und einer Höhe gleich der verwendeten Schichtdick z_i aufgeteilt, für welche Merkmale berechnet werden. Die Block-Dimension w stellt einen zu optimierende Hyperparameter dar. Vereinfachend werden in dieser Arbeit nur Blöcke betrachtet, die sich vollständig im Bauteil befinden.

Da in dieser Arbeit zunächst die Eignung der MK-OT Daten zur KI-gestützten Porositätsvorhersage untersucht werden soll, werden nur eher einfache statistische Verfahren zur Merkmalsextraktion verwendet. Feng et al. [4] konnte z.B. zeigen, dass mittlere und maximale monochromatische OT-Grauwerte schon als Merkmale für sein Random Forest Modell gute Ergebnisse erzielen konnten.

Für die Blöcke werden die Grauwerte der Einzelkanäle und die T_{max} -Werte beider Temperaturbilder mittels folgender statistischer Werte ausgewertet und diese dann als Merkmale verwendet: Lagemaße: Mittelwert, Median und Modus, Streumaße: Standardabweichung und Spannweite, Extremwerte: Minimum und Maximum, Formmaße: – Schiefe und Wölbung. Um Muster und Strukturen in den Daten quantitativ zu erfassen, werden auch die Haralick Features [9] für die Blöcke berechnet.

Da bekannt ist, dass insbesondere Keyholeporen im unteren Teil des mehreren Lagen tiefen Schmelzbades entstehen und somit deutlich unterhalb der aktuell gefertigten Schicht auftreten [10] [4], werden die Merkmale mehrerer übereinanderliegender Blöcke dem Model als Input übergeben. Simon et al. spricht hier von der Relevanz von drei oberhalb liegenden und bis zu zwölf unterhalb liegenden Schichten [1]. Feng et al. bezieht zehn Schichten unterhalb der aktuell betrachteten Schicht mit ein [4].

Im Gegensatz zur zeitlich aufgelösten und das Schmelzbad direkt abbildenden SWIR-Thermografie ist der Informationsgehalt der MK-OT-Daten deutlich reduziert. Merkmale wie der zeitliche Temperaturverlauf eines Ortspunktes und die meisten geometrischen Merkmale des Schmelzbades sind aus dem MK-OT Daten nicht bestimmbar. Als einziger geometrischer Wert ließe sich die Schmelzbahnbreite anhand der hochauflösenden OT-Daten abschätzen. Da die korrespondierende Schmelzbadbreite *MPW* laut Oster et al. [1] nur sehr geringe Aussagekraft besitzt, wird dieses Merkmal im Rahmen dieser Arbeit nicht betrachtet.

Auch Informationen über im Prozess auftretende Spritzerpartikel könnten den OT-Daten entnommen werden [12]. Da auch diese laut Oster et al. nur eine geringe Relevanz bezüglich der Vorhersagen von Porosität aufweisen [1], werden sie im Rahmen dieser Arbeit nicht betrachtet.

3 Ai gestützte Auswertung

Im Rahmen der Untersuchung soll zunächst ein Random-Forest-Modell (RF) für den Anwendungsfall getestet werden, da sowohl in [1] als auch in [4] damit bereits gute Ergebnisse erzielt wurden. Ein simplerer Ansatz in Form einer linearen Regression scheint nicht sinnvoll, da wahrscheinlich keine linearen Zusammenhänge im komplexen betrachteten PBF-LB/M Prozess vorliegen [1].

RF weist eine hohe Generalisierungsfähigkeit sowie ein geringes Risiko für Overfitting auf. Darüber hinaus ermöglicht es eine Bewertung des Einflusses der einzelnen Merkmale. Im Rahmen der Arbeit soll sowohl ein Regressions- als auch ein Klassifikationsmodell erstellt werden. Für die Klassifizierung wird der Schwellwert für poröse Blöcke des XCT-Datensatzes hierbei zunächst mit $\rho = 0,1$ % Porosität pro betrachteten Block definiert. Der Schwellwert wird zugleich als Hyperparameter betrachtet, der anhand der Validierungsergebnisse zur Optimierung der Modellgüte angepasst werden kann.

Als zentrale Hyperparameter des RF werden die Anzahl der Bäume, die maximale Tiefe sowie das Splitting-Kriterium (Klassifikation: Gini/Entropie; Regression: MSE/MAE) betrachtet. Die Optimierung erfolgt aufgrund der großen Datenmenge mittels Bayesian Optimization.

Da zwei identische Probekörper vorliegen, werden von Probekörper eins 80 % der Daten für das Training und 20 % für die Validierung verwendet. Probekörper zwei dient vollständig als unabhängiger Testdatensatz [1]. Ein Tausch der beider Datensätze ist zusätzlich möglich.

Zur Bewertung der Klassifikation werden Accuracy, Precision, Recall und F1-Score berechnet. Aufgrund der unausgeglichene Datenbasis (mehr porenfreie als poröse Bauteilbereiche) und da das Übersehen von Poren gravierender ist als falsch-positive Vorhersagen, sind Recall und F1-Score besonders relevant. Für die Regression werden R^2 , RMSE und MAE genutzt.

Die Ergebnisse werden mit der Literatur, z.B. [1] [3] [4] [13], verglichen. Es wird die Python-Bibliothek scikit-learn verwendet.

4 Ziel und Ausblick

Primäres Ziel dieser Arbeit ist die Ermittlung geeigneter Hyperparameter, Features und einer passenden Modellform, um die Eignung der MK-OT-Daten zur Porositätsvorhersage und deren Potenzial als Werkzeug zur Qualitätskontrolle im metallischen AM zu bewerten.

Zudem erfolgt ein Vergleich der beiden MK-OT-Varianten sowie eine Gegenüberstellung mit der industriell eingesetzten monochromatischen OT (simuliert über den 905-nm-Kanal der 2K-OT), um mögliche Vorteile hinsichtlich der Vorhersagegüte zu identifizieren. Ein weiteres Ziel ist die Untersuchung, ob eine aufwendige, in der Praxis kaum umsetzbare Oberflächenverformungskorrektur bei einfachen Geometrien tatsächlich erforderlich ist.

Für zukünftige Arbeiten erscheinen neben alternativen Modellansätzen wie k-Nearest Neighbor (kNN) oder Gradient Boosting Trees (GBT) auch komplexere Methoden vielversprechend. Hierzu zählen neuronale Netzwerke (z. B. CNNs, U-Nets, SOM-basierte Clusterverfahren) oder Rohdatenmodelle wie IMSEQR. Darüber hinaus könnten erweiterte Merkmale, etwa die Analyse der Spatterbildung, zusätzliche Erkenntnisse liefern.

Literatur

- [1] S. Oster, N. Scheuschner, K. Chand, S. J. Altenburg. *Additive Manufacturing* Volume 95, <https://doi.org/10.1016/j.addma.2024.104502>. 2024.
- [2] T. Becker, P.P. Breese, C. Metz, S.J. Altenburg 2023 *IOP Conf. Ser.: Mater. Sci. Eng.* 1296 012008. <https://dx.doi.org/10.1088/1757-899X/1296/1/012008>. 2023.
- [3] O. Ero, K. Taherkhani and E. Toyserkani *Additive Manufacturing, Volume 78* <https://doi.org/10.1016/j.addma.2023.103894> 2023
- [4] S. Feng, Z. Chen, B. Bircher, Z. Ji, L. Nyborg, S. Bigot *Materials & Design, Volume 222* <https://doi.org/10.1016/j.matdes.2022.111115>. 2022

- [5] S. Oster, T. Fritsch, A. Ulbricht, G. Mohr, G. Bruno, C. Maierhofer, S.J. Altenburg *Metals* 2022, 12, 947. <https://doi.org/10.3390/met12060947> 2022
- [6] H. Tomsik „Kalibrierung mittels Bias-, Dark- und Flatframe“ 2011
- [7] T. Becker, S.J. Altenburg, N. Scheuschner, P.P. Breese, C. Metz, K. Hilgenberg, C. Maierhofer *Procedia CIRP, Volume 111* <https://doi.org/10.1016/j.procir.2022.08.035> 2022
- [8] T. Becker, T. Stark, M. Arduini, J. Manara, S.J. Altenburg *Prog Addit Manuf* 9, 695–704 <https://doi.org/10.1007/s40964-024-00665-2> 2024
- [9] R. M. Haralick, K. Shanmugam and I. Dinstein *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610-621 doi: 10.1109/TSMC.1973.4309314. 1973
- [10] S.M.H. Hojjatzadeh, N.D. Parab, Q. Guo, M. Qu, L. Xiong, C. Zhao, L.I. Escano, K. Fezzaa, W. Everhart, T. Sun, L. Chen *International Journal of Machine Tools & Manufacture* <https://doi.org/10.1016/j.ijmachtools.2020.103555> 2020
- [11] M. Grasso und B.M. Colosimo *Meas. Sci. Technol.* 28 044005 <https://dx.doi.org/10.1088/1361-6501/aa5c4f> 2017
- [12] Z. Chen, A. Raza, E. Hryha *Journal of Manufacturing Processes, Volume 81* <https://doi.org/10.1016/j.jmapro.2022.07.031>. 2022
- [13] S. Oster, P.P. Breese, A. Ulbricht, G. Mohr, S.J. Altenburg *J Intell Manuf* 35, 1687–1706 <https://doi.org/10.1007/s10845-023-02117-0> 24

Modellgestützte Identifikation von Prozessparametern für die Additive Fertigung von Metallen

Thomas Niendorf¹, Anna Engelhardt¹, Matthias Kahl², Julia Richter¹, Philipp Krooß¹, Thomas Wegener¹, Andreas Kroll²

¹Institut für Werkstofftechnik, Metallische Werkstoffe, Universität Kassel
Mönchebergstraße 3, 34125 Kassel, Deutschland
E-Mail: niendorf@uni-kassel.de

²Institut für Analyse und Regelung technischer Systeme, Mess- und Regelungstechnik, Universität Kassel
Mönchebergstraße 7, 34125 Kassel, Deutschland
E-Mail: andreas.kroll@mrt.uni-kassel.de

Motivation

Mit Blick auf aktuelle Herausforderungen wie den Klimawandel gewinnt die Herstellung von Leichtbau-Komponenten zunehmend an Bedeutung. Allerdings können die teils äußerst komplexen Geometrien solcher Komponenten meist nicht mit herkömmlichen Fertigungsverfahren hergestellt werden, weshalb Verfahren der Additiven Fertigung von Metallen zunehmend in den Fokus der Industrie rücken [1]. Ein häufig verwendetes Verfahren der Additiven Fertigung ist das pulverbettbasierte Laserschmelzen (PBF-LB/M), auch genannt selektives Laserschmelzen, bei dem Bauteile mit Hilfe eines Lasers schichtweise auf Basis von Metallpulver gefertigt werden [2]. Hierdurch wird die Herstellung von Bauteilen mit komplexen Geometrien ermöglicht. Allerdings ist das Erreichen einer hohen Bauteilqualität nach wie vor eine große Herausforderung im PBF-LB/M [3, 4], da viele Effekte zur Bildung von Defekten führen können, die die Bauteileigenschaften negativ beeinflussen können [5]. Daher stellt die Suche nach Prozessparametern, die zu möglichst defektfreien Bauteilen führen, i. d.

R. den ersten Schritt bei der Implementierung des PBF-LB/M dar [6]. Bei der Suche nach geeigneten Prozessparametern werden oftmals nur die drei zentralen Parameter Laserleistung, Scangeschwindigkeit und Hatch betrachtet, wobei für jeden der drei Parameter große Wertebereiche untersucht werden müssen. Eine Untersuchung dieser großen Wertebereiche ausschließlich auf Basis von Experimenten kann schwierig und zeitintensiv sein, besonders wenn die entsprechenden Prozessfenster, z. B. aufgrund der Verwendung einer hohen Schichtdicke, sehr klein sind. Ansätze, die experimentelle Daten und datengetriebene Modellierung verbinden, sind daher ein vielversprechender Weg um PBF-LB/M Prozessparameter, die zu möglichst defektfreien Bauteilen führen, zu identifizieren.

1 Experimentelle Daten

Zur Identifikation von Prozessparametern für die Herstellung von Bauteilen aus der Aluminiumlegierung AlSi10Mg im PBF-LB/M unter Verwendung hoher Schichtdicken s , wurde ein entsprechender Ansatz, der experimentelle Daten und datengetriebene Modellierung verbindet, verwendet (siehe [7]). Dazu wurden für $s = 60\mu\text{m}$ und $s = 90\mu\text{m}$ jeweils 60 Proben mit unterschiedlichen Parameterkombinationen hergestellt (Details zur Herstellung siehe [7]). Die dabei verwendeten Parameterkombinationen basieren auf einem raumfüllenden Versuchsplan (latin hypercube design), bei dem die Laserleistung P , die Scangeschwindigkeit v_s und der Hatch h als Faktoren berücksichtigt wurden. Die jeweiligen Wertebereiche der Prozessparameter sind in Tabelle 1 angegeben. Der Versuchsplan wurde zusätzlich um weitere Parameterkombinationen, u. a. Wiederholungen, ergänzt. Für jede Probe wurde auf Basis einer Bildanalyse von Schlibfbildern die Porosität, d. h. der prozentuale Anteil von Defekten an der betrachteten Fläche des Schlibfbilds, bestimmt (Daten siehe [8]).

2 Datengetriebene Modellierung

Auf Basis der experimentellen Daten wurde für jede Schichtdicke mit Hilfe der multiplen linearen Regression ein Modell berechnet. Dabei wurde eine

Tabelle 1: Wertebereiche der Prozessparameter, die zur Erstellung der Versuchspläne verwendet wurden. Nach [7].

s in μm	P in W	v_s in mm/s	h in mm
60	[250, 400]	[370, 1570]	[0, 1, 0, 4]
90	[300, 400]	[370, 1570]	[0, 1, 0, 4]

Tabelle 2: Werte der Bewertungskriterien für die berechneten Modelle für $s = 60\mu\text{m}$ und $s = 90\mu\text{m}$. Nach [7].

Output	R^2	$RMSE$ in %	R^2_{LOOCV}	$RMSE_{LOOCV}$ in %
$y_{s=60\mu\text{m}}$	0,96	0,30	0,92	0,43
$y_{s=90\mu\text{m}}$	0,98	0,25	0,95	0,39

schrittweise Regression durchgeführt, und Haupt- und Interaktionsterme bis einschließlich 3. Ordnung wurden berücksichtigt. Als Input wurden die Prozessparameter P , v_s und h , als Output die Porosität verwendet. Zur Beurteilung der Modelle wurden der Standardfehler $RMSE$ und das Bestimmtheitsmaß R^2 berechnet sowie eine Leave-one-out Kreuzvalidierung (LOOCV) durchgeführt. [7]

Eine Verwendung aller Datenpunkte bei der Modellierung war nicht möglich, da einige Proben eine bestimmte Defektart (Streifenporen genannt) aufwiesen, die zu einer inhomogenen Verteilung der Defekte innerhalb der Probe und damit einer hohen Streuung der Porositätswerte bei den Wiederholungen führte. Da dies die Modellierungsergebnisse negativ beeinflusste, wurden Proben mit der entsprechenden Defektart aus dem Datensatz entfernt, so dass 40 Proben bei $s = 60\mu\text{m}$ und 37 Proben bei $s = 90\mu\text{m}$ berücksichtigt wurden. Die so berechneten Modelle führten für den vorhandenen Anwendungsfall zu guten Werten der verwendeten Bewertungskriterien (siehe Tabelle 2). Auf Basis von Vorhersagen der Modelle konnten daher Konturkarten erstellt und mögliche Prozessfenster identifiziert werden (siehe Bild 1). [7]

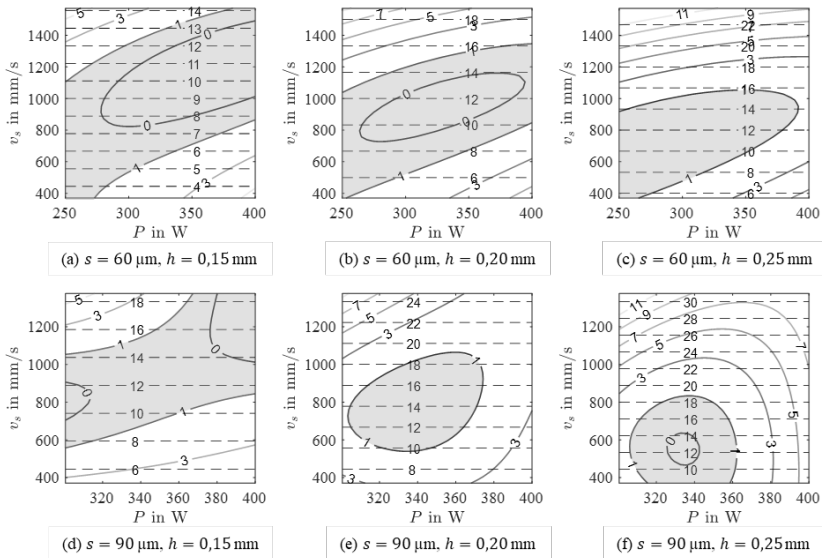


Bild 1: Konturkarten für $s = 60 \mu\text{m}$ und $s = 90 \mu\text{m}$, die auf Basis von Modellvorhersagen für verschiedene Hatchwerte erstellt wurden. Die identifizierten Prozessfenster sind hervorgehoben. Nach [7].

3 Validierung der datengetriebenen Modellierung

Um die mittels datengetriebener Modellierung identifizierten Prozessfenster zu validieren, wurden weitere Proben gefertigt und hinsichtlich ihrer Porosität untersucht. Der dazu verwendete Versuchsplan setzte sich aus mehreren Teilen zusammen, u. a. den Eckpunkten des untersuchten Wertebereichs und vollfaktoriellen Versuchsplänen in den Gebieten, in denen die identifizierten Prozessfenster lagen. Ein Vergleich der experimentellen Daten und der zugehörigen Modellvorhersagen zeigte, dass in vielen Fällen Abweichungen kleiner 1 % vorliegen. Höhere Abweichungen traten fast ausschließlich bei Proben mit Streifenporen bzw. bei Eckpunkten auf, d. h. in Bereichen, zu denen bei der Modellierung keine Informationen vorlagen. Außerdem zeigte sich, dass zwar Proben mit geringer Porosität in den identifizierten Prozessfenstern liegen, die Form und Größe der Prozessfenster jedoch nicht korrekt vorhergesagt wurde. Die Gründe

für die Abweichungen können auf Basis der Versuchsrandbedingungen erklärt werden, so dass eine weitergehende Verbesserung der Modellgüten möglich scheint. Diese soll in zukünftigen Arbeiten umgesetzt werden.

Literatur

- [1] N.T. Aboulkhair, N.M. Everitt, I.A. Ashcroft und C.J. Tuck. „Reducing porosity in AlSi10Mg parts processed by selective laser melting“. In: *Additive Manufacturing, Volumes 1–4*, S. 77-86. 2014.
- [2] N.T. Aboulkhair, M. Simonelli, L. Parry, I.A. Ashcroft, C.J. Tuck und R.J. Hague. „3D printing of Aluminium alloys: Additive Manufacturing of Aluminium alloys using selective laser melting“. In: *Progress in Materials Science, Volume 106*. 2019.
- [3] G. Tapia und A. Elwany. „A Review on Process Monitoring and Control in Metal-Based Additive Manufacturing“. In: *Journal of Manufacturing Science and Engineering, Volume 136, Issue 6*. 2014.
- [4] M. Grasso und B.M. Colosimo. „Process defects and in situ monitoring methods in metal powder bed fusion: a review“. In: *Measurement Science and Technology, Volume 28, Number 4*. 2017.
- [5] A. Sola und A. Nouri. „Microstructural porosity in additive manufacturing: The formation and detection of pores in metal parts fabricated by powder bed fusion“. In: *Journal of Advanced Manufacturing and Processing, Volume 1, Issue 3*. 2019.
- [6] S. Sheikh, B. Vela, V. Attari, X. Huang, P. Morcos, J. Hanagan, C. Acemi, I. Karaman, A. Elwany und R. Arroyave. „Exploring chemistry and additive manufacturing design spaces: a perspective on computationally-guided design of printable alloys“. In: *Materials Research Letters, Volume 12, Issue 4*. 2024.
- [7] A. Engelhardt, M. Kahl, J. Richter, P. Krooß, A. Kroll, T. Niendorf. „Investigation of processing windows in additive manufacturing of

AlSi10Mg for faster production utilizing data-driven modeling“. In: *Additive Manufacturing, Volume 55*. 2022.

- [8] A. Engelhardt, M. Kahl, J. Richter, P. Krooß, A. Kroll, T. Niendorf. „Investigation of processing windows in additive manufacturing of AlSi10Mg for faster production utilizing data-driven modeling [Dataset]“. In: *DaKS*. 2022.

Direkte datenbasierte prädiktive Regelung nichtlinearer Prozesse unter Verwendung von Takagi-Sugeno-Fuzzy-Systemen

Marcel Zehner, Alessio Cavaterra, Steven Lambeck

FB Elektrotechnik und Informationstechnik, Hochschule Fulda

Leipziger Str. 123, 36037 Fulda

E-Mail: {marcel.zehner,alessio.cavaterra,steven.lambeck}@et.hs-fulda.de

1 Einführung

Direkte datenbasierte prädiktive Regelungsverfahren, wie z. B. Data-Enabled Predictive Control (DeePC) [1] haben sich in den letzten Jahren insbesondere im akademischen Umfeld etabliert. DeePC nutzt vergangene Eingangs- und Ausgangstrajektorien eines Systems, um die optimale Stellgrößenfolge unter Minimierung einer Kostenfunktion sowie unter Berücksichtigung von Regel- und Stellgrößenbeschränkungen zu generieren. Dabei muss für diese Methode im Gegensatz zu modellbasierten prädiktiven Regelungsverfahren¹ (engl.: Model Predictive Control, MPC) [2] kein parametrisches Modell im Vorfeld durch eine Offline-Systemidentifikation bestimmt werden. Dies erleichtert die Implementierung von prädiktiven Regelungsansätzen und macht DeePC daher besonders interessant für praktische Anwendungen. DeePC basiert auf der *behavioral*-Systemtheorie [3] und dem Fundamental Lemma nach Willems [4]. In dieser theoretischen Betrachtungsweise wird ein lineares, zeitinvariantes (LTI)-System als eine Menge an vergangenen Eingangs- und Ausgangstrajektorien interpretiert. Zudem kann jede mögliche Trajektorie eines LTI-Systems aus einer endlichen Anzahl vergangener Trajektorien desselben LTI-Systems konstruiert werden. Basierend auf dieser Idee können Eingangs- und Ausgangstrajektorien

¹ Da bei MPC ein parametrisches Modell identifiziert werden muss, zählt MPC zu den indirekt datenbasierten prädiktiven Regelungsverfahren [5]

als prädiktives nichtparametrisches Modell im Optimalsteuerungsproblem Verwendung finden.

Obwohl es eine fundierte theoretische Grundlage für direkte datenbasierte prädiktive Regelungsansätze linearer Systeme gibt, erfordert die praktische Umsetzung den Einsatz von Methoden, die auch nichtlineare Prozesse berücksichtigen. In der Literatur sind einige Beispiele zu finden, die direkte datenbasierte prädiktive Regelungsmethoden auf nichtlineare Systeme erweitern. So werden in [6] Kernelmethoden genutzt, um eine Erweiterung von DeePC für nichtlineare Systeme zu schaffen. Des Weiteren wird in [7] eine Erweiterung mit Hilfe von Basisfunktion realisiert und in [8] ein direkter datenbasierter prädiktiver Regelungsansatz auf Basis des Koopman-Frameworks umgesetzt. In [9] werden die universellen Approximationseigenschaften von Neuronalen Netzen genutzt, um eine DeePC-Methodik für nichtlineare Systeme umzusetzen. Die Untersuchungen in [10] nutzen neu gemessene Ein- und Ausgangstrajektorien zur Aktualisierung der Datengrundlage, wobei lokale lineare Approximationen des zugrunde liegenden Systems ausgenutzt werden. Für einen grundlegenden Überblick über die nichtlinearen Erweiterungen direkter datenbasierter prädiktive Regelungen wird der interessierte Leser auf [11] verwiesen.

In dieser Arbeit wird DeePC mittels Takagi-Sugeno-Fuzzy (TS)-Systemen auf eine Klasse nichtlinearer Systeme erweitert. Mit TS-Systemen ist es möglich, das nichtlineare Gesamtübertragungsverhalten eines Systems mit gewichteten lokal linearen Modellen zu approximieren. Diese Idee wird auf die DeePC-Methodik übertragen, indem die Datengrundlage gemäß einer Partitionierung des Datenraums (reguläre Gitterstruktur und Fuzzy-c-Means) und in Abhängigkeit vom aktuellen Arbeitspunkt gewichtet wird. Hier zeigt sich, dass die arbeitspunktabhängige Gewichtung der Datengrundlage zu einer Verbesserung der Regelgüte im Vergleich mit der Standard-DeePC-Formulierung führt. Das Paper ist folgendermaßen strukturiert: In Abschnitt 2 werden die theoretische Grundlage zu nichtparametrischen Systemdarstellungen und DeePC kurz skizziert. Danach werden in Abschnitt 3 die Erweiterungen von DeePC aufgezeigt. Hierbei wird sowohl eine reguläre Gitterpartitionierung des Datenraums, als auch eine Partitionierung des Datenraums mit dem Fuzzy-c-Means-Algorithmus behandelt. Die nichtlinearen Erweiterungen werden in Abschnitt 4 an einem Hammerstein-System simulativ erprobt und mit dem Standard-DeePC

verglichen. Den Abschluss bildet Abschnitt 5 mit einer Diskussion und einem Ausblick.

2 Direkte datenbasierte prädiktive Regelung

2.1 Nichtparametrische Systemdarstellung

Im Gegensatz zu indirekt datenbasierten prädiktiven Regelungsverfahren verwenden direkte datenbasierte prädiktive Regelungsmethoden keine parametrischen Modellstrukturen, wie beispielsweise lineare Zustandsraummodelle oder ARX-Modelle. Direkte datenbasierte prädiktive Regelungsverfahren setzen nichtparametrische Modelle ein, welche auf der *behavioral* Systemtheorie [3] und dem *Fundamental Lemma nach Willems* [4] beruhen. Mit diesen Ideen kann für das folgende lineare zeitinvariante System (1)

$$\begin{aligned}x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k + Du_k,\end{aligned}\tag{1}$$

mit dem Zustand $x_k \in \mathbb{R}^n$, dem Eingang $u_k \in \mathbb{R}^m$ und dem Ausgang $y_k \in \mathbb{R}^p$ eine nichtparametrische Systemdarstellung basierend auf Eingangs- und Ausgangsdaten von (1) definiert werden. Die im Vorfeld gesammelten Eingangs- und Ausgangstrajektorien werden in Hankelmatrizen angeordnet, wobei die Hankelmatrizen das nichtparametrische Modell darstellen. Für eine Eingangsdatentrajektorie $u = \{u_i\}_{i=1}^{T_d}$ der Länge T_d , mit $T_d \geq L$, lässt sich eine Hankelmatrix mit L Zeilen und $T_d - L + 1$ Spalten konstruieren:

$$\mathcal{H}_L(u) := \begin{pmatrix} u_1 & u_2 & \cdots & u_{T_d-L+1} \\ u_2 & u_3 & \cdots & u_{T_d-L+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_L & u_{L+1} & \cdots & u_{T_d} \end{pmatrix}.\tag{2}$$

Bei T_d handelt es sich um eine dimensionslose Variable, die die Anzahl der Datenpunkte beschreibt. Das Eingangssignal u wird als *persistently exciting* (dt.: persistent anregend) der Ordnung L bezeichnet, wenn die Hankelmatrix $\mathcal{H}_L(u)$

vollen Zeilenrang aufweist. Der Begriff der persistenten Anregung beschreibt Eingangssignale, die das zugrundeliegende System hinreichend anregen, so dass die Ausgangsdaten ein Abbild des charakteristischen Systemverhaltens liefern. Gegeben sind Eingangs- sowie Ausgangsdaten der Form $(u_d, y_d) = \{u_{d_i}, y_{d_i}\}_{i=1}^{T_d}$ von (1), wobei der Index d auf in der Vergangenheit aufgenommene Daten verweist. Wenn das Eingangssignal $u_d = \{u_{d_i}\}_{i=1}^{T_d}$ mit der Länge T_d persistent anregend ist und die dazugehörige Hankelmatrix $\mathcal{H}_L(u_d)$ somit vollen Zeilenrang aufweist, gehören die Eingangs- und Ausgangstrajektorien $(u, y) = \{u_i, y_i\}_{i=1}^L$ zu (1) dann und nur dann, wenn ein $g \in \mathbb{R}^{T_d-L+1}$ existiert, sodass:

$$\begin{pmatrix} \mathcal{H}_L(u_d) \\ \mathcal{H}_L(y_d) \end{pmatrix} g = \begin{pmatrix} u \\ y \end{pmatrix}. \quad (3)$$

Mit diesem Ergebnis ist es möglich, jede Trajektorie eines LTI-Systems aus einer endlichen Anzahl vergangener Trajektorien desselben LTI-Systems zu konstruieren. Dabei werden die Spalten der Hankelmatrizen, welche eine Trajektorie der Länge L des Systems repräsentieren, linear miteinander kombiniert und ergeben somit eine mögliche Trajektorie des Systems. Der Vektor g kann dabei als ein Gewichtungsvektor der einzelnen Spalten in den Hankelmatrizen interpretiert werden. Diese Formulierung ermöglicht es, das Systemverhalten nur auf Basis von aufgenommenen Eingangs- und Ausgangsdaten ohne Zugriff auf ein parametrisches Modell zu prädictieren. Zu diesem Zweck und zur Verwendung in einem Optimalsteuerungsproblem werden die Hankelmatrizen (3) wie folgt in Teilmatrizen aufgeteilt:

$$\begin{pmatrix} U_p \\ U_f \end{pmatrix} := \mathcal{H}_{T_{ini}+N_p}(u_d), \quad \begin{pmatrix} Y_p \\ Y_f \end{pmatrix} := \mathcal{H}_{T_{ini}+N_p}(y_d). \quad (4)$$

Die Matrizen $U_p \in \mathbb{R}^{T_{ini} \times T_d - N_p - T_{ini} + 1}$ und $Y_p \in \mathbb{R}^{T_{ini} \times T_d - N_p - T_{ini} + 1}$ setzen sich dabei aus den ersten T_{ini} Zeilen von $\mathcal{H}_{T_{ini}+N_p}(u_d)$ bzw. $\mathcal{H}_{T_{ini}+N_p}(y_d)$ zusammen. U_f und Y_f bestehen sinngemäß aus den darauffolgenden N_p Zeilen der Hankelmatrix $\mathcal{H}_{T_{ini}+N_p}(u_d)$ bzw. $\mathcal{H}_{T_{ini}+N_p}(y_d)$. Damit gilt ebenso $L = T_{ini} + N_p$. Im Weiteren werden $(u_{ini}, y_{ini}) = \{u_{k-i}, y_{k-1}\}_{i=T_{ini}}^1$ als die zuletzt gemessenen Eingangs- und Ausgangsdaten des Systems definiert. Unter der Verwendung dieser Definition und den in Gleichung (4) festgelegten Teilmatrizen, lässt sich

(3) in erweiterter Form mit Gleichung (5) darstellen:

$$\begin{pmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{pmatrix} g = \begin{pmatrix} u_{ini} \\ y_{ini} \\ u \\ y \end{pmatrix}. \quad (5)$$

Über die ersten zwei Gleichungen in (5) wird eine Schätzung des impliziten Zustands des Systems durchgeführt. Dafür werden die neusten Datentrajektorien (u_{ini}, y_{ini}) der Länge T_{ini} genutzt, von wo aus die Prädiktion des Systems stattfindet [12]. Mit den neusten Datentrajektorien (u_{ini}, y_{ini}) sowie den zuvor definierten Hankelmatrizen U_p, U_f , einem vorgegebenen Eingangssignal u und der dazugehörigen Hankelmatrix U_f kann ein lineares Gleichungssystem aufgestellt werden. Die Lösung dieses Gleichungssystems liefert den Gewichtungsvektor g . Eine rein datenbasierte Prädiktion von y erfolgt über die letzte Zeile von (5) und somit über die Gleichung $y = Y_f g$. Diese nichtparametrische Systemdarstellung führt eine implizite Zustandsschätzung und gleichzeitige Prädiktion durch, womit es in prädiktiven Regelungsansätzen Verwendung finden kann. Durch die Wahl dieser Formulierung kann auf eine separate Zustandsschätzung mittels z. B. Beobachterentwurf oder Kalmanfilter verzichtet werden.

2.2 Data Enabled Predictive Control

Bei DeePC handelt es sich um einen prädiktiven Regelungsansatz, der nur auf Ein- und Ausgangsdaten des zu regelnden Systems basiert und somit eine klassische Offline-Systemidentifikation eines parametrischen Modells umgeht. Gleichung (5) definiert die nichtparametrische Systemdarstellung und ersetzt dabei die parametrischen Modelle, wie z. B. lineare Zustandsraummodelle oder ARX-Modelle, wie sie in modellbasierten prädiktiven Regelungsverfahren explizit vorliegen. Das folgende Optimierungsproblem [1] kann zur prädiktiven Regelung von LTI-Systemen ohne Messrauschen herangezogen werden. Dieses wird unter Minimierung von ℓ_1 und unter Berücksichtigung von Regel- und

Stellgrößenbeschränkungen gelöst:

$$\min_{g, u, y} \sum_{k=0}^{N_P-1} \ell_1(y_k, u_k) \quad (6a)$$

$$\text{s.t.} \quad \begin{pmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{pmatrix} g = \begin{pmatrix} u_{ini} \\ y_{ini} \\ u \\ y \end{pmatrix}, \quad (6b)$$

$$u_k \in \mathcal{U}, \forall k \in \{0, \dots, N_P - 1\}, \quad (6c)$$

$$y_k \in \mathcal{Y}, \forall k \in \{0, \dots, N_P - 1\}. \quad (6d)$$

Der Term ℓ_1 in (6a) betrachtet die Abweichung der Regelgröße y von der Führungsgröße r , sowie die Stellgröße u_k und ist folgendermaßen definiert:

$$\ell_1(y_k, u_k) = \left(\|y_k - r_k\|_Q^2 + \|u_k\|_R^2 \right). \quad (7)$$

Die Abweichung von der vorhergesagten Regelgröße y zur Führungsgröße r wird mit der euklidischen Norm erfasst und über Wichtungsmatrix $Q \in \mathbb{R}^{p \times p}$ im Optimierungsproblem gewichtet. Analog dazu wird die Stellgröße u mit der euklidischen Norm bestraft. Eine Gewichtung dieses Terms findet mit der Wichtungsmatrix $R \in \mathbb{R}^{m \times m}$ statt. Bei N_P handelt es sich um den Prädiktionshorizont, womit die Länge der prädizierten Regel- und Stellgrößen bestimmt wird. T_{ini} hingegen beschreibt die Länge von u_{ini} und y_{ini} , welche den impliziten Zustand des Systems beschreiben. Im Rahmen einer Reglerparametrierung sind somit die Hyperparameter N_P , T_{ini} sowie Q und R festzulegen. Neben der Minimierung der Kostenfunktion werden die Regel- und Stellgrößen über die konvexen Mengen $\mathcal{U}$ und $\mathcal{Y}$ im Optimierungsproblem beschränkt, wie in (6c) und (6d) dargestellt. In (6b) ist die nichtparametrische Darstellung des Systems als Gleichungsnebenbedingung definiert. Diese Formulierung eignet sich nicht für die praktische Umsetzung, da das zu regelnde System oft mit Prozess- oder Messrauschen versehen ist und somit das Fundamental Lemma nach Willems nicht erfüllt ist. Um diesen Sachverhalt entgegenzuwirken, schlägt [1] die Verwendung von Regularisierung und Slackvariablen vor. Dabei ergibt sich das

folgende angepasste Optimierungsproblem:

$$\min_{g, u, y, \sigma_y, \sigma_{\Delta u}} \sum_{k=0}^{N_P-1} \ell_1(y_k, u_k) + \ell_2(g) + \ell_3(\sigma_u, \sigma_y) \quad (8a)$$

$$\text{s.t.} \quad \begin{pmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{pmatrix} g = \begin{pmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \\ y \end{pmatrix} + \begin{pmatrix} \sigma_u \\ \sigma_y \\ 0 \\ 0 \end{pmatrix}, \quad (8b)$$

$$u_k \in \mathcal{U}, \forall k \in \{0, \dots, N_P - 1\}, \quad (8c)$$

$$y_k \in \mathcal{Y}, \forall k \in \{0, \dots, N_P - 1\}. \quad (8d)$$

Zusätzlich zu ℓ_1 werden die Terme ℓ_2 und ℓ_3 in der Kostenfunktion ergänzt. Diese sind wie folgt definiert:

$$\ell_2(g) = \lambda_g \|g\|_2, \quad (9a)$$

$$\ell_3(\sigma_u, \sigma_y) = \lambda_u \|\sigma_u\|_2 + \lambda_y \|\sigma_y\|_2. \quad (9b)$$

Der Term ℓ_2 regularisiert die Optimierungsvariable g innerhalb des Optimierungsproblems, um eine Überanpassung der Prädiktionen zu vermeiden. Hierbei wird g ebenso mit der euklidischen Abstandsnorm bewertet und über λ_g gewichtet. Da bedingt durch z. B. Messrauschen die Gleichungsnebenbedingungen in (6b) nicht erfüllt werden können, müssen die Slackvariablen σ_u sowie σ_y eingeführt, um die Lösbarkeit des Optimierungsproblems in jedem Abtastschritt zu gewährleisten. Beide Slackvariablen werden in der Kostenfunktion über den Term ℓ_3 mit der euklidischen Norm bestraft und einem entsprechenden Gewicht λ_u sowie λ_y bewertet. Dadurch ergeben sich bei der Reglerimplementierung die zusätzlichen Tuningparameter $\lambda_g, \lambda_u, \lambda_y$.

3 Erweiterung von DeePC mittels Takagi-Sugeno-Fuzzy-Systemen

3.1 Partitionierung mit regulärer Gitterstruktur

Die hier vorgestellte Methode wird im Folgenden als „G-DeePC“ bezeichnet. Die in diesem Abschnitt verwendeten Gleichungen folgen aus [13, 15]. Mittels TS-Systemen ist es möglich, das nichtlineare Gesamtübertragungsverhalten eines Prozesses mit lokalen linearen Modellen zu approximieren. Die Daten werden z. B. gemäß einer regulären Gitterstruktur partitioniert und in Abhängigkeit der aktuellen Prämissenvariablen über unscharfe Zugehörigkeitsfunktionen gewichtet. Im Anschluss können die gewichteten nichtparametrischen Systemdarstellungen dem Optimierungsproblem in jedem Abtastschritt übergeben werden. Diese Idee wird nun auf die DeePC-Methodik erweitert. Aus Gründen der besseren Übersicht enthält die Prämisse $\mathbf{z}_k = [p_1 \ p_2]^T$ nur zwei Schedulingvariablen. Theoretisch sind hier aber beliebig viele Schedulingvariablen möglich. Das Regelwerk des TS-Systems besteht aus $i = 1 \dots N_g$ Regeln, wobei die i -te Regel wie folgt definiert ist:

$$\begin{aligned} R_i : \text{ WENN } p_1 \text{ ist } \mu_{i,1}(p_1) \text{ UND } p_2 \text{ ist } \mu_{i,2}(p_2) \\ \text{ DANN } U_p^{(i)}, U_f^{(i)}, Y_p^{(i)}, Y_f^{(i)}. \end{aligned} \quad (10)$$

Ferner stellen $U_p^{(i)}, U_f^{(i)}, Y_p^{(i)}, Y_f^{(i)}$ die gewichteten Hankelmatrizen des entsprechenden nichtparametrischen Teilmodells dar. Diese müssen im Vorfeld über das Regelwerk berechnet werden. Weiterhin werden Gauß'sche Zugehörigkeitsfunktionen mit den Zentren $c_{i,j}$ und der Standardabweichung $\sigma_{i,j}$ verwendet:

$$\mu_{i,j}(p_j) = \exp\left(\frac{-(p_j - c_{i,j})^2}{2\sigma_{i,j}^2}\right). \quad (11)$$

Die Zentren und Standardabweichungen werden so gewählt, dass eine äquidistante Verteilung der Gaußglocken stattfindet. Anschließend liefert die Auswertung der Prämissen den Erfülltheitsgrad $w_i(\mathbf{z}_k)$ der i -ten Regel. Danach lässt

sich über die folgende Gleichung die Fuzzy-Basisfunktion bestimmen:

$$h_i(\mathbf{z}_k) = \frac{w_i(\mathbf{z}_k)}{\sum_{i=1}^{N_g} w_i(\mathbf{z}_k)}. \quad (12)$$

Alle N_g Fuzzy-Basisfunktionen müssen die Orthogonalitätsbedingung erfüllen:

$$\sum_{i=1}^{N_g} h_i(\mathbf{z}_k) = 1. \quad (13)$$

$h_i(\mathbf{z}_k)$ kann im Rahmen der Datenaufteilung genutzt werden, um die Hankelmatrizen, welche als nichtparametrische Teilmodelle auftreten, zu gewichten. Dafür wird die Hauptdiagonale der Wichtungsmatrix W_i mit den Zahlenwerten der i -ten Fuzzy-Basisfunktion besetzt:

$$W_i = \text{diag}(h_i(\mathbf{z}_1), \dots, h_i(\mathbf{z}_N)) \in \mathbb{R}^{N \times N}. \quad (14)$$

N entspricht dabei der Anzahl an Spalten in den Hankelmatrizen, d. h. es gilt $N = T_d - N_p - T_{ini} + 1$. Dies muss bei der Bestimmung der N_g Teilmodelle gewährleistet sein, da W_i jede Spalte der Hankelmatrizen gewichtet. Eine Berechnung des i -ten nichtparametrischen Modells erfolgt über:

$$U_p^{(i)} = U_p W_i, \quad Y_p^{(i)} = Y_p W_i, \quad U_f^{(i)} = U_f W_i, \quad Y_f^{(i)} = Y_f W_i. \quad (15)$$

Die N_g Teilmodelle werden im Optimierungsproblem von G-DeePC eingesetzt. Auf Basis der aktuellen Prämissenvariable resultiert eine nichtparametrische Systemdarstellung. Diese setzt sich zusammen aus der gewichteten Summe der N_g Teilmodelle. Die folgende Gleichung zeigt die Berechnung für $\bar{U}_p(\mathbf{z}_k)$ im resultierenden nichtparametrischen Modell in Abhängigkeit von der aktuellen Prämissenvariable. Die Berechnung für $\bar{U}_f(\mathbf{z}_k)$, $\bar{Y}_p(\mathbf{z}_k)$ und $\bar{Y}_f(\mathbf{z}_k)$, welche das nichtparametrische Modell vervollständigen, erfolgt analog:

$$\bar{U}_p(\mathbf{z}_k) = \sum_{i=1}^{N_g} h_i(\mathbf{z}_k) \cdot U_p^{(i)}. \quad (16)$$

Mit Verwendung der sich in jedem Abtastschritt neu einstellenden nichtparametrischen Systemdarstellung ergibt sich das folgende Optimierungsproblem:

$$\min_{g, u, y, \sigma_y, \sigma_u} \sum_{k=0}^{N_p-1} \ell_1(y_k, u_k) + \ell_2(g) + \ell_3(\sigma_y, \sigma_u) \quad (17a)$$

$$\text{s.t.} \quad \begin{pmatrix} \bar{U}_p(\mathbf{z}_k) \\ \bar{Y}_p(\mathbf{z}_k) \\ \bar{U}_f(\mathbf{z}_k) \\ \bar{Y}_f(\mathbf{z}_k) \end{pmatrix} g = \begin{pmatrix} u_{\text{ini}} \\ y_{\text{ini}} \\ u \\ y \end{pmatrix} + \begin{pmatrix} \sigma_u \\ \sigma_y \\ 0 \\ 0 \end{pmatrix}, \quad (17b)$$

$$u_k \in \mathcal{U}, \forall k \in \{0, \dots, N_p - 1\}, \quad (17c)$$

$$y_k \in \mathcal{Y}, \forall k \in \{0, \dots, N_p - 1\}. \quad (17d)$$

Dieses unterscheidet sich lediglich in der nichtparametrischen Systemdarstellung zu (8). Der folgende Algorithmus zeigt die Herangehensweise für G-DeePC:

Algorithmus 1.: G-DeePC

Offline:

- 1: Erzeuge U_p, Y_p, U_f, Y_f mit (u_d, y_d) und (2), definiere die Reglerparameter $T_{\text{ini}}, N_p, Q, R, \lambda_g, \lambda_y, \lambda_u$ sowie die Parameter des TS-Systems $N_g, c_{i,j}, \sigma_{i,j}$
- 2: Definiere die Prämissenvariablen p_1, p_2 in $\mathbf{z}_k$
- 3: Erzeuge die Wichtungsmatrizen mit (14)
- 4: Erzeuge die N_g nichtparametrischen Teilmodelle $U_p^{(i)}, Y_p^{(i)}, U_f^{(i)}, Y_f^{(i)}$ mit (15)

Online:

- 5: Messe $u_{\text{ini}}, y_{\text{ini}}$ und erzeuge $\mathbf{z}_k$
 - 6: Erzeuge $\bar{U}_p(\mathbf{z}_k), \bar{Y}_p(\mathbf{z}_k), \bar{U}_f(\mathbf{z}_k), \bar{Y}_f(\mathbf{z}_k)$ mit (16)
 - 7: Löse das Optimierungsproblem (17) für g^*
 - 8: Berechne die optimale Stellgrößenfolge mit $u^* = \bar{U}_f(\mathbf{z}_k)g^*$
 - 9: Schalte das erste Element von u^* auf den Prozess und setze k zu $k + 1$
 - 10: Gehe zu Zeile 5 und starte von vorne
-

3.2 Partitionierung mit Fuzzy-c-Means

Im Gegensatz zu einer regulären gitterartigen Partitionierung kann der Prämissenraum auch mittels des Fuzzy-c-Means (FCM)-Algorithmus aufgeteilt werden. Die Gleichungen zu FCM sind aus [14, 15] entnommen. Bei FCM handelt es

sich um einen unscharfen Clustering-Algorithmus, wobei jedem Datenpunkt des Merkmalraums eine Zugehörigkeit $\mu_i(\mathbf{z}_k)$ zu einem der $i = 1 \dots N_c$ Cluster zugeschrieben wird. Im Rahmen von FCM wird folgendes Zielfunktional minimiert:

$$J(N_c, v) = \sum_{k=1}^N \sum_{i=1}^{N_c} \mu_i^v(\mathbf{z}_k) \cdot \|\mathbf{z}_k - \mathbf{v}_i\|^2. \quad (18)$$

Dabei ist N_c die Anzahl der Cluster, $\mathbf{z}_k$, $k = 1 \dots N$ stellt den k -ten Datenpunkt des Merkmalraums dar, $\mathbf{v}_i$ zeigt das i -te Clusterzentrum und v ist ein Unschärfe-parameter, mit welchem die Unschärfe der Clusterzuordnung bestimmt wird. Als Nebenbedingung des Minimierungsproblems gilt die Orthogonalitätsbedingung, sodass alle Zugehörigkeiten eines Datenpunktes folgende Gleichung erfüllen:

$$\sum_{i=1}^{N_c} \mu_i(\mathbf{z}_k) = 1. \quad (19)$$

Die Clusterzentren werden im Minimierungsproblem über folgende Berechnungsvorschrift ermittelt:

$$\mathbf{v}_i = \frac{\sum_{k=1}^N \mu_i^v(\mathbf{z}_k) \cdot \mathbf{z}_k}{\sum_{k=1}^N \mu_i^v(\mathbf{z}_k)}. \quad (20)$$

Eine Berechnung der Zugehörigkeiten der Datenpunkte zu den jeweiligen Clustern erfolgt über folgende Gleichung:

$$\mu_i(\mathbf{z}_k) = \left[\sum_{j=1}^{N_c} \left(\frac{\|\mathbf{z}_k - \mathbf{v}_i\|}{\|\mathbf{z}_k - \mathbf{v}_j\|} \right)^{\frac{2}{v-1}} \right]^{-1}. \quad (21)$$

Der FCM-Algorithmus generiert neben den Clusterzentren v die sogenannte Partitionsmatrix U . Diese hat bezogen auf die hier vorgestellte Methode N_c Zeilen sowie N Spalten und enthält die Zugehörigkeiten jedes Datenpunktes zu den N_c Clustern:

$$U = [\mu_i(\mathbf{z}_k)] \in \mathbb{R}^{N_c \times N}. \quad (22)$$

Mit der Partitionsmatrix U ist es nun möglich, die Spalten der Hankelmatrizen U_p, U_f, Y_p, Y_f gemäß FCM zu gewichten. Dazu wird die i -te Zeile in U genutzt,

um die Hauptdiagonalelemente einer Wichtungsmatrix W_i zu besetzen:

$$W_i = \text{diag}(\mu_i(\mathbf{z}_1), \dots, \mu_i(\mathbf{z}_N)) \in \mathbb{R}^{N \times N}. \quad (23)$$

Über diese Wichtungsmatrizen können die nichtparametrischen Teilmodelle nun analog zu der in Kapitel 3.1 vorgestellten Methode über Gleichung (15) berechnet werden. Da FCM orthogonale Zugehörigkeitsfunktionen [15] liefert, sind sie identisch mit den Fuzzy-Basisfunktionen und können zur Gewichtung herangezogen werden. Die Berechnung des resultierenden nichtparametrischen Modells in Abhängigkeit von der aktuellen Prämissenvariable erfolgt mit Hilfe von Gleichung (21). Ergänzend mit den nichtparametrischen Teilmodellen ist die Berechnung des aktuell gültigen Modells möglich über:

$$\bar{U}_p(\mathbf{z}_k) = \sum_{i=1}^{N_c} \mu_i(\mathbf{z}_k) \cdot U_p^{(i)}. \quad (24)$$

Die Berechnung von $\bar{U}_f(\mathbf{z}_k)$, $\bar{Y}_p(\mathbf{z}_k)$ und $\bar{Y}_f(\mathbf{z}_k)$ findet analog statt. Im Anschluss finden die Hankelmatrizen in (17) Verwendung. Die in diesem Abschnitt behandelte Methode wird im weiteren Verlauf mit „FCM-DeePC“ abgekürzt. Die Funktionsweise von FCM-DeePC ist im folgenden Algorithmus veranschaulicht.

Algorithmus 2.: FCM-DeePC

Offline:

- 1: Erzeuge U_p, Y_p, U_f, Y_f mit (u_d, y_d) und (2), definiere die Reglerparameter $T_{\text{ini}}, N_p, Q, R, \lambda_g, \lambda_y, \lambda_u$ sowie die Parameter von FCM N_c, v
- 2: Definiere den Merkmalsraum und führe FCM durch
- 3: Erzeuge die Wichtungsmatrizen mit (22), (23)
- 4: Erzeuge die N_c nichtparametrischen Teilmodelle $U_p^{(i)}, Y_p^{(i)}, U_f^{(i)}, Y_f^{(i)}$ mit (15)

Online:

- 5: Messe $u_{\text{ini}}, y_{\text{ini}}$ und erzeuge $\mathbf{z}_k$
 - 6: Erzeuge $\bar{U}_p(\mathbf{z}_k), \bar{Y}_p(\mathbf{z}_k), \bar{U}_f(\mathbf{z}_k), \bar{Y}_f(\mathbf{z}_k)$ mit (24)
 - 7: Löse das Optimierungsproblem (17) für g^*
 - 8: Berechne die optimale Stellgrößenfolge mit $u^* = \bar{U}_f(\mathbf{z}_k)g^*$
 - 9: Schalte das erste Element von u^* auf den Prozess und setze k zu $k+1$
 - 10: Gehe zu Zeile 5 und starte von vorne
-

4 Simulationsergebnisse

4.1 Hammerstein-System

Die beiden DeePC-Erweiterungen werden im folgenden Kapitel an einem nichtlinearen Hammerstein-System evaluiert, das in [13] beschrieben ist. Die folgende Gleichung zeigt die zeitdiskrete nichtlineare Differenzengleichung:

$$\begin{aligned} y(k) = & 0.0187 \arctan(u(k-1)) \\ & + 0.01746 \arctan(u(k-2)) \\ & + 1.7826y(k-1) - 0.8187y(k-2). \end{aligned} \quad (25)$$

Hammerstein-Systeme zeichnen sich durch eine statische Nichtlinearität aus, auf die ein lineares System folgt. In diesem Beispielsystem wird die Nichtlinearität durch eine $\arctan(\cdot)$ -Funktion beschrieben. Das nachgeschaltete lineare System entspricht einem gedämpften, schwingungsfähigen PT_2 -Glieder mit der Verstärkung $K = 1$, der Zeitkonstante $T = 5s$ und einer Dämpfung $D = 0.5$. Zur Anregung des Systems wird ein Amplitude-Modulated Pseudo-Random Binary Signal (APRBS) im Wertebereich von $[0, 3]$ eingesetzt. Sowohl das Anregungssignal u als auch die Systemantwort y sind in Abbildung 1 zu sehen. Die Länge des Anregungssignals beläuft sich auf 400s. Bei einer Abtastzeit von $T_s = 1s$ ist $T_d = 400$. Diese Daten werden genutzt, um die Hankelmatrizen zu besetzen und dienen als initiales nichtparametrisches Modell.

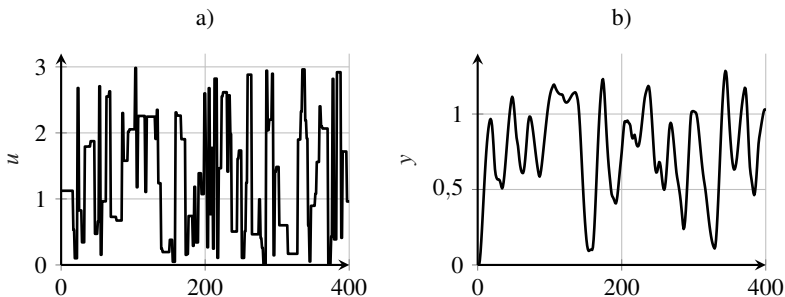


Bild 1: **a)** Anregungssignal u im Wertebereich von $[0, 3]$. **b)** Systemantwort y des Hammerstein-Systems.

4.2 Ergebnisse der Regelung und Vergleich

Da die Nichtlinearität lediglich in den Eingängen vorhanden ist, findet die Partitionierung in $u(k-1)$ und $u(k-2)$ statt. Dadurch ergibt sich die Prämissenvariable $\mathbf{z}_k = [u(k-1) \ u(k-2)]^T$. Für die Partitionierung gemäß der regulären Gitterstruktur und des FCM werden dafür die erste und zweite Zeile der Hankelmatrix U_p verwendet, wodurch $N = T_d - N_p - T_{ini} + 1$ gilt. Wie in [13] beschrieben ist, spielt es keine Rolle, in welcher Dimension von $\mathbf{z}_k$ eine Partitionierung stattfindet. Dies liegt daran, dass sich $u(k-1)$ und $u(k-2)$ in einem Großteil der Daten kaum voneinander unterscheiden. Aus diesem Grund wird die reguläre Partitionierung nur in der Dimension $u(k-1)$ durchgeführt. FCM hingegen partitioniert weiter in $u(k-1)$ und $u(k-2)$. Im Rahmen einer Simulationsstudie werden die beiden DeePC-Erweiterungen miteinander verglichen, wobei aus Gründen der Vergleichbarkeit ein grundlegender Parametersatz für die Reglerparameter festgelegt wird:

$$\begin{aligned} T_d &= 400, N_p = 10, T_{ini} = 8, \lambda_g = 0.2, \\ \lambda_u &= 1 \cdot 10^6, \lambda_y = 1 \cdot 10^6, Q = 2, R = 0.1. \end{aligned} \quad (26)$$

In der Simulationsstudie wird die Folgeregung einer zuvor definierten Referenztrajektorie mittels Algorithmus 1 und Algorithmus 2 untersucht. Hierbei wird sowohl die Anzahl der Regeln in $u(k-1)$ als auch die Anzahl der Cluster

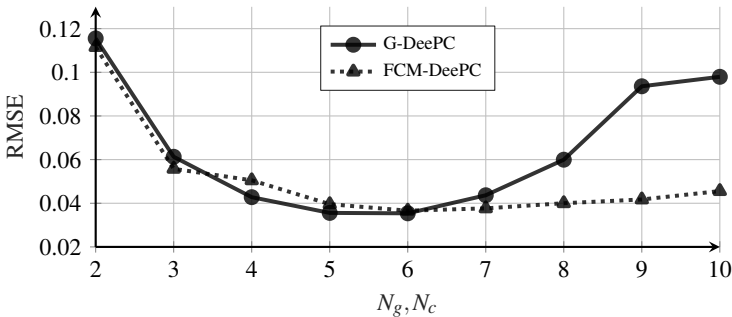


Bild 2: Einfluss der Regelanzahl N_g und Clusteranzahl N_c auf die Regelgüte. Dargestellt ist der RMSE für G-DeePC (durchgezogene Linie mit Kreisen) und FCM-DeePC (gestrichelte Linie mit Dreiecken).

in $u(k-1)$ und $u(k-2)$ zwischen $[2, 10]$ variiert. Für den FCM-DeePC wird der Unschärfeparameter auf $v = 2$ gesetzt. Der Regelfehler wird in Form des Root Mean Squared Error (RMSE) bewertet. Die Simulationsstudie wird mit MATLAB durchgeführt und die Lösung des Optimierungsproblems erfolgt mit qpOASES². Dabei wird ersichtlich, dass beide Ansätze den kleinsten Fehler im Sinne des RMSE bei sechs Regeln bzw. Clustern aufweisen. Dieser Sachverhalt ist in Abbildung 2 ersichtlich. Zudem ist zu erkennen, dass mit einer weiteren Steigerung der Regeln/Clustern keine Verbesserung eintritt, sondern bei $N_g, N_c = 6$ das Minimum vorliegt. Für den G-DeePC ergibt sich mit dieser Erkenntnis die in Abbildung 3 vorliegende Partitionierung. Es zeigt sich die Aufteilung in der $u(k-1)$ -Dimension und die äquidistante Verteilung der Fuzzy-Zugehörigkeitsfunktionen. Im Gegensatz dazu, partitioniert der FCM-DeePC in beiden Dimensionen von $\mathbf{z}_k$. FCM setzt die Clusterzentren entlang der Winkelhalbierenden. Dies zeigt, dass auch FCM die geringen Unterschiede zwischen $u(k-1)$ und $u(k-2)$ erkennt und eine Aufteilung in einer Dimension ausreichend ist. Die Partitionierung mittels FCM ist in Abbildung 4 zu sehen. Die Ellipsen in der Abbildung zeigen die Niveaulinien der Zugehörigkeiten zu den Clusterzentren. Um die Funktionalität der beiden Ansätze zu zeigen,

² <https://github.com/coin-or/qpOASES>

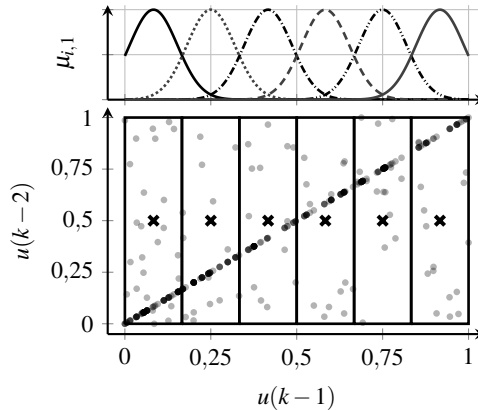


Bild 3: Regulierte Gitterpartitionierung des Prämisseusraums mit $N_g = 6$ Regeln. Die Markierung innerhalb der Teilbereiche verdeutlicht das Zentrum der Gauß'schen Zugehörigkeitsfunktion. Graue Punkte zeigen die Daten. Dunklere Datenpunkte veranschaulichen Datenanhäufungen.

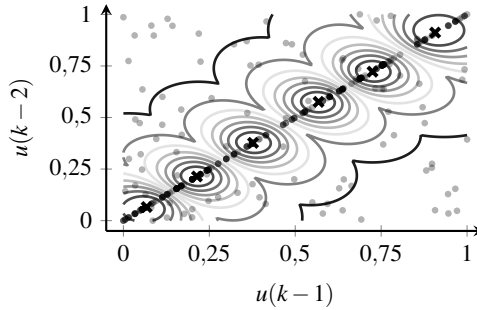


Bild 4: Partitionierung des Prämissenraums mittels FCM mit $N_c = 6$ Clustern. Die Markierungen verdeutlichen die Zentren der Cluster. Die Ellipsen sind Niveaulinien der Zugehörigkeiten um die Clusterzentren. Die Abbildung ist mit der Fuzzy Clustering and Data Analysis Toolbox³ erstellt worden.

erfolgt ein Vergleich mit der Standard-DeePC-Formulierung in Gleichung (8). Der Standard-DeePC arbeitet mit denselben Anregungsdaten wie der G-DeePC sowie FCM-DeePC und wird mit den Werten aus (26) parametrisiert. Da eine Anpassung von λ_g zu einer Verbesserung des Standard-DeePC-Ergebnisses führt, wird dieser Parameter für den Standard-DeePC auf $\lambda_g = 5$ gesetzt. Die Betrachtung der Fehlermaße aufgelistet in Tabelle 1 verdeutlicht die verbesserte Regelgüte der beiden DeePC-Erweiterungen im Vergleich zum Standard-DeePC. In Abbildung 5 werden die Ergebnisse der Folgeregulierung für den DeePC, den G-DeePC und den FCM-DeePC dargestellt. Neben der Regelgröße y ist die Regelabweichung $y - r$, die Stellgröße u und die Zugehörigkeitsfunktion μ des FCM-DeePC sowie die Fuzzy-Basisfunktion h des G-DeePC zu sehen. Bei

³ <https://www.abonyilab.com/fclusttoolbox>

Tabelle 1: Vergleich der Regelgüte von DeePC, G-DeePC und FCM-DeePC. Angegeben sind der *Root Mean Squared Error* (RMSE) und der *Sum of Squared Errors* (SSE)

Formulierung	RMSE	SSE
DeePC	0.0405	0.9819
G-DeePC	0.0354	0.7515
FCM-DeePC	0.0366	0.8037

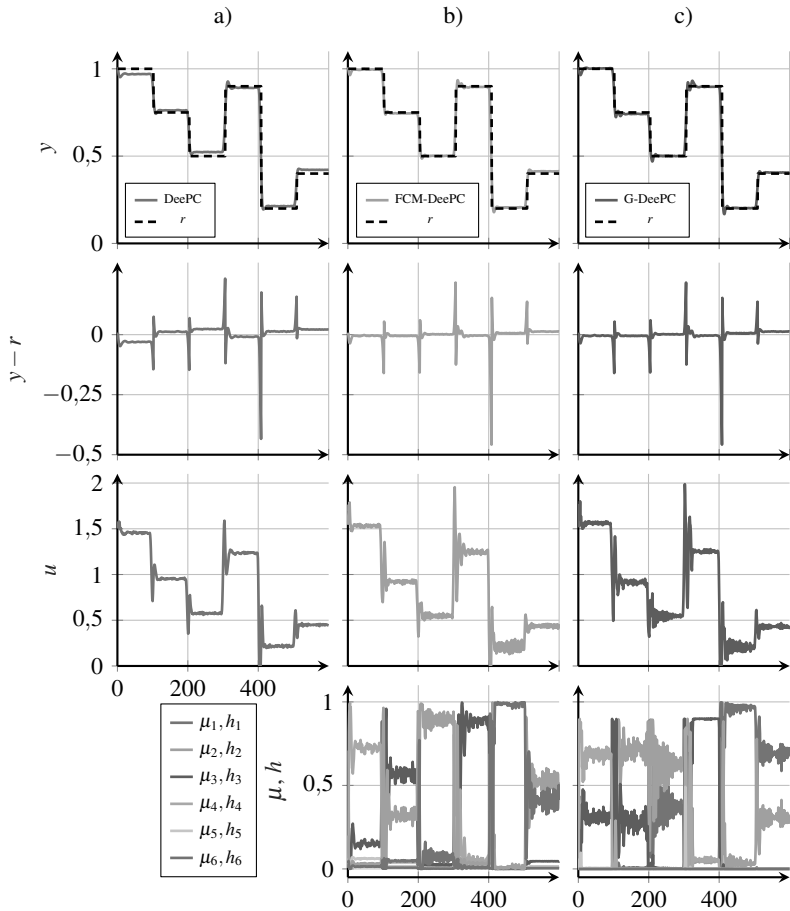


Bild 5: Vergleich der Regelungsergebnisse für **a)** DeePC, **b)** FCM-DeePC, **c)** G-DeePC. Gezeigt sind von oben nach unten die Regelgröße y , die Regelabweichung $y - r$, die Stellgröße u sowie die Gewichte der einzelnen Teilmodelle μ_i, h_i für die beiden DeePC-Erweiterungen.

genauerer Betrachtung der Regelabweichung ist zu erkennen, dass beide DeePC-Erweiterungen eine verbesserte Regelgüte durch Partitionierung der Daten und das gezielte Einbringen von Prozesswissen aufweisen. Der Standard-DeePC zeigt in einem Großteil der Arbeitsbereiche einen Offset. Gleichzeitig reagieren jedoch die Stellgrößen der beiden Erweiterungen empfindlich, was zu einem leichten Überschwingen bei einigen Sollwertwechseln führt.

5 Diskussion und Ausblick

In diesem Artikel werden zwei Erweiterungen von DeePC für eine Klasse nichtlinearer Systeme vorgestellt. Dabei werden die Spalten der Hankelmatrizen, welche im DeePC Verwendung finden, entsprechend der aktuellen Prämissenvariable gewichtet. Die Gewichtung kann sowohl über ein TS-Regelwerk mit regulärer Gitterstruktur (G-DeePC) als auch mit FCM (FCM-DeePC) stattfinden. An einem Hammerstein-System wird gezeigt, dass die nichtlinearen Erweiterungen von DeePC die Regelgüte im Vergleich zu der Standard-DeePC-Formulierung verbessern. Dies lässt sich damit erklären, dass die nichtlinearen Erweiterungen die Matrixspalten stärker gewichten, welche den aktuellen Arbeitspunkt repräsentieren. Spalten, die entfernt vom aktuellen Arbeitspunkt liegen, werden nahe Null gesetzt. Dadurch arbeitet das Optimierungsproblem in jedem Arbeitspunkt nur mit den zu diesem Zeitpunkt relevanten Spalten der Hankelmatrizen, während die Standard-DeePC-Formulierung alle Spalten nutzt. Diese Art der Spaltenauswahl bzw. -gewichtung kann als lokale Linearisierung interpretiert werden. Durch diese Erweiterung kann die DeePC-Methodik nun vorhandenes Prozesswissen mit in die Regelung einbringen, was bei der Standard-DeePC-Formulierung nicht ohne Weiteres möglich ist. In den hier vorgestellten Ansätzen muss keine parametrische Modellstruktur vorgegeben werden und keine Offline-Systemidentifikation stattfinden. Trotz einer Verbesserung der Regelgüte und prinzipiellen Anwendbarkeit der vorgestellten Ansätze haben diese Limitierungen, welche in zukünftigen Arbeiten näher untersucht werden müssen. Beide Erweiterungen gewichten die Spalten der Hankelmatrizen, was nicht garantiert, dass die resultierenden Hankelmatrizen vollen Zeilenrang besitzen. Diese Bedingung ist Grundvoraussetzung für das Fundamental Lemma nach Willems. Außerdem muss sich genauer mit dem Einfluss des nichtparametrischen Modells auseinander gesetzt werden, vor allem mit Hinblick auf Robustheit und Stabilitätsbetrachtungen der entwickelten Regler. Zukünftige Arbeiten sollten sich hier mit theoretischen Robustheits- und Stabilitätsuntersuchungen beschäftigen. Ein weiterer Nachteil ist, dass die Datenlänge T_d die Größe des Optimierungsproblems bestimmt und dadurch die Aufzeichnung großer Datenmengen limitiert. Dies führt dazu, dass bei einer großen Anzahl an Regeln bzw. Clustern nur wenige Datenpunkte pro Regel bzw. Cluster vorhanden sind, was zu einer Verschlechterung der Reglerperformance

führen kann. Eine Möglichkeit, diesem Problem entgegenzuwirken, wird in [14] vorgeschlagen. Statt alle gewichteten Spalten des i -ten Teilmodells zu berücksichtigen, werden nur die Spalten ausgewählt, die dem i -ten Teilmodell mit einer Zugehörigkeit größer als einem festgelegten Schwellenwert zugeordnet sind. Damit kann die Größe des Optimierungsproblems verkleinert und ein größerer Datensatz zur Partitionierung genutzt werden. Zukünftige Arbeiten sollten zudem die Anwendbarkeit auf komplexere nichtlineare Systeme sowie die praktische Realisierung untersuchen.

Literatur

- [1] J. Coulson, J. Lygeros und F. Dörfler. „Data Enabled Predictive Control: In the Shallows of the DeePC“. 2019 European Control Conference (ECC), S. 307-312. 2019.
- [2] J. Maciejowski „Predictive Control with Constraints“. Prentice Hall, England. 2022.
- [3] J. W. Polderman und J.C. Willems „Introduction to mathematical systems theory: a behavioral approach“. Springer-Verlag. 1997.
- [4] J.C.Willems, P. Rapisarda, I. Markovsky und B.L.M. De Moor. „A note on persistence of excitation“. Systems & Control Letters, 54(4), S. 312-329. 2005.
- [5] F. Dörfler et. al. „Bridging Direct and Indirect Data-Driven Control Formulations via Regularizations and Relaxations“. IEEE Transactions on Automatic Control 68(2), S. 883-897. 2023.
- [6] L. Huang, J. Lygeros und F. Dörfler. „Robust and Kernelized Data-Enabled Predictive Control for Nonlinear Systems“. IEEE Transactions on Control Systems Technology 32(2), S. 611-624. 2024.
- [7] M. Lazar „Basis-Functions Nonlinear Data-Enabled Predictive Control: Consistent and Computationally Efficient Formulations“. 2024 European Control Conference (ECC), S. 888-893. 2024.

- [8] Y. Lian und C. N. Jones „Koopman based data-driven predictive control“. arXiv:2102.05122. 2021.
- [9] M. Lazar „Neural Data-Enabled Predictive Control“. IFAC-PapersOnLine 58(15), S. 91-96. 2024.
- [10] J. Berberich et al. „Linear Tracking MPC for Nonlinear Systems—Part II: The Data-Driven Case“. IEEE Transactions on Automatic Control 67(9), S. 4406-4421. 2022.
- [11] J. Berberich und F. Allgöwer „An Overview of Systems-Theoretic Guarantees in Data-Driven Model Predictive Control “. Annual Review of Control, Robotics, and Autonomous Systems 8. 2025.
- [12] I. Markovsky und P. Rapisarda „Data-driven simulation and control“. International Journal of Control, 81(12), S. 1946-1959. 2008.
- [13] O. Nelles, „Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models“. Springer. 2020.
- [14] R. Babuska „Fuzzy Modeling for Control“. Springer Science. 1998.
- [15] A. Kroll, „Computational Intelligence. Probleme, Methoden und technische Anwendungen.“ De Gruyter Studium. 2016.

Large-Signal Takagi-Sugeno Sliding Mode Observer for Reconstruction of Parameter Changes in DC-DC Converters

Felix Dietrich¹, Horst Schulte¹

¹Control Engineering Research Group, School of Engineering – Energy and Information, University of Applied Sciences HTW Berlin, Wilhelminenhofstraße 75A, 12459 Berlin, Germany

{felix.dietrich, schulte}@htw-berlin.de

Abstract

This paper proposes a non-invasive method for the online detection of parameter variations in DC-DC power converters. The method is computationally efficient and only requires standard current and voltage sensors. The proposed model-based design method consists of several steps. First, general models of converters are presented based on results in [6]. Next, the general design of a Takagi-Sugeno (TS) sliding mode (SM) observer is described, along with the method for fault reconstruction. The T-S SM observer is designed using a buck converter as an example to detect parameter changes in critical components, such as the main capacitor and longitudinal inductance, simultaneously. Finally, simulations involving accelerated aging of converter are performed to validate the observer design. The research takes place in the context of public funded research project INDEX-WR [2]. The aim is to improve product comparability by evaluating the reliability and performance of photovoltaic string converters using assessment criteria. There is a wide variety of inverter topologies available on the market; therefore, it is essential to establish generally valid criteria for their comparison. One criterion is the change in component parameters over the entire life cycle. For this reason, a robust observer approach was used, which is robust to unmodeled dynamics by model reduction (topology

invariant formulation) but is also able to account for physically well describable nonlinearities. The latter property is necessary due to the large operating range in which DC-DC systems are used within photovoltaic systems with fast frequency reserve (FFR) behavior. For this purpose, large-gain models and large-gain observers must be used because the systems are operated not only in the local Maximum Power Point (MPP) region but also the in Demanded Power Point Tracking global region with fast change of the fed-in power for FFR [8]. The main contribution of this work is the formulation of T-S SM observers for DC-DC converters, which are used to simultaneously estimate the change of two parameters in the nonlinear averaging state space model of switched DC-DC converters. This is an extension of the single-parameter estimation proposed in [4].

1 Introduction

A buck converter is a specific type of DC-DC converter that lowers its input voltage while increasing its output current. Buck converters are commonly used as energy processing units in the field of renewable energy, e.g., in combination with photovoltaic power cells [5]. Stresses caused by temperature cycling and mechanical vibrations may cause wear of the electronic components during long-term operation, resulting in parameter drift [6]. Such altered conditions reduce converter performance and ultimately lead to failure. Hence, for health monitoring and lifetime forecasting of converters, parameter measurement is essential. However, parameters cannot be measured directly and must instead be estimated. Offline methods, such as [7], are generally accurate. Nevertheless, they require the interruption of the converter's operation, which is usually undesirable. They also depend on extensive measurement equipment, which can be quite costly. To address these limitations, online methods have also been developed using the converters' built-in sensors to estimate parameter variations during operation. For instance, [8] proposes a method to estimate the Equivalent Series Resistance (ESR) of buck converters. It does so by sampling the output voltage and then fitting the current ripple, which is derived from the output voltage ripple. The ESR is deduced in the process. A scheme to derive the inductance value of DC-DC converters is proposed in [9]. In [10], the ripples in

the converters' voltage and current signals are also analysed, and by including the calculation of the converter's power loss, its parameters are derived. Finally, in [11], a multilayer neural network is used to identify malfunctions in converter operation. In this paper, a nonlinear sliding mode observer is applied to estimate parameter changes in a buck converter. Sliding mode observers have been used to reconstruct actuator and sensor faults. In [12], for example, actuator and sensor faults are estimated in a Modular Multilevel Converter (MMC).

To account for the wide operating range of the buck converter, the Takagi–Sugeno framework is applied for observer synthesis of the nonlinear buck converter model. The Takagi–Sugeno framework, based on fuzzy logic, has been widely applied in observer synthesis, as in [13]. Similarly, in [14], actuator and sensor faults in a wind turbine are reconstructed using a Takagi–Sugeno sliding mode observer. This method is then used to implement a fault-tolerant control scheme for the turbine. Thus, for reconstructing parameter changes in the buck converter, this paper employs a Takagi–Sugeno Edward–Spurgeon sliding mode observer design and formulates the parameter change as an actuator fault to be reconstructed.

In the following, a buck converter model is first described, along with the formulation of its parameter uncertainty as an actuator fault. The stability of the sliding mode observer is then derived for linear models. Next, the Takagi–Sugeno framework is introduced and applied to extend the linear sliding mode observer synthesis to nonlinear systems. Finally, simulation results from an experimental setup are presented.

2 The Buck Converter Model

The ideal buck converter is composed of a diode D , a capacitor C , and an inductor L , see Figure 1. A DC voltage source supplies the circuit with the input voltage V_{in} . At the output, an ohmic load, R , consumes the circuit's energy. The current flow in the circuit is controlled by switching a power semiconductor device. When the switch is closed, it is in an "ON" state, and when open, in an "OFF" state. Within each switching period, the ratio d of the time the switch is

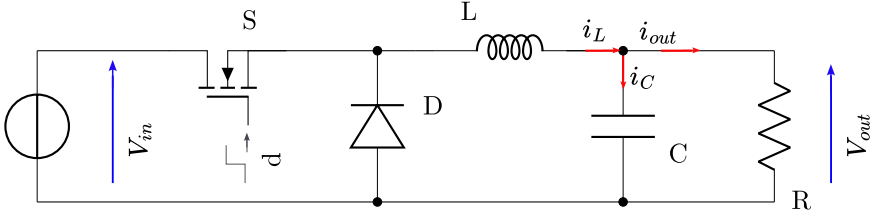


Figure 1: The DC-DC buck converter. While i_L is the inductance's current, V_{in} and V_{out} represent the converters' input and output voltage, R its ohmic load. L represents the converter's inductance, D its diode and C its capacitance. The power semiconductor S can be either open or in an "ON" state or closed in an "OFF" state

in the ON state is defined as the duty cycle. Switching periods generally occur at frequencies ranging from 100 kHz up to several MHz [15]. During operation, this duty cycle is continuously adapted to regulate the buck converter's output voltage and current. As mentioned in the introduction, the focus of this paper, however, lies in observer design rather than the design of the buck converter controller. By applying Kirchhoff's Voltage and Current Laws, we now establish the governing equations of the currents and voltages of the buck converters in both "ON" and "OFF" states:

ON State

$$V_{out} = V_{in} - L \frac{di_L}{dt} \quad (1)$$

$$i_L = C \frac{dV_{out}}{dt} + \frac{V_{out}}{R} \quad (2)$$

OFF State

$$V_{out} = L \frac{di_L}{dt} \quad (3)$$

$$i_L = C \frac{dV_{out}}{dt} + \frac{V_{out}}{R} \quad (4)$$

whereby i_L represents the current flowing through the inductance L , V_{in} and V_{out} the input and output voltages, respectively, and C the converter's capacitance.

2.0.1 State Space Representation of the Average Model

The state-space averaging method is now applied so that both the ON and OFF states of the buck converter can be summarized in a single state-space representation. The currents and voltages from Equations (1) to (4) are averaged over one switching period using the duty cycle $d \in [0, 1]$. This yields the averaged model given in Equation (5):

$$\dot{x} = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix} x + \begin{bmatrix} \frac{d}{L} \\ 0 \end{bmatrix} u \quad (5)$$

where the states are defined as $x = \begin{bmatrix} i_L \\ V_{out} \end{bmatrix}$, the input as $u = V_{in}$.

Finally, we assume that both state variables can be measured; the measured output y can therefore be expressed as:

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x \quad (6)$$

The fact that both states of the system can be measured makes this an unusual application of an observer, whose typical purpose is to reconstruct non-measurable

states. In our case, however, the objective is not state recovery per se, but rather the reconstruction of parameter changes in the system.

3 The Linear Sliding Mode Observer Design for Parameter Change Estimation

In this section, an Edwards–Spurgeon sliding mode observer for uncertain linear systems is presented. The observer is synthesized under the assumption that all system parameters in Equation (5), including the duty cycle d and the load R , are constants, resulting in a linear model. As mentioned in the introduction, the observer design will be expanded to non linear systems in Section 4.

3.1 System Formulation

3.1.1 General Formulation

We use the following general formulation of a linear uncertain dynamical system:

$$\dot{x}(t) = Ax(t) + Bu(t) + M(x, u)f(t) \quad (7)$$

$$y(t) = Cx(t) \quad (8)$$

Where:

$$A \in \mathbb{R}^{n \times n}, \quad B \in \mathbb{R}^{n \times m}, \quad C \in \mathbb{R}^{p \times n}$$

and $M(x, u) \in \mathbb{R}^{n \times q}$ is a time-varying matrix dependent on the system's states and input values.

$f(t) \in \mathbb{R}^q$ represents the unknown but norm-bounded modelling fault that satisfies $\|f(t)\| \leq \gamma$, where γ is a known positive scalar.

We assume that the time-varying matrix $M(x, u)$ can be factorized in a fixed component Λ and a time varying-component N :

$$M(x, u) = \Lambda N(x, u) \quad (9)$$

where $\Lambda \in \mathbb{R}^{n \times q}$ and $N(x, u) \in \mathbb{R}^{q \times q}$. It is further assumed that $N(x, u)$ is invertible for all x and u , and that the states and inputs present in $N(x, u)$ are measurable at all times.

Now, the function f_v is defined as follows:

$$f_v(t, x, u) := N(x, u)f(t) \quad (10)$$

This function represents the unknown 'virtual faults' arising from model uncertainty, which might be caused by parameter deviations in the real system. It depends on the system's states and inputs. The system's state equation in (7) can now be rewritten as:

$$\dot{x}(t) = Ax(t) + Bu(t) + \Lambda f_v(t, x, u) \quad (11)$$

In a first step, $f_v(t, x, u)$ will therefore be estimated with the help of the sliding mode observer. Then, the parameter variations $f(t)$ can be derived by inverting $N(x, u)$.

3.1.2 Application of the uncertain systems' formulation to the buck converter model

This subsection shows how to formulate the parameter variation of the buck converter model in Section 2 in the form introduced in the previous subsection.

Having formulated the general dynamical systems with model uncertainty in Equation (7) we now define a more specific formulation of a dynamical system involving parameter uncertainty:

$$\dot{x} = (A + \Delta A)x + (B + \Delta B)u \quad (12)$$

$$y = Cx \quad (13)$$

Now, we define $\alpha =: \frac{1}{L}$ and $\beta =: \frac{1}{C}$, where L is the inductance value of the buck converter and C its capacitance (see Section 2). A change in these impedance values with respect to their inverse expression can therefore be formulated as

$$\alpha + \Delta\alpha := \frac{1}{L + \Delta L}, \quad \beta + \Delta\beta := \frac{1}{C + \Delta C} \quad (14)$$

(14) applied to (5) results in the following state and input matrices:

$$A = \begin{bmatrix} 0 & -\alpha \\ \beta & -\beta/R \end{bmatrix}, B = \begin{bmatrix} d \alpha \\ 0 \end{bmatrix} \quad (15)$$

Then the uncertainty matrices are consequently:

$$\Delta A = \begin{bmatrix} 0 & -\Delta\alpha \\ \Delta\beta & -\Delta\beta/R \end{bmatrix}, \Delta B = \begin{bmatrix} d \Delta\alpha \\ 0 \end{bmatrix} \quad (16)$$

In the form of (7) and using the factorization presented in (9), we get

$$\Lambda = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, N(x, u) = \begin{bmatrix} \begin{bmatrix} 0 & -1 \end{bmatrix} x + d u & 0 \\ 0 & \begin{bmatrix} 1 & -1/R \end{bmatrix} x \end{bmatrix} \text{ and } f(t) = \begin{bmatrix} \Delta\alpha \\ \Delta\beta \end{bmatrix} \quad (17)$$

For $N(x, u)$ in (17), to be invertible at all times, we define $N(1, 1) := \varepsilon_1$ if $\| \begin{bmatrix} 0 & -1 \end{bmatrix} x + d u \| < \varepsilon_1$ and $N(2, 2) := \varepsilon_2$ if $\| \begin{bmatrix} 1 & -1/R \end{bmatrix} x \| < \varepsilon_2$ with ε_1 and ε_2 being user-defined positive scalars.

This way, these entries of $N(x, u)$ are always greater than 0, and therefore $N(x, u)$ has always full column rank and is thus invertible.

3.2 The Edward-Spurgeon Observer Stability Criteria

In the following, we demonstrate how to design a sliding mode observer and then derive the estimation of parameter changes. The observer design is based on three lemmas:

The first Lemma introduces a coordinate transform of the linear system. The objective of this transformation is to restructure the system so that it can be partitioned into a non-measurable part of size $n - p$ and a measurable part of size p .

Given a triple (A, M, C) representing a linear system as shown in (7) and (8) with

- $p \leq q$ and
- $\text{rank}(CM) = \text{rank}(M) = q$

the following holds:

Lemma 1 (Edwards et al. (2002)) [16]

There is a nonsingular transformation matrix T , which makes the coordinate transformation $x \rightarrow Tx$, for the system (A, M, C) , have the following structure:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad M = \begin{bmatrix} 0 \\ M_2 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & T \end{bmatrix} \quad (18)$$

where:

$$A_{11} \in \mathbb{R}^{(n-p) \times (n-p)}, \quad A_{21} = \begin{bmatrix} A_{211} & A_{212} \end{bmatrix}^T, \quad A_{211} \in \mathbb{R}^{(p-q) \times (n-p)}, \quad M_2 \in \mathbb{R}^{p \times q}$$

with $M_2 = \begin{bmatrix} 0 & M_o \end{bmatrix}^T$, where $M_o \in \mathbb{R}^{q \times q}$ is a non-singular matrix and $T \in \mathbb{R}^{p \times p}$ is orthogonal.

Lemma 2 (Edwards et al. (2002)) [16]

The pair (A_{11}, A_{211}) is detectable if and only if the invariant zeros of (A, M, C) are Hurwitz.

The sliding mode observer for the uncertain system is now designed based

on the assumptions and lemmas introduced above. For the system given in (18), it takes the following structure:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) - G_I e_y(t) + G_n v(t) \quad (19)$$

$$\hat{y}(t) = C\hat{x}(t) \quad (20)$$

where $\hat{x}(t)$ is the estimated state of $x(t)$ and $e_y(t) = \hat{y}(t) - y(t)$ denotes the output estimation error. The term $v(t)$ represents a nonlinear switching function that depends on the output estimation error e_y :

$$v(t) = \begin{cases} -\rho \frac{P_o e_y(t)}{\|P_o e_y(t)\|} & \text{if } e_y(t) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (21)$$

Where ρ is a user-defined scalar. In (19), G_I and $G_n \in \mathbb{R}^{n \times p}$ are two gain feedback matrices that must also be determined by the user.

In practice, however, the nonlinear term $v(t)$, is challenging to compute. Therefore, we introduce the equivalent output error injection, $v_{eq}(t)$, which approximates the nonlinear switching term $v(t)$ as follows:

$$v_{eq}(t) = -\rho \frac{P_o e_y(t)}{\|P_o e_y(t)\| + \delta} \quad \text{if } e_y \neq 0 \quad (22)$$

where δ is a small positive user-defined scalar. Note that the smaller the value of δ is chosen, the more accurate the estimation of $v(t)$ by $v_{eq}(t)$ becomes.

Now, in the coordinate system of (18), let the nonlinear gain matrix G_n have the form of

$$G_n := \begin{bmatrix} -L \\ I_p \end{bmatrix} T^T P_o^{-1} \quad (23)$$

where $L = \begin{bmatrix} L^o & 0 \end{bmatrix}$, with $L^o \in \mathbb{R}^{(n-p) \times (n-p)}$ and $P_o \in \mathbb{R}^{p \times p}$ are to be determined.

How these design variables are synthesised will be discussed in Section 3.4.

The state estimation error system is defined as

$$e(t) := \hat{x}(t) - x(t) \quad (24)$$

and

$$A_o := A - G_I C \quad (25)$$

where A and C have the structure of (18); By inferring (7) in (19) and (8) in (20) the error system becomes:

$$\dot{e}(t) = A_o e(t) + G_n v(t) - M f_v(t, x, u) \quad (26)$$

Now, let us introduce a new coordinate transformation using the following non-singular matrix:

$$T_L = \begin{bmatrix} I_{n-p} & L \\ 0 & T \end{bmatrix} \quad (27)$$

Applying the coordinate transformation of (27) to (18) results in:

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \end{bmatrix}, \mathcal{M} = \begin{bmatrix} 0 \\ \mathcal{M}_2 \end{bmatrix}, \mathcal{C} = \begin{bmatrix} 0 & I_p \end{bmatrix} \quad (28)$$

with $\mathcal{A}_{21} = T A_{21}$, $\mathcal{M}_2 = T M_2$ and $\mathcal{A}_{11}$ defined as:

$$\mathcal{A}_{11} = A_{11} + L^o A_{211} \quad (29)$$

Corollary (Tan and Edwards (2002)) [16]

A sliding motion takes place on the sliding surface $\mathcal{S} = \{e : Ce = 0\}$ with a suitable ρ in finite time. The sliding motion is governed by (29), see proof in [16].

Now, assuming (A_{11}, A_{211}) are detectable, it is possible to stabilize $\mathcal{A}_{11}$ by choosing the right L^o .

3.3 Parameter Drift Recovery

In the new coordinates of (28), the error system (26) can now be partitioned into a non-measurable error state $e_1 \in \mathbb{R}^{n-p}$ and a measurable error state $e_y \in \mathbb{R}^p$:

$$\dot{e}_1(t) = \mathcal{A}_{11}e_1(t) + (\mathcal{A}_{21} - \mathcal{G}_{l,1})e_y(t) \quad (30)$$

$$\dot{e}_y(t) = \mathcal{A}_{21}e_1(t) + (\mathcal{A}_{22} - \mathcal{G}_{l,2})e_y(t) + P_o^{-1}v - \mathcal{M}_2f_v(t, x, u) \quad (31)$$

where $\begin{bmatrix} \mathcal{G}_{l,1} \\ \mathcal{G}_{l,2} \end{bmatrix} = T_L G_L$.

Assuming an operational sliding mode observer, once the sliding motion is achieved, the measurable output estimation error becomes $e_y(t) = \dot{e}_y(t) = 0$. Therefore, (30) and (31) can be simplified to:

$$\dot{e}_1(t) = \mathcal{A}_{11}e_1(t) \quad (32)$$

$$0 = \mathcal{A}_{21}e_1(t) + P_o^{-1}v(t) - \mathcal{M}_2f_v(t, x, u) \quad (33)$$

By choosing L^o appropriately, (29) becomes stable, therefore $e_1(t) \rightarrow 0$ in finite time. Then, (33) results in (34):

$$P_o^{-1}v(t) \approx \mathcal{M}_2f_v(t, x, u) \quad (34)$$

The error signal can therefore be reconstructed by using the equivalent output error injection signal defined in (22) and by taking the left pseudoinverse of the error distribution matrix. In the coordinates of (18), the virtual fault can be estimated as shown in the following:

$$\hat{f}_v(t, x, u) =: (M_2^T M_2)^{-1} M_2^T T^T P_o^{-1} v_{eq}(t) \quad (35)$$

Finally by inferring (9) in (35) the model error can be derived from the virtual fault:

$$\hat{f}(t) = N(x, u)^{-1} \hat{f}_v(t, x, u) \quad (36)$$

Note that $N(x, u)$ is assumed to be invertible and all its occurring states to be measurable.

3.3.1 Parameter Drift Recovery Applied to the Buck Converter Model

As shown in (6) all system states are measurable. Consequently, when applying (35) to our buck converter model, we obtain $T = I_2$, and thus, $M_2 = M$. Now we already deduced $M = \Lambda N(x, u)$ of the buck converter model in (17). Applying this result to (35) and (36) it is possible to reconstruct the parameter change in the buck converter as follows:

$$\hat{f}(t) = \begin{bmatrix} \widehat{\Delta\alpha} \\ \widehat{\Delta\beta} \end{bmatrix} = \begin{bmatrix} \frac{1}{-x_2 + d u} & 0 \\ 0 & \frac{1}{x_1 - x_2/R} \end{bmatrix} P_o^{-1} v_{eq} \quad (37)$$

To be invertible at all times, $N(t, x, u)^{-1}$ is defined as:

$$N^{-1}(1, 1) := \begin{cases} 1/(-x_2 + d u) & \text{if } \|-x_2 + d u\| > \varepsilon_1 \\ 1/\varepsilon_1 & \text{otherwise} \end{cases} \quad (38)$$

$$N^{-1}(2, 2) := \begin{cases} 1/(x_1 - x_2/R) & \text{if } \|x_1 - x_2/R\| > \varepsilon_2 \\ 1/\varepsilon_2 & \text{otherwise} \end{cases} \quad (39)$$

3.4 LMI Methodology for Linear Observer Synthesis

This subsection presents a method for synthesizing an Edwards-Spurgeon sliding mode observer for the linear system of the form (19) and (20). This is done by using Linear Matrix Inequalities (LMIs) as shown in [17]. This approach defines the observer synthesis as a convex optimisation problem. Generally, it defines a positive definite Lyapunov function as $V = e^T P e$ with e being the observer error. The goal then is to find a Lyapunov matrix P such that the differentiation of the Lyapunov function with respect to time $\dot{V} = \dot{e}^T P e + e^T P \dot{e}$ is negative definite. For the sliding mode observer, we define its error system in (26) and subsequently A_o in (25). The following Lemma is now introduced:

Lemma 3 (Tan and Edwards (2002)) [16]

If there exists a gain matrix G_l and a Lyapunov matrix P with the structure:

$$P = \begin{bmatrix} P_1 & P_1 L \\ L^T P_1 & T^T P_o T + L^T P_1 L \end{bmatrix} > 0 \quad (40)$$

where

$$P_1 \in \mathbb{R}^{(n-p) \times (n-p)}$$

that satisfies

$$P A_o + A_o^T P < 0 \quad (41)$$

and $\rho \leq \|P_o C M\| \alpha + \eta_0$, where α and η_0 are positive scalars, then the state estimation error $e(t)$ from (26) is asymptotically stable (see proof in [?]).

For the observer design it is now necessary to find appropriate matrices for L^o , G_l , P_o and P_1 .

To this end, (41) will now be reformulated. Inferring (25) in (41) results in:

$$P(A - G_l C) + (A - G_l C)^T P < 0 \quad (42)$$

By defining:

$$G_l := P^{-1} C^T V^{-1} \quad (43)$$

one obtains for (42):

$$P A + A^T P - C^T V^{-1} C + P W P < 0 \quad (44)$$

$W \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{p \times p}$ are positive symmetric matrices that must be designed by the user. As in LQG control theory, W represents the observers' performance weighting matrix and V denotes the covariance matrix of the system sensor noise.

A solution for P can be obtained by minimizing the trace of P^{-1} subject to (44) and $P > 0$.

For better software implementation, one can apply the Schur complement [?] to

(44), resulting in the equivalent matrix inequality:

$$\begin{bmatrix} PA + A^T P - C^T V^{-1} C & P \\ P & -W^{-1} \end{bmatrix} < 0 \quad (45)$$

Now, let us have an $X \in \mathbb{R}^{n \times n}$ that is positive definite. Then, $X > P^{-1}$ is equivalent to:

$$\begin{bmatrix} -P & I_n \\ I_n & -X \end{bmatrix} > 0 \quad (46)$$

The Lyapunov matrix $P \in \mathbb{R}^{n \times n}$ of (40) can be implemented in software as:

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{bmatrix} > 0 \text{ with } P_{12} = \begin{bmatrix} P_{121} & 0 \end{bmatrix} \quad (47)$$

Hence, to solve the LMI, one can minimize the trace of X with respect to P_{11} , P_{121}, P_{22} and X subject to (45), (46) and (47).

Once the LMI solved, the following matrices can be equated by comparing P in (47) and (40):

$$P_1 = P_{11} \quad (48)$$

$$L^o = P_{11}^{-1} P_{121} \quad (49)$$

$$P_o = T(P_{22} - P_{12}^T P_{11}^{-1} P_{12}) T^T \quad (50)$$

Finally, the observer parameters L^o and P_o can be obtained from (49) and (50), respectively, and G_l and G_n from (62) and (63).

4 Non-Linear Sliding Mode Design

In this section, the linear sliding mode observer design discussed in the previous section will be extended to nonlinear systems.

In general, sliding mode observers can be deployed with nonlinear systems as they are capable of reconstructing system states even in the presence of

nonlinearities within such systems [18].

Therefore, one might ask why expand their design to nonlinear systems? The goal of this paper is not merely state reconstruction but, more importantly, the estimation of parameter variations. As shown in (35) and (37), the parameter variation is captured by the observer's sliding term $v_{eq}(t)$, also known as equivalent output error rejection. In the case of nonlinear systems, however, this sliding term simultaneously captures both the nonlinear plant dynamics and the effects of the model's parameter variation, leading to poor reconstruction performance of the latter.

We therefore present a nonlinear expansion of the sliding mode observer design.

4.1 The Takagi-Sugeno Framework

A useful approach for controller and observer design of non-linear models is provided by the Takagi-Sugeno (TS) framework which was originally introduced in the context of fuzzy systems [19]. Takagi-Sugeno models approximate nonlinear systems by forming a weighted combination of linear submodels. The original non-linear systems can either be derived from analytical models of non-linear systems or from input-output data using system identification techniques [19, 20].

There are various methods for obtaining linear submodels from nonlinear systems. In one approach, Taylor linearization is applied at several user-defined operating points. Another method is the so-called sector nonlinearity approach [21] in which the nonlinearities of a state-space realization are absorbed into the membership functions, leaving the remaining subsystems linear. The sector nonlinearity approach will be used in our application in the following section. As shown in [9] linear matrix inequalities (LMIs) can be applied for the design of stable controller and observer for TS models. This feature will be utilized later, building on the previous section's description of linear sliding mode observer design with LMIs.

The general TS structure of a state-space model is of the form

$$\dot{x} = \sum_{i=1}^N h_i(z) (A_i x + B_i u) \quad (51)$$

$$y = \sum_{i=1}^N h_i(z) C_i x \quad (52)$$

The TS model consists of N linear submodels, and the matrices A_i , B_i and C_i describe the respective i th linear submodel.

Depending on the current operating point, TS models blend the linear submodels by forming their weighted sum as shown in (51) and (52). This is conducted using the so-called premise variable z of dimension j_{max} and the fuzzy convex membership functions h_i . The premise variable z may be dependent on the states x , the input u , as well as some external variables. The following conditions must hold for the membership functions:

$$\sum_{i=1}^N h_i(z) = 1 \quad (53)$$

and

$$0 \leq h_i(z) \leq 1 \quad \forall i \in \mathbb{N} \cap [1, N] \quad (54)$$

For $h_i(z)$ standard fuzzy membership functions like triangular, Gaussian, trapezoidal or sigmoid functions can be used [21].

In our application, we employ triangular membership functions. These functions can be constructed by defining triangular weighting functions $w_{l,j}(z,j)$ at the l -th linearization point z_{0l} . The latter corresponds to a linear submodel of the system. j denotes the dimension of z .

$$w_{j,l}(z_j) = \begin{cases} \frac{z_j - z_{0(l-1)}}{z_{0l} - z_{0(l-1)}} & \text{if } z_{0(l-1)} < z_j \leq z_{0l} \\ 1 - \frac{z_j - z_{0l}}{z_{0(l+1)} - z_{0l}} & \text{if } z_{0l} < z_j \leq z_{0(l+1)} \\ 0 & \text{otherwise} \end{cases} \quad (55)$$

In (55), the first case computes the output value of the weighing function if the premise variable z_j lies between the linearisation point z_{0l} and its previous

linearisation point $z_{0(l-1)}$. The weighing value is computed as a function of the rising side of the triangle, starting with its minimum value of 0 at $z_{0(l-1)}$ (the triangle's left corner) and ending with its maximum value of 1 at z_{0l} (its top corner). The second case depicts the function of the falling side of the triangle giving a maximum output value of 1 at z_{0l} all the way to a minimum value of 0 at the next linearization point $z_{0(l+1)}$ (the triangle's right corner). An example of a set of one-dimensional membership functions is shown in Figure 2 in Section 5.

Incidentally, if the premise variable z only has one dimension ($j_{max}=1$), the weighting function $w_l(z)$ is the equivalent to the membership function h_i as well as their sums: $\sum_{i=1}^N h_i(z) = \sum_{l=1}^{j_{max}} w_l(z)$ and their indices i and l then represent the same linearization point. If, however, z has multiple dimensions, the weighing functions are combined by building their respective product :

$$\sum_{i=1}^N h_i(z) = \prod_{j=1}^{j_{max}} \left(\sum_{l=1}^{l_{j,max}} w_{l,j}(z_j) \right) \quad (56)$$

4.2 The sliding mode observer in TS framework

Similar to the general description of a TS system in (51) and (52), the sliding mode observer can be expressed as a weighted sum of N individual sliding mode observers of the form of (19) and (20):

$$\hat{\dot{x}}(t) = \sum_{i=1}^N h_i(z) (A_i \hat{x}(t) + B_i u(t) - G_{l,i} e_y(t) + G_{n,i} v(t)) \quad (57)$$

$$\hat{y}(t) = \sum_{i=1}^N h_i(z) C_i \hat{x}(t) \quad (58)$$

thus the error system of the TS sliding mode observer becomes

$$\dot{e}(t) = \sum_{i=1}^N h_i(z) (A_{o,i} e(t) + G_{n,i} v(t) - M_i f_v(t, x, u)) \quad (59)$$

with

$$A_{o,i} := A_i - G_{l,i}C_i \quad (60)$$

To design a TS sliding mode observer with a stable error system, the problem can be stated as a convex optimization problem, similar to the linear case. Using the same definitions as in the linear case, one can find a Lyapunov matrix P by minimizing the trace of X with respect to P_{11}, P_{12}, P_{21} and X subject to (46) and (47). Now for the Lyapunov matrix P to ensure stability for all N linear subsystems of the TS model, the minimization of the trace of X must be additionally subject to the following set m of matrix inequalities:

$$m = \left\{ \begin{bmatrix} PA_i + A_i^T P - C_i^T V^{-1} C_i & P \\ P & -W^{-1} \end{bmatrix} < 0, \quad \forall i \in [1, N] \cap \mathbb{N} \right\} \quad (61)$$

The gain matrices of the individual sliding mode observers can then be derived in a manner analogous to the linear case. First, by equating (48), (49) and (50) and then by setting

$$G_{l,i} := P^{-1} C_i^T V^{-1} \quad (62)$$

and

$$G_{n,i} = \begin{bmatrix} -L \\ I_p \end{bmatrix} T^T P_o^{-1} \quad (63)$$

for all $i \in [1, N] \cap \mathbb{N}$.

Note that with this design method the respective gain matrices of individual the sliding mode observers will be identical.

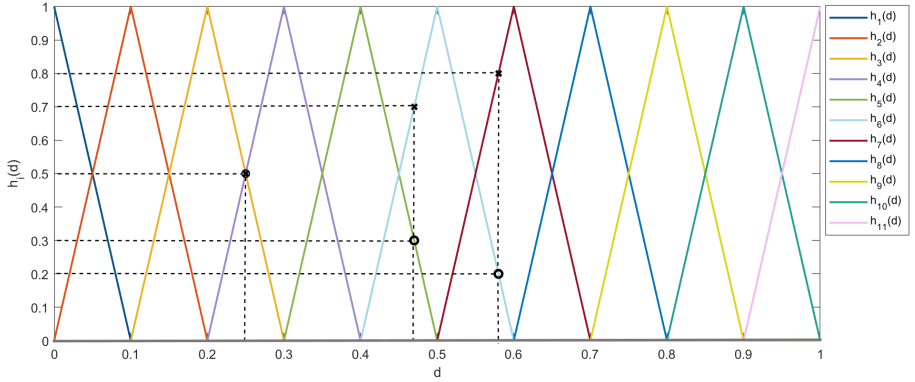


Figure 2: The eleven membership functions for the Takagani-Sugeno model of the buck converter. They are dependent on the premise variable which is the duty cycle d in this case. The plotted points correspond to the non-zero output values of the membership functions for each of the three duty cycle values of the simulation shown in Figure 3. Note that the respective membership functions' output values always add up to one.

4.2.1 Model Deviation Recovery in the TS Framework

By applying the commutative law of summation in (57) and inferring (22)

$$\mathbf{v}_{eq}(t) = - \sum_{i=1}^N h_i(z) \left(\rho \frac{P_{o,i} e_y(t)}{\|P_{o,i} e_y(t)\| + \delta} \right) \quad \text{if } e_y \neq 0 \quad (64)$$

$$\hat{f}_v(t, x, u) := \sum_{i=1}^N h_i(z) \left((M_{2,i}^T M_{2,i})^{-1} M_{2,i}^T T_i^T P_{o,i}^{-1} \mathbf{v}_{eq}(t) \right) \quad (65)$$

As in the linear case, the model error can then be derived from the virtual faults by applying (36).

5 Simulation Results

In this section, we first apply the Takagi-Sugeno framework to a buck converter model of (5) with parameter values shown in Table 1. Then we design a sliding

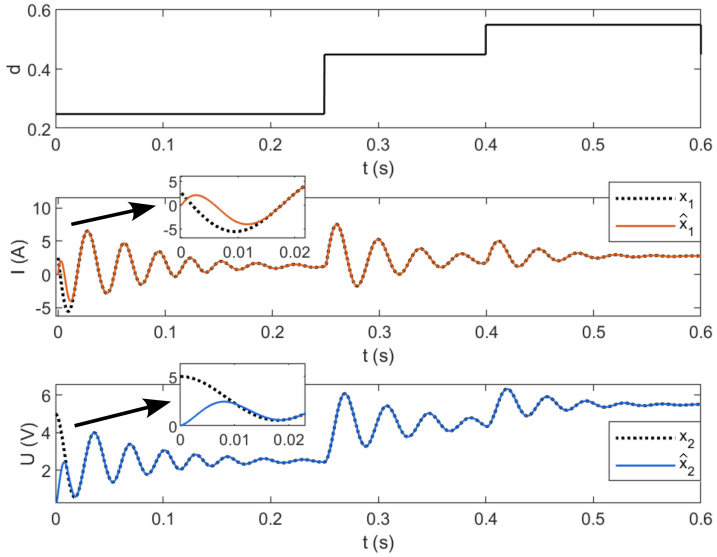


Figure 3: The variable duty cycle d of the simulation is shown in the upper graph, the values of the current i_L (dotted line) are shown in the middle and in the lower graph one can see the values for V_{out} (also dotted line), see the model of the buck converter in Figure 1. In the middle and lower graph one can see how the estimated values of the TS sliding Mode Observer quickly converge to the real state values.

mode observer with regard to the resulting TS model and finally present a simulation to showcase the observers' performance. Note that no controller is used in this setup; instead, we consider an open-loop system.

As already mentioned, we apply the sector nonlinearity approach by choosing the nonlinearities of the model as the premise variable z . In this case we choose the duty cycle, d , of the buck converter to be our premise variable. We take 11 linearisation points with $d \in \{0, 0.1, 0.2 \dots 1\}$. The resulting linear subsystems are listed in the appendix and the membership functions are shown in Figure 2. For this setup, the load R is declared to be constant; however, in another setup, it could also be considered variable, adding a second dimension to the premise variable z .

To design the Sliding Mode Observer the convex optimization with LMIs

Description	Parameter	Value
Inductance	L	1.8 mH
Capacitance	C	17.7 mF
Load	R	2 Ohm

Table 1: The Parameters of the Simulation

described in Section 4.2 is applied.

The design parameters are chosen to be $V = I_2$ and $W = I_2 \times 5 \times 10^5$

The LMIs of (61), (46) and (47) are solved by using the YALMIP toolbox (Lofberg (2004)). The resulting solutions for the Lyapunov matrix is:

$$P = \begin{bmatrix} 0.0001505 & -0.0001207 \\ -0.0001207 & 0.0001447 \end{bmatrix} \quad (66)$$

For all $i \in [1, N] \cap \mathbb{N}$ the following design matrices have been found:

$$P_{o,i} = P \quad (67)$$

and

$$G_{n,i} = G_{l,i} = \begin{bmatrix} 20081 & 16749 \\ 16749 & 20880 \end{bmatrix} \quad (68)$$

with $\text{trace}(X) = 1.8235 \times 10^5$. The variable ρ is set to 1, $\delta = 10^{-5}$ and $\varepsilon_1 = \varepsilon_2 = 10^{-6}$.

In the simulation, the initial state values of the buck converter are set to $x_1(t = 0) = i_L(t = 0) = 2.5A$ and $x_2(t = 0) = V_{out}(t = 0) = 5V$ while both observer states are set to 0. The input state is set to $u = V_{in} = 10V$ throughout the simulation.

The duty cycle d at $t = 0s$ is set to 0.25. At $t = 0.25s$ the duty cycle makes a step to a value of $d = 0.47$ and at $t = 0.4s$ to a value of $d = 0.58$, see Figure 3. For these three values of d , the output values of the membership functions are shown in Figure 2. Figure 3 illustrates that once the simulation commences, the

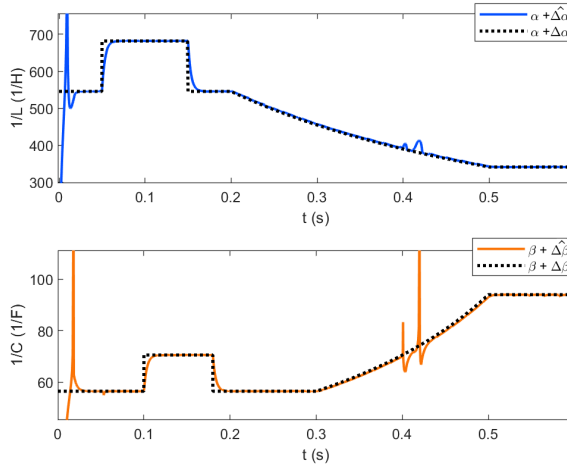


Figure 4: The reconstruction of the parameter change of the buck converter. The parameter change of $\alpha + \Delta\alpha = \frac{1}{L + \Delta L}$ (upper graph) and $\beta + \Delta\beta = \frac{1}{C + \Delta C}$ (lower graph) are depicted with the dotted lines. The reconstruction of the parameter change converges to its real values when the observer states reach the sliding surface.

estimated states rapidly converge to the actual state values. Similar convergence occurs after the two steps in the duty cycle values, but it is barely noticeable due to the high observer gains.

As shown in Figure 4, various steps and ramps are used as parameter fault signals to demonstrate the sliding mode observer's reconstruction ability.

6 Conclusion

A model-based generic approach for online estimation of concentrated electrical parameters in DC-DC converters was presented. This approach employs a sliding mode observer formulated within the Takagi-Sugeno (TS) fuzzy framework to effectively capture and handle the nonlinear dynamics inherent in DC-DC converters during the observer design process and online estimation. It was demonstrated how, by keeping the dynamic observer system on the sliding surface, the time-varying system parameters can be formally treated as unknown

inputs and estimated from input-output measurements. This enables effective parameter estimation even in a large DC-DC operating range. In the future, this will be a key requirement e.g. for DC-DC converters in photovoltaic inverter applications and bidirectional charging of electric car batteries. For example, the voltage load can vary greatly in demanded power point control applications in active distribution networks [8], which is why a large output voltage range (e.g., 4:1) [6] and fine input voltage PV modul control are essential to ensure fast and precise frequency and voltage control. In future work, the method will now be investigated using experimental setups with variable inductances and capacitances with regard to its efficiency and robustness for real-world operation.

References

- [1] H. Schulte, Advanced Control of Grid-Integrated Renewable Energy Power Plants: LMI-based Design in the Takagi-Sugeno Framework, Wiley-IEEE Press, John Wiley & Sons 2024, Hoboken, New Jersey, 2024
- [2] INDEX-WR research project under grant no. 03EE1171B by the German Federal Ministry BMWK
- [3] H. Schulte, M. Resener, Integration of Active Power-controlled Photovoltaic Energy Sources in Distribution Grids by Dynamic Virtual Power Plants, 24th Wind & Solar Integration Workshop, Berlin, 7-10 October 2025
- [4] F. Dietrich, H. Schulte, Sliding Mode Observer Design for Reconstruction of Parameter Change in DC-DC Buck Converters, 6th SysTol Conference, Aiya Napa, Zyprus, 6-8 October 2025
- [5] M. Ettappan, G. Maheswaran, V. Vimala, K. Somasundaram and J. Vishnupriyan, Design of Buck Converter and PV Module for Solar Powered Sailing, Boat, J. Phys.: Conf. Ser. vol 2027, Sep 2021, pp. 012023

- [6] Y. Peng and H. Wang, Parameters Identification of Buck Converter Based on Dynamic Characteristics, 2019 10th International Conference on Power Electronics and ECCE Asia (ICPE 2019 - ECCE Asia), Busan, Korea (South), pp. 3071-3076, 2019
- [7] A. M. AbdelAty, A. Al-Durra, H. Zeineldin and E. F. El-Saadany, Offline Parameter Estimation of a Fractional-Order Buck Converter Model, 2023 IEEE PES Conference on Innovative Smart Grid Technologies - Middle East (ISGT Middle East), Abu Dhabi, United Arab Emirates, pp. 1-5, 2023
- [8] L. Ren, J. Li, M. Jiang, An Online Equivalent Series Resistance Estimation Method for Output Capacitor of Buck Converter Based on Inductor Current Ripple Fitting. *Electronics* 2025, 14, 1037. <https://doi.org/10.3390/electronics14051037>
- [9] V. Michal, Z. Nicolò and A. Matteo, Low-complexity inductance estimation for switched-mode power converters using peak-current mode control, *IET Power Electronics*, vol. 13, pp. 2269-2273, 2020
- [10] H. Givi, E. Farjah and T. Ghanbari, A Comprehensive Monitoring System for Online Fault Diagnosis and Aging Detection of Non-Isolated DC–DC Converters' Components, in *IEEE Transactions on Power Electronics*, vol. 34, no. 7, pp. 6858-6875, July 2019
- [11] M. Bindi, F. Corti, I. Aizenberg, F. Grasso, G.M. Lozito, A. Luchetta, M.C. Piccirilli, A. Reatti, Machine Learning-Based Monitoring of DC-DC Converters in Photovoltaic Applications, *Algorithms* 2022, vol 15, no 74., 2022
- [12] Y. Zhang, X. Cheng, Z. Liu, Y. Zheng, C. Cheng, "Sliding Mode Observer Based Robust Fault Reconstruction for Modular Multilevel Converter with Actuator and Sensor Fault", *IFAC-PapersOnLine*, vol 53, no 2, pp. 13365-13370, 2020
- [13] F. Poeschke, E. Gauterin, H. Schulte: LMI region-based non-linear disturbance observer with application to robust wind turbine control. In: O Boubaker, Q Zhu, M S Mahmoud, J Ragot, H R Karimi, J D'avila,

eds. New Trends in Observer-based Control, p. 35-75. Academic Press.
2019. DOI: 10.1016/B978-0-12-817034-2.00015-0

- [14] S. Georg, Fault diagnosis and fault-tolerant control of wind turbines : nonlinear Takagi-Sugeno and sliding mode techniques, Universität Rostock, 2015, DOI: https://doi.org/10.18453/rosdok_id00001468
- [15] D. Butnicu, D. O. Neacsu and A. Cracan, "A Study of Switching Frequency Impact on Reliability of DC-DC PoL Converters With Discrete Transistors," in IEEE Transactions on Industry Applications, vol. 59, no. 5, pp. 6373-6383, Sept.-Oct. 2023
- [16] C. Tan, C. Edwards, Sliding Mode Observers for Robust Fault Detection & Reconstruction, IFAC Proceedings Volumes, vol. 35, no. 1, pp. 347-352, 2002
- [17] C. Scherer, S. Weiland, Linear Matrix Inequalities in Control. The Control Systems Handbook: Control System Advanced Methods, Second Edition 3, 2011. 1178*, 2011
- [18] V.I. Utkin: Variable structure systems with sliding-modes. IEEE Transactions on Automatic Control 22, 212–222 (1977)
- [19] T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-15, no. 1, pp. 116-132, Jan.-Feb. 1985, doi: 10.1109/TSMC.1985.6313399
- [20] M Sugeno, G.T Kang, Structure identification of fuzzy model, Fuzzy Sets and Systems, Volume 28, Issue 1, 1988, Pages 15-33, ISSN 0165-0114, [https://doi.org/10.1016/0165-0114\(88\)90113-3](https://doi.org/10.1016/0165-0114(88)90113-3)
- [21] Z Lendek, T M Guerra, R Babuska, and B De Schuer. Stability Analysis and Nonlinear Observer Design Using Takagi-Sugeno Fuzzy Models. Springer-Verlag Berlin Heidelberg, 2010
- [22] K. Tanaka, H. Wang, Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach John Wiley & Sons, 2001, Hoboken, New Jersey, 2001

Appendix A

The Linear Submodels of the Simulation Example

$$\forall i \in [1, 11] \cap \mathbb{N}: \quad A_i = \begin{bmatrix} 0 & -545.5866 \\ 56.3985 & -28.1992 \end{bmatrix}$$
$$B_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, B_2 = \begin{bmatrix} 54.5587 \\ 0 \end{bmatrix}, B_3 = \begin{bmatrix} 109.1173 \\ 0 \end{bmatrix}, B_4 = \begin{bmatrix} 163.6760 \\ 0 \end{bmatrix}, B_5 = \begin{bmatrix} 218.2346 \\ 0 \end{bmatrix}$$
$$B_6 = \begin{bmatrix} 272.7933 \\ 0 \end{bmatrix}, B_7 = \begin{bmatrix} 327.3519 \\ 0 \end{bmatrix}, B_8 = \begin{bmatrix} 381.9106 \\ 0 \end{bmatrix}, B_9 = \begin{bmatrix} 436.4693 \\ 0 \end{bmatrix}$$
$$B_{10} = \begin{bmatrix} 491.0279 \\ 0 \end{bmatrix}, B_{11} = \begin{bmatrix} 545.5866 \\ 0 \end{bmatrix}$$

Adaptive Model Reference Control for Multivariable Takagi-Sugeno Fuzzy Systems

Horst Schulte

Control Engineering Group, School of Engineering – Energy and
Information, University of Applied Sciences, HTW Berlin, Germany
schulte@htw-berlin.de

Abstract

This paper presents a novel model reference methodology for multi-input multi-output controller design within the Takagi-Sugeno framework. The principal contribution is the unified representation of both the process and the reference models in Takagi-Sugeno form, thereby enabling a consistent and fully integrated fuzzy-model-based control synthesis for MIMO systems. The reference model may utilize different premise variables than the process model. This flexibility allows the closed-loop dynamics to be shaped based on external or supervisory variables, enabling the design of adaptable control systems that can respond to higher-level decision-making or context-dependent objectives.

1 Introduction

Model Reference Control (MRC) is a widely-used approach for shaping the closed-loop behavior of controlled systems. In previous work, a linear reference model is specified for controller design within the Takagi-Sugeno framework [9]. This approach is particularly effective when nonlinear systems modeled by Takagi-Sugeno systems are forced to behave like an LTI system over the entire large signal range using a complementary T-S controller. The author analyzed this approach and transferred it to renewable generation systems, such as wind turbines [2] and photovoltaic (PV) systems [7], in order to impose linear

behavior with regard to active power feed-in as a function of the measured frequency at the common coupling point. Such linear behavior makes sense if the grid load conditions do not change. In the case of grid congestion, however, adaptive automatic behavior of the distributed generation systems for flexible AC grids would be desirable. Instead of LTI systems, it is in principle possible to use T-S systems as reference models, where the premise variables are set by higher-level event-based controllers.

2 Problem Formulation

This work extends conventional model reference control from linear to nonlinear reference models within the Takagi–Sugeno (T-S) framework and assesses the feasibility of the proposed approach. Following the nomenclature and methodology of the linear reference model presented in [5], the linear model is substituted by a Takagi–Sugeno (T-S) model

$$\dot{x}^r = \sum_{j=1}^{N_j} h_j(z^r) (A_j^r x^r + E_j^r w), \quad y^r = \sum_{j=1}^{N_j} h_j(z^r) (C_j^r x^r + F_j^r w). \quad (1)$$

For further investigation, the plant model¹

$$\dot{x} = \sum_{k=1}^{N_k} h_k(z) (A_k x + B_k u + E_k w), \quad y = \sum_{k=1}^{N_k} h_k(z) (C_k x + F_k w) \quad (2)$$

is used that distinguishes between the controller input $u \in \mathbb{R}^m$ and an external reference signal $w \in \mathbb{R}^p$. For both Takagi–Sugeno models, i.e., the reference model (1) and the plant model (2), it is assumed that the membership functions $h_{\{j,k\}}$ for $j, k = 1, \dots, N_{\{j,k\}}$ satisfy the convex sum property, ensuring their values are nonnegative and sum to one

$$h_j \geq 0, \quad h_k \geq 0, \quad \sum_{j=1}^{N_j} h_j(z) = 1, \quad \sum_{k=1}^{N_k} h_k(z) = 1. \quad (3)$$

¹ T-S Models can be obtained from given nonlinear systems using the Sector Nonlinearity concept [4] or by data-driven methods.

For a complete synthesis in an LMI framework, the sets of membership functions $h_j(z^r)$, $j = 1, \dots, N_j$ and $h_k(z)$, $k = 1, \dots, N_k$ are combined into a common set $h_i(z^r, z)$, $i = 1, \dots, N_i$ utilizing the following sum expression

$$\sum_{i=1}^{N_i} \tilde{h}_i(z^r, z) = \sum_{j=1}^{N_j} h_j(z^r) \sum_{k=1}^{N_k} h_k(z), \quad (4)$$

where $N_i = N_j N_k$. It follows from (3) that the convex sum property also applies to $\tilde{h}_i(z^r, z)$. Without loss of generality, the reference and plant model can now be related to the common set of $\tilde{h}_i$ -functions. Attention should only be drawn to the appropriate mapping of the individual local LTI models to the membership functions.

Analogous to the structure in [5], a suitable control law is specified.

$$u = - \sum_{i=1}^{N_i} \tilde{h}_i(z^r, z) \underbrace{\begin{pmatrix} K_{x,i} & K_{x^r,i} & -K_{I,i} \end{pmatrix}}_{\bar{K}_i} \underbrace{\begin{pmatrix} x \\ x^r \\ x_I \end{pmatrix}}_{\bar{x}} \quad (5)$$

with the auxiliary state vector

$$x_I := \int_0^t (y^r(\tau) - y(\tau)) d\tau = \int_0^t \varepsilon(\tau) d\tau, \quad (6)$$

where y^r denotes the reference value and $\varepsilon = y^r - y$ the matching error.

An augmented plant model for controller synthesis is defined based on the extended state vector $\bar{x}$ as in (5) and matching error $\varepsilon = \dot{x}_I = y^r - y$:

$$\begin{aligned} \dot{\bar{x}} &= \sum_{i=1}^{N_i} \tilde{h}_i(z^r, z) \left[\underbrace{\begin{pmatrix} A_i & 0 & 0 \\ 0 & A_i^r & 0 \\ -C_i & C_i^r & 0 \end{pmatrix}}_{\bar{A}_i} \bar{x} + \underbrace{\begin{pmatrix} B_i \\ 0 \\ 0 \end{pmatrix}}_{\bar{B}_i} u + \underbrace{\begin{pmatrix} E_i \\ E_i^r \\ F_i^r - F_i \end{pmatrix}}_{\bar{E}_i} w \right], \\ \varepsilon &= \underbrace{\begin{pmatrix} -C_i & C_i^r & 0 \end{pmatrix}}_{\bar{C}_i} \bar{x} + \underbrace{\begin{pmatrix} F_i^r - F_i \end{pmatrix}}_{\bar{F}_i} w, \end{aligned} \quad (7)$$

which results directly from the combination of (1), (2), and (6).

3 LMI criteria for Controller Synthesis

Linear Matrix Inequality (LMI) criteria [1] are widely used in controller synthesis for Takagi-Sugeno systems to ensure stability and performance of fuzzy-model-based controllers [3, 9]. The approach involves representing nonlinear systems as convex combinations of linear models, each associated with a membership function, and then formulating control design conditions as LMIs. With the control objective of minimizing the output error $\varepsilon = y^r - y$, the control problem can be cast as an H_∞ optimization task. This approach not only ensures asymptotic stability of the closed-loop system but also provides a systematic criterion for the calculation of the controller gain matrices $\bar{K}_i$. Formally, the goal is to design a controller that minimizes the induced H_∞ norm γ from the exogenous input (reference signal) w to the matching error ε of (7):

$$\begin{aligned} & \text{minimize} && \gamma \\ & \text{subject to} && \sup_{\|w\|_2 \neq 0} \frac{\|\varepsilon\|_2}{\|w\|_2} \leq \gamma, \end{aligned} \quad (8)$$

where $\|\cdot\|_2$ denotes the L_2 -norm.

The following LMI formulation, stemming from the bounded real lemma, enables the design of stabilizing controllers by solving a convex optimization problem that minimizes the H_∞ norm of the closed-loop transfer function while guaranteeing internal stability and robust performance.

$$\Gamma_{ij} = \begin{pmatrix} \bar{A}_i X + X \bar{A}_i^T - \bar{B}_i \bar{M}_j - \bar{M}_j^T \bar{B}_i^T & \bar{E}_i & X \bar{C}_i^T \\ \bar{E}_i^T & -\gamma^2 I & \bar{F}_i^T \\ \bar{C}_i X & \bar{F}_i & -I \end{pmatrix}, \quad X \succ 0 \quad (9)$$

where $X = X^T$ with the relaxed condition

$$\begin{aligned} & \Gamma_{ij}(X, M_j) + \Gamma_{ji}(X, M_i) \prec 0, \\ & \Gamma_{ii}(X, M_i) \prec 0, \text{ for all } i = 1, 2, \dots, N_i, \quad j = i+1, i+2, \dots, N_i. \\ & \text{s.t. } \tilde{h}_i(z) \tilde{h}_j(z) \neq 0, \exists \end{aligned}$$

Note that, to formulate the LMI problem (8) for controller synthesis, the auxiliary term

$$\bar{M}_j = \bar{K}_j X, \quad \bar{M}_j^T = X^T \bar{K}_j = X \bar{K}_j \quad (10)$$

is used. Once a solution has been found, the gain matrices $\bar{K}_i$ are determined for the control law (5) via

$$\bar{K}_j = \bar{M}_j X^{-1}, \quad j = 1, \dots, N_i. \quad (11)$$

4 Conclusion

A design method was presented within the Takagi-Sugeno framework to synthesize model reference controllers, in which the dynamics of the reference model can be adjusted via external signals. This capability arises because the premise variable z' , which serves as an external control input, can be set independently from the premise variable z of the plant model. The actual controller design is done by solving a convex linear matrix inequality. All necessary mathematical expressions are specified in the paper. Further investigations will explore the suitability of the proposed method for the aforementioned applications, specifically in the field of power systems and, in particular, active distribution networks [8].

References

- [1] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, Philadelphia, PA, USA: SIAM, 1994.
- [2] J. Brunner, H. Schulte, *Takagi-Sugeno based Model Reference Control for Wind Turbine Systems in Frequency Containment Scenarios*, (accepted for publication), In: at - Automatisierungstechnik, 2025.

- [3] G. Feng, *A Survey on Analysis and Design of Model-Based Fuzzy Control Systems*, IEEE Transaction on Fuzzy Systems, vol. 14, no. 5, October 2006.
- [4] H. Ohtake, K. Tanaka, and H. O. Wang. *Fuzzy Modeling via Sector Nonlinearity Concept*. In Joint 9th IFSA World Congress and 20th NAFIPS International Conference, pp. 127-132, Vancouver, Canada, 2001.
- [5] H. Schulte, Comparison between Pole Region Design and Model Reference Control for Multivariable TS Fuzzy Systems. In: Proceedings 32. Workshop Computational Intelligence. Karlsruhe, Germany: KIT Scientific Publishing, pp. 257-266, 2022.
- [6] H. Schulte, *Advanced Control of Grid-Integrated Renewable Energy Power Plants*, LMI-based Design in the Takagi-Sugeno Framework, Wiley-IEEE Press, John Wiley & Sons 2024, Hoboken, New Jersey, 2024
- [7] H. Schulte, Takagi-Sugeno Model Reference Control of Photovoltaic Power Plants to Provide Instantaneous Reserve. In: Proceedings of IEEE International Conference on Fuzzy Systems 2025 (FUZZ-IEEE), Reims, France, July 2025
- [8] H. Schulte, M. Resener, Integration of Active Power-controlled Photovoltaic Energy Sources in Distribution Grids by Dynamic Virtual Power Plants, 24th Wind & Solar Integration Workshop, Berlin, 7-10 October 2025
- [9] K. Tanaka and H.O. Wang. *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. John Wiley & Sons, Inc, 2001.

Space-Filling Regularization for Modeling Hydraulic Excavator with Nonlinear State Space Models

Jonas Powell, Hermann Klein, Oliver Nelles

Universität Siegen, Department Maschinenbau,
Institut für Mechanik und Regelungstechnik – Mechatronik
Paul-Bonatz-Str. 9-11, 57068, Siegen, Germany
E-Mail: {jonas.powell, hermann.klein, oliver.nelles}@uni-siegen.de

1 Introduction

Front-attachment assistance functions for hydraulic excavators support operators in tasks such as leveling and angled grading. The front attachment is actuated by three hydraulic cylinders, each controlled by its corresponding joystick. The performance of the assistance functions is critically dependent on the accuracy of the underlying dynamic models, which are utilized for control in a feed-forward and feed-back manner. Each joystick input affects the velocity of its corresponding cylinder; the assistance function aims to modify these inputs to enable precise motion along prescribed trajectories.

Nonlinear state space models are a powerful class of representation of dynamical systems. The Local Model State Space Network (LMSSN) [2] is an effective way to identify these. Its main advantages are local interpretability and explainability grounded in the well-known foundations of linear system theory. Recent advances in space-filling regularization demonstrate improvements in model accuracy and robustness, as well as reduced training time [5]. This is because the regularization ensures a better data coverage of the state space during training.

This work investigates the application of space-filling regularization to modeling the dynamics of a hydraulic excavator. We evaluate both the regularization

strength and the number of states with respect to test error, model complexity, and training time.

2 Excavator Setup

The test platform is a JCB Hydradig 110W equipped with a dual-variable-displacement pump driven by a diesel engine. The hydraulic system is load-sensing with load-pressure-independent flow distribution and a maximum working pressure of 280 bar [4]. For the assistance functions, we focus on the front attachment, which comprises three cylinders (boom, arm, bucket). Cylinder velocities are commanded by joystick displacements. Each cylinder can be actuated individually via a dedicated joystick (Figure 1).

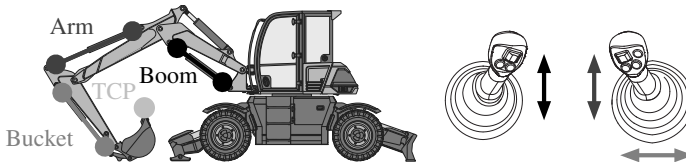


Figure 1: Schematic of the excavator front attachment showing the cylinders of interest (boom, arm, bucket; left) and the corresponding joysticks (right). Adapted from [1].

Three classes of data were collected for system identification and model testing: amplitude-modulated pseudo-random binary sequence (APRBS), multisine, and leveling profiles (Figure 2). The leveling signal is less dynamically challenging, but it is useful for evaluating application-relevant performance.

For training, we combine all signal types to cover the target operating regime while ensuring broad excitation of the system dynamics. For testing, the multisine and leveling sets are evaluated separately to assess (i) generalization to unseen data and (ii) performance in practical application scenarios. An overview of the datasets is given in Table 1.

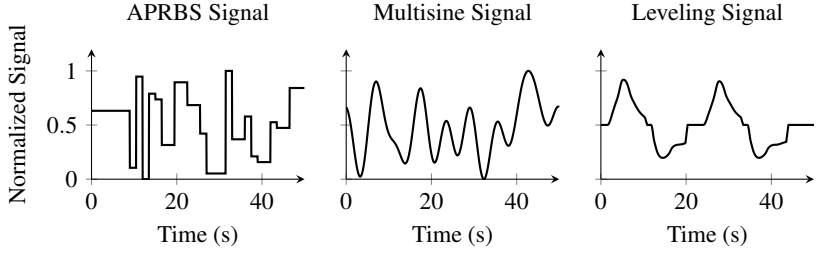


Figure 2: Comparison of the three signal types used for system identification: APRBS (left), multisine (middle), and leveling (right).

	Signal type	Samples	Duration
Training	APRBS & multisine & leveling	35 000	≈ 29 min
Validation	Multisine	10 000	≈ 8 min
Test (multisine)	Multisine	10 000	≈ 8 min
Test (leveling)	Leveling	38 000	≈ 32 min

Table 1: Datasets used for training, validation, and testing.

3 Nonlinear State Space Modeling with Space-Filling Regularization

We consider nonlinear discrete-time state space models whose state evolves within a known admissible set $\mathcal{X} \subset \mathbb{R}^{n_x}$.

The Local Model State Space Network (LMSSN) [2] represents the global dynamics by blending local affine models with normalized radial basis function (NRBF) validity weights. Concretely,

$$\hat{\underline{x}}(k+1) = \sum_{j=1}^{n_{\text{LM},x}} \Phi_j^{[x]}(k) \left(\underline{A}_j \hat{\underline{x}}(k) + \underline{B}_j \underline{u}(k) + \underline{o}_j \right), \quad (1)$$

$$\hat{\underline{y}}(k) = \sum_{j=1}^{n_{\text{LM},y}} \Phi_j^{[y]}(k) \left(\underline{C}_j \hat{\underline{x}}(k) + \underline{D}_j \underline{u}(k) + \underline{p}_j \right), \quad (2)$$

where $\underline{A}_j \in \mathbb{R}^{n_x \times n_x}$, $\underline{B}_j \in \mathbb{R}^{n_x \times n_u}$, $\underline{C}_j \in \mathbb{R}^{n_y \times n_x}$, $\underline{D}_j \in \mathbb{R}^{n_y \times n_u}$, $\underline{a}_j \in \mathbb{R}^{n_x}$, and $\underline{p}_j \in \mathbb{R}^{n_y}$ are the parameters of the j -th local affine model. The basis functions $\Phi_j^{[x]}(k)$ and $\Phi_j^{[y]}(k)$ satisfy $\sum_j \Phi_j^{[x]}(k) = 1$ and express the j -th local validity function. Local models are constructed using the Local Linear Model Tree (LOLIMOT) algorithm, which iteratively splits the worst local model into two submodels (see [3]). The number of local models is selected based on validation performance.

Parameter estimation proceeds by minimizing a closed-loop output-error objective J_{data} . To encourage uniform coverage of the admissible state space during training, we augment the objective with a space-filling regularizer [5]. Let $\{\underline{g}_j\}_{j=1}^{n_g} \subset \mathbb{R}^{n_x}$ be a uniformly distributed set of support points in the state space. Then the mean minimum distance between the predicted states $\hat{\underline{x}}(k; \underline{\theta})$ and the support points is defined as

$$\psi(\underline{\theta}) = \frac{1}{n_g} \sum_{j=1}^{n_g} \min_{1 \leq k \leq N} \left\| \underline{g}_j - \hat{\underline{x}}(k; \underline{\theta}) \right\|_2. \quad (3)$$

This regularizer is added to the objective function as a weighted penalty term, yielding

$$J(\underline{\theta}) = J_{\text{data}}(\underline{\theta}) + \lambda \psi(\underline{\theta}), \quad \lambda \geq 0, \quad (4)$$

where the parameter λ balances data fit and state space coverage. This regularization biases training toward trajectories that populate $\mathbb{R}^{n_x}$ uniformly, thereby improving robustness and stabilizing extrapolation without altering the LMSSN architecture.

4 Results

Hydraulic cylinders are commonly modeled as well-damped second-order systems [6]. However, leveling is a quasi-static task, i.e., dynamic effects are not dominant. We therefore consider two alternative state dimensions, $n_x \in \{3, 6\}$. The input and output dimensions are $n_u = n_y = 3$.

We first study the influence of the regularization strength λ (see Fig. 3). When increasing λ , the test error initially does not change significantly, but after

a certain point the test error increases. This can be observed for both state dimensions. The sum of the multisine and leveling test errors is minimized at $\lambda = 0.5$ for $n_x = 3$ and at $\lambda = 0.05$ for $n_x = 6$.

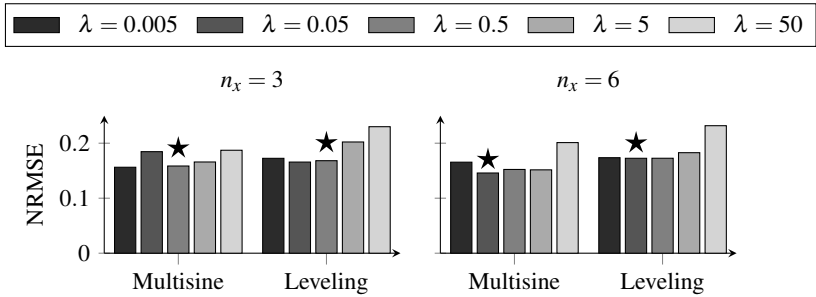


Figure 3: NRMSE on the test sets for different regularization strengths λ with $n_x = 3$ (left) and $n_x = 6$ (right). The star ★ markers denote the selected best model (lowest combined test error).

Compared with their unregularized counterparts, the regularized models achieve substantially lower test errors and shorter training times (Table 2). For the unregularized $n_x = 3$ model, training was terminated after 10 splits due to unstable gradients that prevented further progress. For the unregularized $n_x = 6$ model, training was stopped before the first split because the optimizer failed to converge after 480 h of computation. The higher test errors of the unregularized models are thus attributable to the lack of convergence to satisfactory solutions. The regularized models all converged reliably to a satisfactory solution within reasonable time. This is because the regularization ensures that data is disturbed more uniformly in the state space, which allows each local model to have sufficiently many active data points, and therefore stable gradients and a more reliable optimization.

Between the regularized models, the one with $n_x = 6$ yields an 1.3 % lower NRMSE on the multisine test set compared to $n_x = 3$, but an 0.4 % higher NRMSE on the leveling test set. This indicates that quasi-static leveling can be modeled sufficiently well with three states, whereas more dynamic tasks (e.g., the multisine experiment) benefit from a higher state dimension.

In the unregularized case, increasing the model order from $n_x = 3$ to $n_x = 6$ results in a significant increase in computational effort by a factor of ≈ 2.07 .

Regularization	n_x	λ	n_{LM}	Training	NRMSE Test	
				Time [h]	Multisine	Leveling
Unregularized	3	0	10	480+	0.1928	0.2341
Regularized	3	0.5	24	40.50	0.1585	0.1682
Unregularized	6	0	1	480+	0.3224	0.4738
Regularized	6	0.05	14	83.85	0.1458	0.1726

Table 2: Summary of the best models for $n_x \in \{3, 6\}$ with and without space-filling regularization.

This underlines the importance of regularization for higher-order models. It can also be observed that the regularized models allow for a higher number of local models, which improves the ability to capture nonlinear behavior. This may be attributed to the more uniform state space coverage, which mitigates the risk of encountering unstable gradients and convergence issues.

5 Conclusion

This work demonstrates that space-filling regularization improves nonlinear, data-driven state space models of hydraulic excavator dynamics. We evaluate two different state dimensions, $n_x \in \{3, 6\}$, across a range of regularization strengths and compare the best regularized models with unregularized baselines. Models trained with space-filling regularization have multiple advantages. They (1) show significantly faster training times, (2) allow for higher-order models that better capture dynamic behavior, and (3) more reliably converge to satisfactory solutions by avoiding unstable gradients and allowing for more splits.

Acknowledgments

We thank Dr. Benjamin Hartmann, Dr. Ozan Demir (both researchers at Robert Bosch GmbH) for providing the excavator measurements.

References

- [1] JCB Deutschland GmbH. *HYDRADIG 110W Online Brochure*. 2024. Available at: www.jcb.com/dfsmedia/261086efe15a46f5afb%95d093ef038ea/63726-source Accessed: 2024-12-10.
- [2] Max Schüssler. *Machine Learning with Nonlinear State Space Models*. PhD thesis, Universität Siegen, July 2022.
- [3] Oliver Nelles. *Nonlinear System Identification Results*. Springer International Publishing, Cham, 2020.
- [4] Ozan Demir et al. Learning Based Feed-forward Control for Advanced Excavator Assistance Functions. In: *Proceedings of the 13th International Fluid Power Conference (13. IFK)*, Aachen, Germany, March 21–23, 2022.
- [5] Hermann Klein et al. Space-Filling Regularization for Robust and Interpretable Nonlinear State Space Models. *arXiv*, 2025. doi: 10.48550/arXiv.2507.07792.
- [6] Dongik Sun et al. Lever Control for Position Control of a Typical Excavator in Joint Space Using a Time Delay Control Method. *Journal of Intelligent & Robotic Systems*, 102(3):63, July 2021.

Direct Model Reference Adaptive Control – Practical Examples with Evolving Controller

Goran Andonovski, Igor Škrjanc

University of Ljubljana, Faculty of Electrical Engineering
Tržaška cesta 25, Ljubljana, Slovenia

E-Mail: goran.andonovski@fe.uni-lj.si, igor.skrjanc@fe.uni-lj.si

Summary

We present a Direct Model Reference Adaptive Control (DMRAC) scheme that integrates an evolving fuzzy controller with nonparametric, data–cloud antecedents (RECCo). The controller grows its rule base online via a novelty test in a normalized cloud space and updates local PID–R parameters by an error-driven law with dead zone, projection, and leakage for robustness. Range-based rules size the adaptation gains directly from input/output limits and the reference model, yielding consistent adaptation across operating points without plant identification. A post-learning pruning step removes weakly supported clouds to improve parsimony. In a nonlinear DC motor case study with dead-zone, current saturation, and friction, both “adding only” and “adding + pruning” variants achieve comparable tracking; pruning yields a smaller rule base with slightly lower squared-error metrics and reduced saturation exposure.

1 Introduction

Real-world processes are rarely stationary: operating points drift, disturbances vary, and dynamics change over time. Fixed-structure controllers often underperform in such settings unless they are repeatedly retuned. Evolving fuzzy and neuro-fuzzy methods address this challenge by learning from streaming data while keeping models compact and auditable. Early neuro-fuzzy formulations showed how fuzzy rules with learnable consequents can approximate nonlinear

dynamics and enable adaptive control [1–3]. Subsequent work accelerated the move from fixed membership functions to data-driven structures and online adaptation mechanisms [4–9].

A central idea is to replace predefined antecedent memberships with nonparametric, data-driven constructs (e.g., clusters, data clouds, Gaussian, etc.), so that structure grows only when novelty is detected and otherwise remains sparse and interpretable [10–14]. To sustain performance under drift and noise, evolving frameworks combine structural operators (redundancy elimination, split–merge, and rule splitting) with procedures that preserve interpretability and enhance robustness of parameter learning [5, 15, 16]. Streaming-aware components such as active learning and uncertainty sampling prioritize informative samples, while self-adjusting thresholds stabilize operation in nonstationary regimes [17, 18].

Recent advances extend evolving fuzzy systems toward data-efficient identification and federated learning. An evolving neuro-fuzzy design of experiments was proposed to actively generate informative inputs for nonlinear process identification [19], later enhanced with fuzzy confidence intervals to guide safe online experimentation under uncertainty [20]. In parallel, evolving Gaussian clustering has been adapted for decentralized settings, first with eFedGauss for federated clustering [21], then with an unsupervised scheme for discovering shared structures across silos [22]. Most recently, evolving Gaussian systems have been formulated as a general framework for federated regression, combining interpretability with adaptability to nonstationary data [23].

Evolving fuzzy control has progressed from theoretical adaptation laws [24] and evolving reference controllers [25–28] to practical plug-and-play deployments. This includes robust evolving cloud-based control (RECCo) [29, 30], extensions to process industries such as heat exchangers and surge tanks [31, 32], and applications to nonlinear plants such as overhead cranes [33]. The focus is not only on stability and robustness but also on interpretability and reduced operator burden during online adaptation [5, 34]. In summary, evolving control reduces reliance on global plant models while retaining an auditable, rule-based structure and enabling straightforward deployment.

2 Background and Related Work

One of the first neuro-fuzzy approaches applied to control was presented in [1], where an Adaptive-Network-based Fuzzy Inference System (ANFIS) was trained via backpropagation/hybrid learning to synthesize fuzzy controllers. Despite the term *adaptive*, classic ANFIS adapts parameters in an epoch-based, batch manner on fixed datasets and assumes a preset rule structure. It neither performs incremental, streaming updates nor supports online structural evolution. Hence, in the sense adopted in this paper—adaptation from data streams with evolving structure—ANFIS is not an *evolving* controller. In [2], the Fuzzy Adaptive Learning Control Network (FALCON) was proposed as a connectionist structure for fuzzy logic controllers; two online learning algorithms, FALCON-FSM and FALCON-ART, dynamically construct and optimize fuzzy rules, membership functions, and partitions. Flexibility and scalability are achieved via neural learning, but the design remains largely heuristic, lacking closed-loop guarantees and explicit constraint handling (e.g., bounded inputs, actuator saturation) and showing sensitivity to structural hyperparameters, such as ART vigilance. The absence of a state-space formulation further hinders integration with modern robust/optimal synthesis and complicates the treatment of drift.

An evolving controller based on the Self-Organizing Neural Fuzzy Inference Network (SONFIN) was proposed in [3], enabling flexible rule creation, on-line learning, and simultaneous structural and parametric optimization. Rule consequents are adapted via Least Mean Squares (LMS) or Recursive Least Squares (RLS), while antecedent memberships are tuned using backpropagation. A related advancement in [4] introduced an online methodology for adaptive control based on an evolving Takagi–Sugeno fuzzy model, where antecedents evolve through online clustering and consequents are estimated using RLS.

Subsequent research focused on simplified fuzzy models with nonparametric antecedents based on data clouds (the AnYa model) [35]. One of the earliest controllers using AnYa was presented in [36], where the self-evolving, parameter-free fuzzy rule-based controller (SPARC) autonomously learned control actions online without predefined rules, memberships, or explicit plant models. Later, self-evolving controllers adopting AnYa antecedents with PID- and MRC-type

consequents were introduced in [37,38], with evolving mechanisms driven by local and global cloud densities.

The first RECCo (Robust Evolving Cloud-based Controller) appeared in [38]. An improved version was presented in [39], including a simulation study on two processes. The adaptive law for rule consequents employed gradient learning, while the evolving law relied on local density and a threshold parameter; RECCo can be classified as a direct model reference adaptive control approach (see Fig. 1). Practical implementations followed in [31,40], introducing a normalized data space and an improved adaptation law that enabled more stable initial settings and more reliable parameter updates. Testing on the same plant as in [37] demonstrated significantly more stable and robust adaptation [41]. Further developments and applications are reported in [30,42–44].

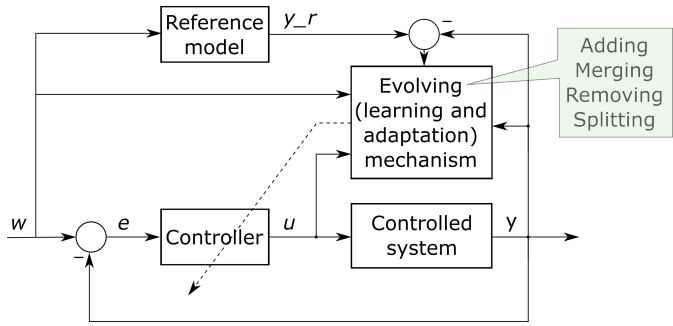


Figure 1: Schematic representation of a direct (model reference) adaptive control scheme with an evolving mechanism.

In [45], an Online Self-Evolving Fuzzy Controller (OSEFC) was introduced, comprising a parameter-learning (PL) block and a self-evolving (SE) block. The PL block integrates local and global learning, while the SE topology selects the input variable with the largest Index of Responsibility to receive a new membership function; the new function's position is set by the center of gravity of that variable's error distribution, after which a new rule is initialized. A modification based on a Taylor Series Neuro-Fuzzy model, termed OSENF-TaSe, was proposed in [46].

For benchmarking, interval-model-based PID design provides a robust fixed-structure baseline [47], while data-driven MPC illustrates trade-offs when

accurate models and solvers are available [48]. Overall, evolving fuzzy control positions itself between classical MRAC/PID and optimization-based MPC: it preserves interpretability, reduces model dependence, and sustains performance under nonstationarity via online structural adaptation [5, 29, 30].

3 Methodology

The proposed controller is the Robust Evolving Cloud-based Controller, which extends the model reference adaptive control concept by embedding evolving fuzzy structures and robust adaptation mechanisms. RECCo belongs to the ANYA family of fuzzy rule-based systems with nonparametric antecedents, where the membership of incoming data samples is defined through data clouds formed directly from observations. The controller evolves its structure online and adapts PID-type parameters without requiring an explicit model of the plant.

3.1 Controller structure

RECCo consists of three main components: a reference model, an evolving law, and an adaptation law. Each incoming sample is associated with the most relevant data cloud, the parameters of the corresponding fuzzy rule are updated, and new rules are introduced if novelty is detected. The global control action is obtained via weighted aggregation of the local PID rules.

3.2 Reference model

The desired closed-loop dynamics are defined by a discrete-time first-order reference model,

$$y_{k+1}^r = a_r y_k^r + (1 - a_r) r_k, \quad 0 < a_r < 1, \quad (1)$$

where r_k is the reference signal and a_r determines the desired response speed. The control objective is to minimize the tracking error,

$$\varepsilon_k = y_k^r - y_k. \quad (2)$$

3.3 Evolving law

The antecedent part of RECCo is based on nonparametric data clouds. For a data sample $\mathbf{x}_k$, the local density with respect to the i -th cloud is

$$\gamma_k^i = \frac{1}{1 + \frac{\sum_{j=1}^{M^i} \|\mathbf{x}_k - \mathbf{x}_j^i\|^2}{M^i}}, \quad (3)$$

where M^i is the number of samples in the cloud. The normalized association is

$$\lambda_k^i = \frac{\gamma_k^i}{\sum_{j=1}^c \gamma_k^j}, \quad i = 1, \dots, c. \quad (4)$$

3.3.1 Adding mechanism

If $\max_i \gamma_k^i < \gamma_{\max}$, the sample is considered novel and a new cloud is added. Due to the normalized cloud space, the novelty threshold is fixed at $\gamma_{\max} = 0.93$, independent of the process. This ensures that the evolving structure can flexibly capture new regions of the input–output space as soon as the existing clouds fail to represent them adequately.

3.3.2 Post-learning pruning mechanism

Finish pruning. After the learning phase, an additional *post-learning pruning* step is performed to remove clouds that accumulated little data overall. Let

$$M_{\text{tot}} = \sum_{i=1}^c M^i$$

be the total number of assignments across all clouds. A pruning threshold M_{thr} is defined as

$$M_{\text{thr}} = \max(m_{\text{count}}, \lfloor \alpha M_{\text{tot}} \rfloor),$$

where α is the minimal fraction of the overall data that a cloud must capture, and m_{count} is a user-defined absolute minimum count. In our experiments we used $\alpha = 0.02$ (i.e., 2% of all assignments) and $m_{\text{count}} = 1$.

All clouds with $M^i \leq M_{\text{thr}}$ are considered for removal. To preserve stability, at least k_{keep} of the most active clouds are always retained, regardless of their size. We set $k_{\text{keep}} = 2$. If all clouds fall below the threshold, the single largest cloud (in terms of M^i) is kept.

This final pruning step ensures that only informative and well-populated clouds remain for the testing and deployment phases, thereby improving controller robustness and interpretability.

3.4 Adaptation law

Each cloud i maintains PID parameters $\theta_k^i = [P_k^i, I_k^i, D_k^i, R_k^i]^\top$. Parameters are updated only for the active cloud according to

$$\theta_k^i = \theta_{k-1}^i + \Delta \theta_k^i, \quad (5)$$

with increments

$$\Delta P_k^i = \alpha_P G_{\text{sign}} \lambda_k^i \frac{|e_k \varepsilon_k|}{1 + r_k^2}, \quad (6)$$

$$\Delta I_k^i = \alpha_I G_{\text{sign}} \lambda_k^i \frac{|e_k \Delta_k^\varepsilon|}{1 + r_k^2}, \quad (7)$$

$$\Delta D_k^i = \alpha_D G_{\text{sign}} \lambda_k^i \frac{|e_k \Delta_k^\varepsilon|}{1 + r_k^2}, \quad (8)$$

$$\Delta R_k^i = \alpha_R G_{\text{sign}} \lambda_k^i \frac{\varepsilon_k}{1 + r_k^2}, \quad (9)$$

where $e_k = r_k - y_k$ is the control error, ε_k is the tracking error, Δ_k^ε its discrete derivative, and G_{sign} is the known process gain sign.

Local and global control signals. The local control law for rule i is

$$u_k^i = P_k^i \varepsilon_k + I_k^i \Sigma_k^\varepsilon + D_k^i \Delta_k^\varepsilon + R_k^i, \quad (10)$$

with Σ_k^ε and Δ_k^ε denoting the integral and derivative of ε_k . The final control signal is

$$u_k = u_{\min} + \frac{\sum_{i=1}^c \gamma_k^i u_k^i}{\sum_{i=1}^c \gamma_k^i}. \quad (11)$$

Range-based selection of adaptation gains. To make the adaptation speed consistent across operating points and processes, we size the adaptation gains $\alpha = \{\alpha_P, \alpha_I, \alpha_D, \alpha_R\}$ directly from the input/output ranges and the reference model:

$$\begin{aligned} U_{\text{rng}} &:= U_{\max} - U_{\min}, & Y_{\text{sc}} &:= \frac{1}{2} (Y_{\max} - Y_{\min}), \\ a_r &:= 1 - \frac{T_s}{\tau}, & \Delta Y_{\text{sc}} &:= (1 - a_r) Y_{\text{sc}}. \end{aligned}$$

Let $e_{\text{sc}} \approx Y_{\text{sc}}$, $\varepsilon_{\text{sc}} \approx Y_{\text{sc}}$, and $\Delta_{\text{sc}}^\varepsilon \approx \Delta Y_{\text{sc}}$ denote “large but plausible” magnitudes of $e_k, \varepsilon_k, \Delta_k^\varepsilon$. Choose a small fraction $\rho \in [0.01, 0.05]$ representing the maximum portion of the input range that a *single* parameter update is allowed to induce. Then set

$$\alpha_P = \rho \frac{U_{\text{rng}}}{e_{\text{sc}} \varepsilon_{\text{sc}}}, \quad \alpha_I = \rho \frac{U_{\text{rng}}}{e_{\text{sc}} \Delta_{\text{sc}}^\varepsilon}, \quad \alpha_D = \rho \frac{U_{\text{rng}}}{e_{\text{sc}} \Delta_{\text{sc}}^\varepsilon}, \quad \alpha_R = \rho \frac{U_{\text{rng}}}{\varepsilon_{\text{sc}}}. \quad (12)$$

With this choice, if $|e_k| \leq e_{\text{sc}}$, $|\varepsilon_k| \leq \varepsilon_{\text{sc}}$, and $|\Delta_k^\varepsilon| \leq \Delta_{\text{sc}}^\varepsilon$, then the one-step change in u_k caused by each parameter channel is bounded by ρU_{rng} . Because $\Delta_{\text{sc}}^\varepsilon \ll \varepsilon_{\text{sc}}$ for slow references ($\tau \gg T_s$), (12) naturally yields larger α_I, α_D than α_P ; the resulting *control* move remains balanced once multiplied by their respective regressors.

3.5 Robustness mechanisms

Several mechanisms guarantee safe adaptation. A dead zone suppresses parameter updates when $|\varepsilon_k|$ is below a specified threshold d_{dead} . Projection confines the parameters within predefined bounds $\underline{\theta} \leq \theta_k^i \leq \bar{\theta}$. Leakage prevents parameter

drift by applying a forgetting factor, and adaptation is interrupted whenever u_k exceeds actuator limits.

3.6 Normalized cloud space

To ensure transferability across processes, the cloud space is normalized:

$$\mathbf{x}_k = \begin{bmatrix} \varepsilon_k / \Delta \varepsilon \\ (y_k^r - r_{\min}) / \Delta r \end{bmatrix}, \quad (13)$$

where $\Delta r = r_{\max} - r_{\min}$ and $\Delta \varepsilon = \Delta r / 2$.

Algorithm 1 summarizes crucial steps of RECCo controller. It begins with no rules, constructs clouds online, adapts PID parameters locally, and employs robustness mechanisms. The normalized design ensures plug-and-play deployment across nonlinear and time-varying processes.

4 Nonlinear DC Motor Control Problem

4.1 Process description

We consider a DC motor with armature dynamics and shaft mechanics. The manipulated variable is the armature voltage $u(k) \in \mathbb{R}$ and the measured output is the angular speed $y(k) = \omega(k)$ in rad/s. The plant exhibits several relevant nonlinearities for control design: (i) voltage dead-zone around zero, (ii) armature-current saturation, and (iii) friction modeled as viscous plus smooth Coulomb friction. Additive sensor noise is assumed at the output.

Algorithm 1.: RECCo Control Algorithm

Voraussetzung: T_s : sampling period; $u_{\min}, u_{\max}$: actuator limits; $r_{\min}, r_{\max}$: reference range; $\gamma_{\max}$: novelty threshold; $\alpha_P, \alpha_I, \alpha_D, \alpha_R$: adaptation gains; $d_{\text{dead}}, \sigma_L$: robustness parameters; $\underline{\theta}, \bar{\theta}$: parameter bounds; $\alpha_{\text{fp}}, m_{\text{count}}, k_{\text{keep}}$: finish-pruning parameters

```
1: function RECCO CONTROLLER
2:   initialize rule count  $c \leftarrow 0$ ; reference state  $y_r^0$ 

3:   for  $k = 0, 1, 2, \dots$  do                                 $\triangleright$  Main control loop (every  $T_s$  seconds)
4:     Measure plant output  $y_k$ ; read reference  $r_k$ 
5:     Update reference model  $y_k^r$  using (1)
6:      $\varepsilon_k \leftarrow y_k^r - y_k$                                  $\triangleright$  Model-tracking error
7:      $e_k \leftarrow r_k - y_k$                                  $\triangleright$  Reference-tracking error
8:      $\mathbf{x}_k \leftarrow [\varepsilon_k / \Delta \varepsilon, (y_k^r - r_{\min}) / \Delta r]^\top$   $\triangleright$  Normalized feature
9:     if  $c = 0$  then
10:       Create first cloud centered at  $\mathbf{x}_k$ ;  $c \leftarrow 1$ 
11:       Initialize PID parameters  $\theta^1$ 
12:     else
13:       Compute local densities  $\gamma_k^i$  for  $i = 1, \dots, c$ 
14:       if  $\max_i \gamma_k^i < \gamma_{\max}$  then                             $\triangleright$  Novelty test
15:         Create new cloud at  $\mathbf{x}_k$ ;  $c \leftarrow c + 1$ ; initialize  $\theta^c$ 
16:          $i^* \leftarrow \arg \max_i \gamma_k^i$                              $\triangleright$  Winning cloud
17:         Update cloud statistics (e.g.,  $M^{i^*}$ ) with  $\mathbf{x}_k$ 
18:         Compute normalized associations  $\lambda_k^i \leftarrow \gamma_k^i / \sum_{j=1}^c \gamma_k^j$ 
19:         Adaptation law: Update  $\theta_k^{i^*}$  using (5) and apply robustness checks.
20:         Compute local control outputs  $u_k^i$  via (10)
21:         Aggregate final control  $u_k$  via (11) using  $\{\lambda_k^i, u_k^i\}$ 
22:         Saturate  $u_k$  to  $[u_{\min}, u_{\max}]$  if necessary
23:         Send  $u_k$  to the plant

                                      $\triangleright$  Post-learning finish pruning
24:       Compute  $M_{\text{tot}} \leftarrow \sum_{i=1}^c M^i$ 
25:        $M_{\text{thr}} \leftarrow \max(m_{\text{count}}, \lfloor \alpha_{\text{fp}} M_{\text{tot}} \rfloor)$ 
26:        $\mathcal{S} \leftarrow \{i \in \{1, \dots, c\} : M^i \leq M_{\text{thr}}\}$          $\triangleright$  Prune candidates
27:       Let  $\mathcal{K}$  be indices of the  $k_{\text{keep}}$  clouds with largest  $M^i$ 
28:       Remove all clouds in  $\mathcal{S} \setminus \mathcal{K}$  and keep  $\mathcal{K}$ 
29:       Shrink parameter matrix  $\Theta$  to rows corresponding to kept clouds
```

4.2 Continuous-time model

Let $i(t)$ denote the armature current and $\omega(t)$ the shaft speed. The nominal (unconstrained) dynamics are

$$L \dot{i}(t) = u_{\text{eff}}(t) - R i(t) - K_e \omega(t), \quad (14)$$

$$J \dot{\omega}(t) = K_t i(t) - b \omega(t) - \tau_c \tanh\left(\frac{\omega(t)}{\omega_\epsilon}\right) - \tau_{\text{load}}(t), \quad (15)$$

with standard motor parameters: resistance $R > 0$, inductance $L > 0$, back-EMF constant $K_e > 0$, torque constant $K_t > 0$, inertia $J > 0$, viscous friction $b \geq 0$, and Coulomb friction magnitude $\tau_c \geq 0$ smoothed by $\tanh(\cdot)$ with $\omega_\epsilon > 0$.

Nonlinear input and state constraints. A voltage dead-zone of width $V_{\text{dead}} > 0$ is modeled by

$$u_{\text{eff}}(t) = \begin{cases} 0, & |u(t)| \leq V_{\text{dead}}, \\ u(t) - V_{\text{dead}} \operatorname{sgn}(u(t)), & |u(t)| > V_{\text{dead}}. \end{cases} \quad (16)$$

Armature current saturation is enforced as $|i(t)| \leq i_{\text{max}}$. A (possibly unknown) load torque $\tau_{\text{load}}(t)$ can be used to inject disturbances.

4.3 Discretization and measurement model

Using an Euler forward scheme with integration step $d_t > 0$, we obtain

$$i(k+1) = \operatorname{sat}_{[-i_{\text{max}}, i_{\text{max}}]} \left[i(k) + d_t \frac{u_{\text{eff}}(k) - R i(k) - K_e \omega(k)}{L} \right], \quad (17)$$

$$\omega(k+1) = \omega(k) + d_t \frac{K_t i(k) - b \omega(k) - \tau_c \tanh\left(\frac{\omega(k)}{\omega_\epsilon}\right) - \tau_{\text{load}}(k)}{J}. \quad (18)$$

The measured output is

$$y(k) = \omega(k) + v(k), \quad v(k) \sim \mathcal{N}(0, \sigma_y^2). \quad (19)$$

4.4 Operating ranges and sampling

The admissible input and output domains are

$$u(k) \in [U_{\min}, U_{\max}], \quad y(k) \in [Y_{\min}, Y_{\max}], \quad (20)$$

with symmetric voltage limits allowing reversal: $U_{\min} = -12 \text{ V}$, $U_{\max} = 12 \text{ V}$, and a wide speed window $Y_{\min} = -300 \text{ rad/s}$, $Y_{\max} = 300 \text{ rad/s}$. The control sampling time is T_s , while the internal integration step d_t in (17)–(18) can be chosen as a submultiple of T_s if needed.

In the experimental setup and simulations we use the following parameters (scaled for numerically well-conditioned simulations):

$$\begin{aligned} R &= 0.5 \, \Omega, \, L = 0.2 \, \text{H}, \, K_e = 0.02, \, K_t = 0.45, \, J = 0.01 \, \text{kg m}^2, \\ b &= 0.015 \, \text{Nms/rad}, \, \tau_c = 0.02 \, \text{Nm}, \, \omega_\varepsilon = 1.0 \, \text{rad/s}, \, V_{\text{dead}} = 1.0 \, \text{V}, \\ i_{\text{max}} &= 10.0 \, \text{A}, \, T_s = 1 \, \text{s}, \, d_t = 0.02 \, \text{s}, \, \sigma_y = 0.02 \, \text{rad/s}. \end{aligned}$$

These values yield visible dead-zone effects, saturating current transients, and friction-induced nonlinearity around $\omega \approx 0$, which are useful to benchmark adaptive/evolving controllers. The open-loop characteristic of the process is plotted in Fig. 2, where the nonlinearities of the process are clearly visible.

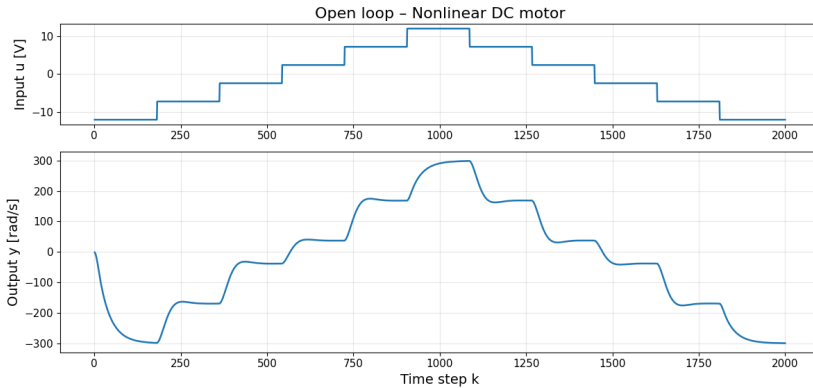


Figure 2: Open-loop experiment.

5 Experimental results

In this section we evaluate RECCo on a randomized step-tracking experiment over the full operating range. The controller starts from scratch and evolves during the learning phase by *adding* new data clouds (rules) whenever the local-density test is not satisfied. We then compare two variants in the subsequent frozen-structure test phase: (*Experiment 1*) using only the adding mechanism, and (*Experiment 2*) using adding together with a post-learning *pruning* of weak/rarely-used clouds.¹ The number of data clouds generated during learning for the two evolving strategies is summarized in Fig. 3. The reference is a staircase of uniformly sampled random setpoints; after learning, structure growth is disabled and only the parameters continue adapting under the usual protection mechanisms (dead zone, projection, leakage). Performance is assessed over $K = 2000$ samples using standard tracking and actuation-smoothness criteria.

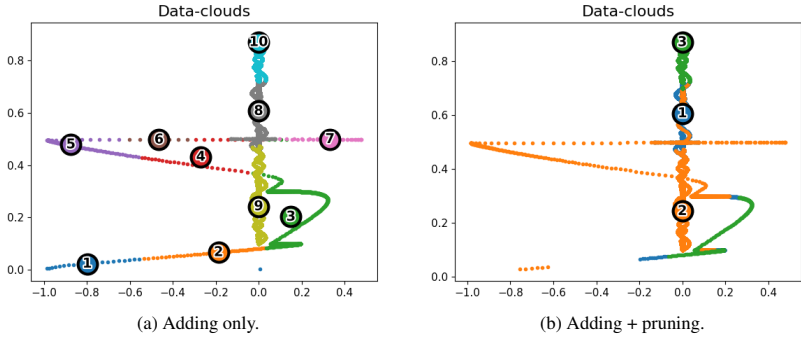


Figure 3: Number of data clouds generated during learning for the two evolving strategies.

Closed-loop results for the testing phase are reported in Figs. 4 and 5. Figure 4 shows Experiment 1, where only the cloud-adding mechanism is active, while Fig. 5 illustrates Experiment 2, where post-learning pruning is additionally applied. Each figure provides the reference trajectory, model reference, and plant output (top panel), the control input $u(k)$ (middle panel), the tracking error $\varepsilon(k)$ (third panel), and the cloud activations (bottom panel).

¹ Pruning is applied once at the end of learning to eliminate clouds with insufficient support while preserving the most informative ones.

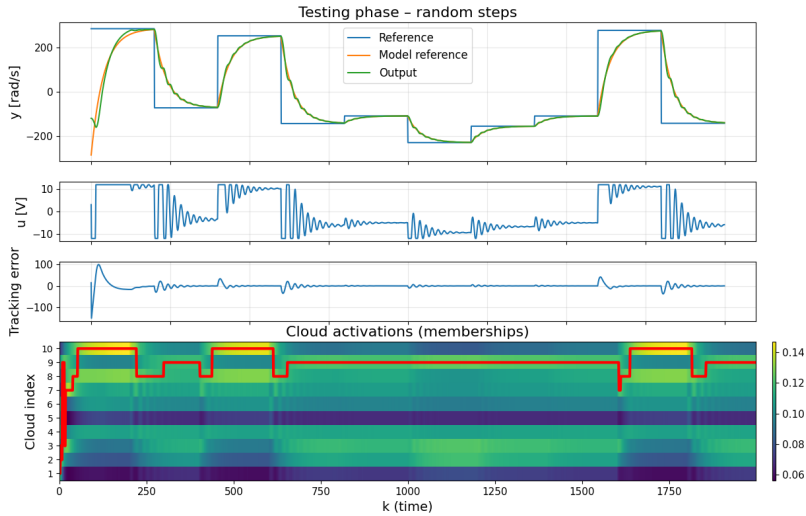


Figure 4: (*Experiment 1: Adding mechanism*) Closed-loop control performance during the testing phase with random step references when only cloud *adding* is enabled. Top: reference signal, model reference, and system output. Middle: control signal $u(k)$. Next: tracking error $\varepsilon(k)$. Bottom: cloud activations (memberships) with the winning cloud trajectory shown in red.

A quantitative comparison of both strategies is given in Table 1, which summarizes the tracking and control-effort metrics computed over the $K = 2000$ random steps. The table refers directly to the results in Figs. 4 and 5.

Discussion. Both variants achieve similar tracking quality; differences are marginal. Light post-learning pruning slightly improves squared-error metrics (ISE, RMSE, NRMSE) and reduces actuator saturation, while the “adding only” variant yields marginally smoother moves (Δu RMS, total variation). This is consistent with pruning away small/rarely-activated clouds that can introduce occasional sharp blends, while retaining enough local structure to preserve accuracy. Overall, the removal step offers a minor accuracy gain with essentially unchanged closed-loop behavior, and can be enabled when model parsimony or lower saturation exposure is desired.

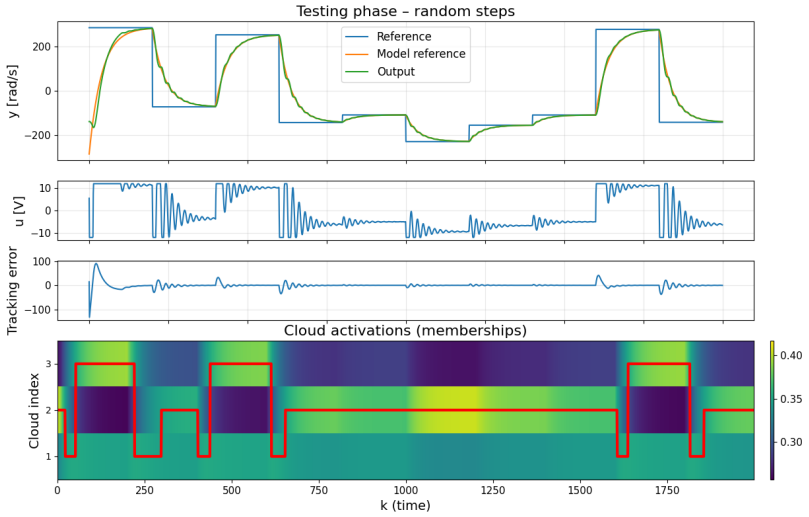


Figure 5: (*Experiment 2: Adding + post-learning pruning*) Closed-loop control performance during the testing phase with random step references when *adding* and *pruning* is enabled. Top: reference signal, model reference, and system output. Middle: control signal $u(k)$. Next: tracking error $\varepsilon(k)$. Bottom: cloud activations (memberships) with the winning cloud trajectory shown in red.

6 Conclusion

This paper presented the Robust Evolving Cloud-based Controller (RECCo) as a Direct Model Reference Adaptive Control (DMRAC) framework that integrates evolving fuzzy structures with robust adaptation mechanisms. The controller begins without predefined rules and incrementally builds its structure online through the addition of data clouds, while adaptation laws update local PID-R parameters within robust bounds. A post-learning pruning step was introduced to remove rarely used or weakly supported clouds, thereby simplifying the rule base without compromising performance.

The nonlinear DC motor case study demonstrated that RECCo can successfully handle nonstationary and nonlinear dynamics using default design parameters, without extensive tuning. Both variants—adding only, and adding with pruning—achieved comparable tracking performance, with pruning yielding

Table 1: Performance over the random-steps experiment (Figs. 4–5). The “Goal” column indicates whether higher (↑) or lower (↓) is better.

Metric	Goal	Just Adding	With Pruning
<i>Tracking performance</i>			
IAE	↓	1.058×10^5	1.058×10^5
ISE	↓	2.155×10^7	2.150×10^7
ITAE [s]	↓	8.532×10^7	8.541×10^7
RMSE	↓	103.8	103.7
NRMSE (vs span)	↓	0.173	0.1728
Tracking score [0..1]	↑	0.827	0.8272
<i>Control effort / smoothness</i>			
u RMS	↓	8.338	8.330
Δu RMS	↓	1.343	1.407
Total variation (u)	↓	1004	1039
Avg. jitter $ \Delta u $	↓	0.502	0.5195
Saturation ratio [%]	↓	16.85	15.55

a slightly more compact structure and marginal improvements in certain error and actuation metrics. At the same time, the adaptation law ensured bounded transients and reliable operation under actuator limits.

In summary, RECCo provides a practical, self-organizing control solution that combines the interpretability of fuzzy systems with the adaptability of data-driven methods. Its ability to evolve online and maintain robustness across operating regimes makes it suitable for processes characterized by nonlinearities and time-varying behavior.

References

- [1] J.-S. R. Jang. Self-learning fuzzy controllers based on temporal backpropagation. *IEEE Transactions on Neural Networks*, 3(5):714–723, 1992.

- [2] C. T. Lin, C. J. Lin, and C. S. G. Lee. Fuzzy adaptive learning control network with on-line neural learning. *Fuzzy Sets and Systems*, 71(1):25–45, 1995.
- [3] C. F. Juang and C. T. Lin. An on-line self-constructing neural fuzzy inference network and its applications. *IEEE Transactions on Fuzzy Systems*, 6(1):12–32, 1998.
- [4] P. Angelov. A fuzzy controller with evolving structure. *Information Sciences*, 161(1-2):21–35, 2004.
- [5] I. Škrjanc, J. A. Iglesias, A. Sanchis, D. Leite, E. Lughofer, and F. A. C. Gomide. Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey. *Information Sciences*, 490:344–368, 2019.
- [6] P. Angelov. Evolving Takagi–Sugeno fuzzy systems from streaming data (eTS+). In *Evolving Intelligent Systems: Methodology and Applications*, pages 21–50, 2010.
- [7] D. Leite, I. Škrjanc, and F. Gomide. An overview on evolving systems and learning from stream data. *Evolving Systems*, 11(2):181–198, 2020.
- [8] S. Blažič and A. Zdešar. An implementation of an evolving fuzzy controller. *IEEE Conference on Evolving and Adaptive Intelligent Systems*, 2017-May, 2017.
- [9] D. Leite, G. Andonovski, I. Škrjanc, and F. Gomide. Optimal rule-based granular systems from data streams. *IEEE Transactions on Fuzzy Systems*, 28(3):583–596, 2020.
- [10] P. Angelov and R. Yager. A new type of simplified fuzzy rule-based system (AnYa). *International Journal of General Systems*, 41:163–185, 2012.
- [11] G. Andonovski, G. Mušič, S. Blažič, and I. Škrjanc. Evolving model identification for process monitoring and prediction of non-linear systems. *Engineering Applications of Artificial Intelligence*, 68:214–221, 2018.
- [12] I. Škrjanc. Cluster-volume-based merging approach for incrementally evolving fuzzy gaussian clustering (egauss+). *IEEE Transactions on Fuzzy Systems*, 28:2222–2231, 2020.

- [13] I. Škrjanc. An evolving concept in the identification of an interval fuzzy model of wiener–hammerstein nonlinear dynamic systems. *Information Sciences*, 581:73–87, 2021.
- [14] D. Leite and F. Gomide. Evolving Linguistic Fuzzy Models from Data Streams. In Enric Trillas, Piero P. Bonissone, Luis Magdalena, and Janusz Kacprzyk, editors, *Combining Experimentation and Theory - Studies in Fuzziness and Soft Computing*, volume 271, pages 209–223. Springer, Berlin, Heidelberg, 2012.
- [15] E. Lughofer. On-line assurance of interpretability criteria in evolving fuzzy systems – achievements, new concepts and open issues. *Information Sciences*, 251:22–46, 2013.
- [16] E. Lughofer. Improving the robustness of recursive consequent parameters learning in evolving neuro-fuzzy systems. *Information Sciences*, 545:555–574, 2021.
- [17] D. Ge and X. J. Zeng. Learning data streams online — an evolving fuzzy system approach with self-learning/adaptive thresholds. *Information Sciences*, 507:172–184, 2020.
- [18] D. Leite and I. Škrjanc. Ensemble of evolving optimal granular experts, OWA aggregation, and time series prediction. *Information Sciences*, 504:95–112, 2019.
- [19] M. Ozbot, E. Lughofer, and I. Škrjanc. Evolving neuro-fuzzy systems-based design of experiments in process identification. *IEEE Transactions on Fuzzy Systems*, 31(6):1995–2005, 2023.
- [20] M. Ozbot and I. Škrjanc. Online design of experiments with fuzzy confidence interval for identification of nonlinear dynamical processes. In *IEEE Conference on Evolving and Adaptive Intelligent Systems*, pages 1–8, 2024.
- [21] M. Ožbot, S. Ozawa, and I. Škrjanc. eFedGauss: A federated approach to fuzzy multivariate Gaussian clustering. In *IEEE International Conference on Fuzzy Systems*, pages 1–8, 2024.

- [22] M. Ožbot, S. Ozawa, and I. Škrjanc. Unsupervised federated learning based on evolving Gaussian clustering. In *2025 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1–6, 2025.
- [23] M. Ožbot, P. V. C. Souza, and I. Škrjanc. Evolving Gaussian systems as a framework for federated regression problems. *IEEE Transactions on Fuzzy Systems*, pages 1–11, 2025.
- [24] G. Andonovski, S. Blažič, P. Angelov, and I. Škrjanc. Analysis of Adaptation Law of the Robust Evolving Cloud-based Controller. In *2015 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–7, 2015.
- [25] D. Dovžan, S. Blažič, and I. Škrjanc. Towards evolving fuzzy reference controller. In *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, 2014.
- [26] Ž. Stržinar and I. Škrjanc. Self-tuned Model-Based Predictive Control Using Evolving Fuzzy Model of a Non-linear Dynamic Process. In J. Rayz, V. Raskin, S. Dick, and V. Kreinovich, editors, *Lecture Notes in Networks and Systems*, volume 258, pages 406–421, Cham, 2022. Springer.
- [27] A. Zdešar, D. Dovžan, and I. Škrjanc. Self-tuning of 2 DOF control based on evolving fuzzy model. *Applied Soft Computing Journal*, 19:403–418, 2014.
- [28] A. Zdešar, D. Dovžan, and I. Škrjanc. A 2 DOF predictive control based on evolving fuzzy model. *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2014 - Conference Proceedings*, pages 1–6, 2014.
- [29] G. Andonovski, P. Angelov, S. Blazic, and I. Škrjanc. Robust Evolving Cloud-based Controller (RECCo). In *IEEE Conference on Evolving and Adaptive Intelligent Systems*, volume 2017-May, 2017.
- [30] G. Andonovski, B. S. J. Costa, S. Blažič, and I. Škrjanc. Robust evolving controller for simulated surge tank and for real two-tank plant. *at - Automatisierungstechnik*, 66(9):725–734, 2018.

- [31] G. Andonovski, P. Angelov, S. Blažič, and I. Škrjanc. A practical implementation of Robust Evolving Cloud-based Controller with normalized data space for heat-exchanger plant. *Applied Soft Computing*, 48:29–38, 2016.
- [32] O. Cartagena, M. Ožbot, D. Sáez, and I. Škrjanc. Evolving fuzzy prediction interval for fault detection in a heat exchanger. *Applied Soft Computing*, 145:110625, 2023.
- [33] C. Aguiar, D. Leite, D. Pereira, G. Andonovski, and I. Škrjanc. Nonlinear modeling and robust LMI fuzzy control of overhead crane systems. *Journal of the Franklin Institute*, 358(2):1376–1402, 2021.
- [34] S. Blažič, I. Škrjanc, and D. Matko. A robust fuzzy adaptive law for evolving control systems. *Evolving Systems*, 5(1):3–10, 2014.
- [35] P. Angelov and R. Yager. Simplified fuzzy rule-based systems using nonparametric antecedents and relative data density. In *IEEE SSCI 2011 - Workshop on Evolving and Adaptive Intelligent Systems*, pages 62–69, 2011.
- [36] P. Sadeghi-Tehran, A. B. Cara, P. Angelov, H. Pomares, I. Rojas, and A. Prieto. Self-evolving parameter-free rule-based controller. In *IEEE International Conference on Fuzzy Systems*, pages 10–15, 2012.
- [37] B. Costa, I. Škrjanc, S. Blažič, and P. Angelov. A practical implementation of self-evolving cloud-based control of a pilot plant. In *2013 IEEE International Conference on Cybernetics (CYBCONF)*, pages 7–12, 2013.
- [38] P. Angelov, I. Škrjanc, and S. Blažič. Robust evolving cloud-based controller for a hydraulic plant. In *Proceedings of the 2013 IEEE Conference on Evolving and Adaptive Intelligent Systems*, pages 1–8, 2013.
- [39] I. Škrjanc, S. Blažič, and P. Angelov. Robust evolving cloud-based PID control adjusted by gradient learning method. In *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, 2014.
- [40] G. Andonovski, A. Bayas, and S. Doris. Robust Evolving Cloud-based Control for the Distributed Solar Collector Field. In *2016 IEEE*

International Conference on Fuzzy Systems (FUZZ-IEEE), pages 1570–1577, 2016.

- [41] G. Andonovski and B. S. J. Costa. Robust evolving control of a two-tanks pilot plant. In *IEEE Conference on Evolving and Adaptive Intelligent Systems*, volume 2017-May, 2017.
- [42] J. V. Anand and P. S. Manoharan. Decentralized robust evolving cloud-based controller for two input two output systems. *Transactions of the Institute of Measurement and Control*, 44(5):1056–1069, 2022.
- [43] O. Lamraoui and H. Habbi. Experimental investigation of evolving cloud-based fuzzy control of a pilot thermal exchanger under a decentralized framework. *Applied Soft Computing*, 148:110883, 2023.
- [44] S. Chellamuthu Kalaimani and V. Jeyakumar. Hardware design for blood glucose control based on the Sorensen diabetic patient model using a robust evolving cloud-based controller. *Computer Methods in Biomechanics and Biomedical Engineering*, 0(0):1–22, 2023.
- [45] A. B. Cara, H. Pomares, I. Rojas, Z. Lendek, and R. Babuška. Online self-evolving fuzzy controller with global learning capabilities. *Evolving Systems*, 1(4):225–239, 2010.
- [46] A. B. Cara, L. J. Herrera, H. Pomares, and I. Rojas. New online self-evolving neuro fuzzy controller based on the TaSe-NF model. *Information Sciences*, 220:226–243, 2013.
- [47] G. Karer and I. Škrjanc. Interval-model-based global optimization framework for robust stability and performance of PID controllers. *Applied Soft Computing*, 40:526–543, 2016.
- [48] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer. Linear tracking MPC for nonlinear systems – part II: The data-driven case. *IEEE Transactions on Automatic Control*, 67:4406–4421, 2022.

Zur Multi-Start-Optimierung mit vorzeitiger Terminierung zur effizienten Lösung nichtlinearer Regressionsprobleme mittels MLP-Netzen

Konrad Lange, Andreas Kroll

Fachgebiet Mess- und Regelungstechnik, Universität Kassel

Mönchebergstr. 7, 34125 Kassel, Germany

E-Mail: {konrad.lange, andreas.kroll}@mrt.uni-kassel.de

1 Einführung

Die datengetriebene Approximation statischer nichtlinearer Systeme führt bei verschiedenen Modellarten zu nichtlinearen Optimierungsproblemen. Multi-Layer-Perceptron-(MLP-)Netze mit mindestens einer verdeckten Schicht und ausreichend vielen Neuronen mit nichtlinearer Aktivierungsfunktion können beliebige stetige Funktionen auf kompakten Definitionsmengen beliebig genau approximieren [1]. Optimierungsalgorithmen quasi-zweiter Ordnung, wie Levenberg-Marquardt (LM) [2, Kap. 10.3] oder Broyden-Fletcher-Goldfarb-Shanno (BFGS) [2, Kap. 6.1] konvergieren bei der Bestimmung der MLP-Modellparameter nach vergleichsweise wenigen Epochen zu lokalen Minima.

Bei Multi-Start-(MS-)Strategien wird eine Vielzahl von Modellkandidaten mit zufälligen Startwerten im Parameterraum initialisiert, um die Chance auf ein global besseres Ergebnis zu erhöhen [3]. Da die vollständige iterative Optimierung aller Kandidaten einen erheblichen Rechenaufwand erfordert, können Methoden zur MS-Optimierung das Training von Kandidaten möglichst früh abbrechen, wenn eine unzureichende finale Güte zu erwarten ist. Die Terminierung von MS-Modellkandidaten mit geringer erwarteter Modellgüte wird beispielsweise zur effizienten Hyperparameteroptimierung untersucht [4,5]. Zur vorzeitigen Terminierung bei der Optimierung der MLP-Modellparameter

zur Lösung von nichtlinearen Regressionsproblemen mit LM oder BFGS fehlen in der Literatur ausführliche Analysen. Ziel dieses Beitrags ist die Untersuchung modellfreier Terminierungsstrategien.

2 Methoden

2.1 Multi-Start-Optimierungsverfahren

Als Kostenfunktion f der Modellparameteroptimierung des MLP-Netzes soll der mittlere quadratische Fehler (MSE):

$$f(\boldsymbol{\theta}) = \text{MSE}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^N (r_i(\mathbf{x}_i, \boldsymbol{\theta}))^2 = \frac{1}{N} \sum_{i=1}^N (y_i(\mathbf{x}_i) - \hat{y}_i(\mathbf{x}_i, \boldsymbol{\theta}))^2 \quad (1)$$

mit den Residuen r_i verwendet werden. Dabei sind y_i die wahren Zielwerte der N Trainingsdaten $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ und $\hat{y}_i(\boldsymbol{\theta})$ die Prädiktionen des Modells für die m Eingaben $\mathbf{x}_i = [x_{i,1}, \dots, x_{i,m}]^T$, abhängig von den Modellparametern $\boldsymbol{\theta}$. Aufgrund der nichtlinearen Aktivierungsfunktionen der Neuronen ist das Optimierungsproblem zur Schätzung der Modellparameter nichtlinear in Bezug auf die Parameter. Somit ergibt sich ein nichtlineares (mittlere) Kleinste-Quadrate-Problem, das iterativ gelöst werden muss.

In diesem Beitrag werden das LM- und das BFGS-Verfahren betrachtet, die für Netze kleiner bis mittlerer Größe als State-of-the-Art gelten [6]. Die zwei Optimierungsverfahren quasi-zweiter Ordnung nutzen die Krümmungsinformationen der Zielfunktion zur effizienten Konvergenz, indem die Hesse-Matrix approximiert wird. Bei dem LM-Verfahren wird dabei zwischen Gauss-Newton-Algorithmus und Gradientenabstieg interpoliert. Dadurch ist das LM-Verfahren robuster als der klassische Gauss-Newton-Algorithmus und konvergiert in der Regel in deutlich weniger Schritten als Methoden erster Ordnung. Die Approximation der Hesse-Matrix $\mathbf{H}$ mit der Jacobi-Matrix $\mathbf{J}$ erfolgt dabei nach:

$$\mathbf{H} \approx \mathbf{J}^T \mathbf{J} + \lambda \mathbf{I} \quad \text{mit} \quad \mathbf{J}(\mathbf{x}) = \left[\frac{\partial r_i(\mathbf{x}_i)}{\partial x_j} \right]_{\substack{i=1, \dots, N \\ j=1, \dots, m}}. \quad (2)$$

Für Probleme mit vielen Trainingsdaten und vielen Modellparametern wird $\mathbf{J}$ sehr groß und die Berechnung teuer. Die Optimierung mit LM wird deshalb für Probleme mit wenigen hundert Modellparametern aufgrund des Vorteils der schnellen Konvergenz eingesetzt [7]. Das BFGS-Verfahren ist ein Quasi-Newton-Verfahren, das die Approximation der Inversen der Hesse-Matrix $\mathbf{H}^{-1}$ iterativ bestimmt, ohne die Jacobi-Matrix explizit zu berechnen. Ein Vorteil von BFGS liegt in der Verwendbarkeit für beliebige glatte, skalare Zielfunktionen, während LM die Struktur nichtlinearer Kleinste-Quadrate-Probleme nutzt und nur für diese verwendbar ist.

Die Kostenfunktion bei der Optimierung der Modellparameter eines MLP-Netzes ist im Allgemeinen nicht konvex, sodass mehrere lokale Minima existieren können. Dennoch können viele dieser lokalen Minima gute Approximationen des globalen Optimums darstellen. Durch die breitere Exploration des Parameterraums bei der MS-Optimierung kann diese Approximation verbessert werden.

2.2 Vorzeitige Terminierung

Bei der Optimierung von Modellparametern wird eine vorzeitige Terminierung (Early Stopping) als Mittel verwendet, um Overfitting zu vermeiden [8]. Overfitting kann typischerweise in einer späteren Phase des Trainings auftreten, hierbei verbessert sich die Modellgüte auf dem Trainingsdatensatz weiterhin, während die Generalisierungsfähigkeit abnimmt und die Modellevaluation auf unabhängigen Daten schlechtere Ergebnisse liefert. Um dies zu verhindern, wird in der Regel die Modellgüte auf einem Validierungsdatensatz ausgewertet und das Training gestoppt, sobald diese wieder abnimmt.

Strategien zur vorzeitigen Terminierung mit dem Ziel der Effizienzsteigerung werden in der Literatur in verschiedenen Bereichen untersucht. Ein Hauptanwendungsgebiet liegt dabei in der Hyperparameteroptimierung von Modellen [5, 9, 10] und Algorithmen [4, 11].

Guo und Fu untersuchen Strategien zur Ressourcen-Allokation zu vielversprechenden Kandidaten bei der Multi-Start-Initialisierung des Gradientenabstiegsverfahrens für globale Optimierungsprobleme [12].

In [11] werden Methoden mit dem Ziel einer effizienteren automatischen Konfiguration der Hyperparameter von Optimierungsalgorithmen untersucht. Dabei werden Hüllkurven aus vorherigen Durchführungen berechnet, um Abbruchbedingungen für neue Ausführungen festzulegen. In verschiedenen Optimierungsszenarien konnten die vorgeschlagenen Methoden im Durchschnitt eine Einsparung von 20 % bis 40 % des Konfigurationsaufwands bei Ermittlung einer Konfiguration gleicher Qualität erzielt werden.

Ebenfalls im Rahmen der Hyperparameteroptimierung untersuchen Marinov und Karapetyan in [13] die lineare Regression der Modellgüte zu einer Epoche m zur finalen Modellgüte, um aus einem Subset an vollständigen Trainings zu bestimmen, welche Kandidaten im Training der gesamten Kandidatenmenge nach m Epochen abgebrochen werden sollen.

Jamieson schlägt in [5] den Algorithmus des Successive-Halving für die Hyperparameteroptimierung einer Klassifizierungsaufgabe vor. Dabei wird gezeigt, dass durch die sukzessive Halbierung der Kandidatenmenge die Rechenzeit zum Erreichen vergleichbarer Testfehler zum uniformen Optimieren aller Kandidaten um eine Größenordnung reduziert werden kann.

In [4] verwenden Karapetyan und Stützle im Rahmen einer Conditional Markov Chain Search für kombinatorische Probleme Volltrainings von 10 % der Kandidaten, um die schlechteste Lernkurve der besten 1 % oder 5 % der Kandidaten als Cutoff-Grenze für das anschließende Training der restlichen Multi-Start-Kandidaten zu definieren. Diese Performance-Profile ermöglichten eine Identifikation der Top-Modelle, mit einer Verbesserung der Rechenzeit um einen Faktor ~ 2 bis ~ 9 , abhängig vom untersuchten Problem.

Kostenko [14] untersucht für MS-Optimierung bei der Regression mit MLP-Netzen für ein festes Budget die Auswirkungen der Anzahl an Terminierungsschritten auf die finale Modellgüte. Dabei werden zweidimensionale Regressionsprobleme ohne Rauschen betrachtet und die maximal verwendeten vier Terminierungsschritte als vorteilhaft gegenüber weniger Schritten identifiziert. Die Anzahl terminierter Kandidaten und die Terminierungszeitpunkte werden jedoch nicht benannt.

2.3 Vorgeschlagenes Verfahren für Regressionsprobleme

Da die genaue benötigte Anzahl an Optimierungsepochen allgemein nicht vor dem Start des Modelltrainings bekannt ist, wird ein zweistufiges Verfahren vorgeschlagen. In einem ersten Schritt werden wenige Kandidaten für eine der Problemdimension entsprechend ausreichende Epochenanzahl ohne vorzeitigen Abbruch trainiert. Aus diesen Lernkurven wird ein approximierter Konvergenzzeitpunkt bestimmt, um die folgenden Terminierungsstrategien auf das konkret vorliegende Problem zu skalieren und Overfitting zu vermeiden. Anschließend wird in parallelisierter Durchführung die MS-Kandidatenmenge trainiert und Kandidaten mit schlechter Modellgüte werden entsprechend der nachfolgend erläuterten Heuristiken vorzeitig terminiert. Für beide der erläuterten Strategien wird die Modellgüte durch den MSE auf dem Validierungsdatensatz evaluiert, eine Erweiterung um die Verbesserungsrate brachte in den vorliegenden Beispielen keine Vorteile.

2.3.1 Abschätzung des Konvergenzzeitpunkts

Zur Abschätzung des Konvergenzzeitpunkts wird im Trainingsverlauf der Zeitpunkt identifiziert, an dem ein signifikanter Fortschritt der Optimierung nicht mehr zu erwarten ist. Hierfür wird die Epoche k^* bestimmt, bei der der letzte der vollständig trainierten Kandidaten erstmals einen definierten Schwellenwert von 10 % über seinem minimalen MSE erreicht hat. Diese Wahl des Schwellenwertes wurde als geeignet beobachtet und nicht als Hyperparameter betrachtet, um die Anzahl an untersuchten Varianten zu begrenzen. Die Bestimmung ist bewusst einfach gehalten, da in diesem Beitrag besonders die Betrachtung der Terminierungsstrategien von Interesse ist. In Bild 1a ist die Bestimmung am Beispiel des Trainings mit dem LM-Algorithmus für das Problem R3 (siehe Abschnitt 3.1) mit 441 Trainingsdatenpunkten und einem normalverteilten unkorrelierten Rauschen mit $\sigma = 0,01$ dargestellt. Durch die zufällige Wahl der Volltrainings gibt es für dasselbe Experiment Schwankungen in der bestimmten Konvergenzepoche.

2.3.2 Angepasstes Successive-Halving

In der klassischen Formulierung der Successive-Halving-Strategie nach [5] wird die Kandidatenmenge in mehreren Runden evaluiert. Nach jeder Runde wird ein fester Anteil der Kandidaten mit der geringsten Modellgüte, hier mit dem Terminierungsfaktor γ_{cut} bezeichnet, eliminiert. In der darauffolgenden Runde wird das Ressourcenbudget der verbleibenden Kandidaten, etwa in Form von Trainingsepochen, mit dem Faktor $1/\gamma_{\text{cut}}$ erhöht.

Für die vorliegende Anwendung ist es sinnvoll, verschiedene Terminierungsfaktoren γ_{cut} des Anteils terminierter Kandidaten zu untersuchen. Ebenfalls ist es sinnvoll, eine gleichmäßig verteilte Reduzierung der Kandidatenmenge durch z.B. Terminierung in den Epochen $k_i = n \cdot i, n \in N$ zu definieren. Dies ist durch eine beobachtete schlechte Korrelation der Modellgüten der Kandidaten von sehr frühen Epochen zur finalen Güte begründet. In diesem Beitrag werden die Terminierungsschritte k_i außerdem mittels Faktoren relativ zu der zuvor bestimmten approximierten Konvergenzepoche festgelegt:

$$k_i = \lfloor k^* \cdot \alpha_i \rfloor, \quad \text{mit } \alpha_i \in \mathbb{R}^+, \quad \alpha_0 < \dots < \alpha_r. \quad (3)$$

wobei α_i Skalierungsfaktoren sind. Im Folgenden wird hierfür die Bezeichnung „angepasstes Successive-Halving“ (ASH) verwendet. Die ASH-Strategie ist zur Verdeutlichung an einem Beispiel in Bild 1b dargestellt.

2.3.3 Relative MSE-Obergrenze

Für die Definition der Terminierung mittels einer Obergrenze (OG) in Relation zu der besten bekannten Validierungs-Modellgüte $\text{MSE}_{\text{val,min}}^{(k_i)}$ werden zunächst gleichverteilte Epochen k_i der Modellevaluation definiert. MS-Kandidaten werden nur dann weitertrainiert, wenn die aktuelle Modellgüte unterhalb der Obergrenze über dem Referenzwert $\text{MSE}_{\text{val,min}}^{(k_i)}$ liegt:

$$\text{MSE}_{\text{val,cand}}^{(k_i)} \leq \text{MSE}_{\text{val,min}}^{(k_i)} \cdot (1 + \delta_i). \quad (4)$$

Der Initialwert des Toleranzfaktors δ_i wird verhältnismäßig groß festgelegt (z.B. $\delta_0 = 10$), um zu Beginn einen signifikanten Teil der Kandidatenmenge zu

erfassen. Anschließend wird δ_i im Verlauf des Trainings bis zur approximierten Konvergenzepoche k^* schrittweise reduziert:

$$\delta_i(k_i) = \delta_0 \cdot \left(1 - \left(\frac{k_i}{k^*} \right)^\beta \right). \quad (5)$$

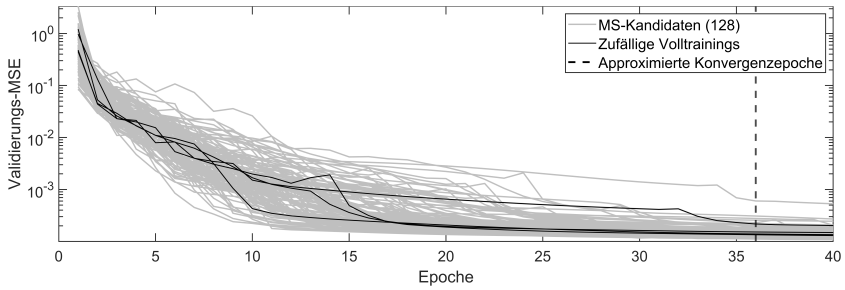
Mit $\beta = 1$ entspricht dies einer linearen Abnahme des Toleranzfaktors δ_i , allerdings erfordert die schnelle Abnahme von $\text{MSE}_{\text{val,min}}^{(k_i)}$ zu Beginn eine Wahl von $\beta < 1$. Die OG-Strategie ist zur Verdeutlichung an einem Beispiel in Bild 1c dargestellt.

3 Fallstudien

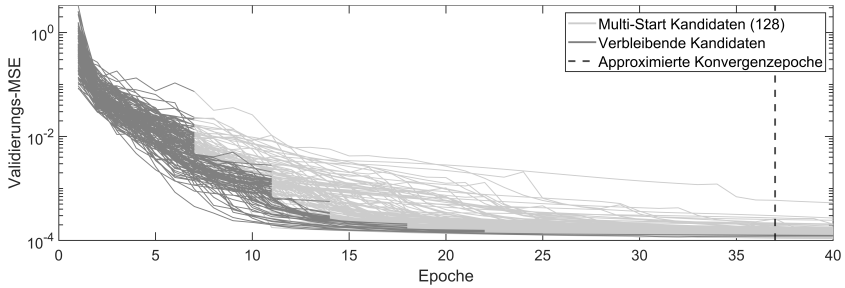
3.1 Testfunktionen

Im Folgenden sollen die beschriebenen Methoden zur modellfreien Terminierung bei der Multi-Start-Optimierung an mittels mehrerer Testfunktionen erzeugten synthetischen Daten evaluiert werden. Zur Auswertung werden drei Testfunktionen (Bild 2a, 2b und 2c), sowie ein Datensatz zu dem Kennfeld eines einstufigen Axialkompressors (Bild 2d) mit zwei Eingängen betrachtet.

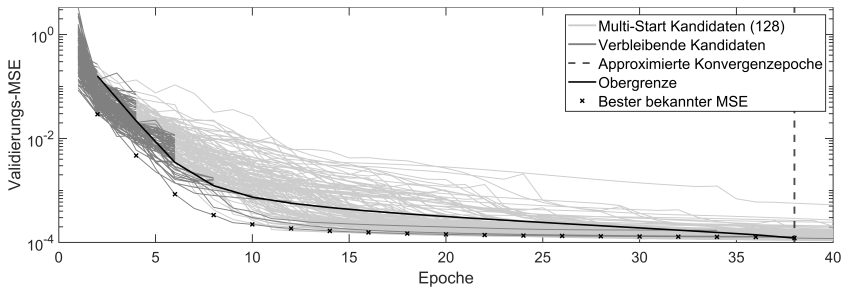
Dabei basiert der aus [15] übernommene Kompressor-Datensatz auf nachbearbeiteten Daten aus [16] und wird im Folgenden als Problem R5 bezeichnet.



(a) Beispiel der Bestimmung einer approximated Konvergenzepoch k^* aus 4 Volltrainings mittels eines Schwellenwertes zum jeweils besten beobachteten Validierungs-MSE.



(b) Multi-Start-Training mit vorzeitiger Terminierung durch den ASH-Algorithmus. Der Terminierungsfaktor ist $\gamma_{\text{cut}} = 0,5$, die Evaluation erfolgt in Schritten von 10% zwischen 20% und 70% der Trainingsdauer bis zur Konvergenzepoch k^* .



(c) Multi-Start-Training mit vorzeitiger Terminierung durch den OG-Algorithmus. Der initiale Toleranzfaktor ist $\delta_0 = 10$, der Exponent ist $\beta = 0,2$.

Bild 1: Darstellung der Bestimmung der Konvergenzepoch k^* und Anwendung der Terminierungsmethoden beim MS-Training eines MLP-Netzes mit dem LM-Algorithmus zur Approximation der Funktion R3 (siehe Abschnitt 3.1) mit 441 Trainingsdatenpunkten und einem normalverteilten unkorrelierten Rauschen von $\sigma = 0,01$.

Die zweidimensionalen Beispiele R1 bis R3 sind definiert durch:

$$f_{R1}(\mathbf{x}) = 10 \sin(\pi x_1) x_2; \quad \mathbf{x} \in [0, 2]^2, \quad (6)$$

$$f_{R2}(\mathbf{x}) = \cos\left(\frac{1}{1+x_2^2}\right) \exp\left(\frac{x_2}{10} \sin(x_1)\right) + \exp\left(\frac{x_1}{10} \cos^2(x_2)\right); \quad \mathbf{x} \in [-4, 4]^2, \quad (7)$$

$$f_{R3}(\mathbf{x}) = x_1^2 + x_2^2 + 10 \sin^2(\pi x_1) + 10 \sin^2(\pi x_2); \quad \mathbf{x} \in [0, 2]^2. \quad (8)$$

Außerdem wird als höherdimensionales Problem die Friedman-Funktion [17]:

$$f_{R4}(\mathbf{x}) = 10 \sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5; \quad \mathbf{x} \in [0, 1]^5, \quad (9)$$

als Regressionsbeispiel mit fünf Eingängen betrachtet.

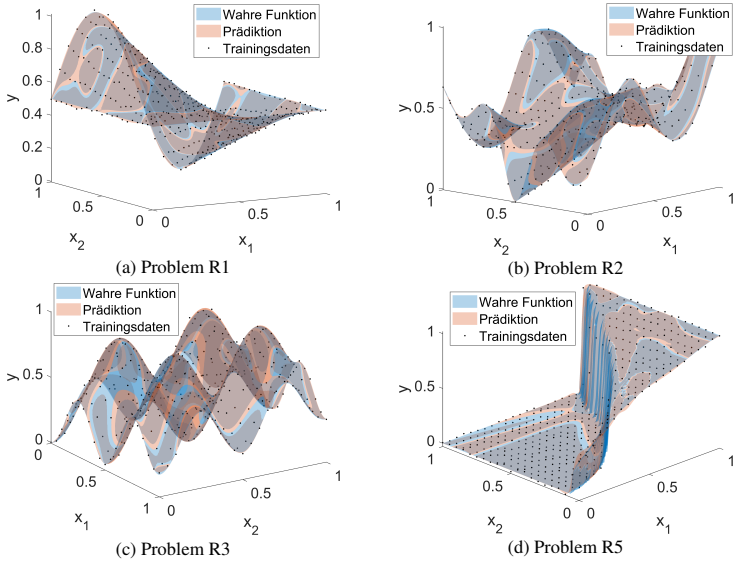


Bild 2: Betrachtete zweidimensionale Regressionsprobleme und Verifikation der genutzten MLP-Netz-Strukturen. Dabei wurden die MLP-Modelle mittels LM optimiert und 441 Trainingsdaten mit einem normalverteilten unkorrelierten Rauschen mit $\sigma = 0,01$ verwendet. Für die Beispiele R1-R3 wurde die MLP-Struktur [2-10-10-1] verwendet, für das Beispiel R5 die Struktur [2-5-5-1].

Die erzeugten Daten werden mit einem normalverteilten unkorrelierten Rauschen mit Mittelwert 0 und Varianz σ^2 überlagert:

$$y = f(\mathbf{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2). \quad (10)$$

Dabei werden verschiedene Rauschstärken mit Standardabweichungen von

$$\sigma \in \{0,005; 0,01; 0,02; 0,05; 0,1\}$$

und unterschiedlich große Trainingsdatensätze betrachtet. Die Trainingsdaten liegen dabei für die Probleme R1 bis R3, sowie R5 auf homogenen Rastern von 21^2 und 51^2 Punkten, für R4 auf Rastern der Größe 3^5 , 5^5 und 7^5 . Sowohl Eingangs- als auch Ausgangsgrößen werden dabei auf den Bereich $[0, 1]$ normiert. Zusätzlich werden Validierungsdaten generiert, deren Anzahl 30 % der Größe des jeweiligen Trainingsdatensatzes entspricht. Diese werden durch eine gleichmäßige Zufallsstichprobe im Eingangsraum erzeugt. Die betrachteten Regressionsprobleme sind in Tabelle 1 aufgelistet.

3.2 MLP-Struktur

Die gewählten MLP-Strukturen sind in Tabelle 1 aufgeführt und erzielen bei der Verwendung des hyperbolischen Tangens als Aktivierungsfunktion der verdeckten Neuronen sehr gute Modellgüten. Auf eine ausführliche Bestimmung der für die jeweiligen Probleme optimalen Modellstrukturen (Anzahl Neuronen, Anzahl verdeckter Schichten, Aktivierungsfunktion der Neuronen) wird verzichtet, da das Ziel dieses Beitrages in der Untersuchung des Trainingsprozesses liegt.

Generell ist die Untersuchung der MS-Terminierung auch für verschieden gut geeignete Modellstrukturen interessant, da eine effizientere Optimierung der Strukturkandidaten die Hyperparameter-Optimierung deutlich verbessern kann. Für genauere Betrachtungen der Topologie-Ermittlung des beispielhaften Anwendungsfalls der Kompressordaten sei auf [18] verwiesen. Die Neuronenanzahl für das Problem R4 orientiert sich an in der Literatur verwendeten MLP-Strukturen [19]. Eine visuelle Prüfung von Bild 2 bestätigt die Eignung der gewählten MLP-Modellstrukturen.

Tabelle 1: Untersuchte Regressionsprobleme und verwendete MLP-Strukturen.

Nr.	Regressions- problem	Anzahl Trainingsdaten	MLP-Struktur	Parameter- anzahl
R1	2D Testfunktion	441,2601	2 – 10 – 10 – 1	151
⋮				
R3				
R4	5D Testfunktion	243,3125,16807	5 – 20 – 1	141
R5	Kompressor- kennfeld (2D)	441,2601	2 – 5 – 5 – 1	51

Die Umsetzung erfolgt mit der Funktion *feedforwardnet* der Deep Learning Toolbox in MATLAB 2024b. Die Optimierung erfolgt mit den Funktionen *trainlm* (LM) und *trainbfg* (BFGS) bis auf die Abbruchbedingungen mit Standardeinstellungen.

Für alle Versuche werden die Verbindungsgewichte und Schwellenwerte der Netze mit der Nguyen-Widrow Methode nach [20] initialisiert. Diese Initialisierung führt für flache Neuronale Netze mit sigmoiden Aktivierungsfunktion der verdeckten Neuronen zu einem schnelleren Lernfortschritt als die Initialisierung mit kleinen Zufallszahlen [21]. Dabei werden die Verbindungsgewichte und Schwellenwerte so mit Zufallszahlen initialisiert, dass die aktiven Bereiche der Neuronen gleichmäßig über den Eingaberaum verteilt sind. So werden Symmetrien vermieden und ein Start des Trainings im ungesättigten Bereich der Neuronen sichergestellt. Das ist für die MS-Optimierung mit Terminierung besonders relevant, da ein schneller initialer Lernfortschritt und qualitativ ähnliche Lernkurven für die Probleme der betrachteten Art auftreten.

3.3 Untersuchte Terminierungsstrategien

Um die Modellgüte der durch die vorgestellten Strategien gefundenen MS-Kandidaten beurteilen zu können, werden alle Modellkandidaten für eine feste Anzahl von 40 (LM) bzw. 400 (BFGS) Epochen optimiert. Anschließend wer-

den die Strategien mit verschiedenen Parametrierungen auf den gespeicherten Gütewerten zu den Epochen simuliert.

Die Vergleiche in Bild 3 zeigen den besten Validierungs-MSE aus 128 MS-Kandidaten über die Epochen (Bild 3a) und über die Zeit (Bild 3b). Für die vorliegenden Probleme konvergiert LM in einer um etwa eine Größenordnung geringeren Epochenanzahl als BFGS.

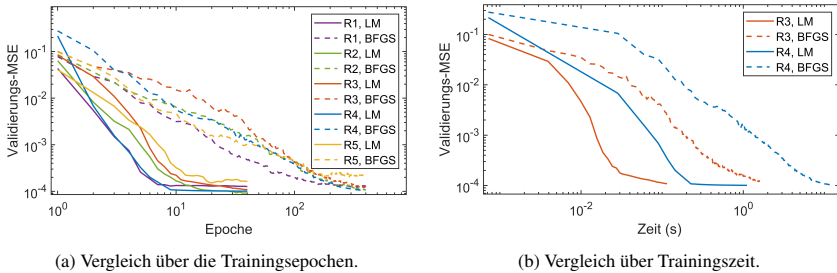


Bild 3: Vergleich des minimalen Validierungs-MSE während der parallelisierten MS-Optimierung von 128 MLP-Modellkandidaten mit LM und BFGS. Dabei wurden 441 (R1-R3, R5) und 16807 (R4) Trainingsdaten mit einem normalverteilten unkorrelierten Rauschen mit $\sigma = 0,01$ verwendet.

Bei der OG-Strategie wird dabei der Faktor β variiert, bei der ASH-Strategie werden verschiedene Anzahlen und Zeitpunkte der Terminierungsschritte mit den jeweiligen Terminierungsfaktoren untersucht.

Dabei werden bei der ASH-Strategie für eine stärkere Terminierung entsprechend weniger Terminierungszeitpunkte festgelegt. Die Auswahl von ASH1 bis ASH6 erfolgt so, dass ein Vergleich zwischen einer frühen, geringeren, kontinuierlichen Terminierung bis hin zu einer spät beginnenden, starken Terminierung möglich ist. Eine Übersicht der gewählten Strategien ist in Tabelle 2 dargestellt.

Die Modellgüten werden bei der OG-Strategie alle 10 Epochen (BFGS) bzw. alle 2 Epochen (LM) ausgewertet. Um die Abhängigkeit der Strategien von der aus den vollständigen Optimierungen der zufälligen Kandidaten approximierten Konvergenzepoche zu berücksichtigen, werden für jede Parametrierung 32 Wiederholungsversuche auf den gespeicherten MS-Trainingsdaten durchgeführt. Für alle Experimente werden 128 MS-Kandidaten trainiert. Diese Anzahl wurde so

Tabelle 2: Untersuchte Parametrierungen der angepassten ASH- und der OG-Strategie.

Name	Terminierungs- faktor γ_{cut}	Skalierungsfaktoren α_i (Terminierung)
ASH1	0,3	0,1; 0,2; 0,3; 0,4; 0,5; 0,6; 0,7; 0,8; 0,9
ASH2	0,5	0,1; 0,2; 0,3; 0,4; 0,5; 0,6
ASH3	0,5	0,2; 0,3; 0,4; 0,5; 0,6; 0,7
ASH4	0,7	0,2; 0,3; 0,4
ASH5	0,7	0,2; 0,4; 0,6
ASH6	0,7	0,4; 0,5; 0,6
Name	Faktor β	
OG1	0,05	
OG2	0,1	
OG3	0,2	
OG4	0,5	
OG5	0,7	

gewählt, dass bei der ASH-Terminierung mindestens zwei Kandidaten erhalten bleiben. Die relativ hohe Zahl an Kandidaten erlaubt eine breite Exploration des Parameterraums. Die Untersuchung verschiedener Kandidatenmengen, insbesondere kleinerer, sollte Gegenstand zukünftiger Untersuchungen sein.

3.4 Ergebnisse

Bevor die Terminierungsstrategien ausgewertet werden, soll betrachtet werden, wie stark sich die erreichten MSE-Werte der Modellkandidaten bei der MS-Optimierung der untersuchten Fälle unterscheiden. Dazu sind in Bild 4 die erreichten Mittelwerte der Kandidaten des MSE auf dem jeweiligen Validierungsdatensatz für die verschiedene Probleme dargestellt.

Bei der Betrachtung des Faktors γ_{MSE} zwischen dem MSE des besten Kandidaten und dem Mittelwert der Kandidatenmenge wird deutlich, dass besonders für Probleme mit keinem oder geringem Rauschen auf dem Datensatz deutliche Verbesserungen durch die MS-Optimierung möglich sind. Der Vergleich mit 32 Wiederholexperimenten auf dem gleichen Datensatz zeigt, dass die Kan-

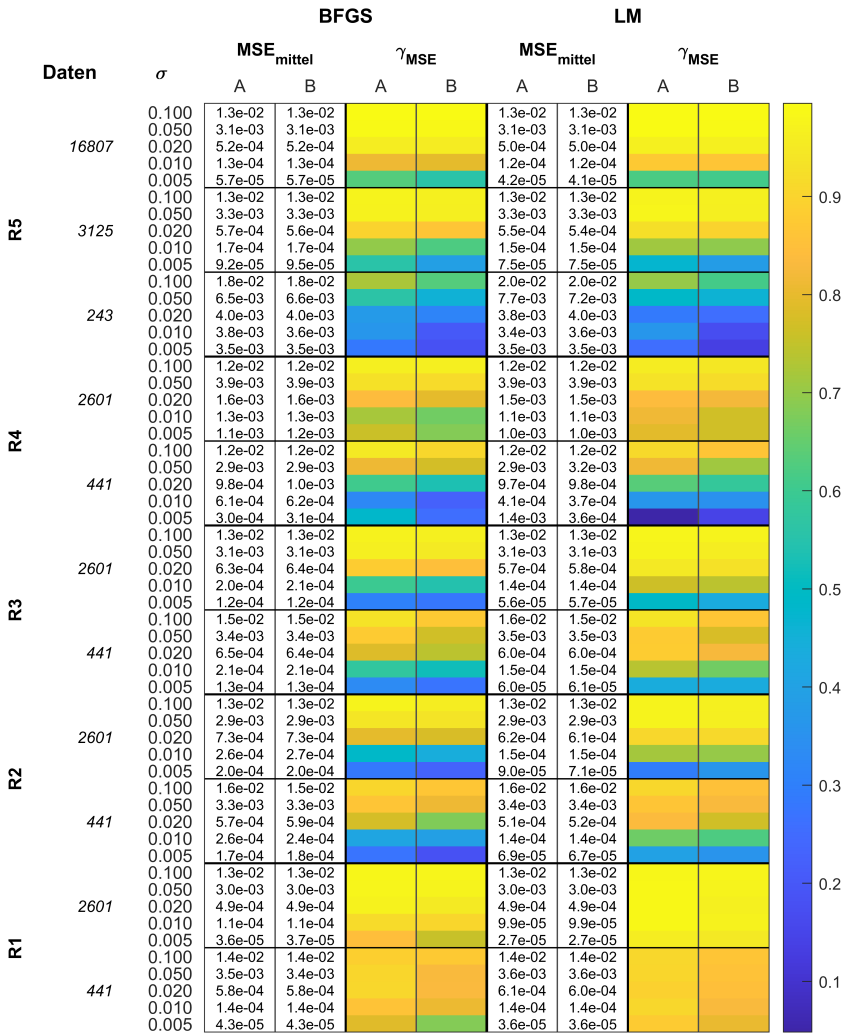


Bild 4: Minimale erreichte MSE-Werte auf dem jeweiligen Validierungsdatensatz bei der Multi-Start-Optimierung mit dem LM- und dem BFGS-Algorithmus und 128 Modellkandidaten. Betrachtet werden verschiedene große Trainingsdatensätze und Rauschlevel. MSE_{mittel} ist der Mittelwert aller 128 Kandidaten, γ_{MSE} ist der Faktor zwischen dem Kandidaten mit minimalem MSE-Wert und dem Mittel aller Kandidaten. Unter A ist jeweils der Wert für ein zufälliges Experiment, unter B der Wert für 32 Wiederholungsversuche der MS-Optimierung auf dem gleichen Datensatz dargestellt.

didatenmenge von 128 Modellen für die meisten der betrachteten Probleme eine ausreichende Exploration des Parameterraums ermöglicht und durch eine größere Kandidatenmenge keine deutlichen Verbesserungen zu erwarten sind. Für die Probleme R4 und R5 tritt mit wenigen Daten Overfitting auf, in diesen Fällen wird beobachtet, dass die minimalen Validierungs-MSE der Kandidaten breiter streuen. Bei dem vergleichsweise einfachen Problem R1 konvergieren durch die verhältnismäßig große betrachtete Epochenanzahl auch schlechte Modelle noch sehr langsam zum besten Kandidaten.

Für die Auswertung der Terminierungsstrategien sind vor allem zwei Aspekte von Interesse. Zum einen sollte möglichst einer der besten Kandidaten des Kandidatenpools nach den vorzeitigen Terminierungen erhalten bleiben. Zum anderen sollte das Budget im Vergleich zu einem vollständigen Training aller MS-Kandidaten möglichst stark reduziert werden.

Zur Einschätzung der Einsparung des Rechenbudgets durch die vorzeitige Terminierung wird die benötigte Anzahl an kumulativen Epochen mit der gesamten MS-Kandidatenmenge bis zur approximierten Konvergenzepoche verglichen. Dabei werden die initialen Trainings der Kandidaten zum Abschätzen der Konvergenzepoche in das Budget der Terminierungsstrategien aufgenommen.

In Bild 5 ist die prozentuale Reduktion des Rechenbudgets dargestellt. Die OG-Strategie zeigt für die verschiedenen Probleme deutliche Unterschiede in der erzielten Reduktion des Rechenbudgets, da keine feste Anzahl an Kandidaten terminiert wird. Für Probleme, bei denen der beste bekannte Kandidat bereits in frühen Epochen stärker vom Mittelwert abweicht, kann somit ein größerer Anteil an Kandidaten terminiert werden. Das ist vor allem bei Problemen mit wenig Rauschen der Fall, welche auch in der finalen Evaluation der MSE-Werte eine größere Verbesserung im Vergleich zum Mittelwert aufzeigen (Bild 6). Im Gegensatz dazu sind die Budgetreduktionen bei der ASH-Strategie vor allem bei geringem Rauschen und besonders bei der BFGS-Optimierung deutlich weniger stark verteilt. Das ist zu erwarten, da das Budget vorab feststeht und Abweichungen nur durch die Anpassung der Terminierungszeitpunkte auf die bestimmte Konvergenzepoche auftreten.

In Bild 6 ist die durch die Terminierungsstrategie erreichte Veränderung des Validierungs-MSE im Vergleich zum Mittelwert der MS-Kandidaten dargestellt. Dabei wird für den finalen Kandidaten mit Terminierung der Zeitraum bis zur

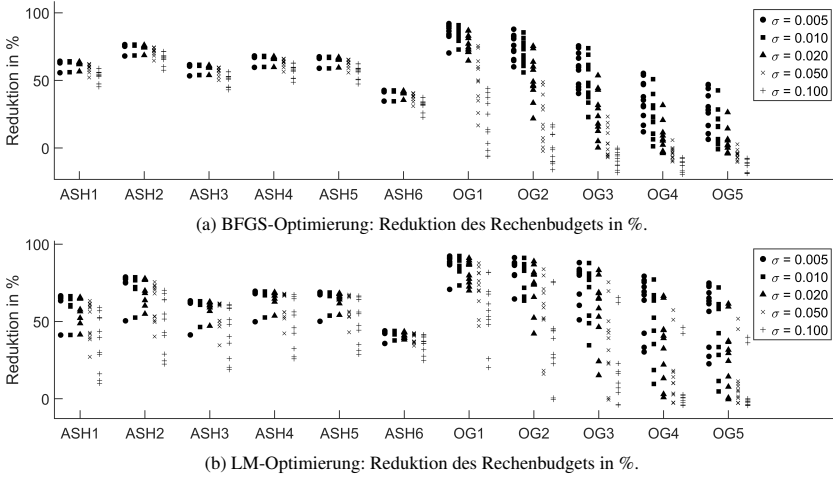


Bild 5: Reduktion des Rechenbudgets in % der Terminierungsstrategie im Vergleich zum vollständigen Training aller Kandidaten bis zur approximierten Konvergenzepoche. Es wird jeweils der Mittelwert für jede Datenmenge der Probleme R1 bis R5 getrennt nach der Standardabweichung σ des Rauschens dargestellt.

Epoche k^* im Vergleich zum Mittelwert des MSE bei vollständigem Training aller MS-Kandidaten betrachtet. Dies ist der ungünstigste Fall und stellt jeweils den schlechtesten erreichten Wert aus je 32 Wiederholungen der ASH- und OG-Strategien auf 10 gespeicherten MS-Trainings dar. Hinsichtlich der Verbesserung des finalen Kandidaten nach der Terminierung sind die Parametrierungen ASH1, und OG2 bis OG5 am robustesten. Diese erzielten in einem Großteil der Probleme, bei denen nach Bild 4 Verbesserungen möglich sind, einen deutlich geringeren MSE im Vergleich zum Mittelwert der MS-Kandidaten. Schlechte Ergebnisse werden vor allem bei LM mit den Parametrierungen erzielt, bei denen an frühen Zeitpunkten ein größerer Teil der Kandidaten terminiert wird.

Um die Strategien im Vergleich zu der Optimierung der gesamten MS-Kandidatenmenge zu beurteilen, ist in Bild 7 der prozentuale Anteil der Experimente, bei denen der finale Kandidat in den besten 5 % der gesamten MS-Kandidatenmenge liegt, dargestellt.

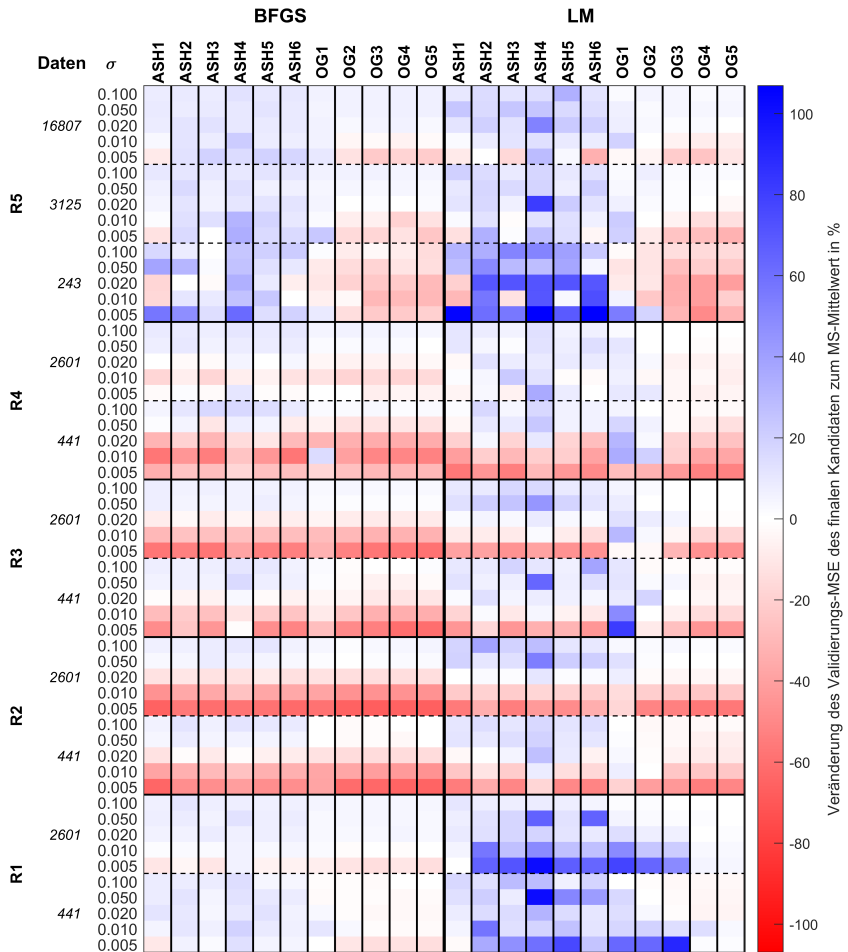


Bild 6: Veränderung des Validierungs-MSE des finalen Kandidaten nach der vorzeitigen Terminierung im Vergleich zum Mittelwert aller voll trainierten MS-Kandidaten in %. Dargestellt wird dabei jeweils der schlechteste erreichte Wert aus je 32 Wiederholungen der ASH- und OG-Strategien auf 10 gespeicherten MS-Trainings.

Die besten Ergebnisse erzielen hier die ebenfalls die Strategien ASH1 und OG2 bis OG5. Im Vergleich zu Bild 6 ist interessant, dass deutliche Verbesserungen der durch den MSE beschriebenen Modellgüte auch dann erreicht werden

können, wenn der finale Kandidat nicht in den besten 5 % der Kandidatenmenge liegt.

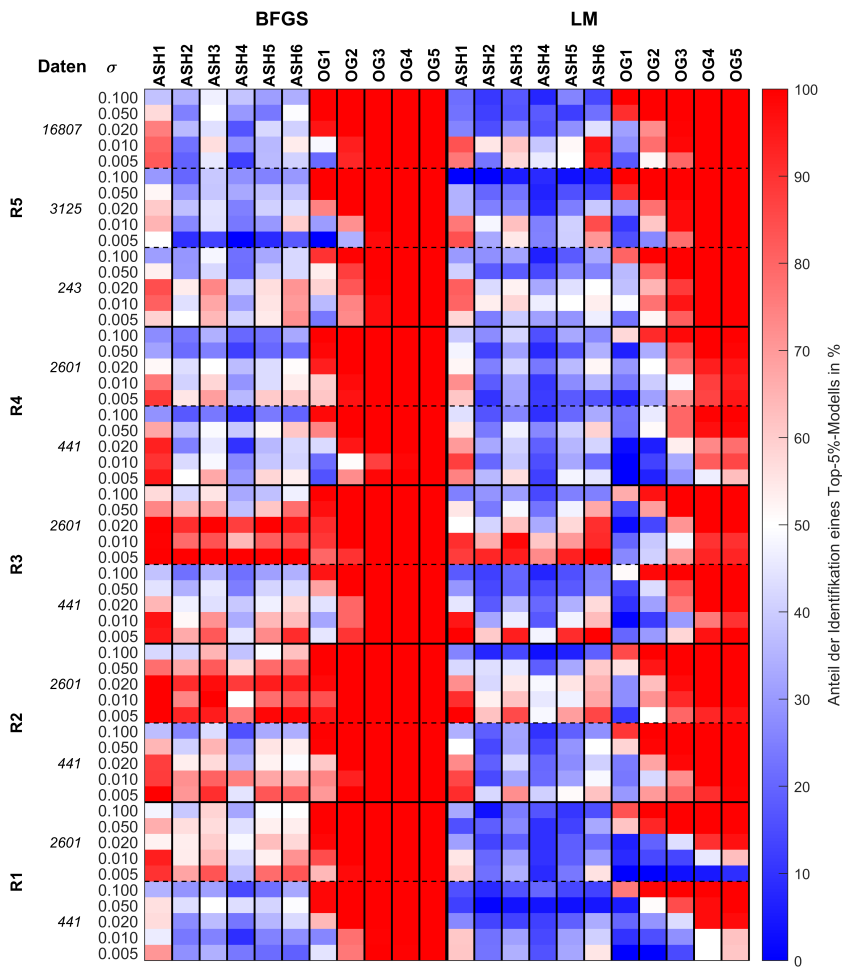


Bild 7: Anteil der Wiederholungsversuche in % bei denen der MSE des besten nicht terminierten Kandidaten zur approximateden Konvergenzepoche in den besten 5 % der gesamten Kandidatenmenge liegt.

3.5 Zusammenfassende Diskussion

Zusammenfassend lässt sich feststellen, dass die Parametrierungen ASH1 und OG2 bis OG5 für die vorliegenden Probleme am besten geeignet sind. Dabei ist die Parametrierung ASH1 dem Ansatz der relativen Obergrenze am ähnlichsten, da kontinuierlicher terminiert wird. Die Parametrierungen OG4 und OG5 erzielen die stabilste Identifikation guter Kandidaten, vor allem bei der BFGS-Optimierung.

Bei einigen Problemen, besonders bei stärkeren Rauschen, benötigen sie jedoch ein deutlich größeres Rechenbudget als ASH1. Einen Kompromiss zwischen Reduktion des Budgets und stabiler Identifikation eines guten Kandidaten bietet OG3.

Der Nutzen der Terminierungsstrategien hängt vom jeweiligen Problem und der gewählten Parametrierung ab. Dabei ist in den hier betrachteten Problemen die vorzeitige Terminierung besonders für die BFGS-Optimierung mit der OG-Strategie lohnenswert und ermöglicht für verschiedene Parametrierungen die Identifikation eines sehr guten Kandidaten. Die gewählte Parametrierung hat dabei jedoch starke Auswirkungen auf die mögliche Reduktion des Rechenbudgets. Bei der LM-Optimierung ist für Parametrierungen mit starker Reduktion des Rechenbudgets keine zuverlässige Identifikation eines der besten Kandidaten erzielt worden. Aufgrund der geringeren Zahl an benötigten Trainingsepochen ist hier die vorzeitige Terminierung weniger lohnenswert.

4 Zusammenfassung und Ausblick

In diesem Beitrag wurde gezeigt, dass durch die Verwendung modellfreier, parallelisierbarer Multi-Start-Terminierungsstrategien das Rechenbudget, gemessen in Trainingsepochen, im Vergleich zum Training aller MS-Kandidaten für die Lösung nichtlinearer Regressionsprobleme mittels MLP-Netzen deutlich reduziert werden kann. Die untersuchten Strategien zur vorzeitigen Terminierung eignen sich besonders bei verhältnismäßig geringem Rauschen auf den verwendeten Daten. Voraussetzung ist, dass die Modellgüten der Kandidaten mit zunehmender Epochenanzahl stärker mit der finalen Güte korrelieren.

Um die Effizienz der Strategien vollständig bewerten zu können, sollte in zukünftigen Untersuchungen auch die Rechenzeit bei der Anwendung der Terminierungsmethoden betrachtet werden. In weiterführenden Arbeiten sollte zudem die Anzahl der untersuchten Probleme erhöht werden. Außerdem sollte geprüft werden, ob sich die Terminierungsstrategien auch für die effizientere Optimierung hinsichtlich des Rechenaufwands komplexerer Probleme eignen. Dabei ist vor allem die Identifikation dynamischer Systeme interessant. Durch die vorzeitige Terminierung bei der MS-Optimierung verschiedener Modellstrukturen bei gleichbleibender Problemstellung sollte untersucht werden, ob die Methoden im Rahmen einer Topologieermittlung einsetzbar sind.

5 Danksagung

Gefördert durch die Deutsche Forschungsgemeinschaft (DFG) - Projektnummer 541311230.

Literatur

- [1] K. Hornik, “Approximation capabilities of multilayer feedforward networks”. *Neural Networks*, Bd. 4, Nr. 2, S. 251–257, 1991.
- [2] J. Nocedal und S. Wright, “Numerical Optimization”. 2. Auflage, New York: Springer, 2006.
- [3] A. György und L. Kocsis, “Efficient multi-start strategies for local search algorithms”. *Journal of Artificial Intelligence Research*, Bd. 41, S. 407–444, 2011.
- [4] D. Karapetyan, A. J. Parkes und T. Stützle, “Algorithm configuration: Learning policies for the quick termination of poor performers”. In: *Learning and Intelligent Optimization*, S. 220–224, 2019.
- [5] K. Jamieson, “Non-stochastic best arm identification and hyperparameter optimization”. In: *Proceedings of Machine Learning Research*, S. 240–248, 2016.

- [6] O. Nelles, “Neural networks”. In: *Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes*. 2. Auflage, Kap. 11, Cham: Springer International Publishing, S. 279–345, 2020.
- [7] M. Hagan und M. Menhaj, “Training feedforward networks with the Marquardt algorithm”. *IEEE Transactions on Neural Networks*, Bd. 5, Nr. 6, S. 989–993, 1994.
- [8] C. Bishop, “Regularization in Neural Networks”. In: *Pattern Recognition and Machine Learning*. Kap. 5.5, New York: Springer, S. 256–271, 2006.
- [9] R. Wang, X. Chen, M. Cheng, X. Tang und C.-J. Hsieh, “Rank-nosh: Efficient predictor-based architecture search via non-uniform successive halving”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, S. 10 357–10 366, 2021.
- [10] T. Domhan, J. T. Springenberg, F. Hutter *et al.*, “Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves”. *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI’15)*, S. 3460–3468, 2015.
- [11] M. de Souza, M. Ritt und M. López-Ibáñez, “Capping methods for the automatic configuration of optimization algorithms”. *Computers & Operations Research*, Bd. 139, 2022.
- [12] P. Guo und M. C. Fu, “Bandit-based multi-start strategies for global continuous optimization”. In: *Proceedings of the Winter Simulation Conference*. S. 3194–3205, 2022.
- [13] D. Marinov und D. Karapetyan, “Hyperparameter optimisation with early termination of poor performers”. In: *Proc. 11th Computer Science and Electronic Engineering (CEECE)*, S. 160–163, 2019.
- [14] V. A. Kostenko, “Multi-start method with cutting for solving problems of unconditional optimization”. *Optical Memory and Neural Networks*, Bd. 29, Nr. 1, S. 30–36, 2020.
- [15] A. Kroll und A. Dürrbaum, “On optimal experiment design for identifying premise and conclusion parameters of Takagi-Sugeno models:

- Nonlinear regression case”. *Applied Soft Computing*, Bd. 60, S. 407–422, 2017.
- [16] J. Flynn, M. Keenan und D. Sulam, “Single-stage evaluation of highly-loaded high mach-number compressor stages”. NASA, Technical Report, 1970.
 - [17] J. H. Friedman, “Multivariate Adaptive Regression Splines”. *The Annals of Statistics*, Bd. 19, Nr. 1, S. 1–67, 1991.
 - [18] A. Kroll, “Kennfläche eines Axialkompressors (MLP)”. In: *Computational Intelligence: Probleme, Methoden und technische Anwendungen*. 2. Auflage, Kap. 15.1, Berlin: De Gruyter Oldenbourg, S. 281–283, 2016.
 - [19] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan und G. Amaratunga, “Ensemble deep learning for regression and time series forecasting”. In: *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)*, S. 1–6, 2014.
 - [20] D. Nguyen und B. Widrow, “Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights”. In: *1990 IJCNN International Joint Conference on Neural Networks*, S. 21–26, 1990.
 - [21] K. Wong, R. Dornberger und T. Hanne, “An analysis of weight initialization methods in connection with different activation functions for feedforward neural networks”. *Evolutionary Intelligence*, Bd. 17, Nr. 3, S. 2081–2089, 2024.

Evolving Clustering of Time Series for Unsupervised Analysis of Industrial Data Streams

Žiga Stržinar¹, Boštjan Pregelj¹, Igor Škrjanc²

¹ Jozef Stefan Institut

Jamova 39, Ljubljana, Slovenia

E-Mail: ziga.strzinar@ijs.si, bostjan.pregelj@ijs.si

² University of Ljubljana, Faculty of Electrical Engineering

Tržaška cesta 25, Ljubljana, Slovenia

E-Mail: igor.skrjanc@fe.uni-lj.si

Summary

We propose sERAL (Streaming Error in Aligned Series), a novel online clustering method for real-time industrial time series monitoring. sERAL aligns and averages streaming data while preserving its shape, offering more accurate, interpretable cluster prototypes than DTW-based methods. Integrated into an evolving clustering framework, it adaptively updates both cluster assignments and cluster count in real time. Evaluated on industrial sensor data, sERAL delivers superior prototype fidelity, scalability, and efficiency. A Python package is provided to support reproducibility and practical adoption.

1 Introduction

The proliferation of sensor data in domains like automotive, manufacturing, healthcare, and finance has elevated the importance of time series analysis, where the temporal order of observations requires specialized methods.

Key subfields include forecasting and clustering [2, 14, 21, 39], and classification [1, 11, 17]. Time series clustering, especially for unlabeled data, supports applications ranging from stock market analysis [28, 42, 45], grid event detection [12], and motor health monitoring [18], to industrial vehicle analysis [14] and process optimization [21].

Traditional clustering approaches are static and rely on offline data, making them poorly suited for streaming scenarios. In contrast, single-pass, evolving clustering algorithms continuously adapt to changing data distributions without retraining, enabling real-time monitoring, fault detection, and edge deployment [3, 4, 6–8, 10, 15, 19, 20, 23, 25, 26, 30–33, 38, 46–53, 58].

This work makes the following contributions. We introduce sERAL (Streaming Error in Aligned Series), a streaming adaptation of the ERAL alignment and averaging method tailored for single-pass clustering. We compare alignment averaging techniques, contrasting DTW-based methods with our shape preserving approach. We develop an evolving time series clustering algorithm that uses sERAL for dynamic centroid formation. We validate our approach using a real-world industrial dataset [54]. Finally, we release the full implementation as source code and a Python package to facilitate reproducibility.

The rest of the paper is structured as follows: Section 2 reviews related work; Section 3 details sERAL and its integration into evolving clustering; Section 4 reports experimental results, and Section 5 concludes and outlines future directions.

2 Related Work

2.1 Time Series Clustering

Time series clustering typically falls into two categories: clustering individual data points within a time series and clustering entire time series.

Clustering of measurements Some methods cluster individual points based on local context, often used for change point detection [?, 57] and anomaly detection [16].

Clustering of whole time series Other methods treat whole time series as clustering objects, using distance measures like DTW [13] combined with k-Means. Our work focuses on this approach.

2.2 k-Shape

k-Shape [39] is a widely used time series clustering method based on shape similarity, using the Shape-Based Distance (SBD) measure. It iteratively refines clusters but requires multiple dataset passes and a fixed number of clusters. It applies to time series of equal length and has been used in stock market analysis [45], energy systems [12,35], and vehicle monitoring [14].

SBD is invariant to scaling, translation, and time shifting, making it robust to common transformations in time series.

2.3 Evolving Clustering

Evolving systems continuously adapt to incoming data, supporting applications in control [5], security [37], power systems [22], and industrial monitoring [9,43]. Evolving clustering algorithms update clusters in real time, adjusting their number, size, and structure without retraining.

Key features include incremental learning, automatic cluster management, and adaptability to concept drift [24,59]. However, most existing evolving clustering methods operate on feature vectors, discarding temporal structure. Our method preserves the full time series, ensuring pattern fidelity.

2.4 Research Gaps

Despite their advantages, evolving clustering methods are rarely applied to full time series. For instance, k-Shape [39] and its streaming variant k-ShapeStream [12] do not support dynamic cluster creation or removal. Similarly, ERAL [56] is designed for batch use and requires a fixed number of clusters.

Our work addresses this gap by introducing a fully evolving time series clustering method that: operates in a single pass, adapts to new data and structure, and does not require predefined cluster counts.

This paper presents the required components to enable real-time, evolving time series clustering.

2.5 Contributions

We present a streaming adaptation of the ERAL alignment and averaging algorithm, enabling single-pass clustering of time series. We further propose an evolving time series clustering algorithm that integrates sERAL for real-time, unsupervised analysis of industrial data streams. Our method is evaluated on real-world data, demonstrating its effectiveness for dynamic monitoring scenarios.

3 Material and Methods

3.1 Time Series Alignment and Averaging

Clustering algorithms typically represent clusters via centroids, computed as simple averages. However, averaging time series is more complex due to varying lengths and distortions. DTW [13] is a standard technique for time series alignment and averaging, with methods such as DBA [40] and SSG [44], and meta-heuristics [27].

DBA has shown strong benchmark performance but is sensitive to initialization, often producing unrealistic prototypes when handling variable-length or distorted series [?, 29, 36, 41]. These issues limit its applicability in domains requiring accurate shape preservation, such as industrial data clustering [56].

To address this, alternatives like k-Shape's shape extraction [39] and the ERAL method [56] have been proposed.

3.2 Error in Aligned Series (ERAL)

ERAL [56] refines a prototype iteratively by aligning it to input series and updating it via a weighted average. Given time series $\underline{x}$ and $\underline{y}$ of lengths N and M :

$$\underline{x} = [x_0, \dots, x_{N-1}] \quad (1)$$

$$\underline{y} = [y_0, \dots, y_{M-1}] \quad (2)$$

A lagged version of $\underline{x}$ is:

$$\underline{x}_\tau = [x_{0-\tau}, x_{1-\tau}, \dots, x_{N-1-\tau}] \quad (3)$$

The ERAL score measures the alignment error:

$$\varepsilon_{x,y}(\tau) = v_{x,y}(\tau)^{-1} \sqrt{\sum_k (x_k - y_{k-\tau})^2} \quad (4)$$

$$v_{x,y}(\tau) = \begin{cases} \sqrt{M - |\tau|}, & \tau < 0 \\ \sqrt{N - |\tau|}, & \tau > N - M \\ \sqrt{M}, & \text{otherwise} \end{cases} \quad (5)$$

$$\tau \in [-M + 1, N - 1] \quad (6)$$

Figure 1a shows the alignment of $\underline{y}$ to $\underline{x}$ with a shift $\tau = 7$. The resulting error is computed as in (4). Fig. 1b visualizes how the error varies with τ .

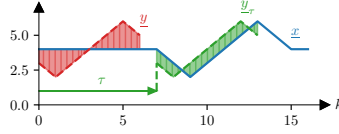
The optimal lag and corresponding score are:

$$\tau_{x,y}^* = \arg \min_{\tau} \varepsilon_{x,y}(\tau) \quad (7)$$

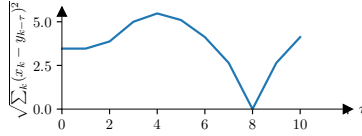
$$\varepsilon_{x,y}^* = \varepsilon_{x,y}(\tau_{x,y}^*) \quad (8)$$

ERAL applies an EM-style update to the prototype $\underline{p}_j$ using weighted, aligned time series:

$$\underline{p}_{j+1} = \underline{p}_j + \lambda \sum_{i=1}^H w_{i,j} \cdot (\underline{x}_{i*,j} - \underline{p}_j) \quad (9)$$



(a) $\underline{x}$ (blue), $\underline{y}$ (red), and shifted $\underline{y}_{-\tau}$ (green) at $\tau = 7$. Shaded area represents error.



(b) ERAL score for $\tau \in [0, N - M]$.

Figure 1: Visual representation of ERAL alignment error ((4)).

Here, λ is the learning rate, H the number of input series, $w_{i,j}$ the weight for each series,

$$w_{i,j} = \frac{\frac{1}{\epsilon_{x_i, p_j}^*}}{\sum_{q=1}^H \frac{1}{\epsilon_{x_q, p_j}^*}} = \left(\epsilon_{x_i, p_j}^* \sum_{q=1}^H \frac{1}{\epsilon_{x_q, p_j}^*} \right)^{-1}, \quad (10)$$

and $\underline{x}_{i*,j}$ the optimally shifted version of $\underline{x}_i$ aligned to $\underline{p}_j$.

3.3 Streaming ERAL

ERAL, as defined in [56], is a batch algorithm requiring full re-computation when new time series are added. To enable online processing, we propose Streaming ERAL (sERAL), which incrementally updates cluster centroids [34].

3.3.1 Prototype Shape and Confidence Vectors

sERAL introduces the Prototype Shape Vector (PSV) $\underline{r}$ and Prototype Confidence Vector (PCV) $\underline{q}$, defined after j processed series as:

$$\underline{r}_j = [r_{j,0}, r_{j,1}, \dots, r_{j,R_j-1}], \quad \underline{q}_j = [q_{j,0}, q_{j,1}, \dots, q_{j,R_j-1}] \quad (11)$$

The prototype $\underline{p}_j$ is a segment of $\underline{r}_j$:

$$\underline{p}_j = [r_{j,k}, \dots, r_{j,k+P_j}], \quad 0 \leq k \leq R_j - P_j \quad (12)$$

PSV is updated only in the regions where the new sample x_j^* and the previous PSV $\underline{r}_{j-1}$ overlap after alignment, i.e., where $0 \leq i_{j,\min} + i - \tau_{x,p}^* \leq N$, with N the length of $\underline{x}_j$. The updated values are calculated as a weighted average, with the iteration count j acting as the weighting factor:

$$r_{j+1,i} = \begin{cases} \frac{j}{j+1} r_{j,i} + \frac{1}{j+1} x_{j,i}^*, & \text{if } 0 \leq i_{j,\min} + i - \tau_{x,p}^* \leq N \quad (\text{overlap region}) \\ r_{j,i}, & \text{otherwise} \end{cases} \quad (13)$$

Since new input data is incorporated in the overlap regions, the confidence values expressed by PCV are updated accordingly:

$$q_{j+1,i} = \begin{cases} \frac{j}{j+1} q_{j,i} + \frac{1}{j+1}, & \text{if } 0 \leq i_{j,\min} + i - \tau_{x,p}^* \leq N \quad (\text{overlap region}) \\ \frac{j}{j+1} q_{j,i}, & \text{otherwise} \end{cases} \quad (14)$$

The prototype $\underline{p}_j$ is extracted using confidence threshold α :

$$i_{j,\min} = \min i \mid q_{j,i} > \alpha, \quad i_{j,\max} = \max i \mid q_{j,i} > \alpha \quad (15)$$

$$\underline{p}_j = [r_{j,i} \mid i_{j,\min} \leq i \leq i_{j,\max}] \quad (16)$$

3.3.2 Evolving PSV and PCV

To align a new input $\underline{x}_j$ with the previous prototype $\underline{p}_{j-1}$, the ERAL score is computed:

$$\varepsilon_{x_j, p_{j-1}}(\tau) = v_{x_j, p_{j-1}}^{-1} \sqrt{\sum_k (x_{j,k} - p_{j-1,k-\tau})^2 \tau_{x,p}^*} = \min_{\tau} \varepsilon_{x,p}(\tau) \quad (17)$$

Aligned input $\underline{x}_j^?$ is then used to update PSV and PCV as above. When alignment extends PSV/PCV beyond their current length, they are padded with values from $\underline{x}_j^?$ and confidence $\frac{1}{j+1}$.

3.3.3 Merging

To merge two clusters represented by sERAL prototypes, let $\underline{r}^1$, $\underline{q}^1$, n^1 and $\underline{r}^2$, $\underline{q}^2$, n^2 denote the PSVs, PCVs, and sample counts of each cluster.

The procedure is as follows:

1. Extract prototypes $\underline{p}^1$ and $\underline{p}^2$ using (13)–(16).
2. Compute optimal alignment lag:

$$\tau^{1,2} = \arg \min_{\tau} \left(v_{1,2}^{-1} \sqrt{\sum_k (p_{1,k} - p_{2,k-\tau})^2} \right) \quad (18)$$

3. Align $\underline{r}^1$, $\underline{q}^1$ and $\underline{r}^2$, $\underline{q}^2$ using $\tau^{1,2}$.
4. Compute the merged PSV and PCV:

$$r_i^M = \begin{cases} \frac{n^1}{n^1 + n^2} r_i^1 + \frac{n^2}{n^1 + n^2} r_{i+\tau^{1,2}}^2, & \text{if both defined,} \\ r_i^1, & \text{if only } \underline{r}^1 \text{ defined,} \\ r_i^2, & \text{if only } \underline{r}^2 \text{ defined.} \end{cases} \quad (19)$$

$$q_i^M = \begin{cases} \frac{n^1}{n^1 + n^2} q_i^1 + \frac{n^2}{n^1 + n^2} q_{i+\tau^{1,2}}^2, & \text{if both defined,} \\ \frac{n^1}{n^1 + n^2} q_i^1, & \text{if only } \underline{q}^1 \text{ defined,} \\ \frac{n^2}{n^1 + n^2} q_{i+\tau^{1,2}}^2, & \text{if only } \underline{q}^2 \text{ defined.} \end{cases} \quad (20)$$

$$n^M = n^1 + n^2. \quad (21)$$

The resulting $\underline{r}^M$, $\underline{q}^M$, and n^M define the merged prototype.

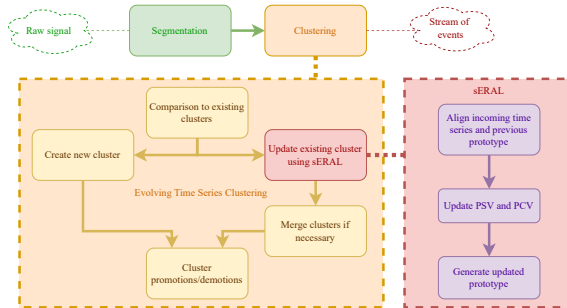


Figure 2: Diagram of the evolving time series clustering approach with sERAL. Raw signals are segmented into time series and passed to the clustering algorithm. The evolving logic is shown in orange, and the sERAL-based prototype update in red and purple.

3.4 Evolving Clustering

This section presents a simple evolving clustering algorithm based on sERAL.

The main features are:

- Operates in a single-pass manner, processing time series samples as they arrive and discarding them afterward.
- Maintains a dynamic set of clusters, each with a centroid and sample count.
- The first sample initializes the first cluster.
- Each new sample is compared to existing centroids. If the minimum distance is below a threshold, it's added to the nearest cluster; otherwise, a new cluster is formed.
- Clusters are merged if their prototypes converge.
- Clusters are categorized as *established* (sufficient samples) or *outlier* (few samples).

Figure 2 illustrates the complete flow.

3.4.1 Distance Calculation

The distance between a sample $\underline{x}$ and prototype $\underline{p}$ is given by:

$$d(\underline{p}, \underline{x}) = \min_{\tau} (\epsilon_{x,p}(\tau)) \cdot v_{x,p} \cdot \left(\frac{N \cdot P}{O^2} \right)^2 \quad (22)$$

where N, P are lengths of $\underline{x}$ and $\underline{p}$, O is the valid overlap after optimal shift $\tau_{*,p}$, and $\epsilon_{x,p}(\tau)$ is the denormalized ERAL score (4), defined in (??).

4 Experiment and Results

To evaluate the proposed clustering algorithm, we use a real-world industrial dataset [55], available at [54].

4.1 Use Case

Many industrial machines—especially older or custom-built—lack automated diagnostics due to outdated interfaces, budget constraints, or limited analytics capabilities. This work aims to equip such machines with lightweight diagnostic capabilities.

We focus on an end-of-line testing station, a critical unit performing product validation. Malfunctions can cause shipping delays or faulty results. To monitor its behavior, we installed a pneumatic pressure sensor at its supply line.

4.1.1 Segmentation

Analysis showed that actuator activity causes abrupt pressure drops. We developed a simple segmentation algorithm that detects these drops and filters out low-variance segments (inactive states). The segmented signals (Fig. 3b) are presumed to correspond to distinct machine actions, forming the basis for unsupervised time series clustering.

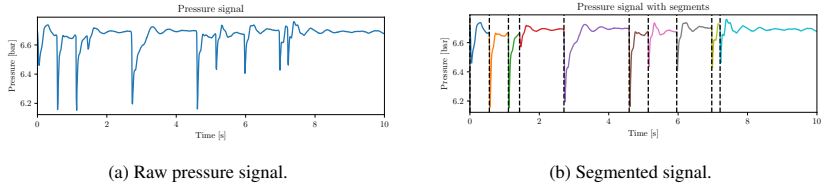


Figure 3: Pressure signal sampled at 100 Hz. Each segment likely corresponds to a machine action.

4.2 Evaluation of sERAL

4.2.1 Demonstrative Example

We evaluate sERAL on a synthetic dataset of 24 noisy, variable-length time series. The samples are added sequentially to sERAL, updating the prototype at each step using Equations (13)–(16), with $\alpha = 0.5$.

4.2.2 Comparison to other methods

We compare sERAL with SBD [39], DBA [40], and ERAL [?], on a subset of the industrial dataset [?, 54, 55]. Fig. 4 shows selected class examples.

ERAL and sERAL yield similar, accurate prototypes. DBA suffers from issues noted in prior works [?, ?, 29, 36, 41]. SBD is limited to equal-length series; we truncated the data to meet this constraint. Observed SBD drawbacks: prototypes are overly short, scaling artifacts are present, and shapes may not reflect underlying data.

4.3 Evolving Clustering

This section presents our main contribution: an evolving clustering algorithm that integrates the sERAL method for alignment and averaging.

4.3.1 Demonstrative Example

Fig. 5 illustrates the algorithm’s core behavior on a simple example.

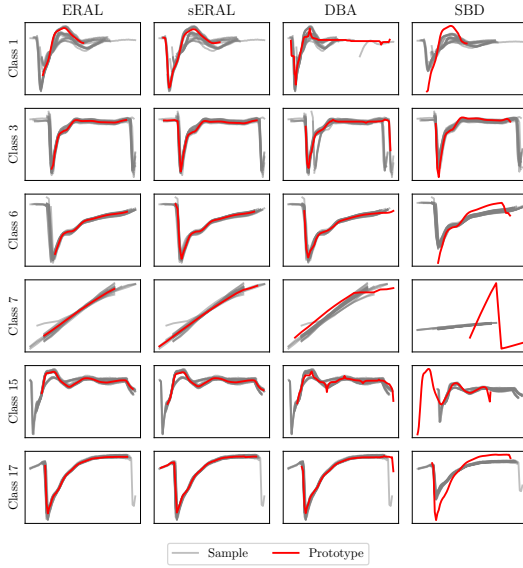


Figure 4: Prototype comparison: ERAL, sERAL, DBA, and SBD. Samples are aligned to prototypes for clarity [54].

The top row shows incoming time series. The second row displays distances to current cluster prototypes, highlighting the closest match. If no match is found, a new cluster is created. The middle section shows cluster evolution, with each row representing a cluster. Prototypes (orange) update when new samples (blue) are added. The bottom graph tracks the number of clusters over time.

4.3.2 Evolving Clustering Results

Using the unlabeled dataset from Section 4.1, we applied the clustering algorithm with sERAL.

Final established clusters are visualized in Fig. 6, showing strong alignment between prototypes (red) and samples (gray).

These results confirm that sERAL enables accurate, adaptive clustering in evolving environments.

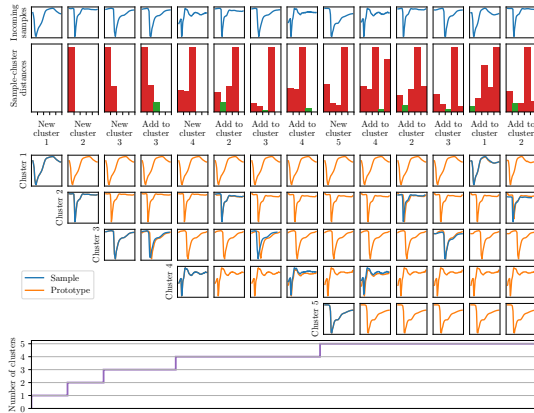


Figure 5: Evolution of clusters for the first 16 time series. Rows represent clusters; columns represent new samples. Orange: prototypes. Blue: assigned samples.

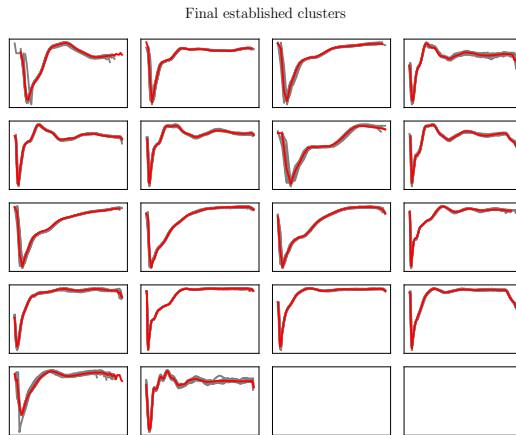


Figure 6: Final established clusters.

5 Conclusion

The experimental evaluation demonstrates that sERAL offers a robust, adaptive solution for unsupervised clustering in evolving, unlabeled environments.

Using a labeled dataset, sERAL's clusters align well with ground truth, underlining its ability to extract meaningful groupings without supervision.

Overall, sERAL stands out as an accurate, interpretable, and scalable framework for adaptive clustering and anomaly detection in non-stationary, real-time data streams.

References

- [1] Amaia Abanda, Usue Mori, and Jose A Lozano. A review on distance based time series classification. *Data Mining and Knowledge Discovery*, 33(2):378–412, 2019.
- [2] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering—A decade review. *Information Systems*, 53:16–38, 2015.
- [3] Goran Andonovski, Plamen Angelov, Saso Blazic, and Igor Skrjanc. A practical implementation of robust evolving cloud-based controller with normalized data space for heat-exchanger plant. *Appl. Soft Comput.*, 48:29–38, 2016.
- [4] Goran Andonovski, Bruno Sielly Jales Costa, Saso Blazic, and Igor Skrjanc. Robust evolving controller for simulated surge tank and for real two-tank plant. *Autom.*, 66(9):725–734, 2018.
- [5] Goran Andonovski, Bruno Sielly Jales Costa, Sašo Blažič, and Igor Škrjanc. Robust evolving controller for simulated surge tank and for real two-tank plant. *at-Automatisierungstechnik*, 66(9):725–734, 2018.
- [6] Goran Andonovski, Edwin Lughofer, and Igor Skrjanc. Evolving fuzzy model identification of nonlinear wiener-hammerstein processes. *IEEE Access*, 9:158470–158480, 2021.
- [7] Goran Andonovski, Gasper Music, Saso Blazic, and Igor Skrjanc. Evolving fuzzy model based performance identification for production control. In *2016 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2016, Natal, Brazil, May 23-25, 2016*, pages 85–91. IEEE, 2016.

- [8] Goran Andonovski, Gasper Music, Saso Blazic, and Igor Skrjanc. Evolving model identification for process monitoring and prediction of non-linear systems. *Eng. Appl. Artif. Intell.*, 68:214–221, 2018.
- [9] Goran Andonovski, Gašper Mušič, Sašo Blažič, and Igor Škrjanc. Evolving model identification for process monitoring and prediction of non-linear systems. *Engineering applications of artificial intelligence*, 68:214–221, 2018.
- [10] Milos Antic, Andrej Zdesar, and Igor Skrjanc. Segmentation of stereo-camera depth image into planar regions based on evolving principal component clustering. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI 2021, Karlsruhe, Germany, September 23-25, 2021*, pages 1–5. IEEE, 2021.
- [11] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.
- [12] Mohini Bariya, Alexandra von Meier, John Paparrizos, and Michael J Franklin. k-shapestream: Probabilistic streaming clustering for electric grid events. In *2021 IEEE Madrid PowerTech*, pages 1–6. IEEE, 2021.
- [13] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pages 359–370. AAAI Press, 1994.
- [14] Silvia Buccafusco, Luca Cagliero, Andrea Megaro, Francesco Vaccarino, Riccardo Loti, and Lucia Salvatori. Learning industrial vehicles’ duty patterns: A real case. *Computers in Industry*, 145:103826, 2023.
- [15] Oscar Cartagena, Miha Ozbot, Doris Sáez, and Igor Skrjanc. Evolving fuzzy prediction interval for fault detection in a heat exchanger. *Appl. Soft Comput.*, 145:110625, 2023.
- [16] C. Catalano, L. Paiano, F. Calabrese, M. Cataldo, L. Mancarella, and F. Tommasi. Anomaly detection in smart agriculture systems. *Computers in Industry*, 143:103750, 2022.

- [17] Hoang Anh Dau, Anthony J. Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn J. Keogh. The UCR time series archive. *CoRR*, abs/1810.07758, 2018.
- [18] Miromar Jose de Lima, Cesar David Paredes Crovato, Rodrigo Ivan Goytia Mejia, Rodrigo da Rosa Righi, Gabriel de Oliveira Ramos, Cristiano André da Costa, and Giovanni Pesenti. Healthmon: An approach for monitoring machines degradation using time-series decomposition, clustering, and metaheuristics. *Computers & Industrial Engineering*, 162:107709, 2021.
- [19] Dejan Dovzan, Vito Logar, and Igor Skrjanc. Implementation of an evolving fuzzy model (efumo) in a monitoring system for a waste-water treatment process. *IEEE Trans. Fuzzy Syst.*, 23(5):1761–1776, 2015.
- [20] Cristiano Garcia, Daniel F. Leite, and Igor Skrjanc. Incremental missing-data imputation for evolving fuzzy granular prediction. *IEEE Trans. Fuzzy Syst.*, 28(10):2348–2362, 2020.
- [21] Zhaoke Huang, Chunhua Yang, Xiaofang Chen, Xiaojun Zhou, and Weihua Gui. Time series clustering method with cluster validation to identify unknown local cell conditions in the aluminum reduction cell. *Computers & Industrial Engineering*, 174:108790, 2022.
- [22] Rashpinder Kaur Jagait, Mohammad Navid Fekri, Katarina Grolinger, and Syed Mir. Load forecasting under concept drift: Online ensemble learning with recurrent neural network and arima. *IEEE Access*, 9:98992–99008, 2021.
- [23] Gregor Klancar and Igor Skrjanc. Evolving principal component clustering with a low run-time complexity for LRF data mapping. *Appl. Soft Comput.*, 35:349–358, 2015.
- [24] Daniel Leite, Igor Škrjanc, and Fernando Gomide. An overview on evolving systems and learning from stream data. *Evolving systems*, 11(2):181–198, 2020.

- [25] Daniel F. Leite and Igor Skrjanc. Ensemble of evolving optimal granular experts, OWA aggregation, and time series prediction. *Inf. Sci.*, 504:95–112, 2019.
- [26] Daniel F. Leite, Igor Skrjanc, and Fernando A. C. Gomide. An overview on evolving systems and learning from stream data. *Evol. Syst.*, 11(2):181–198, 2020.
- [27] Pablo Leon-Alcaide, Luis Rodriguez-Benitez, Ester Castillo-Herrera, Juan Moreno-Garcia, and Luis Jimenez-Linares. An evolutionary approach for efficient prototyping of large time series datasets. *Information Sciences*, 511:74–93, 2020.
- [28] Tristan Lim and Chin Sin Ong. Portfolio diversification using shape-based clustering. *Journal of Financial Data Science*, 3(1):111, 2021.
- [29] Yutao Liu, Yong-An Zhang, Ming Zeng, and Jie Zhao. A novel shape-based averaging algorithm for time series. *Engineering Applications of Artificial Intelligence*, 126:107098, 2023.
- [30] Fabricio P. Lucas, Pyramo Costa, Rose M. S. Batalha, Daniel F. Leite, and Igor Skrjanc. Fault detection in smart grids with time-varying distributed generation using wavelet energy and evolving neural networks. *Evol. Syst.*, 11(2):165–180, 2020.
- [31] Edwin Lughofer, Mahardhika Pratama, and Igor Skrjanc. Incremental rule splitting in generalized evolving fuzzy systems for autonomous drift compensation. *IEEE Trans. Fuzzy Syst.*, 26(4):1854–1865, 2018.
- [32] Edwin Lughofer, Mahardhika Pratama, and Igor Skrjanc. Online bagging of evolving fuzzy systems. *Inf. Sci.*, 570:16–33, 2021.
- [33] Edwin Lughofer and Igor Skrjanc. Evolving error feedback fuzzy model for improved robustness under measurement noise. *IEEE Trans. Fuzzy Syst.*, 31(3):997–1008, 2023.
- [34] Edwin Lughofer and Igor Skrjanc. Online active learning for evolving error feedback fuzzy models within a multi-innovation context. *IEEE Trans. Fuzzy Syst.*, 32(11):5998–6011, 2024.

- [35] Igor Manojlović, Goran Švenda, Aleksandar Erdeljan, and Milan Gavrić. Time series grouping algorithm for load pattern recognition. *Computers in Industry*, 111:140–147, 2019.
- [36] Marion Morel, Catherine Achard, Richard Kulpa, and Séverine Dubuisson. Time-series averaging using constrained dynamic time warping with tolerance. *Pattern Recognition*, 74:77–89, 2018.
- [37] Riyadh Rahef Nuiiaa, Selvakumar Manickam, Ali Hakem Alsaeedi, and Dhiah Eadan Jabor Al-Shammary. Evolving dynamic fuzzy clustering (edfc) to enhance drdos_dns attacks detection mechnism. *International Journal of Intelligent Engineering & Systems*, 15(1), 2022.
- [38] Miha Ozbot, Edwin Lughofer, and Igor Skrjanc. Evolving neuro-fuzzy systems-based design of experiments in process identification. *IEEE Trans. Fuzzy Syst.*, 31(6):1995–2005, 2023.
- [39] John Paparrizos and Luis Gravano. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1855–1870, 2015.
- [40] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern recognition*, 44(3):678–693, 2011.
- [41] Irati Rasines, Anthony Remazeilles, Miguel Prada, and Itziar Cabanes. Minimum cost averaging for multivariate time series using constrained dynamic time warping: A case study in robotics. *IEEE Access*, 2023.
- [42] Seokhyun Ryu, Kiho Kwak, and Sungjoo Lee. The growth patterns of firms in the industrial value chain using the hidden markov model: Evidence from the korean semiconductor industry. *Computers & Industrial Engineering*, 192:110187, 2024.
- [43] Mailson Ribeiro Santos, Bruno Sielly Jales Costa, Clauber Gomes Bezerra, Goran Andonovski, and Luiz Affonso Guedes. An evolving approach for fault diagnosis of dynamic systems. *Expert Systems with Applications*, 189:115983, 2022.

- [44] David Schultz and Brijnesh Jain. Nonsmooth analysis and subgradient methods for averaging in dynamic time warping spaces. *Pattern Recognition*, 74:340–358, 2018.
- [45] Yukari Shirota and Akane Murakami. Long-term time series data clustering of stock prices for portfolio selection. In *2021 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, pages 1–6. IEEE, 2021.
- [46] Igor Skrjanc. Evolving fuzzy-model-based design of experiments with supervised hierarchical clustering. *IEEE Trans. Fuzzy Syst.*, 23(4):861–871, 2015.
- [47] Igor Skrjanc. egauss+ evolving clustering in classification. In *15th IEEE International Symposium on Applied Computational Intelligence and Informatics, SACI 2021, Timisoara, Romania, May 19-21, 2021*, pages 533–538. IEEE, 2021.
- [48] Igor Skrjanc. An evolving concept in the identification of an interval fuzzy model of wiener-hammerstein nonlinear dynamic systems. *Inf. Sci.*, 581:73–87, 2021.
- [49] Igor Skrjanc, Goran Andonovski, José Antonio Iglesias, María Paz Sesmero, and Araceli Sanchis. Evolving gaussian on-line clustering in social network analysis. *Expert Syst. Appl.*, 207:117881, 2022.
- [50] Igor Skrjanc, Goran Andonovski, Agapito Ledezma, Oscar Sipele, José Antonio Iglesias, and Araceli Sanchis. Evolving cloud-based system for the recognition of drivers’ actions. *Expert Syst. Appl.*, 99:231–238, 2018.
- [51] Igor Skrjanc, Saso Blazic, Edwin Lughofer, and Dejan Dovzan. Inner matrix norms in evolving cauchy possibilistic clustering for classification and regression from data streams. *Inf. Sci.*, 478:540–563, 2019.
- [52] Igor Skrjanc, José Antonio Iglesias, Araceli Sanchis, Daniel F. Leite, Edwin Lughofer, and Fernando A. C. Gomide. Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey. *Inf. Sci.*, 490:344–368, 2019.

- [53] Igor Skrjanc, Seiichi Ozawa, Tao Ban, and Dejan Dovzan. Large-scale cyber attacks monitoring using evolving cauchy possibilistic clustering. *Appl. Soft Comput.*, 62:592–601, 2018.
- [54] Žiga Stržinar, Boštjan Pregelj, Janko Petrovčič, Igor Škrjanc, and Gregor Dolanc. Pneumatic pressure and electrical current time series in manufacturing, April 2024.
- [55] Žiga Stržinar, Boštjan Pregelj, Janko Petrovčič, Igor Škrjanc, and Gregor Dolanc. Time series insights from the shopfloor: A real-world dataset of pneumatic pressure and electrical current in discrete manufacturing. *Data in Brief*, page 110619, 2024.
- [56] Žiga Stržinar, Boštjan Pregelj, and Igor Škrjanc. Non-elastic time series fuzzy clustering for efficient analysis of industrial data sets. *Applied Soft Computing*, 167:112398, 2024.
- [57] Wangshu Tu, Utkarsh J Dang, and Sanjeena Subedi. Change point detection via gaussian mixture model. *Journal of Statistical Research*, 58(1):197–219, August 2024.
- [58] Andrej Zdesar, Dejan Dovzan, and Igor Skrjanc. Self-tuning of 2 DOF control based on evolving fuzzy model. *Appl. Soft Comput.*, 19:403–418, 2014.
- [59] Alaettin Zubaroğlu and Volkan Atalay. Data stream clustering: a review. *Artificial Intelligence Review*, 54(2):1201–1236, 2021.

Evaluating Time-Series Foundation Models for Cooling Demand Forecasting with Little Data

Alexander Kreusel, Matthias Hertel, Moritz Noskiewicz, Heiko Maaß, Ralf Mikut, Veit Hagemeyer

Karlsruhe Institute of Technology, E-Mail: ralf.mikut@kit.de

Abstract

Modern buildings increasingly integrate local energy generation, consumption, and storage, often involving multiple energy carriers such as electricity and thermal energy. This complexity creates opportunities for cost-efficient operation based on accurate forecasts of key energy parameters such as the cooling demand. However, generating such forecasts on a building-level can be challenging, as individual buildings likely exhibit highly specific consumption patterns and often offer only limited historical data.

Time Series Foundation Models (TSFMs) offer a promising solution to this problem due to their ability to generalise across forecasting tasks and adapt to new domains with minimal data via fine-tuning. This study evaluates three state-of-the-art TSFMs (MOIRAI, MOIRAI-MoE and Chronos) in both zero-shot and fine-tuned settings. The models are tested on real-world energy consumption data from a split-type cooling unit used in a server room, spanning a period of less than four months. Outdoor air temperature is included as a covariate to assess its impact on prediction accuracy. Results are compared to two baseline models.

Our findings show that Chronos, when incorporating outdoor air temperature, achieves a substantial improvement in forecasting accuracy for three-day ahead forecasts, reducing the Mean Absolute Percentage Error (MAPE) to as low as 2.57 %. In contrast, MOIRAI and MOIRAI-MoE show no significant benefit from the inclusion of temperature information. Overall, the study demonstrates that TSFMs represent a promising alternative for cooling demand forecasting

in buildings, particularly in scenarios where only limited historical data is available.

1 Introduction

The heating and cooling sector accounts for more than 50 % of Germany's total final energy consumption, making it a central focus of the energy transition [17,21]. Accurate demand forecasting is essential for efficient energy management, yet a major challenge stems from the limited or non-existent historical consumption data in most buildings. This impedes the training of building-specific forecasting models and constrains their practical applicability. In recent years, TSFMs have emerged as a promising solution to address the challenge of forecasting under limited historical data [11]. These models are pre-trained on diverse collections of time series and can be applied to a wide range of forecasting tasks without task-specific training. They can also be adapted to domain-specific problems with only limited data through fine-tuning. While this makes TSFMs well suited for time series forecasting under data-scarce conditions, their application for cooling demand forecasting remains largely unexplored. A major challenge is that external covariates, such as outdoor temperature, strongly influence the efficiency of cooling systems and thus the cooling demand, yet strategies for effectively integrating such auxiliary information into TSFMs remain underdeveloped. This study addresses this gap by systematically evaluating the potential of TSFMs for cooling demand prediction with the integration of covariate information. Three TSFMs are tested on a real-world time series of a split-type AC unit at both 15-minute and hourly resolutions. The models are compared in zero-shot and fine-tuned settings, with and without outdoor temperature as a covariate, and using two context lengths to assess the impact of model configurations on prediction performance. Two baseline models are included for benchmarking.

This study is structured as follows. Section 2 provides an overview of current studies on chiller energy consumption prediction. Section 3 describes the methodology, including the dataset, preprocessing steps, the models used, and the experimental design. Section 4 presents the results, which are discussed

in Section 5. Section 6 summarises the findings and outlines future research directions.

2 Related Work

In the prediction of chiller systems, a distinction is commonly made between physics-based and data-driven models [24]. As this work focuses on data-driven approaches, the following provides an overview of recent research in this field. Tien et al. [20] present a comprehensive survey of chiller system prediction across spatial and temporal scales, highlighting the diversity and rapid development of approaches. Wang et al. [22] emphasise the challenges posed by high variability and non-stationarity and propose a hybrid method combining wavelet decomposition, Long Short-Term Memory (LSTM) networks, RAdam optimisation, and Pearson correlation-based feature selection [22]. Their results, based on factory data from Xiamen, show improved prediction accuracy compared to baseline models such as linear regression and feed-forward networks. Sulaiman et al. [18] apply a fixed forward Neural Network (NN) optimised with the Teaching-Learning-Based Optimisation (TLBO) metaheuristic to predict energy consumption in commercial buildings, demonstrating significant performance gains. In subsequent work, Sulaiman et al. [19] extend this approach using Kolmogorov-Arnold Networks (KANs), which outperform both TLBO-based and conventional NN models. While previous studies focus on electrical consumption, others address the prediction of the Coefficient of Performance (COP), which requires estimating the ratio of cooling capacity to electrical input. Ho et al. [6] employ a hybrid ARIMA-regression model and identify the part load ratio as most relevant, while Deng et al. [5] use a dynamic graph convolutional network and highlight cooling water return temperature as the key variable. These differing findings underline the system-specific nature of COP modelling.

3 Methodology

3.1 Data and Pre-Processing

Cooling Demand Time Series The cooling demand time series contains electrical energy consumption data of a Panasonic R410A split type air conditioning (AC) unit in the server room of the (office) Building 445 at Karlsruhe Institute of Technology (KIT) and covers February 23 to May 31, 2025. Based on the electrical energy consumption of the AC unit, the actual cooling demand can be calculated with the temperature dependent COP. The raw series is non-equidistant with a temporal resolution of 4 seconds to 3 minutes and 4 seconds and contains several missing segments. Gaps up to 30 minutes are filled via linear interpolation, while longer gaps are imputed with the value observed 135 minutes earlier, corresponding to one hysteresis-control cycle. A more detailed description of the tested imputation methods, as well as the data pre-processing and analysis, can be found in [10].

The series exhibits a daily pattern and a recurring 135-minute cycle from the unit's hysteresis control. As a prediction of the shorter cycle is not necessary to predict the overall cooling demand, the series is smoothed using a rolling window with a window size of 135 minutes. It is further resampled to 15-minute and hourly resolution. Finally, the time series is divided into training, validation, and test sets, with the training set covering data until March 31, 2025, the validation set comprising April 2025, and the test set comprising May 2025. The processed time series is shown in Figure 1. Despite the short time span, seasonal effects causing rising average outdoor air temperatures introduce a concept drift, complicating the forecasting task.

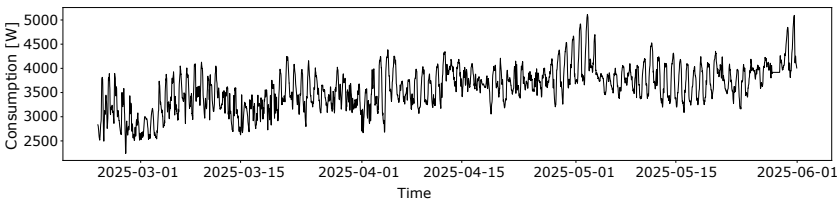


Figure 1: Pre-processed cooling demand time series in **15 minute** resolution.

Outdoor Air Temperature Time Series The Outdoor Air Temperature time series contains measurements recorded every 15 minutes by a weather station on the roof of Building 445. The recording period and the division into training, validation, and test sets are identical to those of the cooling demand series. Different imputation methods were evaluated for the missing segments in the raw series. The selected strategy fills gaps of up to four hours via linear interpolation, while longer gaps are imputed using measurements from the nearby Station 4177 in Rheinstetten provided by the German Weather Service (DWD)¹. Further information on the compared imputation methods is provided in [10]. Finally, the 15-minute series is resampled to derive an additional hourly-resolution series. Figure 2 shows the relationship between the chiller’s energy consumption and the outdoor temperature, with each point colour-coded according to the time of day. At lower temperatures, the relationship exhibits considerably more variance compared to higher temperature ranges.

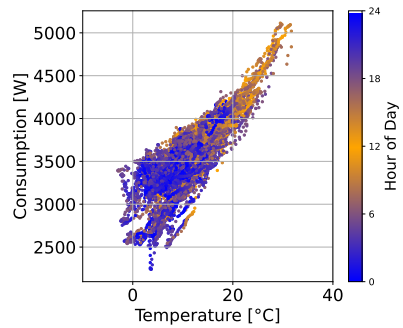


Figure 2: Energy consumption versus outdoor Temperature.

3.2 Compared Models

MOIRAI [23] is built on an encoder-only Transformer and uses a patch-based approach, dividing each time series into non-overlapping segments that are transformed via Multi-Patch-Size Input Projection layers, with patch sizes chosen according to the series frequency (e.g., hourly, daily). MOIRAI is developed as a multivariate model, supporting the integration of both past and future covariates. It is trained on the Large-scale Open Time Series Archive (LOTSA) dataset [23], containing around 27 billion observations from over

¹ Data from DWD Station 4177: https://opendata.dwd.de/climate_environment/CDC/observations_germany/climate/10_minutes/air_temperature/recent/

100 univariate and multivariate time series. However, the included covariates are limited to past covariates, with no future covariates provided. The model is available in Small, Base, and Large variants with 14 to 311 million parameters. This study employs `MOIRAI-1.1R-small`. `MOIRAI` is pre-trained with the Negative Log-Likelihood loss [23]. As fine-tuning on the cooling demand time series showed no improvement with this loss function, it is changed to the packed Mean Squared Error (MSE). More details on this can be found in [10]. `MOIRAI` generates multiple forecast samples, and point predictions are computed as the mean of these samples, as also done in [3].

MOIRAI-MoE [12] is a decoder-only Transformer-based TSFM that extends the `MOIRAI` architecture. Instead of using multiple heuristic projection layers for different frequencies, it employs a single projection layer and addresses time series variability through a sparse Mixture-of-Experts (MoE) mechanism within the Transformer layers. As `MOIRAI`, `MOIRAI-MoE` is developed as a multivariate model, trained on the LOTSA dataset and is available in two sizes (Small and Base). In this study, the `MOIRAI-MoE-1.0R-small` version with 117 million parameters is applied [12]. `MOIRAI-MoE` generates multiple forecast samples, and point predictions are computed as the median of these samples. This approach is used because `MOIRAI-MoE` occasionally produces extreme outlier samples on 15-minute data, which are effectively ignored when using the median. Further details can be found in [10].

Chronos [1] is a framework for probabilistic time series forecasting that adapts language models by scaling and quantizing values into a fixed vocabulary for tokenization. It is primarily based on the T5 encoder-decoder transformer architecture [16] but can also be applied to decoder-only architectures such as GPT-2 [15]. Originally developed as a univariate model, covariates are incorporated via a separate regression model predicting the target variable, while `Chronos` models the residuals between predictions and actual values. `Chronos` is available in different model sizes, and in this study, the `Chronos-Bolt-Mini` variant with 21 million parameters is used. `Chronos` generates multiple forecast samples, with point predictions computed by default as the mean of these samples [1].

Linear Regression As the first baseline model, a linear regression is trained on the training set using the relationship between the consumption of the AC unit and the outdoor temperature. The prediction of the target variable is then obtained by applying the model to the outdoor temperature values within the forecasting window.

Last Day As a second baseline model, the value from the previous day at the same time is used for each corresponding time step within the forecasting horizon. This approach is motivated by the pronounced daily patterns in the data.

3.3 Experiment Design

All models are employed to generate rolling forecasts of the cooling demand time series for the next three days, using a stride of one. Two context lengths, one week and four weeks, are used and compared. Since no fine-tuning script for MOIRAI-MoE is available at the time of this study, this model is applied in a zero-shot (ZS) setting only. In addition to zero-shot evaluation, MOIRAI and Chronos are also fine-tuned (FT) on the training set. The model configurations used for fine-tuning are presented in Tables 1 and 2 and are either recommended in the respective paper or were determined experimentally (see [10]). For all fine-tuned setups, the context length is restricted to one week, as extending it to four weeks would drastically reduce the size of the training dataset.

All models are evaluated both with and without the integration of outdoor air temperature as covariate, in order to quantify the impact of additional information. While the MOIRAI paper suggests support for incorporating future covariates during training, this functionality is not implemented in the available fine-tuning script. Consequently, only Chronos is fine-tuned with covariate information.

Table 1: MOIRAI fine-tuning settings

Parameter	Value
Learning Rate	1×10^{-6}
Batch Size	32
Patch Size ²	32
Batches per Epoch	100
Patience	100
Loss Function	PackedMSE
Optimizer	AdamW

Table 2: Chronos fine-tuning settings

Parameter	Value
Learning Rate	1×10^{-6}
Batch Size	32
Max. Training Steps	10,000
Regression Model ³	Linear
Loss Function	Cross-entropy
Optimizer	AdamW

3.4 Evaluation Metrics

Mean Absolute Error

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i| \quad (1)$$

The Mean Absolute Error (MAE) is the average of the absolute differences between the actual values y_i and the predicted values $\hat{y}_i$, with N denoting the number of data samples times the forecast length⁴. Taking each error without sign prevents over- and underestimations from cancelling out. MAE is computationally efficient, relatively robust to outliers, and easy to interpret due to its consistent units. However, it is non-differentiable at zero, complicating gradient-based optimisation [8].

Mean Absolute Percentage Error

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|y_i - \hat{y}_i|}{y_i} \times 100\% \quad (2)$$

The Mean Absolute Percentage Error (MAPE) measures forecasting accuracy by averaging the absolute percentage differences between predicted and actual

² This patch size is also used during inference.

³ The linear regression model is applied for all model settings with covariate integration (zero-shot and fine-tuned).

⁴ The notation of N , y_i and $\hat{y}_i$ will remain the same for all error metrics and will therefore not be explained again in the following.

values. Errors are relative to the actual value and taken without sign. MAPE is intuitive and allows easy comparison across time series. However, it is sensitive to small denominators, as values near zero can produce undefined or excessively large errors [8].

Mean Squared Error

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (3)$$

The MSE measures the average squared deviation between the actual value and the prediction. By squaring errors, it smooths the gradient for small errors, aiding optimisation. However, this also makes MSE highly sensitive to outliers, which can significantly affect model performance and cause large gradient jumps during backpropagation. The MSE is always positive and ranges from 0 to infinity [8].

Continuous Ranked Probability Score

$$CRPS(F, y) = \int_{\mathbb{R}} (F(z) - \mathbf{1}_{\{y \leq z\}})^2 dz \quad (4)$$

The Continuous Ranked Probability Score (CRPS) is a scoring rule for probabilistic predictions and is defined as in Equation 4. Here, $F(z)$ denotes the predicted cumulative distribution function, $\mathbf{1}_{\{y \leq z\}}$ is the indicator function that equals 1 if $y \leq z$ and 0 otherwise, and $\mathbb{R}$ represents the domain of all real numbers over which the integral is evaluated [9] [13]. To reduce computational effort, the CRPS is commonly approximated as

$$CRPS(F, y) \approx \frac{2}{N} \sum_{i=1}^N \Lambda_{\alpha_i}(q_{\alpha_i}, y), \quad (5)$$

where $\alpha_i \in (0, 1)$ are the quantile levels, q_{α_i} the corresponding quantile predictions, and $\Lambda_{\alpha}(q, y)$ is the quantile loss function defined as

$$\Lambda_{\alpha}(q, y) = (\alpha - \mathbf{1}_{\{y < q\}})(y - q). \quad (6)$$

This approximation is used, for example, in [4] and in [23], and is applied in this study.

3.5 Hardware and Software

The experiments are conducted on a workstation equipped with an Intel Core i7-10700 CPU and an NVIDIA RTX 3070 GPU with 8 GB VRAM. They are performed in a *Python 3.11* environment using libraries such as *Pandas*, *NumPy*, and *Matplotlib*.

4 Experiment Results

Table 3: Forecasting Results of the **15 minute** *Cooling Demand* time series on the test set.

Model	Cov.	Context [Weeks]	Type	MSE [W ²]	MAPE [%]	MAE [W]	CRPS [W]
Lin. Regression	Yes	-	-	62,866.47	5.42	214.53	-
Last Day	No	-	-	81,026.19	5.42	208.69	-
MOIRAI	No	1	ZS	131,741.85	7.56	284.33	220.89
MOIRAI	No	4	ZS	94,476.82	6.51	245.08	187.00
MOIRAI	Yes	1	ZS	91,996.97	6.36	241.90	189.68
MOIRAI	Yes	4	ZS	91,862.53	6.27	240.02	186.06
MOIRAI-MoE	No	1	ZS	125,425.10	7.35	280.98	232.11
MOIRAI-MoE	No	4	ZS	131,008.77	7.49	278.64	227.19
MOIRAI-MoE	Yes	1	ZS	101,789.40	6.66	252.11	237.80
MOIRAI-MoE	Yes	4	ZS	107,645.66	6.62	249.50	218.22
Chronos	No	1	ZS	71,034.98	5.15	195.99	167.68
Chronos	No	4	ZS	69,672.95	4.98	189.32	162.04
Chronos	Yes	1	ZS	18,477.34	2.73	103.03	89.17
Chronos	Yes	4	ZS	17,956.99	2.67	100.50	86.43
MOIRAI	No	1	FT	82,778.91	5.89	222.62	168.31
Chronos	No	1	FT	73,438.70	5.13	195.44	168.69
Chronos	Yes	1	FT	20,095.80	2.88	108.41	93.87

Tables 3 and 4 present the test set results for the 15-minute and hourly time series. On the 15-minute time series (Table 3), Chronos achieves the lowest error across all evaluation metrics in the zero-shot setting when using a four-week context and the temperature covariate. Shortening the context to one week under otherwise identical settings results in only a slightly higher error. Compared to

Table 4: Forecasting Results of the **hourly Cooling Demand** time series on the test set.

Model	Cov.	Context [Weeks]	Type	MSE [W ²]	MAPE [%]	MAE [W]	CRPS [W]
Lin. Regression	Yes	-	-	62,008.75	5.39	213.41	-
Last Day	No	-	-	79,763.28	5.37	207.12	-
MOIRAI	No	1	ZS	66,448.49	5.10	196.72	153.25
MOIRAI	No	4	ZS	60,577.64	4.92	189.15	146.59
MOIRAI	Yes	1	ZS	59,829.38	4.87	188.52	147.26
MOIRAI	Yes	4	ZS	55,855.61	4.69	180.85	142.14
MOIRAI-MoE	No	1	ZS	76,899.71	5.36	207.35	165.91
MOIRAI-MoE	No	4	ZS	65,786.63	5.03	194.23	152.68
MOIRAI-MoE	Yes	1	ZS	63,030.09	4.92	189.85	151.20
MOIRAI-MoE	Yes	4	ZS	59,359.82	4.81	185.51	145.52
Chronos	No	1	ZS	57,716.64	4.72	179.55	143.17
Chronos	No	4	ZS	52,007.28	4.57	173.77	137.56
Chronos	Yes	1	ZS	17,943.55	2.70	101.90	82.90
Chronos	Yes	4	ZS	16,225.05	2.57	96.81	78.05
MOIRAI	No	1	FT	75,852.56	5.56	211.82	160.01
Chronos	No	1	FT	59,535.35	4.80	182.18	145.20
Chronos	Yes	1	FT	17,944.00	2.69	101.68	81.98

the linear regression baseline model, the best Chronos configuration reduces the MSE by up to 71 %. Fine-tuning Chronos results in a slightly decreased performance, compared to the respective zero-shot counterpart. All Chronos models without covariate integration perform significantly worse and fail to surpass the linear regression baseline in both zero-shot and fine-tuned settings.

MOIRAI achieves its best performance in the fine-tuned setting using a one-week context without covariates, although it does not outperform either baseline model. In the zero-shot setting, performance with a one-week context and no covariates is significantly worse, but increasing the context length improves the results. Adding temperature as a covariate leads to a substantial performance gain for the one-week context and a modest improvement for the four-week context. Nonetheless, none of the MOIRAI configurations surpass the weakest Chronos model.

MOIRAI-MoE does not outperform either baseline model in any configuration and performs on par with the worst MOIRAI model. Increasing the context length reduces the error only marginally while incorporating the covariate

noticeably lowers the error. Still, the performance remains worse than that of the best MOIRAI model and all Chronos configurations.

The results on the hourly time series (Table 4) largely confirm those observed on the 15-minute data. Chronos achieves the best performance in the zero-shot setting with covariates and a four-week context, closely followed by the same configuration with a one-week context and the fine-tuned model with a one-week context and covariate. All configurations without covariate perform significantly worse, but unlike in the 15-minute setting, they still outperform both baseline models. Interestingly, fine-tuning both with and without covariates leads to a decrease in model performance.

MOIRAI achieves its best performance in the zero-shot setting using a four-week context combined with covariates. All zero-shot configurations outperform both baseline models, and within this setting, extending the context length and incorporating temperature as a covariate both contribute positively to forecasting accuracy. Compared to Chronos, all zero-shot MOIRAI settings reach similar performance levels to Chronos configurations that exclude covariates. In contrast, fine-tuning with a one-week context results in a significant drop in accuracy, aligning with the degradation observed in Chronos under the same conditions.

Moirai-MoE performs worse on average than both MOIRAI and Chronos. In its best configuration, which uses a four-week context and includes covariates, it outperforms both baseline models. As observed with MOIRAI, extending the context length and incorporating temperature as a covariate lead to slight improvements in model performance.

Table 5: Fine-tuning duration of the **15 min** time series.

Model	Cov.	Context [Weeks]	Learning Rate	Total [min]	Best [min]	Best [Steps]
MOIRAI	No	1	1×10^{-6}	65.35	1.40	300
Chronos	No	1	1×10^{-6}	7.79	0.37	600
Chronos	Yes	1	1×10^{-6}	6.55	1.35	2,200

Table 6: Fine-tuning duration of the **hourly** time series.

Model	Cov.	Context [Weeks]	Learning Rate	Total [min]	Best [min]	Best [Steps]
MOIRAI	No	1	1×10^{-6}	77.97	11.14	1,700
Chronos	No	1	1×10^{-6}	7.20	0.00	100
Chronos	Yes	1	1×10^{-6}	6.85	0.95	1,500

Tables 5 and 6 present the training times and number of training steps for all evaluated model configurations. Notably, both models complete very few training steps on the 15-minute time series when covariates are not included, while Chronos fails to show any training improvement in the corresponding configuration of the hourly time series. The reasons for this behaviour will be discussed in the following section. It is also noteworthy that MOIRAI requires significantly more training time than Chronos, even when the number of training steps is similar.

5 Discussion

The following section presents a detailed analysis of various effects to support the interpretation of the results in Tables 3 and 4. To illustrate these effects, figures showing forecasts from different model configurations are provided and compared.

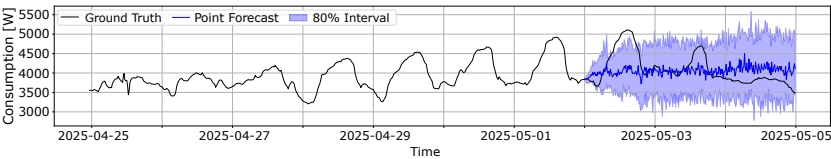


Figure 3: **Resolution:** 15 min; **Model:** MOIRAI; **Covariate:** No; **Type:** ZS.

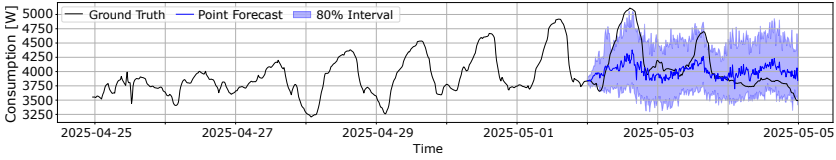


Figure 4: **Resolution:** 15 min; **Model:** MOIRAI; **Covariate:** No; **Type:** FT.

Resolution-Specific Effects (15-Minute Data) Figure 3 and Figure 4 show predictions of the 15-minute time series generated by MOIRAI in the zero-shot and fine-tuned settings, respectively. In both plots, the context length is set to one week and no covariate is used. In the zero-shot scenario, both MOIRAI and MOIRAI-MoE fail to capture the daily pattern of the time series, resulting in the high errors presented in Table 3. Instead, their predictions resemble noise fluctuating around the mean of the input window, lacking any meaningful temporal structure. The limited performance of the two models is likely due to the small proportion of 15-minute data in the LOTSA dataset, which contains only four time series at this resolution [23]. However, this finding contradicts the claim by the authors that MOIRAI-MoE, through its Mixture of Experts approach, should enable frequency-independent forecasting [12]. Accordingly, the daily pattern of the time series should at least be captured in the predictions, since such patterns are present in the model’s training data, though at a different temporal resolution.

In contrast to the zero-shot setting, the fine-tuned MOIRAI model (Figure 4) is able to partially learn the daily pattern of the time series, resulting in a slightly lower error. However, the performance improvement achieved through fine-tuning is relatively modest and the model fails to predict the magnitude of the series.

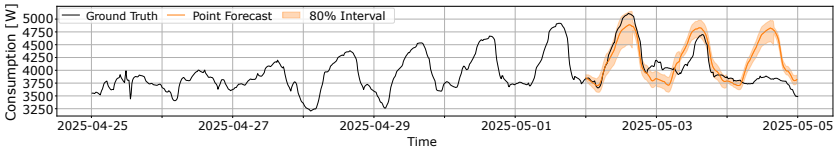


Figure 5: **Resolution:** 15 min; **Model:** Chronos; **Covariate:** No; **Type:** ZS.

Figure 5 displays a prediction of the same forecast window generated by Chronos in the zero-shot setting without covariates. In contrast to MOIRAI and MOIRAI-MoE, Chronos is able to capture the underlying daily pattern of the time series, whereby the predictions are strongly influenced by the temporal dynamics within last days of the input window. This leads to a substantial error on the third day of the forecast horizon, when the time series encounters a structural shift due to a change of outdoor temperature that cannot be inferred from the input data. As a result, MOIRAI and MOIRAI-MoE achieve lower prediction errors on that specific day. When evaluating model performance across the entire test set (see Table 3), this leads to only a small advantage for the Chronos configurations without covariates over the MOIRAI and MOIRAI-MoE variants in terms of average error. However, such aggregated metrics obscure important differences in model behaviour, as the closer examination of individual forecast windows reveals that Chronos is capable of learning and reproducing meaningful temporal patterns, whereas the predictions generated by MOIRAI and MOIRAI-MoE lack clear structure. Notably, despite capturing temporal dynamics well, Chronos tends to be overconfident in its forecasts, as the predicted uncertainty band appears too narrow to adequately reflect the true variability in the data.

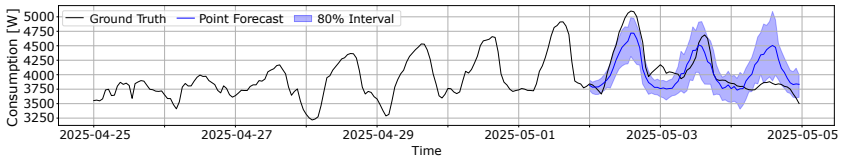


Figure 6: **Resolution:** Hourly; **Model:** MOIRAI; **Covariate:** No; **Type:** ZS.

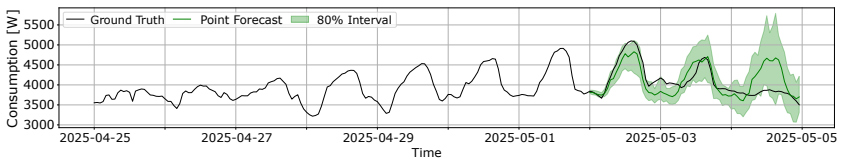


Figure 7: **Resolution:** Hourly; **Model:** MOIRAI-MoE; **Covariate:** No; **Type:** ZS.

Zero-Shot Performance Figures 6 and 7 display predictions generated by MOIRAI and MOIRAI-MoE in zero-shot setting without covariates, applied to the hourly-resolved version of the time series. Both models are now able to capture and reproduce the daily pattern, resulting in a significantly reduced prediction error compared to the 15-minute resolution. However, the models encounter the same issue as Chronos on the third day of the forecast horizon, where an abrupt change in the time series leads to noticeable deviations from the ground truth. Chronos benefits slightly from the lower temporal resolution, likely because resampling filters out high-frequency noise, leading to a modest reduction in prediction error.

Impact of Fine-Tuning As the results in Table 4 demonstrate, fine-tuning MOIRAI and Chronos without the inclusion of covariates does not lead to improved performance, but rather results in a decline in prediction accuracy. This suggests that the time series lacks meaningful intrinsic autoregressive structure, which is further supported by the very low number of training steps observed for the model variants without covariates (see Tables 5 and 6). Since the efficiency of the AC unit, and therefore its energy consumption, is strongly affected by temperature, which follows a diurnal cycle, the visible autocorrelations reflect external periodicity rather than genuine internal dependencies. Without access to temperature data, the model cannot effectively exploit this structure, making fine-tuning without exogenous inputs largely ineffective.

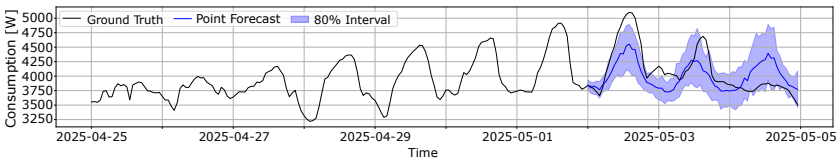


Figure 8: **Resolution:** Hourly; **Model:** MOIRAI; **Covariate:** Yes; **Type:** ZS.

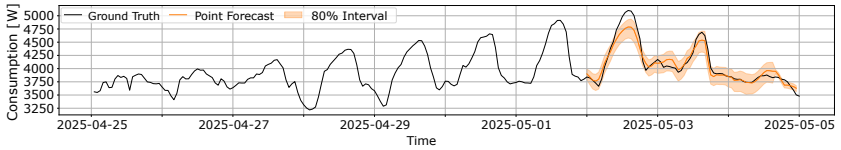


Figure 9: **Resolution:** Hourly; **Model:** Chronos; **Covariate:** Yes; **Type:** ZS.

Effect of Covariate Integration Figures 8 and 9 present predictions from MOIRAI and Chronos in a zero-shot setting, using one week of context and temperature as a covariate. Compared to the equivalent setup without covariates, MOIRAI’s forecasts exhibit noticeable differences, accompanied by a slight reduction in average prediction error across the entire test period. A similar effect is observed for MOIRAI-MoE, indicating that both models incorporate the provided future covariate information into their forecasts despite not having encountered such covariates during pre-training or fine-tuning. Nevertheless, the exact way in which this information is utilised remains unclear. While incorporating the covariate results in a slightly improved performance on the cooling demand series, both models show decreased performances on other time series tested in [10].

In contrast to MOIRAI, Chronos achieves a significant improvement in forecasting accuracy when incorporating the temperature covariate. Several noteworthy observations emerge when evaluating model performance. First, the comparison to the linear baseline is of particular interest, as Chronos incorporates the covariate using a linear regression model that is fit on the training set. Chronos considerably outperforms this baseline, suggesting that it captures additional patterns within the context length, besides the temperature dependency. These patterns may arise from latent relationships involving other variables, such as interactions between the cooling system’s energy consumption and the server room load, which could induce autoregressive structures in the target time series. Secondly, fine-tuning Chronos with covariate integration leads to decreased performance compared to the zero-shot setting, showing the model’s limited ability to generalise the learned temperature dependency across different seasonal conditions. The linear regression model used to incorporate the temperature covariate is trained on data from the colder months of February and March. During this period, the correlation between outdoor temperature and

cooling energy consumption exhibits higher variance (see Figure 2), indicating a less stable linear relationship. In contrast, the test period in May features warmer temperatures and a more consistent correlation pattern, requiring the model to extrapolate beyond the temperature range seen during training. As fine-tuning is conducted on the residuals between the ground truth and the linear regression model's predictions, forecasting accuracy may decline if the residual patterns learned during training do not generalise to the test period. As a result, the model may overfit to residual patterns specific to the training data, which are not representative of the test set.

6 Conclusion

Directly forecasting cooling demand rather than estimating it as part of total building energy consumption provides several advantages, as it enables more targeted energy management, can act as a proxy for larger cooling systems, and offers a lever for demand-side management or grid-friendly cooling strategies. Against this background, this study investigates the use of TSFMs for cooling demand forecasting with outdoor air temperature as covariate. Three TSFMs and two reference models are evaluated on real-world data with less than four months of history, reflecting realistic conditions where only limited data is available. Since none of the models handle preprocessing or missing values internally, and no alternative TSFM that does is available to date, the data is first preprocessed and missing values filled. Afterwards, three-day-ahead forecasts are generated with different model configurations. The results show that Chronos benefits significantly from covariate integration, clearly outperforming baseline models. In contrast, MOIRAI and MOIRAI-MoE do not gain from future covariate information, most likely due to the absence of such data in their pre-training corpora. This highlights both the potential of TSFMs for cooling demand forecasting and the need for improved approaches to incorporating covariates.

This study provides a foundation for further research. Future experiments on a larger dataset, where training data better reflects the temperature ranges of the validation and test sets, could enable fine-tuning with covariates to outperform zero-shot performance. Moreover, while the current evaluation uses

measured temperature data as covariate input, this does not reflect a realistic application scenario. Using weather forecast data as covariate input would allow assessment of its impact on model performance, introducing additional uncertainty. Fine-tuning the models on weather forecast data is also of interest to investigate its effect on probabilistic predictions under realistic operating conditions. Furthermore, Chronos was originally developed as a univariate model and incorporates covariates via a tabular regression approach, meaning it does not directly learn relationships between variables. However, recent advances such as ChronosX [14], which extends Chronos to capture inter-variable dependencies, and COSMIC [2], a multivariate pre-trained model supporting zero-shot forecasting with covariates, have emerged as promising alternatives. Consequently, comparisons with these models, as well as with TabPFN [7], which demonstrates strong performance even on relatively small datasets, is of particular interest.

References

- [1] A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. Sundar Rangapuram, S. Pineda Arango, S. Kapoor, J. Zschiegner, D. C. Maddix, H. Wang, M. W. Mahoney, K. Torkkola, A. Gordon Wilson, M. Bohlke-Schneider, and Y. Wang. Chronos: Learning the Language of Time Series, 2024. URL: <https://arxiv.org/abs/2403.07815>.
- [2] A. Auer, R. Parthipan, P. Mercado, A. F. Ansari, L. Stella, B. Wang, M. Bohlke-Schneider, and S. S. Rangapuram. Zero-Shot Time Series Forecasting with Covariates via In-Context Learning, 2025. URL: <https://arxiv.org/abs/2506.03128>.
- [3] M. Beichter, N. Friederich, J. Pinter, D. Werling, K. Phipps, S. Beichter, O. Neumann, R. Mikut, V. Hagenmeyer, and B. Heidrich. Decision-focused fine-tuning of time series foundation models for dispatchable feeder optimization. *Energy and AI*, 21:100533, 2025. doi:10.1016/j.egyai.2025.100533.

- [4] J. Berrisch and F. Ziel. Multivariate probabilistic CRPS learning with an application to day-ahead electricity prices. *International Journal of Forecasting*, 40(4):1568–1586, 2024. doi:10.1016/j.ijforecast.2024.01.005.
- [5] Q. Deng, Z. Chen, W. Zhu, Z. Li, Y. Yuan, and W. Gui. A performance prediction method for on-site chillers based on dynamic graph convolutional network enhanced by association rules. *Building Simulation*, 17:1213–1229, 2024. doi:10.1007/s12273-024-1136-3.
- [6] W.T. Ho and F.W. Yu. Predicting chiller system performance using ARIMA-regression models. *Journal of Building Engineering*, 33:101871, 2021. doi:10.1016/j.jobe.2020.101871.
- [7] S. B. Hoo, S. Müller, D. Salinas, and F. Hutter. From Tables to Time: How TabPFN-v2 Outperforms Specialized Time Series Forecasting Models, 2025. URL: <https://arxiv.org/abs/2501.02945>.
- [8] A. Jadon, A. Patil, and S. Jadon. A Comprehensive Survey of Regression-Based Loss Functions for Time Series Forecasting. In N. Sharma, A. C. Goje, A. Chakrabarti, and A. M. Bruckstein, editors, *Data Management, Analytics and Innovation*, pages 117–147, Singapore, 2024. Springer Nature Singapore.
- [9] A. Jordan, F. Krüger, and S. Lerch. Evaluating probabilistic forecasts with scoringRules. *Journal of Statistical Software*, 90:1–37, 2019. doi:10.18637/jss.v090.i12.
- [10] A. Kreusel. Integrating Covariates into Time Series Foundation Models for Building Energy Forecasting. Master’s thesis, Karlsruhe Institute of Technology, 2025.
- [11] Y. Liang, H. Wen, Y. Nie, Y. Jiang, M. Jin, D. Song, S. Pan, and Q. Wen. Foundation Models for Time Series Analysis: A Tutorial and Survey. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 6555–6565. ACM, 2024. doi:10.1145/3637528.3671451.
- [12] X. Liu, J. Liu, G. Woo, T. Aksu, Y. Liang, R. Zimmermann, C. Liu, S. Savarese, C. Xiong, and D. Sahoo. Moirai-MoE: Empowering Time

- Series Foundation Models with Sparse Mixture of Experts, 2024. URL: <https://arxiv.org/abs/2410.10469>.
- [13] F. Marchesoni-Acland, R. Alonso-Suárez, A. Herrera, J. Kherroubi, J.-M. Morel, and G. Facciolo. A CRPS Loss for Deep Probabilistic Regression. In *2024 IEEE URUCON*, pages 1–4, 2024. doi:10.1109/URUCON63440.2024.10850406.
 - [14] S. Pineda Arango, P. Mercado, S. Kapoor, A. F. Ansari, L. Stella, H. Shen, H. Senetaire, C. Turkmen, O. Shchur, D. C. Maddix, M. Bohlke-Schneider, Y. Wang, and S. S. Rangapuram. ChronosX: Adapting Pretrained Time Series Models with Exogenous Variables, 2025. URL: <https://arxiv.org/abs/2503.12107>.
 - [15] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
 - [16] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, 2020. URL: <http://jmlr.org/papers/v21/20-074.html>.
 - [17] Statista. Verteilung des Endenergieverbrauchs zu Wärmeezwecken nach Verbrauchssektoren im Jahr 2022, 2024. Accessed: 19. January 2025. URL: <https://de.statista.com/statistik/daten/studie/1463468/umfrage/verteilung-waermeverbrauch-nach-sektoren/>.
 - [18] M. H. Sulaiman and Z. Mustaffa. Chiller energy prediction in commercial building: A metaheuristic-Enhanced deep learning approach. *Energy*, 297:131159, 2024. doi:10.1016/j.energy.2024.131159.
 - [19] M. H. Sulaiman, Z. Mustaffa, M. S. Saealal, M. M. Saari, and A. Z. Ahmad. Utilizing the Kolmogorov-Arnold Networks for chiller energy consumption prediction in commercial building. *Journal of Building Engineering*, 96:110475, 2024. doi:10.1016/j.jobe.2024.110475.
 - [20] P. W. Tien, S. Wei, J. Darkwa, C. Wood, and J. Kaiser Calautit. Machine Learning and Deep Learning Methods for Enhancing Building Energy

Efficiency and Indoor Environmental Quality – A Review. *Energy and AI*, 10:100198, 2022. doi:10.1016/j.egyai.2022.100198.

- [21] Umweltbundesamt. Energieverbrauch für fossile und erneuerbare Wärme, 2024. Accessed: 19. January 2025. URL: <https://www.umweltbundesamt.de/daten/energie/energieverbrauch-fuer-fossile-erneuerbare-waerme>.
- [22] Y. Wang, H. Cheng, H. Chen, M. Ye, Y. Ren, and C. Yang. A hybrid model based on wavelet decomposition and LSTM for short-term energy consumption prediction of chillers. *Journal of Building Engineering*, 99:111539, 2025. doi:10.1016/j.jobbe.2024.111539.
- [23] G. Woo, C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo. Unified training of universal time series forecasting transformers. In *Proceedings of the 41st International Conference on Machine Learning, ICML'24*. JMLR.org, 2024.
- [24] F.-W. Yu, W.-T. Ho, and C.-F. Jeff Wong. Improved energy management of chiller system with AI-based regression. *Applied Soft Computing*, 150:111091, 2024. doi:10.1016/j.asoc.2023.111091.

Vision-Language Model Embodiment for Robot Task Decomposition and Zero-Shot Control via Prompt Engineering

Gautham Ganesh Prasad¹, Darshana Anil Hosurkar¹, Heiko Renz¹, Frank Hoffmann¹, Torsten Bertram¹

¹Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund
E-Mail: gautham.ganesh@tu-dortmund.de

Abstract

This work develops and evaluates a task-planner-based framework for general-purpose robot systems that integrates foundational Vision-Language Models (VLM) with Generalist Robot Policies (GRP) [2]. The VLM decomposes high-level human instructions into executable subtasks using visual feedback, while GRPs translate these into low-level robot actions; verification and recovery modules further enhance robustness. In our implementation, Gemma 3 (27B) is selected as the foundation VLM task-planner over competitive Large Language Model (LLM) and VLM candidates, while Octo [2] and CrossFormer [3] (GRPs) act as executors. The framework reliably achieves task decomposition and supervision, though overall task success is constrained by GRP limitations. To explore alternatives, we adapt Gemma 3 (12B) into a zero-shot Vision-Language-Action (VLA) controller via structured prompting, without finetuning. Task-space prompting produces consistent and adaptive motion, whereas joint-space prompting, though shows intuitive motion, is less reliable, highlighting the promise of foundation VLMs as future visuomotor agents.

1 Introduction

Robotic systems capable of performing diverse tasks, from household chores to complex manipulation, remain a long-standing goal of Artificial Intelligence (AI) and robotics research. Achieving such versatility requires general-purpose robots that can interpret high-level instructions, reason about dynamic environments, and translate goals into physical actions without task-specific programming.

Recent advances in foundation models [3] provide an opportunity to achieve this goal. These large-scale models are trained on diverse datasets covering broad knowledge domains, enabling generalization across tasks with minimal supervision. Two families are particularly relevant: LLMs, which specialize in textual reasoning, and VLMs, which extend this capability to multimodal reasoning by jointly interpreting images and text. Both are typically transformer-based architectures, with VLMs offering direct grounding in visual context, making them especially suited for robotics applications.

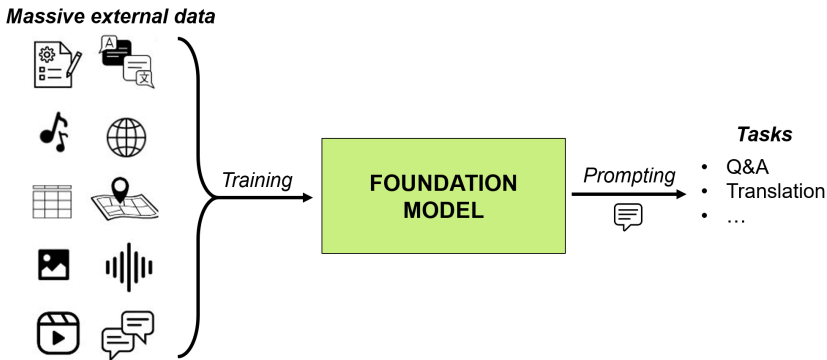


Figure 1: Foundation Models [3]

In digital domains, foundation models already act as versatile assistants, answering questions, summarizing documents, or interpreting images. However, their reasoning remains confined to virtual spaces. Robots, in contrast, often rely on narrowly trained learning-based policies or predefined control scripts. Such controllers limit robots to predictable environments, preventing them from

handling ambiguous instructions, novel layouts, or disturbances, and leaving them unable to carry out high-level goals like “set the table for two” in a context-aware manner.

Recent state-of-the-art systems explore coupling foundation models with embodied agents to enable generalist robotic behavior. SayCan [4] uses a language model (PaLM) as a high-level planner, grounding instructions in a library of pretrained low-level robot skills. Other approaches adopt VLA models, which jointly learn perception, language understanding, and action generation, allowing robots to execute multi-step tasks end-to-end. PaLM-E [5] extends this idea by integrating visual and proprioceptive inputs into a multimodal model capable of producing structured high-level plans. These works demonstrate the potential of combining multimodal reasoning with robot control, while also highlighting a key challenge: effectively executing decomposed plans through low-level controllers. Current strategies either rely on fine-tuned action primitives or pretrained VLA policies, and balancing these approaches remains an open problem in developing scalable, generalist robotic systems.

To bridge this gap, we leverage Generalist Robot Policies (GRPs) [2]. GRPs are pre-trained, goal-conditioned transformer-based models that generalize across diverse manipulation tasks by mapping natural-language instructions and visual input directly to low-level robot actions. Conceptually, these policies allow high-level instructions to be translated into robot actions without task-specific retraining. Representative GRPs in this research include Octo [2] and CrossFormer [3], illustrating the potential of this approach for versatile robotic control.

This work investigates whether VLMs can embed semantic reasoning into robot control and coordinate effectively with GRPs. Specifically, we ask: **Can a general-purpose VLM interpret human instructions, reason about a scene, and guide a robot to execute multi-step tasks in dynamic environments?**

We explore a modular framework in which a VLM serves as a high-level task-planner, decomposing instructions into subtasks, while GRPs handle subtask-level execution. Importantly, this study leverages pre-existing foundation models rather than training new models from scratch, demonstrating the potential of integrating semantic reasoning with embodied control. Our goal is to show that embedding task-planner model and GRPs into modular robotic frameworks

enables more adaptive, instruction-driven robots that operate in unstructured environments, advancing generalist robotic agents from digital reasoning to embodied execution.

2 General-Purpose Robot System Framework

The goal of building general-purpose robot systems using a task-planner-based approach is illustrated using the example of stacking colored blocks in traffic-light order (red, yellow, green) on a plate, as shown in Figure 2. The robot system in Figure 2 is required to recognize the 'Traffic light' order and determine the correct block stacking sequence that matches the intended pattern.

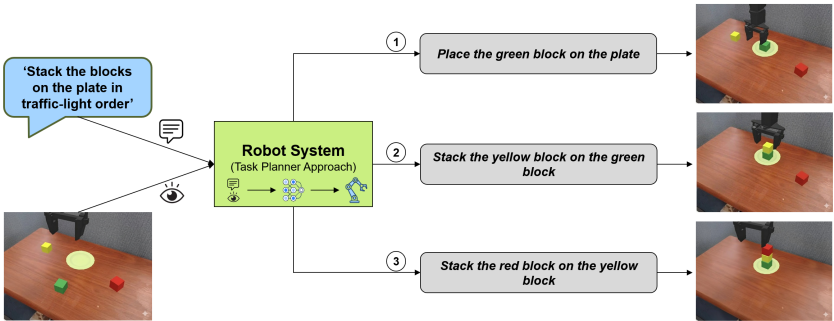


Figure 2: Task Planner Approach: Traffic Light Stack Example

As seen in Figure 3, the central idea is to use a foundation VLM as a high-level planning module that interprets user instructions and visual feedback from the environment, applies real-world knowledge, and generates a logical sequence of subtasks to accomplish the task. The VLM provides these subtasks to a robot control module implemented via a GRP, which translates each subtask into low-level robot actions.

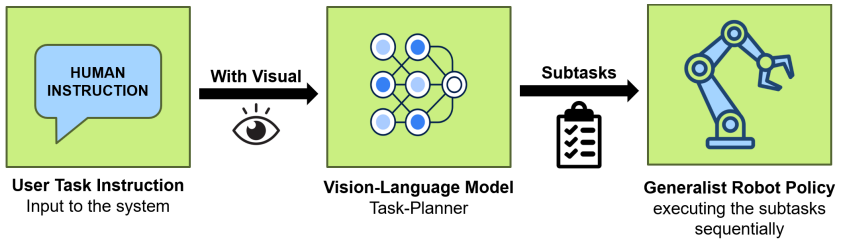


Figure 3: Conceptual Architecture of the Framework

2.1 System Architecture

The system is organized as a closed-loop interaction among the task-planner model, the GRP, and the robot’s environment (see Figure 4). Visual input from the workspace and a high-level natural language instruction are processed by the planner, which decomposes the task into sequential subtask instructions. Each generated natural language subtask instruction is communicated to the GRP, which generates the corresponding low-level robot actions based on current visual feedback.

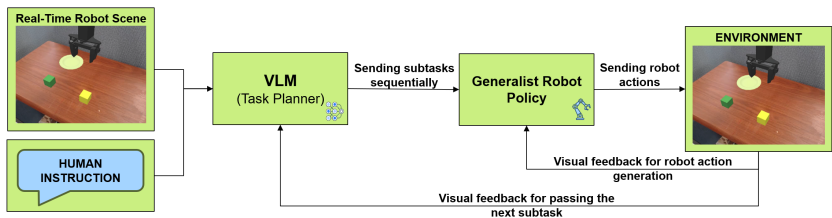


Figure 4: System Architecture

Execution proceeds episodically, with each episode initialized by a randomized workspace configuration. Within an episode, the GRP performs discrete inference-action steps, guided by consecutive image pairs that preserve temporal continuity and allow reasoning over motion and object placement across steps. At fixed intervals, the task-planner VLM monitors progress using visual feedback to advance through the subtask sequence.

2.2 Foundation Model as Task-Planner

A central component of the framework is the task-planner, which leverages a foundation model to decompose high-level human instructions into executable subtasks. The choice of planner directly influences the system’s ability to apply real-world knowledge, interpret ambiguous language, ground reasoning in visual context, and produce reliable action sequences across varying environments.

2.2.1 Candidate Models

Two model families are considered: LLMs and VLMs. While VLMs natively process both images and text, LLMs operate on text only and therefore require an auxiliary vision model (such as BLIP [7], which is used for image descriptions) to function in visuomotor tasks. This modular approach is chosen for comparative analysis because, at the time of this research, foundational LLMs were often superior to their foundational VLM counterparts in terms of general reasoning and performance. The goal of this scientific comparison is to determine the advantages and disadvantages of each paradigm. Candidate LLMs included LLaMa 3.3 (70B) [8], Qwen 2.5 (7B) [9], Qwen QwQ (32B) [10], DeepSeek V2 Distill (7B, Qwen architecture) [12], and DeepSeek V2 Distill (8B, LLaMA architecture) [13]. The integrated VLM candidate is the Gemma 3 (27B) [14] variant.

2.2.2 Evaluation of Candidate Models

Models are evaluated according to the factors in Table 1, with qualitative scores assigned using the rubric in Table 2.

2.2.3 Benchmark Tasks

The following are the benchmark tests used to evaluate model performance across the defined evaluation metrics (see Table 1):

- **Traffic Light Problem:** generate subtasks to stack red, yellow, and green blocks in canonical traffic-light order.

Table 1: Evaluation Criteria for Robotic Prompting Models

Factor	Description
Context Understanding	Correctly interpret user intent and scene context (visual grounding).
Task Decomposition	Breakdown given tasks into logical sequence of subtasks.
Rule Adherence	Follow given instructions for output generation.
General Knowledge	Real world knowledge to inform task planning.
Response Speed	Time taken for response generation.
Consistency	Stable and repeatable outputs without hallucinations.

Table 2: Qualitative Scoring Metric for Model Evaluation

Score	Description
5	Excellent: Ideal performance, highly suitable for direct integration into the robotic instruction pipeline.
4	Good: Near-ideal behavior, with minor issues that are potentially tolerable depending on the task.
3	Acceptable: Integration is possible but would require additional downstream filtering, correction, or support modules.
2	Poor: Significant limitations that reduce applicability in real-world robotic tasks.
1	Unreliable: Extremely poor or random outputs, often misunderstanding the prompt or task completely.

- **Tool Placement Problem:** place multiple tools into target locations while varying prompt formatting to test instruction sensitivity.

2.2.4 Results and Observations

Table 3 summarizes candidate model performance across the six evaluation dimensions, highlighting differences between LLM+BLIP combinations and the integrated VLM.

Table 3: Model Evaluation Across Key Factors (Scores out of 5)

Factor	G3	L3	Q2.5	QwQ	DS-Q	DS-L
Context Understanding	5	5	4	4	3	3
Task Decomposition	5	5	4	2	3	3
Rule Adherence	5	5	3	2	1	1
General Knowledge	5	5	5	3	4	4
Response Speed	5	2	5	5	5	5
Consistency	5	5	3	2	2	2

Model Legend: **G3** - Gemma 3 (27B, VLM), **L3** - LLaMA 3.3 (70B, LLM), **Q2.5** - Qwen 2.5 (7B, LLM), **QwQ** - Qwen QwQ (32B, LLM), **DS-Q** - DeepSeek V2 Distill (Qwen arch., 7B, LLM), **DS-L** - DeepSeek V2 Distill (LLaMA arch., 8B, LLM).

The following are the key observations from the evaluation:

- **LLM + BLIP:** LLaMa 3.3 produces high-quality text reasoning in text-only settings; when paired with BLIP, planning degraded due to occasional caption inconsistencies and increased computational overhead, with response generation time of 65-70 seconds. BLIP remains useful as a generative image captioner but adds latency compared to an integrated VLM.
- **VLM:** Gemma 3 demonstrates stable multimodal grounding, fast response generation (0.6-0.75 seconds per image+prompt cycle), and superior transferability to novel environments and tasks.
- **Comparative outcome:** Gemma 3 exceeds LLM+BLIP combinations in context understanding, task decomposition, rule adherence, real world knowledge, and transferability, while offering better runtime efficiency and operational simplicity.

Based on these observations, Gemma 3 is selected as the framework’s task-planner for integration with robotic control modules.

2.3 Integration with Generalist Robot Policies

Once the task-planner is established, it is integrated with a GRP to execute low-level actions. The VLM-based planner is prompted to generate optimal natural language subtask instructions for the GRP, based on the results from two key preliminary studies.

The first, Performance Benchmarking, tests the GRP’s performance (specifically Octo-Base and Octo-Small) on its own pre-training tasks within a digital twin environment of the pre-training robot setup (SimplerEnv [6]). The second, Instruction Sensitivity, evaluates the GRPs’ performance for different phrasing variations of the same pre-training task to identify the key components of a natural language subtask instruction.

Crucially, these experiments show that Octo-Small generalizes and perform better than its larger counterpart, Octo-Base. This is likely because the finer details of the real-robot pre-training setup are captured by the larger model, reducing its ability to generalize effectively to the digital twin environment. Therefore, Octo-Small is selected for the integration evaluation, and the task-planner VLM is prompted to generate optimally phrased subtask instructions that ensure the best performance from the chosen GRP.

2.4 Features incorporated into the Framework

To mitigate the inherent limitations of the GRP, the framework incorporates several features grouped into three categories: structural, vision-based, and performance-based.

Structural Features enhance the system’s flexibility and execution logic. A key strength is the modular architecture, where the planner and GRP operate as independent components, enabling substitution with alternative models or policies without modifying the overall system. This modularity supports systematic experimentation with different foundation models for task planning

and diverse GRPs for execution, making the framework a flexible testbed for general-purpose robotic systems. Additionally, hierarchical task execution is employed, where the VLM generates a logical sequence of subtask instructions that are executed by the GRP sequentially, establishing a clear execution flow.

Vision-Based Features leverage the full capabilities of the VLM, enabling it to function as a dynamic task manager that actively monitors the environment. This begins with environment state tracking, where the VLM monitors the semantic states of task-relevant objects (e.g., “the green block is on the plate”) using visual feedback and user instruction, rather than relying on precise object pose tracking. This tracking enables the subtask completion tracker, which automatically advances to the next instruction once the semantic state satisfies the current subtask’s conditions. Finally, to ensure robustness against a common flaw observed with learning-based policies, the system continuously monitors the environment state to detect disturbances caused by the random robot actions generated by these policies. If these disturbances compromises a previously completed subtask, the system reverts to the appropriate subtask or prompts the VLM to generate a corrective instruction in real-time, preventing common policy divergence flaws and ensuring stability.

Performance-Based Features enhance the reliability, adaptability, and measurability of the system. A deadlock detection system monitors the robot’s end-effector motion to identify when the GRP fails to generate meaningful actions (near-zero movement). Upon detection, the system prompts the VLM to rephrase the subtask instruction to guide the GRP toward successful completion. Additionally, flexible time limits are implemented, dynamically adjusting the total time allocated per episode based on the number of subtasks required, ensuring fair performance evaluation across varying task complexities. Lastly, a comprehensive logging system records every issue encountered and the time taken for each subtask, providing critical insights for system analysis and performance evaluation.

Integration tests in *SimplerEnv* confirms that the framework reliably executes compound tasks. The selected planner, *Gemma 3*, generates subtask instructions, while the chosen GRPs carries out the corresponding actions under continuous verification. Occasional visual occlusions or misinterpretations are handled by the re-verification mechanism, demonstrating how structural, vision-based, and

performance-based features collectively enable robust, adaptable task execution in dynamic environments.

2.5 Evaluation of the Framework

The framework is evaluated in two robotic implementations to validate modularity, integration, and robustness under realistic task conditions. The first uses *SimplerEnv* [6], a photorealistic digital twin of the pretraining setup for the Octo model. The second implementation uses a Gazebo-based digital twin of the UR3 laboratory robot. This setup is integrated with the CrossFormer model, which has been finetuned for this robot environment by a parallel study on GRPs specifically for integration with the framework. Across both settings, the framework decomposed high-level instructions into structured subtask sequences and leveraged verification, recovery, and performance-oriented mechanisms.

Evaluation focuses on three core metrics reflecting the framework’s objectives: task completion (end-to-end success of instruction execution), task decomposition (quality and logical consistency of VLM-generated subtasks), and operational robustness (stability and adaptability under disturbances or perception errors).

Task completion depend on the combined performance of the VLM task-planner and the GRP. Failures typically arise from (i) VLM misinterpretations of visual input (e.g., false positives due to occlusion) or (ii) GRP limitations, reflecting challenges in zero-shot generalization. Consequently, evaluation emphasizes adaptability, resilience, and robustness rather than raw completion rates.

2.5.1 *SimplerEnv* Implementation

Two compound tasks are evaluated: “*stack the blocks on the plate*” and “*place the blocks on the plate*”. Gemma 3 reliably decomposes high-level instructions into subtasks, while Octo-Small executes the subtasks such as picking and placing. In the *SimplerEnv* evaluation, most failures are observed to result from Octo-Small struggling to complete certain subtasks, particularly those requiring precise placement or stable stacking, while occasional errors stem from VLM misinterpretations of visual input. As illustrated in Figure 5, the planner is seen

prematurely advancing to the next subtask after incorrectly detecting that the yellow block has been placed. These failure cases are effectively mitigated through integrated deadlock detection, re-verification, and corrective instruction mechanisms.

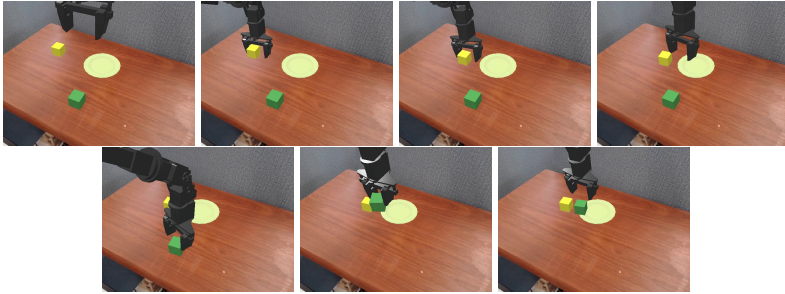


Figure 5: SimplerEnv Implementation Testing of Execution Pipeline

Overall, the SimplerEnv experiments validate the framework’s closed-loop design and modular integration. In terms of the evaluation metrics, the framework demonstrates reliable task completion for simple placements, coherent task decomposition through logical subtask sequencing, and strong operational robustness via its verification and correction mechanisms.

2.5.2 Gazebo Digital Twin Implementation

A Gazebo-based digital twin of the UR3 laboratory setup is used to evaluate the framework in a real-robot-oriented context. The CrossFormer policy, fine-tuned for this robot setup and for pick-and-place task (“*place the blocks on the plate*”), is implemented as the GRP. The fixed environment camera configuration provides clearer visual input, reducing misinterpretation by Gemma 3 compared to SimplerEnv.

CrossFormer executes simple grasp-and-place subtasks reliably, but sometimes shows inconsistencies in precise placement or stacking, as seen in Figure 6 where CrossFormer fails to accomplish the second subtask. Failures such as deadlocks or incomplete subtasks are addressed through deadlock detection, re-verification, and corrective instructions from the VLM.

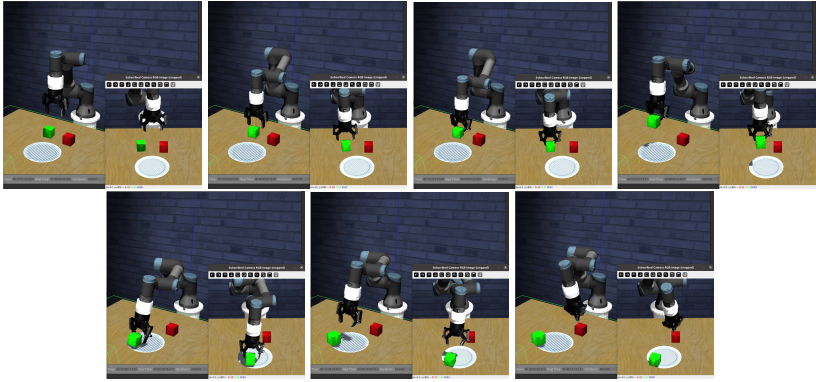


Figure 6: Gazebo Implementation Testing of Execution Pipeline

Overall, these experiments reinforce the VLM’s role as a high-level task manager, maintaining coherent subtask sequencing and adaptive recovery. In terms of the evaluation metrics, the framework demonstrates improved task completion reliability due to clearer visual input, comparable task decomposition quality to the SimplerEnv case, and sustained operational robustness through effective recovery from execution errors and occlusions.

2.5.3 Summary of Evaluation

The evaluation across both robotic environments successfully validates the framework’s core functionalities: translating vague user instructions into structured subtasks, actively managing execution, and adapting to real-time failures. Differences in GRP performance are notable, particularly Octo-Small’s superior generalization compared to Octo-Base and the performance observed with the finetuned CrossFormer model. These variations underscore the critical need for continued development and reliability improvements in GRPs themselves, alongside the necessity of careful GRP selection and structured prompting. Overall, the evaluation confirms that the framework effectively bridges high-level semantic reasoning with low-level robotic control. It also provides a robust and flexible platform for systematic testing and comparison of foundation models and GRPs, highlighting future directions for enhancing general-purpose robotic capabilities.

3 Prompt-Conditioned Action Generation using Gemma 3

Following observations of limited reliability in GRPs, we explore an alternative approach: leveraging the foundational VLM, Gemma 3, to directly generate robot actions, essentially converting it into a zero-shot VLA controller. The aim is to assess the model’s capacity to interpret visual input and produce actionable commands without fine-tuning or explicit knowledge of robot kinematics. Two distinct control modalities are evaluated for action generation: joint-space, where outputs correspond to robot joint angles, and task-space, where outputs specify end-effector positions converted via inverse kinematics.

3.1 Methodology

Gemma 3 is adapted to generate robot actions in place of subtask instructions. Structured prompts provide guidance for visuomotor control, with the VLM receiving consecutive workspace images to iteratively generate incremental motion commands until a subtask is internally considered complete. The model’s large context window (128k tokens) facilitates memory of prior actions and iterative refinement.

Execution follows the incremental approach typical of GRPs (e.g., Octo and CrossFormer), with a motion horizon limited to a single step per iteration. This design choice isolates the effects of prompt engineering and visual input on control. All experiments utilize a UR3 digital twin equipped with an additional environment camera for improved depth perception, integrated via the existing ROS network.

3.2 Experimental Setup

We evaluate two model variants, the 27B parameter model and the 12B parameter model, to assess performance scaling. For each subtask, Gemma 3 receives two consecutive workspace images along with a structured prompt. The model generates incremental actions in either joint-space or task-space until internal completion is indicated. This setup allows a direct comparison of the

two modalities while maintaining consistency with the framework’s evaluation strategy.

3.3 Results and Observations

Joint-space control produces direct incremental joint angles, but is unreliable. While the model intuitively moves the correct joints in the correct direction, indicating task understanding, most executions fail due to frequent collisions or overextensions.



Figure 7: Gemma 3 VLA Joint-Space Control Episode

Task-space control performs significantly better. The end-effector consistently reaches the target within a small tolerance and adapts dynamically to visual feedback. Exploration-oriented prompts further enhance the adaptability and diversity of motion. The modality’s success stems from converting task-space commands to joint angles via inverse kinematics, which allows the VLM to focus on high-level planning while low-level execution is handled externally.

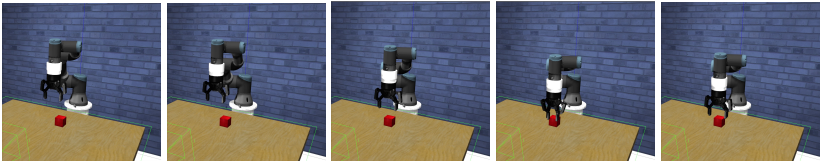


Figure 8: Gemma 3 VLA Task-Space Control Episode

This separation of concerns makes task-space control more suitable for zero-shot visuomotor control. Importantly, both the 12B and 27B Gemma 3 model variants exhibit similar performance in terms of context understanding, rule/instruction following (for action generation) and visual grounding; therefore, the

smaller, more efficient 12B model is preferentially utilized for this direct action generation approach.

3.4 Discussion and Outlook

The experimental results demonstrate the capability of a foundational VLM to effectively function as a zero-shot VLA agent when guided by structured prompting. The strong performance in task-space control confirms that abstraction is key: this modality mitigates the VLM’s limitations in low-level joint-level reasoning by allowing it to focus on high-level planning without requiring explicit knowledge of robot kinematics. This finding is further strengthened by the observation that the smaller, more efficient 12B Gemma 3 model performs comparably to the 27B variant, validating the approach’s practical utility and efficiency.

Conversely, the failures observed in joint-space control underscore the inherent difficulty of managing configuration instability and error accumulation in end-to-end action generation. Even with task understanding, direct joint-angle output requires prompt refinement and sophisticated recovery strategies to manage the cumulative errors inherent in longer-horizon tasks.

Prompt-conditioned action generation with a foundational VLM offers a promising route toward generalist, black-box visuomotor agents. Future directions should focus on visual subgoal chaining for complex multi-step tasks and integrating lightweight motion planning algorithms to enhance feasibility and safety. Furthermore, continued prompt engineering efforts are necessary to improve both error detection and correction mechanisms. Ultimately, this work extends the capabilities of foundation models from general reasoning to effective embodied execution.

4 Conclusion

This work advances general-purpose robotic systems by presenting a modular framework that integrates foundation models, specifically the VLM Gemma 3 as a high-level task-planner, with Generalist Robot Policies (GRPs) such as

Octo-Small and CrossFormer for low-level execution. The framework’s closed-loop design incorporates hierarchical task execution, semantic environment tracking, subtask verification, deadlock detection, and adaptive timing. These structural, vision-based, and performance-oriented mechanisms collectively enhance robustness under uncertainties and dynamic conditions. This design provides an extensible and reproducible platform for instruction-conditioned robotic task execution.

Evaluation in the *SimplerEnv* implementation and *Gazebo-based UR3 digital twin* implementation confirm the framework’s effectiveness. Gemma 3 reliably decomposes high-level instructions, while GRPs execute low-level actions under continuous supervision. The framework successfully mitigates failures, which arise primarily from VLM visual misinterpretations or GRP execution variability, through its verification and recovery mechanisms. This stability demonstrates the value of modular integration for successfully combining high-level semantic reasoning with grounded, embodied control.

Beyond the core framework, we explore Prompt-conditioned Action Generation, showing that Gemma 3 can function as a zero-shot VLA agent without fine-tuning. Structured prompts with visual input enable the direct generation of robot actions. Task-space control yields reliable and adaptive performance, whereas joint-space control exhibits instability, underscoring the benefit of abstracting motion planning from robot-specific kinematics. This approach provides significant insight into prompt-driven robotic control and supports cross-embodiment transfer. Despite GRP limitations constraining overall task success, this research demonstrates that embedding foundation models into modular, prompt-driven frameworks is a promising route toward generalist robotic systems.

References

- [1] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. “Octo:

An Open-Source Generalist Robot Policy”. In: *Proceedings of Robotics: Science and Systems*, Delft, Netherlands. 2024.

- [2] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine. “Scaling Cross-Embodied Learning: One Policy for Manipulation, Navigation, Locomotion and Aviation”. In: *Proceedings of The 8th Conference on Robot Learning*, vol. 270. 2025.
- [3] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. Chatterji, A. Chen, K. Creel, J. Q. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. Gillespie, K. Goel, N. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. Roohani, C. Ruiz, J. Ryan, C. Ré, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang. “On the Opportunities and Risks of Foundation Models.” In: *arXiv preprint arXiv:2108.07258*. 2022. <https://arxiv.org/abs/2108.07258>.
- [4] Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.-H., Levine, S., Lu, Y., Luu, L., Parada, C., Lee, J., Mordatch, I., Nachum, O., Sindhvani, V., Sermanet, P., Tikhonoff, J., Vanhoucke, V., Xia, F., Xu, P., Xiao, T., and Ichter, B. “Do As I Can, Not As I Say: Grounding Language in

- Robotic Affordances.” In: *arXiv preprint arXiv:2204.01691*. April 2022. <https://arxiv.org/abs/2204.01691>.
- [5] Driess, D., Xia, F., Mahdavi, M. S. M., Zeng, A., Rao, K., Wohlhart, P., Xu, P., Gonzalez Arenas, M., Tompson, J., Guadarrama, S., Ichter, B., Chebotar, Y., Sermanet, P., Vanhoucke, V., Hausman, K., Mordatch, I., Levine, S., Greff, K., Dosovitskiy, A., and Florence, P. “PaLM-E: An Embodied Multimodal Language Model.” In: *arXiv preprint arXiv:2303.03378*. March 2023. <https://arxiv.org/abs/2303.03378>.
- [6] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, S. Levine, J. Wu, C. Finn, H. Su, Q. Vuong, and T. Xiao. “Evaluating Real-World Robot Manipulation Policies in Simulation.” In: *arXiv preprint arXiv:2405.05941*. 2024.
- [7] J. Li, D. Li, C. Xiong, and S. Hoi. “BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation.” In: *Proceedings of the 39th International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research, vol. 162, Baltimore, USA. 2022.
- [8] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, *et al.*, and H. Touvron. “The Llama 3 Herd of Models.” In: *arXiv preprint arXiv:2407.21783*. July 2024. doi:10.48550/arXiv.2407.21783.
- [9] Qwen Team. “Qwen2.5: A Party of Foundation Models.” In: *Qwen Technical Blog*. September 2024. <https://qwenlm.github.io/blog/qwen2.5/>.
- [10] Qwen Team. “QwQ-32B: Embracing the Power of Reinforcement Learning.” In: *Qwen Technical Blog*. March 6, 2025. <https://qwenlm.github.io/blog/qwq-32b/>.
- [11] DeepSeek-AI. “DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning.” In: *arXiv preprint arXiv:2501.12948*. 2025. <https://arxiv.org/abs/2501.12948>.
- [12] DeepSeek-AI. “DeepSeek-R1-Distill-Qwen-7B.” In: *Model release note*. 2025. Distilled from DeepSeek-R1 and based on Qwen-2.5

architecture. <https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B>.

- [13] DeepSeek-AI. “DeepSeek-R1-Distill-Llama-8B.” In: *Model release note*. 2025. Distilled from DeepSeek-R1 and based on LLaMA-3.1-8B-Instruct. <https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B>.
- [14] Google DeepMind. “Introducing Gemma 3: Lightweight, open, multi-modal AI models in 1B to 27B sizes.” In: *Google AI Blog*. March 12, 2025. <https://blog.google/technology/developers/gemma-3/>.

An Investigative Study of Transformer-Based Generalist Robot Policies on Physical and Simulated Robotic Embodiments

Darshana Anil Hosurkar¹, Gautham Ganesh Prasad¹, Heiko Renz¹, Frank Hoffmann¹, Torsten Bertram¹

¹Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund
E-Mail: darshana.hosurkar@tu-dortmund.de

Abstract

Robotic manipulation in unstructured environments remains challenging due to the need for policies that generalize across diverse tasks, embodiments, and contexts. Traditional task-specific methods scale poorly, motivating research into Generalist Robot Policies (GRPs) as building blocks for general-purpose robotic systems. This work investigates two transformer-based GRPs, Octo and CrossFormer, which process natural language and visual observations to generate low-level robot actions. Both employ multi-modal embeddings and transformer layers, positioning them as strong candidates for flexible, instruction-conditioned control.

These models are adapted to a UR3 robotic arm for fundamental manipulation tasks, with experiments conducted on both the physical robot and its digital twin using a ROS-based data collection pipeline and fine-tuning strategies tailored to each architecture. Beyond fine-tuning, their ability is evaluated in terms of interpreting language, generalizing to novel tasks, and reasoning over visual attributes such as object color. Results show that CrossFormer consistently achieves higher task success rates and demonstrates greater robustness across environments, while Octo exhibits architectural fragility and limited generalization. These findings highlight the potential of GRPs as building blocks for general-purpose robotic systems, while also underscoring their current limitations in multi-step generalization and embodiment adaptation.

1 Introduction

Robotic systems are increasingly expected to operate beyond rigid industrial workflows, functioning as general-purpose assistants capable of understanding human instructions and performing manipulation tasks in unstructured environments. Achieving such versatility requires control policies that generalize across diverse tasks, embodiments, and contexts, bridging perception, reasoning, and action. Recent advances in foundation models, large pretrained systems based on transformer architectures, have demonstrated strong generalization in language and vision [1]. When adapted to robotics, they give rise to GRPs, which interpret multimodal sensory inputs and translate natural language commands, such as “pick the red block,” into robot actions. GRPs therefore form the core low-level controllers of general-purpose robotic systems, determining how effectively high-level instructions are executed.

This work investigates the potential and limitations of GRPs through three main objectives:

- Evaluate generalization across robot embodiments and environments: Can models like Octo and CrossFormer successfully transfer learned behaviors to new robots and scenes without additional task-specific training?
- Compare strategies for adapting pretrained models to improve task performance: Which parameters should be modified, and how does adapting the models on domain-specific datasets affect previously learned knowledge, including risks of catastrophic forgetting?
- Assess feasibility of integration with vision-language task decomposition frameworks: As future systems increasingly rely on Vision-Language Models (VLMs) to generate subgoals, can GRPs like CrossFormer or Octo robustly execute basic instructed actions such as “pick the red block” as part of larger tasks?

To address these objectives, Octo [2] and CrossFormer [3] are adapted to a UR3 robotic arm, applying fine-tuning strategies on domain-specific datasets and evaluating their performance in both real-world and simulated environments. The study provides empirical insights into the reliability, adaptability, and task-level performance of GRPs as low-level controllers, informing the development

of general-purpose robotic systems capable of interpreting human instructions and executing tasks in dynamic settings.

2 Theoretical Foundations

This section establishes the theoretical background that underpins the study of GRPs as potential low-level controllers for general-purpose robotic systems. It introduces the fundamental principles of neural architectures, attention mechanisms, and highlights how these concepts converge in modern robotics research. The discussion emphasizes the transition from traditional robot control frameworks to learning-based approaches, situating generalist policies like Octo and CrossFormer within this broader context of flexible, instruction-driven robotic autonomy. By outlining these foundations, this section provides the conceptual groundwork necessary for understanding the subsequent methodological choices and experimental design.

2.1 From Reinforcement Learning to Transformers in Robotics

Reinforcement Learning (RL) has long been a central paradigm for robotic control, where agents learn through trial and error to maximize cumulative rewards [4] [5]. RL has enabled progress in locomotion, grasping, and navigation, but real-world deployment remains challenging. Training often requires millions of interaction steps, is unsafe for physical hardware, and yields policies that generalize poorly to new environments or robot embodiments.

To overcome these limitations, robotics research increasingly adopts transformers, first introduced in natural language processing [6]. Transformers leverage self-attention to capture long-range dependencies and fuse multimodal sensory inputs, making them well suited for sequential decision-making. Models such as the Decision Transformer [7] and Trajectory Transformer [8] demonstrate that transformers generate temporally coherent action sequences from multimodal inputs, laying the foundation for their integration into robotic learning and, ultimately, general-purpose systems that require adaptable low-level control.

2.2 From Foundation Models to Generalist Robot Policies

The success of large foundation models in vision and language [9] has inspired analogous developments in robotics. Instead of training task-specific controllers, researchers now pretrain on large, diverse datasets of robot trajectories. Models such as RT-1 [10], RT-2 [11], and π_0 [12] learn multimodal embeddings of images, proprioception, and language. These models are promptable: tasks are specified using natural language or goal images, enabling more flexible deployment than classical policies.

Building on these advances, GRPs, also referred to as Vision-Language-Action (VLA) models, aim to unify task and embodiment generalization. GRPs map high-level instructions to robot actions across diverse tasks and robot morphologies. Many adopt transformer architectures to produce temporally coherent action chunks, which improve motion smoothness and reduce inference latency, supporting their role as low-level controllers in general-purpose robotic systems.

This progression, from RL-based approaches toward transformer-based GRPs, reflects a broader shift from trial-and-error to data-driven multimodal policy learning. Among the most prominent representatives of this new class are Octo and CrossFormer, which take complementary approaches to model design and generalization. The following subsections examine their architectures in detail, highlighting their motivations, structural choices, and implications for deployment.

2.3 Octo: An Open-Source Generalist Robot Policy

Octo, available in Small and Base variants, is a transformer-first, diffusion-decoding generalist manipulation policy that maps language and visual context to temporally chunked continuous control, refer Figure 1. This work focuses on Octo Small, as Octo Base shows poor performance in early experiments. Language instructions are tokenized and embedded by a pretrained T5 encoder [13], images (wrist and third-person) pass through lightweight CNN patch encoders and flatten into patch tokens, and proprioceptive values are linearly projected. All tokens receive positional embeddings and are concatenated into a

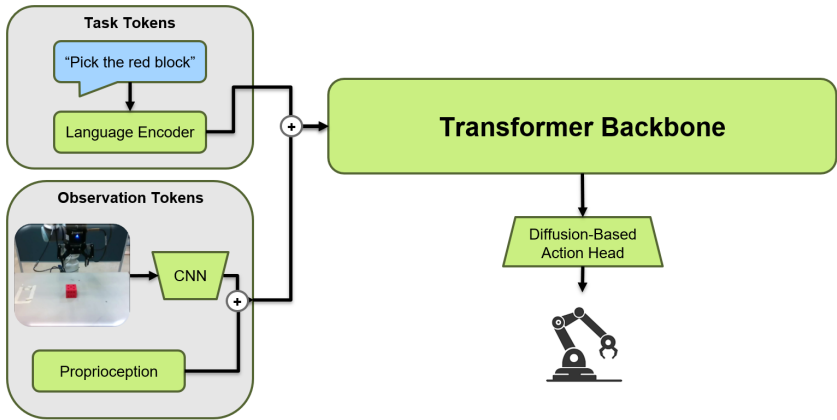


Figure 1: Architecture Overview of Octo

single sequence, together with history tokens when available. That sequence is processed by Octo’s transformer backbone, which uses block-wise masked self-attention to fuse task, history, and perception.

The output is a set of learned readout embeddings acting as the model’s contextual summary and representing a latent space for perception and planning rather than direct motor commands. The readout embeddings are sent to a diffusion action head, implemented as a Denoising Diffusion Probabilistic Model (DDPM)-style denoiser conditioned on the readout embeddings, which performs multi-step denoising to decode a coherent, continuous action chunk. Because decoding is handled in the action head, Octo flexibly adds or swaps heads for different actuation formats (e.g., Cartesian deltas vs. joint targets) without modifying the backbone. Training uses a conditional diffusion loss on action chunks, with gradients backpropagated through both the diffusion head and transformer backbone. At inference time, the pipeline is identical but without gradient updates: the transformer processes the multimodal context once, the diffusion head generates an action chunk through denoising, and the decoded actions execute before the loop repeats with the next observation.

2.4 CrossFormer: A Cross-Embodiment Transformer Policy

CrossFormer is designed as a single, causal (decoder-only) transformer policy for heterogeneous robot learning across many embodiments, refer Figure 2. Inputs are tokenized per modality: workspace, egocentric, and wrist images are encoded with a ResNet-26 encoder [14] and flattened into spatial tokens (goal images, when present, are channel-stacked with current images); proprioceptive inputs are linearly projected to token embeddings; and language instructions are embedded with a frozen T5-XXL [13] encoder. The language embeddings are applied via Feature-wise Linear Modulation (FiLM) [15] conditioning, which modulates intermediate activations of the ResNet-26 visual encoder, thereby injecting task semantics directly into the visual processing stream. Token groups for the observation history, task tokens, and special readout tokens are concatenated into a single sequence and processed by a shared causal self-attention transformer backbone. The readout tokens attend only to past observation and task tokens, producing embeddings that are then routed to small, embodiment-specific action heads.

Unlike Octo’s diffusion decoding, these heads are deterministic linear projection modules that directly map readout embeddings to continuous action vectors, optimized with an L1 regression loss. CrossFormer employs action chunking by matching the number of readout tokens to the action horizon, allowing the same backbone to emit variable-sized outputs for different embodiments: for instance, 2-D waypoints for navigation, Cartesian delta chunks for manipulators, or high-dimensional joint commands for quadrupeds and bimanual robots. The model is trained on a large, cross-embodiment dataset, comprising approximately 900K trajectories across roughly 20 robot types, and the authors report both co-training (single backbone across all robots) and specialized per-robot training. The architecture therefore supports reusing the backbone while adapting or training lightweight heads for new robots. At inference time, the procedure is a forward pass only: observation and language tokens are fused by the backbone, readout embeddings are generated, and the selected action head projects them into executable actions.

Together, Octo and CrossFormer represent two complementary paradigms for GRPs: Octo explores diffusion-based action decoding for smooth manipulation

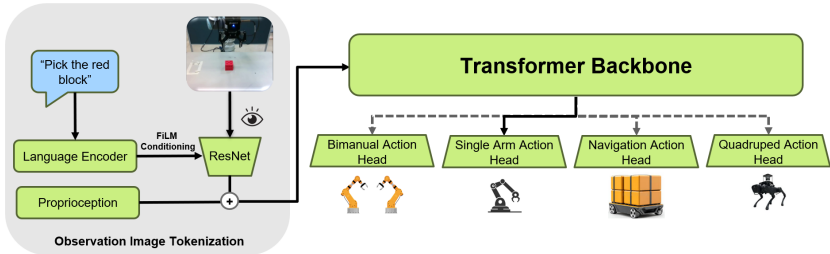


Figure 2: Architecture Overview of CrossFormer

control, while CrossFormer leverages causal sequence modeling with modular action heads for cross-embodiment scalability. Both models highlight the emerging role of GRPs as adaptable low-level controllers within general-purpose robotic systems. The following section describes the experimental setup for evaluating these models on both simulated and real robotic platforms.

3 Experimentation

This section presents the experimental framework developed to evaluate the performance of GRPs on manipulation tasks across real and simulated setups. It details the UR3 and UR5 environments, the dataset collection strategy for grasp, pick, and place tasks, and the pipelines supporting both fine-tuning and inference. The focus is on systematically testing how policies adapt to novel embodiments, task variations, and workspace conditions, with particular attention to training strategies, model capacities, and action representations. Through these experiments, the section reveals both the strengths and limitations of current generalist policies in practical robotic applications.

3.1 Experimental Setup

Experiments evaluate GRPs across physical and simulated environments, forming a coupled physical-digital testbed for fine-tuning and inference. The framework is designed to measure policy adaptability to novel tasks and variations in workspace configurations while maintaining consistency across platforms.

3.1.1 Real-world Setup with UR3 Robot

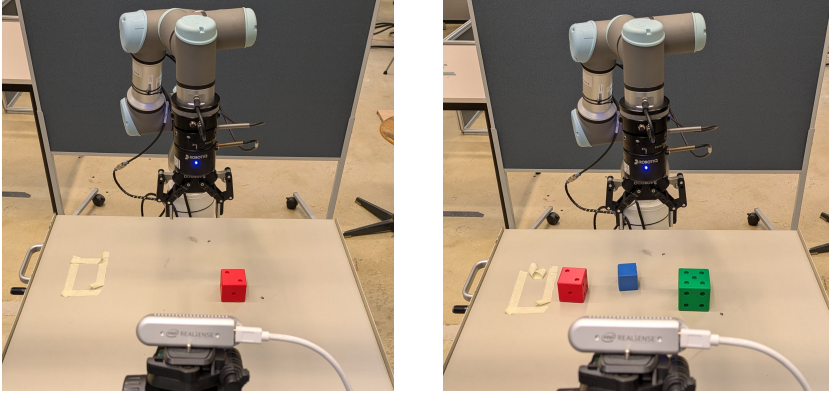


Figure 3: UR3 Real-World Environment Setup

Physical evaluations use a UR3 robotic arm equipped with an adaptive gripper in a fixed workspace for pick-and-place tasks. The primary object is a cube-shaped block, and a fixed camera captures the workspace, see Figure 3. A ROS-based pipeline records joint states, gripper states, end-effector poses, and visual inputs. Standardizing the workspace background reduces visual distractions, and trials with blocks of different colors test policy generalization.

3.1.2 Digital Twin of UR3 in Gazebo

A Gazebo simulation replicates the real-world UR3 setup, including workspace layout, block positions, and background elements, see Figure 4. This digital twin provides a controlled environment for studying model behavior and enables risk-free experimentation and policy testing while reusing the same ROS control scripts and data-collection pipeline. Modifications to the simulation physics ensure consistent grasp execution and support reliable pick-and-place actions.

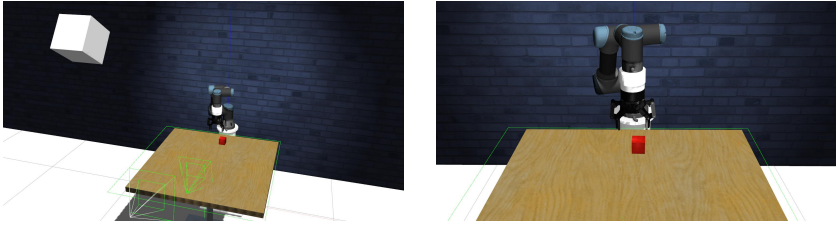


Figure 4: UR3 Digital Twin Environment

3.1.3 UR5 Simulation Setup

To study embodiment transfer and zero-shot performance (evaluating a policy without retraining), a UR5-based simulation mirrors the UR3 workspace, see Figure 5. The UR5’s longer reach and different kinematics allow assessment of pretrained policy performance on a novel robot. The UR5 simulation focuses on reach-dependent behaviors and kinematic generalization, enabling assessment of how differences in reach and joint limits affect zero-shot transfer from UR3-trained policies. By leveraging both UR3 and UR5 setups, the experiments capture differences between real and simulated embodiments, offering a more comprehensive evaluation of policy robustness.

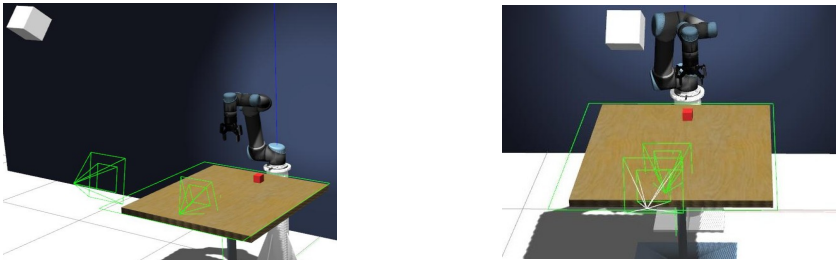


Figure 5: UR5 Simulation Environment Setup

3.2 Data Collection

Data are collected to evaluate the performance and robustness of GRPs on manipulation tasks. The primary tasks are grasp, pick, and place, selected for

their relevance to both research and practical robotic applications.

To systematically assess policy robustness, object poses are varied across three configurations, as seen in Figure 6:

- Fixed pose: standard placement relative to the robot.
- Small perturbations: minor deviations of approximately 2 cm in x-axis and 0.5 cm in y-axis (robot Cartesian coordinates).
- Large perturbations: substantial variations of up to 10 cm in x-axis and 4 cm in y-axis.

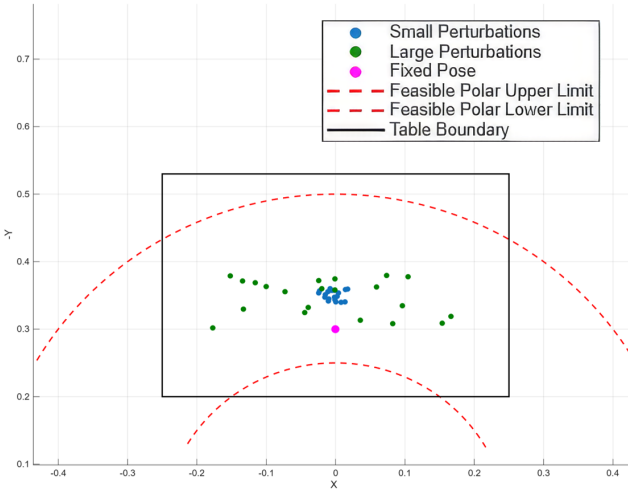


Figure 6: Object Pose Variations for Dataset Collection

For each configuration, 100 episodes are collected at a frequency of 5 Hz, recording joint states, gripper states, end-effector poses, and visual inputs from the fixed camera setup. All datasets are converted to the Reinforcement Learning Datasets (RLDS) [16], a standardized format compatible with several GRPs. This conversion ensures consistent use in fine-tuning pipelines and supports cross-platform evaluation. By including both small and large perturbations, the datasets capture a range of task difficulty levels, enabling assessment of model generalization. Furthermore, these datasets allow for zero-shot evaluation on

new robots or setups, and targeted fine-tuning for improved performance under varying conditions.

4 Fine-Tuning and Inference

This section evaluates GRPs using two complementary strategies: fine-tuning and zero-shot inference. Fine-tuning adapts pretrained models to the specific setup and tasks by leveraging collected demonstrations, enabling tailored performance in the UR3 environment. Zero-shot inference tests the models' ability to generalize to unseen tasks or embodiments without additional training, providing insight into how well pretrained knowledge transfers directly. Together, these strategies establish a balanced framework for assessing adaptability and generalization.

4.1 Evaluation Metrics

Binary success rates are straightforward but fail to capture subtleties in embodied tasks, where failures occur at different execution stages. To provide a more diagnostic view of model behavior, a hierarchical evaluation framework is introduced. This framework spans four levels (L1-L4), reflecting the progression from initiating motion toward the target to completing the full instructed task. Auxiliary checks (L1*, L2*) are included to evaluate critical intermediate aspects, such as gripper alignment and grasp quality, which do not directly cause task failure but affect robustness and repeatability.

This hierarchical structure, summarized in Table 1, moves beyond binary success metrics to reveal where failures occur along the execution pipeline. By separating reaching, grasping, motion, and placement stages, the framework provides a fine-grained view of model performance, highlighting weaknesses and enabling comparison across tasks and policies.

Table 1: Evaluation metrics for task execution

Level	Definition
L1	Initiation of motion toward the target object.
L1*	Verification of gripper alignment to ensure grasp stability.
L2	Execution of grasp through gripper closure.
L2*	Evaluation of grasp stability, centering, and orientation.
L3	Maintenance of a secure grasp during motion.
L4	Successful completion of the instructed task.

4.2 Zero-Shot Evaluation

Zero-shot inference provides an initial benchmark by deploying Octo and CrossFormer directly on the UR3 and UR5 setups without additional training. The UR3 is tested in both real-world and digital twin environments, while the UR5 is evaluated in simulation only. The task in all cases is to pick a block, testing each model’s ability to generalize across embodiments and environments.

Octo performs poorly across all setups: on the UR3, it fails to align or grasp the block, while on the UR5, despite being included in pretraining, it exhibits erratic motion and premature gripper actions. CrossFormer, though unable to complete the task, consistently approaches the block and demonstrates partial spatial and semantic understanding. These outcomes suggest that while pretrained priors enable limited generalization, adaptation through fine-tuning is required for reliable execution. The comparison highlights how zero-shot testing serves less as a deployment-ready solution and more as a diagnostic tool for identifying model biases and limitations. Therefore, all subsequent evaluations focus on fine-tuned results to assess how adaptation improves performance beyond these zero-shot baselines.

4.3 Octo Performance Analysis

The Octo model is fine-tuned on the pick task using the instruction “pick the block”. Two action spaces are examined: delta joint states and delta end-effector

poses. For each configuration, five trials are performed in both simulation and real-world testing. Across configurations, suboptimal performance is observed, with consistent failure modes in simulation and real-world settings. A preliminary evaluation of the grasp task reveals similar limitations, as most trials end in collisions or failed closures, with little reliable behavior observed.

The evaluation results are summarized in Table 2. For the fixed object pose dataset, the block is often not approached (L1). Under small pose perturbations, a single simulation trial reaches L3, where the grasp is maintained after gripper closure; however, this remains an isolated success rather than a repeatable outcome. The model otherwise fails to progress beyond L1, often initiating premature motions or closing the gripper above the object. These behaviors indicate that visual cues are not effectively linked to motion generation, even under minor environmental variations. Performance declines further for large pose changes, where meaningful approaches (L1) are rarely initiated, and most trials end in erratic arm motions or table collisions.

Table 2: Octo’s Inference Results

Dataset Variations	Performance
Fixed Object Pose	Fails to approach the block (L1).
Small Perturbations	Fails to approach the block (L1). Maintains the grasp (L3) once in simulation setup.
Large Perturbations	Fails to approach the block (L1).

Taken together, the results indicate limited generalization across modest task variations. Occasional partial alignments or grasp attempts fail to yield consistent task success, revealing deficiencies in visual grounding and control stability. This outcome is notable since experiments are conducted in a controlled pick-and-place environment with limited workspace complexity. It is concluded that Octo’s pretrained representations and fine-tuned policies remain insufficient for reliable manipulation upon transfer to new robot embodiments and action spaces. The findings highlight the gap between Octo’s promising architecture and practical effectiveness, suggesting that further adaptation strategies or architectural refinements are required for dependable deployment.

4.4 CrossFormer Performance Analysis

CrossFormer demonstrates stronger performance than Octo after fine-tuning, particularly in the delta end-effector pose (task-space) action setting. For each configuration, a total of five trials are conducted in both simulation and real-world testing. A preliminary evaluation on the grasp task shows that the model generally succeeds in closing the gripper around the block, though collisions and weak grasps still occur. On the real UR3, it successfully completes the pick task under both fixed and small pose variations, achieving full task completion (L4). In simulation, it generalizes to larger pose variations, with successful completions observed consistently. These outcomes across dataset variations are summarized in Table 3.

While overall task success is high, partial errors remain. The robot occasionally exhibits misaligned approaches (L1*) and suboptimal grasp quality (L2*). Nevertheless, once the grasp is secured, the object is typically maintained through subsequent stages (L3). Attempts with altered hyperparameters or joint-space action control result in degraded performance, underscoring that the standard task-space configuration (Table 4) provides the most stable outcomes. All explored hyperparameter configurations aimed at improving model performance are listed in Table 5.

Table 3: CrossFormer’s Inference Results

Dataset Variations	Performance
Fixed Object Pose	Successful task completion (L4) for real setup. Fails to approach the block (L1) for simulation setup.
Small Perturbations	Successful task completion (L4) for real setup. Fails to grasp the block (L2) for simulation setup.
Large Perturbations	Fails to approach the block (L1) for real setup. Successful task completion (L4) for simulation setup.

To further examine generalization, different fine-tuning durations are evaluated. Shorter fine-tuning induces a bias toward smaller blocks, whereas longer training improves semantic grounding and enables the robot to identify and pick color-specified blocks. However, in complex multi-step tasks, such as stacking or

placing, the model reliably grasps the correct object but fails to complete placement, reflecting limited generalization beyond single-object pick tasks.

Finally, sim-to-real transfer is explored by deploying a model fine-tuned in simulation on the real UR3. The robot partially reproduces the intended behavior, often approaching and aligning with the block (L1, L1*), but struggles to complete grasps. Common failure modes include table collisions, premature upward motion, and early gripper closure (L2*). These results demonstrate partial but incomplete sim-to-real transfer: CrossFormer retains spatial reasoning capabilities across domains but lacks the precision in action prediction—particularly in gripper control—required for robust real-world manipulation.

4.5 Comparative Analysis

This analysis summarizes key observations from experimental results between Octo and CrossFormer.

Performance Across Embodiments In both real-world and simulated UR3 setups where the embodiment is absent from pretraining, CrossFormer outperforms Octo. Octo often reaches toward the target (L1) but fails to align or grasp the object (L2-L4), indicating limited transfer. CrossFormer achieves L2 and occasionally L3-L4 metrics, demonstrating more stable and adaptable action sequences.

In the UR5 simulation, featuring an embodiment included in pretraining, CrossFormer maintains purposeful object interaction and partial success, whereas Octo fails to approach or grasp the target. Thus, pretraining alone is insufficient for Octo to generalize across embodiments or environments.

Generalization and Robustness CrossFormer demonstrates strong generalization to unseen object poses and workspace variations. It maintains consistent grasping and transport in simplified tasks, achieving up to L3 metrics. However, it occasionally favors smaller objects or misaligns in complex scenarios, reflecting learned visual heuristics rather than deep semantic reasoning.

Octo exhibits poor generalization across domains. Even with pretraining and fine-tuning, it struggles with alignment, stable grasps, and coherent action sequences. Both models are sensitive to restricted fine-tuning, which limits CrossFormer’s performance and prevents Octo from meaningful improvement.

Overall, CrossFormer demonstrates superior task performance, robustness, and adaptability, making it the preferred choice for integration with external task planners. Octo’s limitations highlight challenges in transferring generalist policies across novel robots and environments, emphasizing the need for architectures and training strategies that better support domain adaptation. These findings reinforce that reliable general-purpose robotic systems require strong pretrained priors and well-designed adaptation, motivating CrossFormer’s integration with a task planner to assess its role within modular instruction-driven pipelines.

5 Integration with Task Planner

Following the identification of CrossFormer as the most reliable backbone, its role is explored as a low-level controller within a modular inference pipeline guided by a Vision-Language Model (VLM). The objective is to assess whether a fine-tuned GRP reliably executes subtasks from high-level natural language instructions, thereby bridging task planning and robotic control.

CrossFormer is fine-tuned on a single-block placement task involving the instruction “pick and place the block on the plate” using large pose variations in simulation. This model is integrated with the Gemma-3 VLM, which decomposes the instruction “place the blocks on the plate” into sequential subtasks. The resulting plan requires the same policy to perform two successive placement actions, extending beyond its fine-tuning.

The pipeline demonstrates partial multi-step success. CrossFormer consistently completes the first subtask but fails on the second. These results highlight a limitation: while the VLM provides effective decomposition and sequencing, the GRP generalizes only partially when asked to repeat and extend a learned behavior.

Architecturally, CrossFormer is suited to VLM-driven modularity through its language-conditioned embeddings, vision-proprioception fusion, and token-based temporal structure, supporting flexible task segmentation. In contrast, Octo’s diffusion action head and instability under fine-tuning reduce its suitability for prompt-driven integration. Overall, CrossFormer is a viable robot controller for VLM-GRP pipelines, though precise and reliable multi-step manipulation remains an open challenge.

6 Conclusion

This work investigates the role of Generalist Robot Policies as controllers in modular, language-driven robotic systems. Two state-of-the-art GRPs, Octo and CrossFormer, are systematically evaluated across both simulated and real platforms. Results reveal a consistent adaptability gap: CrossFormer exhibits stronger stability and generalization, while Octo struggles with embodiment mismatches and unstable action predictions.

The integration study further emphasizes both potential and limitations. A CrossFormer model fine-tuned for a single-block placement task successfully executes the first subtask in a two-block scenario but consistently fails on the second, underscoring current challenges in multi-step generalization. Importantly, the integration demonstrates the effectiveness of VLMs for task decomposition, with limitations arising primarily from the policy backbone.

Future work should pursue policies that avoid learning each robot’s kinematics, instead specialize in grasp-trajectory optimization and task-specific execution while deferring kinematic reasoning to adaptation. Develop fine-tuning schemes letting policies attend to new robots’ kinematic structure during adaptation for embodiment-aware joint-space control. Combine such strategies with domain adaptation or real-world augmentations for improved sim-to-real transfer, and explore lightweight GRPs for constrained hardware and VLM integration. These directions can advance GRPs toward dependable, language-driven generalist agents.

7 Appendix

Table 4: Standard Hyperparameter Configurations for Fine-tuning

Hyperparameters Used	Standard Configurations
Fine-tuning Mode	full
Action Space Type	delta end-effector pose
Window Size	2
Fine-tuning Steps	50,000
Warmup Steps	2,000
Learning Rate	3×10^{-4}
Batch Size	8, 10, 12

Table 5: Hyperparameters Explored for Fine-tuning

Hyperparameters Used	Configurations Explored
Fine-tuning Mode	full, head_only, head_mlp_only
Action Space Type	delta end-effector pose and delta joint states
Window Size	1, 2, 5
Fine-tuning Steps	15,000 to 100,000
Warmup Steps	500 to 5,000
Learning Rate	3×10^{-4} , 2×10^{-4} , 1×10^{-4} , 5×10^{-5}
Batch Size	8, 10, 12
Gradient Accumulation Size	4

References

[1] X. Zhang, Y. Zeng, J. Zhang, and H. Li. “Toward Building General Foundation Models for Language, Vision, and Vision-Language Understanding Tasks”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 500–510, 2023.

- [2] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. “Octo: An Open-Source Generalist Robot Policy”. In: *Proceedings of Robotics: Science and Systems*, Delft, Netherlands. 2024.
- [3] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine. “Scaling Cross-Embodied Learning: One Policy for Manipulation, Navigation, Locomotion and Aviation”. In: *Proceedings of The 8th Conference on Robot Learning*, vol. 270. 2025.
- [4] S. Levine, P. Pastor Sampedro, A. Krizhevsky, J. Ibarz, and D. Quillen. “Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection”. In: *The International Journal of Robotics Research*, pp. 421–436. 2018.
- [5] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. “Proximal Policy Optimization Algorithms”. In: *arXiv preprint arXiv:1707.06347*. 2017.
- [6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 6000–6010. 2017.
- [7] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. “Decision Transformer: Reinforcement Learning via Sequence Modeling”. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. 2021.
- [8] M. Janner, Q. Li, and S. Levine. “Offline Reinforcement Learning as One Big Sequence Modeling Problem”. In: *Advances in Neural Information Processing Systems*, vol. 34, pp. 1273–1286. 2021.
- [9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D.

- Amodei. “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901. 2020.
- [10] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-Huei Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu and B. Zitkovich. “RT-1: Robotics Transformer for Real-World Manipulation”. In: *Proceedings of Robotics: Science and Systems*. 2022.
- [11] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. Gonzalez Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. “RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control”. In: *Proceedings of the 7th Conference on Robot Learning*, vol. 229, pp. 2165–2183. 2023.
- [12] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. “ π_0 : A Vision-Language-Action Flow Model for General Robot Control”. In: *Proceedings of Robotics: Science and Systems (RSS)*. 2025.
- [13] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67. 2020.

- [14] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. 2016.
- [15] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville. “FiLM: Visual Reasoning with a General Conditioning Layer”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32. 2018.
- [16] S. Ramos, S. Girgin, L. Hussenot, D. Vincent, H. Yakubovich, D. Toyama, A. Gergely, P. Stanczyk, R. Marinier, J. Harmsen, O. Pietquin, and N. Momchev. “RLDS: An Ecosystem to Generate, Share and Use Datasets in Reinforcement Learning”. In: *arXiv preprint arXiv:2111.02767*. 2021.

Dieser Tagungsband enthält die Beiträge des 35. Workshops Computational Intelligence und präsentiert aktuelle Forschungsarbeiten zu KI-Methoden in Ingenieur- und Steuerungssystemen. Themen reichen von industrieller Fehlererkennung mit Foundation Models, physik-informierten neuronalen Netzen, Benchmark-Datensätzen und Optimierungsalgorithmen bis hin zu Regelungsstrategien, Clustering von Zeitreihen sowie Transformer- und Vision-Language-Modellen für Robotik.