

# **Introduction of an Artificial Intelligence Support System for Production Planning in Manufacturing**

Zur Erlangung des akademischen Grades eines

**DOKTORS DER INGENIEURWISSENSCHAFTEN  
(Dr.-Ing.)**

von der KIT-Fakultät für  
Maschinenbau  
des Karlsruher Instituts für Technologie (KIT)

genehmigte

**DISSERTATION**

von

**Jan Michael Spoor, M.Sc.**

geb. in Ludwigshafen am Rhein

Tag der mündlichen Prüfung:

18.11.2024

Hauptreferent:

Prof. Dr. Dr.-Ing. Dr. h.c. Jivka Ovtcharova

Korreferent:

Prof. Dr. rer. pol. Jens Weber



# Abstract

The development of intelligent manufacturing methods for production planning is driven by the economic objective to decrease the necessary time-to-volume of the production and thus, to achieve a faster market entry of new products. The phase of production planning can be accelerated by implementing AI support systems which enable an evaluation of early planning data. Most useful is the application of AI support systems during the rough planning of production systems within the Digital Factory paradigm since changes can be implemented cost-efficiently during this phase. Knowledge-based systems with embedded ontologies are hereby the foundation for the development of AI solutions in order to enrich data-driven approaches with context and semantic information.

As a result from the review of the body of literature, it is shown that the use cases of process and resource planning utilizing AI models are two under-researched fields in production planning. This research gap is the result of limited data availability, high costs of data pre-processing, and limited interpretability of results. Rule-based or mathematical optimization models are therefore commonly applied in practice but both approaches require high manual efforts and expert knowledge during development. The research question of this thesis therefore focuses on the development of an AI model in the field of process and resource planning which does not rely on manual expert knowledge and is also highly interpretable.

In order to provide a data-driven AI model for the identified use cases, modified Hopfield neural networks for anomaly detection and classification tasks are introduced. The model is based on the main principles of using a database with only correct instances of planning data to circumvent labeling, the usage of ontological

information in order to derive context and similarities between planned and existing production systems, and a high interpretability of the results. The core idea of modified Hopfield neural networks is the modeling of planning data as a graph network. The layout of modified Hopfield neural networks uses the activation of the edges instead of vertices in contrast to conventional neural networks. In order to predict the class membership of objects, the model utilizes an energy definition adapted from the Lenz-Ising model. A high energy is the result of uncommon features of an object and thus, indicates anomalous properties. The weights of the edges are adjusted during the training procedure using a Widrow-Hoff learning rule. As a result, the network layout enables anomaly detection and classification tasks and additionally a recommendation procedure to correct planning data, a similarity measure between classes of the observed production systems, and a highly interpretable decision process.

The model is applied in three use cases from the fields of process planning in manufacturing, synthetic data set generation in production planning and simulation of machinery sequences, and, as an excursus, Leukemia diagnosis using gene expressions. Modified Hopfield neural networks show a high precision across all use cases. In particular, the model outperforms Autoencoders, Isolation Forest, and One-class Support Vector Machines in anomaly detection tasks. In addition, the interpretability and build-in recommendation feature show the practicability of the model which a conventional "black box" model cannot provide.

The application of support systems utilizing a modified Hopfield neural network in the early fault detection during the planning of production systems is therefore recommended. Future research is necessary to prove the modified Hopfield neural network model's applicability across multiple domains and to target a fully autonomous production planning based, among others, on the concepts developed in this thesis.



# Preamble

Firstly, I want to thank Prof. Dr. Dr.-Ing. Dr. h. c. Jivka Ovtcharova, my primary reviewer and supervisor of this thesis. You gave me the necessary guidance but also the space I needed to shape my research. I am very grateful for your provided expertise and your trust in my work.

Secondly, I want to thank Prof. Dr. rer. pol. Jens Weber, my secondary reviewer, supervisor, and colleague at Mercedes-Benz AG. You provided the practical and theoretical knowledge as well as ideas and feedback which enabled my results. Also, your guidance helped me greatly during this process.

Thirdly, I want to thank Matthias Stanka and Dr.-Ing. Dirk Sturzebecher, my superiors at Mercedes-Benz AG. You gave me all required freedom, budget, and support during my research. Without the support by my team at Mercedes-Benz AG, my research project would not have been possible.

Fourthly, there are numerous colleagues, friends, and acquaintances who have supported me in my endeavor with feedback, assistance and contribution during the writing of papers, by providing necessary data sets or competencies during analyses, or with ideas and inspiration. In alphabetical order, I would like to take this opportunity to thank and appreciate the following people: Hélène Arvis, Prof. Dr. rer. pol. Frederik Simon Bäumer, Olaf Buckmann, Sascha Frede, Verena Fröhlich, Markus Gelfgren, Dr.-Ing. Oliver Geißel, Dr. rer. nat. Eileen Giesel, Christian Graewe, Dr.-Ing. Simon Hagemann, Dr. rer. nat. Joschka Kersting, Jens Krknjak, Dr.-Ing. Martin Krockert, Virginie Lerebourg, Prof. Dr.-Ing. Bernd Lüdemann-Ravit, Ulrich Magin, Marvin Matthes, Dr.-Ing. Nicole Schmidt, David Schon, Dawid Stade, Matthias Sinnwell, Quirin Stier, and Benedikt Zimmermann.

Also, a huge thanks to all other people supporting me during this thesis. I am deeply grateful.

Fifthly, I want to thank my family and friends. In particular, I want to thank my parents Maria and Uwe Spoor for supporting me during my academic career and providing the foundation of this thesis by sparking the interest in the sciences during my childhood years.

Lastly and most importantly for me personally, I want to thank my girlfriend Amelie Späthe for your support, patience and love along the way. Thank you for giving me the time and space to accomplish this project. Your contribution is invaluable. I love you. And I hereby ask you:

Will you marry me?

# Table of contents

<b>Abstract</b>	<b>i</b>
<b>Preamble</b>	<b>iii</b>
<b>Abbreviations and Symbols</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation	1
1.2 Background and Initial Situation	4
1.3 Research Objectives	6
1.4 Structure of Doctoral Thesis	8
<b>2 Fundamentals</b>	<b>11</b>
2.1 Production Planning in Manufacturing	12
2.2 The Paradigm of the Digital Factory	23
2.3 Application of Knowledge-based Systems	32
2.4 Fault Detection in Production Systems	39
2.5 Summary	45
<b>3 State-of-the-Art</b>	<b>47</b>
3.1 Methodology of Literature Review	48
3.2 Review of Support System Models	51
3.2.1 Rule-based Models	52
3.2.2 Optimization Models	58
3.2.3 Artificial Intelligence Models	62
3.3 Evaluation of the State-of-the-Art	95
3.4 Open Research Questions	101
3.5 Summary	102

<b>4</b>	<b>Methodology</b>	<b>105</b>
4.1	Design Principles	106
4.2	Objectives	109
4.3	Design of the Support System	110
4.4	Modified Hopfield Neural Networks	114
4.4.1	Neural Network Layout	115
4.4.2	Anomaly Detection Model	140
4.4.3	Classification Model	145
4.4.4	Explainability	152
4.5	Implementation in Production Planning	160
4.6	Summary	168
<b>5</b>	<b>Validation of Methodology</b>	<b>171</b>
5.1	Use Cases	172
5.1.1	Process Planning in the Body-In-White Assembly	173
5.1.2	Simulation of Machinery Sequencing with Synthetic Data	203
5.1.3	<i>Excursus</i> : Classification of Leukemia Subtypes	230
5.2	Discussion	244
5.3	Summary	247
<b>6</b>	<b>Conclusion and Future Work</b>	<b>251</b>
6.1	Conclusion	251
6.2	Future Work	255
<b>A</b>	<b>Appendix</b>	<b>259</b>
	<b>List of Figures</b>	<b>263</b>
	<b>List of Tables</b>	<b>271</b>
	<b>Own Publications</b>	<b>273</b>
	Journal Articles	273
	Conference Papers	273
	<b>References</b>	<b>277</b>

# Abbreviations and Symbols

## Abbreviations

<b>AI</b>	Artificial Intelligence
<b>KPI</b>	Key Performance Indicator
<b>HMI</b>	Human-Machine-Interface
<b>IT</b>	Information Technology
<b>PPR</b>	Product, Process, Resource
<b>ALBP</b>	Assembly Line Balancing Problem
<b>RALBP</b>	Robotic Assembly Line Balancing Problem
<b>RALD</b>	Robotic Assembly Line Design
<b>CIM</b>	Computer Integrated Manufacturing
<b>VDI</b>	Verein Deutscher Ingenieure e.V.
<b>BOM</b>	Bill of Material
<b>CAD</b>	Computer Aided Design
<b>CAP</b>	Computer Aided Planning
<b>BIM</b>	Building Information Modeling
<b>IoT</b>	Internet of Things

<b>OWL</b>	Web Ontology Language
<b>LLM</b>	Large Language Models
<b>AML</b>	Automation Markup Language
<b>DIN</b>	Deutsches Institut für Normung e.V.
<b>ML</b>	Machine Learning
<b>LP</b>	Linear Programming
<b>MIP</b>	Mixed Integer Programming
<b>MILP</b>	Mixed Integer Linear Programming
<b>NN</b>	Neural Network
<b>SVM</b>	Support Vector Machine
<b>ANN</b>	Artificial Neural Network
<b>MLP</b>	Multilayer Perceptron
<b>RBF</b>	Radial Basis Function
<b>DNN</b>	Deep Neural Network
<b>CNN</b>	Convolutional Neural Network
<b>RNN</b>	Recurrent Neural Network
<b>DL</b>	Deep Learning
<b>LSTM</b>	Long Short-Term Memory
<b>SOM</b>	Self Organizing Maps
<b>SVC</b>	Support Vector Classifier
<b>SVR</b>	Support Vector Regressor
<b>KNN</b>	K-nearest-neighbor

<b>LOF</b>	Local Outlier Factor
<b>DBSCAN</b>	Density Based Spatial Clustering of Applications with Noise
<b>HBOS</b>	Histogram-based Outlier Score
<b>ROC</b>	Receiver Operating Characteristic
<b>XAI</b>	Explainable Artificial Intelligence
<b>UX</b>	User Experience
<b>TP</b>	True Positives
<b>TN</b>	True Negatives
<b>FP</b>	False Positives
<b>FN</b>	False Negatives
<b>FPR</b>	False Positives Rate
<b>TPR</b>	True Positives Rate
<b>EMD</b>	Earth Mover Distance
<b>ST</b>	Simple Transition Matrix
<b>TASS</b>	Transition Matrix with Added Sink and Source Conditions
<b>NLP</b>	Natural Language Processing
<b>BIC</b>	Bayesian Information Criterion
<b>MDS</b>	Multi-Dimensional Scaling
<b>AML</b>	Acute Myeloid Leukemia
<b>FAB</b>	French-American-British
<b>APL</b>	Acute Promyelocytic Leukemia

## Latin symbols and variables

$f$	Optimization function
$\mathcal{A}$	Solution space
$\mathbf{x}_i$	Feature vector of object $i$
$P$	Number of inequality constrains
$g_i$	Inequality constrain function $i$
$Q$	Number of equality constrains
$h_i$	Equality constrain function $i$
$x_{i,j}$	Value of feature $j$ of object $i$
$J$	Number of features
$\hat{y}_i$	Predictor of object $i$
$\mathbf{w}$	Weight vector of input neurons
$b$	Bias
$y_i$	True class membership of object $i$
$L$	Loss function
$N$	Number of objects
$\mathcal{D}$	Data set / set of all combinations
$\mathcal{S}$	Mini-batch subset
$P$	Dimensionality of an ANN
$\mathbf{h}_p$	Value of neurons in layer $p$ of an ANN
$\mathcal{P}$	Path of a signal in an ANN in back propagation



$B$	Number of randomly selected data set during bagging
$Z$	Number of nodes in a tree
$\mathcal{R}_z$	Set of objects in node $z$
$s_z$	Threshold for rule of node $z$
$H_z$	Entropy in node $z$
$p_{z,k}$	Percentage of objects from class $k$ in node $z$
$K$	Number of classes
$G_z$	Gini impurity in node $z$
$\mathbf{b}$	Unit vector of hyperplane
$b_0$	Offset of hyperplane
$M$	Margin
$C, C'$	Constant for optimization problem
$h(x)$	Transformation function
$K(x, x')$	Kernel function
$\mathcal{N}_k(x)$	Set containing all $k$ nearest neighbors
$V$	Number of vertices
$E$	Number of edges
$\mathcal{C}$	Set of objects in a combination
$\mathbf{M}$	Connection matrix / tensor
$m_{ij}$	Boolean connection between object $i$ and $j$
$c_i$	Intensity of object $i$
$t_i$	Activation threshold for the intensity of object $i$

$b_{ij}$	Linear threshold parameter between object $i$ and $j$
$b_i$	Linear mitigated threshold parameter of object $i$
$\mathbf{W}$	Weight matrix / tensor
$w_{ij}$	Weight of connection between object $i$ and $j$
$\mathcal{T}$	Training data set
$H$	Energy
$a_{ij}$	Weight for a biased correction value of weight matrix entry $i$ and $j$
$d^t$	Gradient direction in iteration $t$
$\mathbf{w}$	Vectorized nonrecurring entries of weight matrix $\mathbf{W}$
$e_{ij}$	Squared error of weight and connection between object $i$ and $j$
$T$	Number of training data
$G$	Number of ontological groups
$\mathcal{R}$	Set of all relations between objects
$R$	Number of different relations between objects
$c$	Confidence interval factor
$\hat{z}_m$	Predicted anomaly / class assignment of combination $m$
$I_{ij}$	Impact on the energy of a connection between object $i$ and $j$
$\mathcal{K}$	Set of classes
$s_{mk}$	Score of assignment of combination $m$ to class $k$
$\Delta \mathbf{W}_{kl}$	Delta weight matrix between scenario $k$ and $l$
$q_{ijkl}$	Direction score of object $i$ and $j$ for class $k$ and $l$
$\mathbf{D}$	Distance matrix

$C_{ij}$	Ground distance between bin $i$ and $j$
$Q_{ij}$	Source and sink conditions of bin $i$ and $j$
$W_1$	First-order Wasserstein metric
$g_i$	Gene expression intensity of gene $i$

### **Greek symbols and variables**

$\Psi$	Activation function
$\alpha$	Learning rate
$\lambda$	Regularization term
$\eta$	Parameter of RBF function
$\xi_i$	Slack variable of object $i$
$\nu, \mu$	Lagrange multipliers
$\kappa$	Parameter of Tanh function
$\rho$	Local reachability density
$\Phi_i$	Slope of mitigated linear threshold of object $i$
$\theta$	Correction value
$\beta$	Reduction of learning rate
$\gamma$	Decay
$\sigma$	Standard deviation
$\pi_{ij}$	Transition probability from machine $i$ to machine $j$
$\alpha_i$	Supply of bin $i$
$\beta_j$	Demand of bin $j$

$\nu$	Discretized supply distribution
$v$	Discretized demand distribution

## Operators und mathematical symbols

$a, A$	Scalar
$  a  $	Absolute value of scalar $a$
$\mathbf{a}$	Vector
$\mathbf{A}$	Matrix
$\mathbf{A}^{(n)}$	Tensor of order $n$
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$
$\mathcal{A}$	Set
$\  \bullet \ $	Norm / number of entities within a set
sign	Signum function
$\nabla$	Nabla operator
$\partial$	Partial derivation operator
$\langle a, b \rangle$	Inner product of vector $\mathbf{a}$ and $\mathbf{b}$
$d(a, b)$	Distance
$\delta[n]$	Discrete unit sample function
$\mathbf{A} \odot \mathbf{B}$	Hadamard product of matrix $\mathbf{A}$ and $\mathbf{B}$
$\delta_x$	Dirac delta distribution

**General Indices**

smooth	Smoothed loss function
max	Maximum number
stim	Stimulated state
bias	Biased value
eq	Equilibrium state
crit	Critical value



# 1 Introduction

This doctoral thesis is motivated by the economic optimization of production planning by decreasing the error-proneness of production systems during planning and thus fastening the overall timing of the planning process. The first subsection of the following introduction covers the underlying motivation in more detail and also discusses the need for research and industry applications to develop and apply Artificial Intelligence (AI) support systems in the early production planning procedure. This motivation is derived from the body of literature as well as opinions by industry experts and strategic decisions of companies from manufacturing, in particular in the field of automotive manufacturing. The subsequent subsection goes into greater detail regarding the initial situation of this research in production planning enhanced by support systems. Based on this background, the research objectives of this doctoral thesis are specified in the third subsection. Concluding, the overall structure of this thesis is introduced in the fourth subsection.

## 1.1 Motivation

Manufacturing companies from industry sectors such as the automotive industry, energy production, or component manufacturing face the opportunity to utilize an increasing amount of knowledge and information derived from former projects and production operations in their future planning procedures. This is enabled by the availability of larger databases and data lakes at companies and this availability is, in turn, increasingly accelerated by reduced costs of data storage. This kind of Big Data capabilities enable a transition of manufacturing companies towards an intelligent manufacturing approach (Li et al. 2022). Intelligent manufacturing

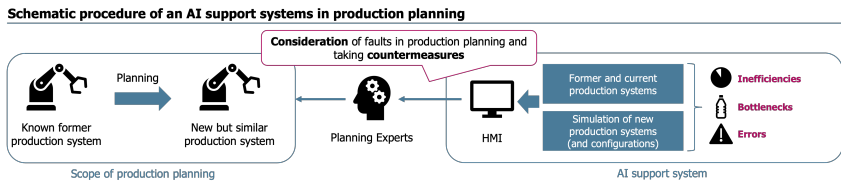
hereby encompasses the utilization of methods from AI, streaming data, and analytics to optimize the production process. Using these data and methods, companies are, in theory, capable of conducting inference procedures for their newly planned production based on known best practices and company-specific knowledge. Methods are applied from the research fields of, among others, statistics, business science, data science and analytics, and data management. However, the practical applied production planning is still rather time-consuming and requires high manual planning efforts because tools are not enrolled at many manufacturing companies due to high implementation costs (Dan Luo and Dolgui 2023). Most notably, the time and resource effort from the start of planning until the so-called time-to-volume, the first mass produced batch of products, is a relevant Key Performance Indicator (KPI) for manufacturing companies (Surbier et al. 2014). Optimizing this KPI enables a faster time-to-market and thus a competitive edge by introducing new products and features earlier. As a result, a fast time-to-volume might result in higher interest by clients due to newly introduced features as a technological market leader or a faster utilization of cost saving or production quantity measures. Furthermore, a reduction of the time-to-volume lowers the overall costs of the production planning. Therefore, the development and implementation of support systems for the early production planning in manufacturing might accelerate the overall planning procedure.

A relevant aspect in this context is the early detection of errors, potential bottlenecks, and production inefficiencies at an early planning stage before more detailed plans or even first constructions of the production facility, production lines, and production cells are erected. If problems and errors are detected in an early phase of production planning and solved prior to a more detailed and thus also time-consuming planning and construction, time is saved by a targeted build-up of an already validated solution and time and costs for remodeling are reduced or even prevented. This is achieved by using prior information about errors, problems, inefficiencies, best practices, and permitted solutions of productions set-ups derived from data of former production lines, cells, and facilities. Surbier et al. (2014) note that experience of former ramp-up of the production should enable faster production ramp-ups in the case that an effective knowledge



management is established within the organization. For the structured evaluation of these data, AI-based approaches are particularly useful since first, they do not require manual evaluation which further decreases planning efforts and second, the necessary data for these kinds of methods is often already available and easily adjusted for the processing by AI methods.

A schematic overview of AI support systems in early production planning is given in Figure 1.1. In summary, if a new production system is planned, the design by planning experts is mostly done based on their knowledge of similar production systems. Similarity is in this context measured by the ability to adopt certain lessons learned or best practices and to reuse prior knowledge. This new but similar production system is then tested by an AI support system for errors, bottlenecks, and inefficiencies based on the aggregated knowledge about all available former production systems and simulations. While the planning is still mostly manually conducted, the validation of the planning is data-driven and automated. The results are returned to the planning experts via a Human-Machine-Interface (HMI).



**Figure 1.1:** Schematic procedure and objectives of an AI support system in early production planning phases in manufacturing.

If such an AI support system is implemented successfully within the early production planning, a manufacturing company could benefit from fast correction of faults during planning and thus, reduce costs by steering the planning procedure as well as reduce costs by preventing faults occurring during stages of planning with a higher degree of finalization of the production system. If faults are detected only late during the production planning procedure, this could increase costs of

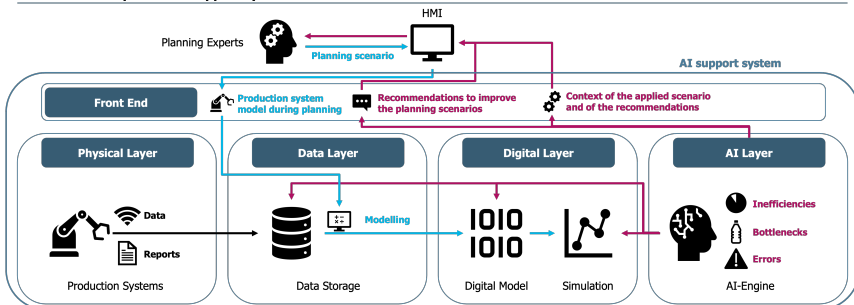
correction and delay the timing of the first batch of production, often named job-number-one.

## 1.2 Background and Initial Situation

In order to utilize the knowledge from prior production systems as discussed, any AI-based method must utilize planning data to recover optimizations and recommendations. One important task during the production planning is the test for consistency within the planned production system and used data model. Thus, authors such as Schmidt et al. (2014) propose increased research efforts in the field of automated consistency checks of engineering results within a tool. In their questionnaire-based survey, Schmidt et al. (2014) found that these tools are often requested by practitioners. Inconsistency of the data model or production system might be a fault or could at least create problems when exchanging data between different planning software and support systems.

As an initial starting point, a support system is broadly described using four layers: the physical layer, the data layer, the digital layer, and the AI layer. The overall architecture is sketched in Figure 1.2.

**Schematic set-up of an AI support system**



**Figure 1.2:** Schematic set-up of an AI support system utilizing a resource, data, digital, and AI layer for the validation of planning scenarios.

The procedure of this support system starts with planning experts designing and modeling new production systems in different scenarios. These planning scenarios are stored and processed within the data layer in order to create a basis for digital models. Thus, the planning scenarios must be processed into a machine-readable format. To also add knowledge from currently used and running production systems, data and reports from multiple production systems are transferred into the data layer. This has been enabled since distributed embedded electronic applications became the norm in industrial production system, in particular in automotive manufacturing (Raptis et al. 2019). Most notably in the field of data management, ontologies provide a machine-readable as well as human understandable way of combining different knowledge domains within a data layer (Feilmayr and Wöß 2016) and can therefore enable support systems in the engineering of production systems, as exemplarily demonstrated by Engel et al. (2018). Using the information from the real and planned configurations of the production system within the planning scenarios, a digital model of the planned production system can be created within a digital layer. This early digital representation can be used to gain further insights into the planned production system. One relevant feature of the digital models of the planned production system is the functionality to run simulations. The resulting simulations, the digital representations of the production system, and the data and knowledge created by the real physical production systems can then be utilized by the AI layer in order to create recommendations to improve the planning scenarios and to provide context to the given scenarios and resulting recommendations. Planning experts can then use this newly created knowledge in order to improve the underlying planning scenarios and production systems during the early planning stages.

This overall conceptualization is used as starting point for the investigation of suitable AI support systems in order to enable a fault detection during the early planning phases. While the final architecture of the support system defined in this doctoral thesis differs in detail from this broadly sketched schematic, Figure 1.2 highlights the background of the underlying thought-process and the initial starting point in order to develop solutions.

## 1.3 Research Objectives

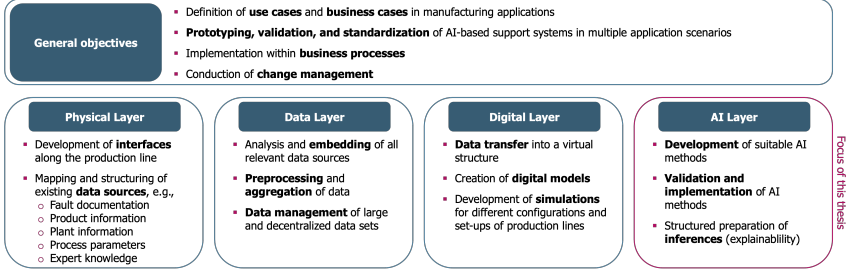
Based on the conceptualization of a support system in order to accelerate production planning, an AI-based fault detection enables use cases which improve the production planning by decreasing construction costs and time of new production systems. Following use cases from an economic business perspective are identified:

1. Improved definition of requirements during the design phase of a production systems, taking into account detected faults during early planning phases
2. System design optimization based on identified causes of faults or optimization of system behavior in the case of occurring faults
3. Improved documentation of the production system behavior
4. Improved risk analysis for fault events
5. Comprehensive analysis of fault events based on detected faults for the prevention of future faults

To fulfill these use cases by developing an AI support system, each identified layer of the system shown in Figure 1.2 has different specific objectives and challenges. While all layers play an important role in the overall performance of this system, this doctoral thesis focuses on the AI layer. Thus, the relevant task becomes the development of suitable AI methods to conduct a fault detection in order to feedback recommendations and useful context to improve the planning scenarios. In addition, it is important to validate these AI methods and prove that they are capable of this task in a practical application. This also includes the requirements for an implementation. Lastly, the developed AI method must yield a high explainability in order to provide structured inference for planning experts using the AI support system. A high explainability is also crucial for an acceptance of the system by users. The relevant objectives and challenges for the development of an AI support system are given in Figure 1.3. In addition to the

discussed objectives of the AI layer, the objectives and requirements of the other layers are also listed.

#### Objectives and challenges along different layers of the AI support system



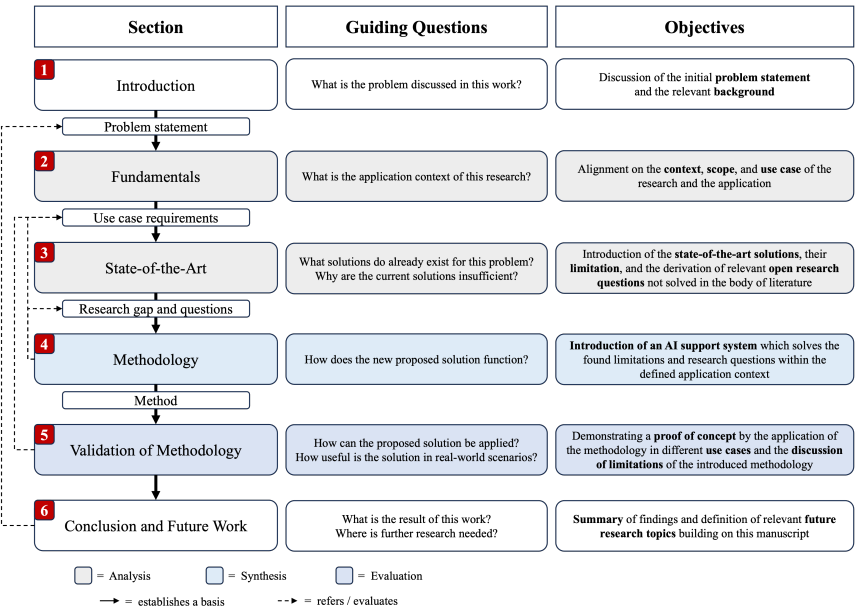
**Figure 1.3:** Overview of important objectives and challenges along general requirements and the different layers of an AI support system.

While all parts of the support system play a crucial role, the AI method are an open field of research of particularly high interest. The listed requirements regarding the physical, data, and digital layer are currently solved in the manufacturing industry by the development of ontologies and by embedding these ontologies into knowledge-based systems. However, the capability to derive useful and comprehensive recommendations using AI methods is still limited and research as well as application are discussing multiple methodologies in order to find explainable as well as precise AI methods. Usuga Cadavid et al. (2020) for example note that the smart manufacturing process design, which would be an essential part of the planning of a production system, is an under-researched field compared to scheduling tasks.

The objective of this thesis can therefore be summarized as the development of an AI method which is capable of a fault detection applied on early planning data. In addition, this method must be embedded into an overall support system and the interfaces to the physical, data, and digital layers must be modeled. An important requirement is hereby to ensure a high explainability of the AI model.

# 1.4 Structure of Doctoral Thesis

In order to answer the defined objectives, this doctoral thesis is structured in six consecutive sections. Each section has different objectives and guiding questions. They build on each other to form a conclusive introduction into the proposed AI support system and show a solution to the presented research question. By answering the corresponding guiding question of each section, a comprehensive understanding of the relevant context of this research, the limitations of the state-of-the-art of AI-based as well as more conventional support systems, the model behind the introduced AI support system, and its application is created. The sections, guiding questions, and objectives are summarized in Figure 1.4.



**Figure 1.4:** Overview of the structure of the thesis and the subsequent guiding questions and objectives.

This Section 1 covers the research questions of this thesis and the relevant background of the underlying initiative funding and supporting this work. Namely, it

introduces the overarching question on how an economic optimization of early production planning using an AI support system can be realized.

Section 2 goes into detail about the application context of this research. Thus, the section gives a brief overview of the necessary background knowledge. First, the basics of production planning are introduced, in particular in the context of manufacturing. Second, the Digital Factory as an important paradigm in modern production planning is presented and discussed for this specific application and use case. The Digital Factory provides the context of overall engineering, Information Technology (IT), and planning systems into which the proposed support system is embedded. Knowledge-based systems are hereby important tools for any kind of information transfer within manufacturing. Based on this consideration, the concept of ontologies for a structured data exchange and documentation within a Digital Factory are presented in the third subsection. Lastly, the identified use case of early fault detection in production planning is discussed and first indications about relevant capabilities of a model as basis for a support system are provided.

Section 3 answers what kind of methods or tools are already developed and implemented for the Digital Factory in research and application. Thus, this section covers the state-of-the-art solutions for fault detection in planning data by support systems in manufacturing processes and their limitations. First, the methodology of the literature review is introduced, followed by an individual introduction of the three main approaches in the body of literature: rule-based models, mathematical optimization model, and AI-based models. These state-of-the-art solutions have different advantages and limitations which are discussed in a third subsection. It is shown that the derived limitations make it necessary to develop a new solution, namely the introduced AI support system. Subsequently, the open research questions based on the limitations of the body of literature are listed in a fourth subsection.

Using the context discussed in Section 2 and based on the limitations of the current state-of-the-art solutions in Section 3, Section 4 introduces the developed novel AI support system. First, overall design principles derived from the limitations of the state-of-the-art solutions are listed and discussed. Based on these design

principles, the objectives of the AI support system as a production planning tool within the Digital Factory are listed. These objectives are derived from the presented fundamentals. Based on the design principles and objectives, the proposed design of the support system is described in detail in the following third subsection. The design comprises six core functionalities of the system along the production planning process which is covered by the support system. As the underlying AI model behind this tool of the Digital Factory, modified Hopfield neural networks are introduced in the fourth subsection. These single-layer neural networks were first introduced by Spoor and Weber (2024) and are here developed in more detail and generalized for broader applications in anomaly detection and classification. In addition, the explainability of modified Hopfield neural networks is discussed. Building on the proposed AI model, the fifth subsection shows how an embedding of the introduced support system into production planning as a further tool of the Digital Factory can be achieved.

Section 5 covers the guiding questions on how the proposed support system can be applied as well as the question about the usefulness of the proposed tool in practice. Using the designed support model (or respectively separable parts of the support model) of Section 4, two relevant use cases within manufacturing from the field of process planning and resource planning are presented in the first subsection. These use cases utilize real world data, and the results are evaluated on their applicability in the real-world scenarios. In addition, the introduced modified Hopfield neural network is tested in a use case from another domain, namely gene expressions of patients diagnosed with leukemia subtypes. Thus, it is shown that modified Hopfield neural networks are applicable in production planning for manufacturing as well as in other domain applications. The results are then critically discussed in the second subsection and limitations of the proposed support system are named.

Lastly, the Section 6 summarizes the results of this work and provides a concluding assessment of the proposed AI support system and the relevant future research topics, which are resulting directly from this work, are presented.



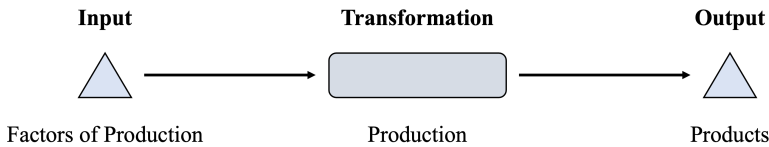
## 2 Fundamentals

As the fundamental application area of the introduced support system, the task of production planning is defined, different phases of the planning procedure are distinguished, and important concepts in an application of production planning are derived in the first section. One important concept in production planning is the Digital Factory which is covered by the subsequent second subsection. The Digital Factory is defined and the classification of an AI support system within the tools of the Digital Factory is conducted. The next fundamental concept of production planning is the application of ontologies and knowledge-based systems which is presented in the third subsection. Thus, the subsection introduces the core idea and application scenarios of ontologies. Lastly, the task of anomaly and fault detection within production planning and the Digital Factory is introduced in the fourth section since these methods are the core function of the AI support system introduced in this thesis.

The fundamental concepts presented in this section provide the readers with more details about the application and research use case of AI support systems in manufacturing and also build a basis for the subsequent discussion of methodological requirements applied in the development of the support system. Therefore, this section covers the important background and design paradigms to understand the use case and set-up of the introduced support system. In addition, this section refines and focuses the research along the discussed use case.

## 2.1 Production Planning in Manufacturing

Since the support system discussed here is part of the production planning procedure, it is important to first define production planning, the corresponding tasks, and the relations to the production process itself. In order to contextualize the tasks of production planning, the higher-level definition of production is given by Dangelmaier (2009) as a transformation process from inputs into outputs. Inputs are the factors of production and outputs are the corresponding products resulting from the transformation. This relation is highlighted in Figure 2.1.

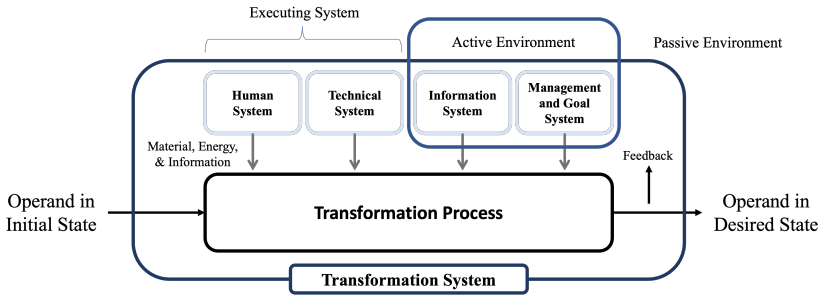


**Figure 2.1:** Production as an input/output transformation process as described by Dangelmaier (2009).

Bellgran and Säfsten (2010) differentiate the transformation process and a transformation system. A transformation system performs a process, which is guided and driven by an operator, on an operand and thus transforms an existing state into a desired state (Hubka and Eder 1988). The transformation as described by Dangelmaier (2009) is not a "black box" but can be analyzed and divided into multiple subsystems of the transformation system, where the operators are the human system, the technical system, and an active environment, which encompasses the information system and the management and goal system. Thus, the occurring transformations are change processes, where value is added to the operands (Bellgran and Säfsten 2010).

The function of the transformation system is the purpose of the system (Hubka and Eder 1988). Production systems are hereby an application of a transformation system, and in the case of a production system the function is the transformation of raw materials into components or products and combinations of these transformations (Bellgran and Säfsten 2010). Thus, each system can generate outputs,

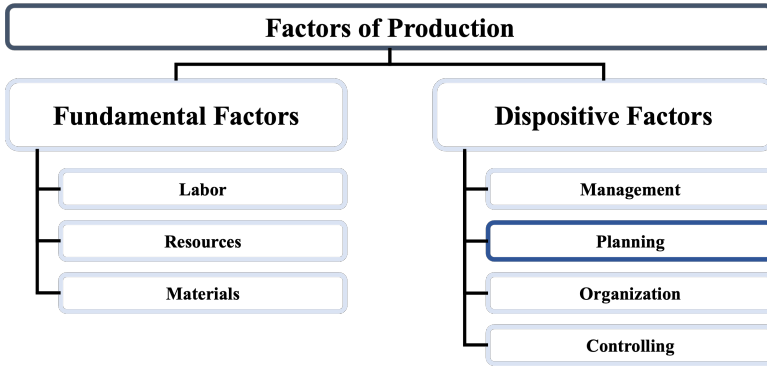
which act as inputs and thus factors of production for subsequent systems. To measure the performance of a production system and thus the economic efficiency of a transformation processes, KPIs such as production rate and throughput can be used for an optimization of the system (Li and Meerkov 2009). The concept by Hubka and Eder (1988) is visualized in Figure 2.2.



**Figure 2.2:** A transformation system model as described by Hubka and Eder (1988).

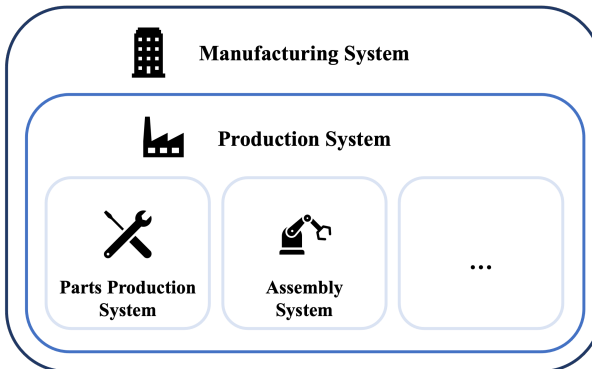
Working with the commonly used definition within the German economic and production research by Gutenberg (1984) and Wöhe et al. (2023), the production factors are divided into the fundamental factors including labor, resources (e.g., robots in the case of manufacturing), and materials (e.g., components as well as additives such as oil or glue in the case of manufacturing) and the dispositive factors which include the management, planning, organization, and controlling tasks of the production process. Thus, the task of production planning is hereby a dispositive factor of production. The categorized factors of production within research are listed in Figure 2.3.

In order to distinguish the priorly used terms of manufacturing system and production system, as well as the in the field of production planning commonly discussed optimization of assembly systems, it is useful to discuss a manufacturing operation using system theory as a "system of systems" as described by Bellgran and Säfsen (2010). The manufacturing system is viewed by Bellgran and Säfsen (2010) as the superior system which comprises the production system but



**Figure 2.3:** Fundamental and dispositive factors of production.

also activities such as design, material supply, planning and production, quality assurance, distribution, management, and marketing. *Vice versa*, a production system as part of the manufacturing system comprises the parts production and the assembly system as well as other possible subsystems. This differentiation and the hierarchical relation are schematically highlighted in Figure 2.4.



**Figure 2.4:** Manufacturing, production, part production, and assembly system based on the differentiation by Belgran and Säfsten (2010).

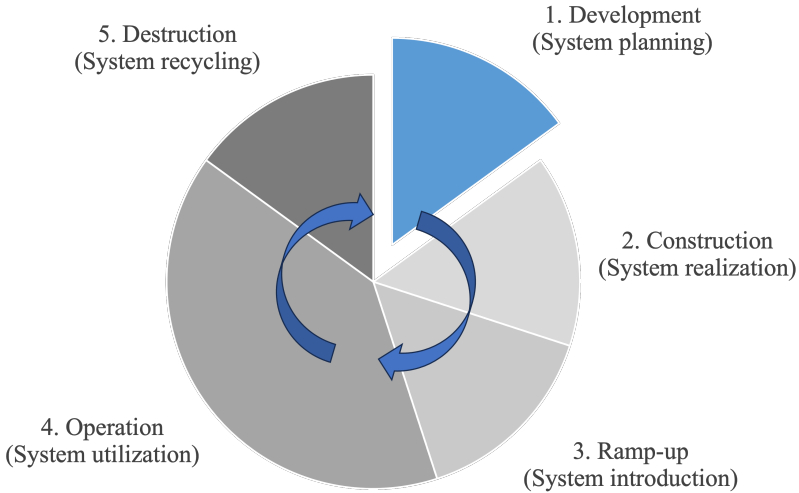
Focusing on the overall manufacturing systems, Gelders and Van Wassenhove (1981) define the production planning as the decision tasks for the usage of production resources in order to satisfy a forecast demand of outputs for a planning horizon and under the consideration of a reasonable amount of accumulated costs of production. Gelders and Van Wassenhove (1981) also differentiate between the mass production of outputs, in particular cars, and the continuous production in bulk of commodity, the batch production in the case of insufficient demand for mass production, or the job production where outputs are produced when ordered.

Thomas and McClain (1993) define production planning as the process of determining a tentative plan. This plan encompasses the amount of production within a planning horizon and determines expected inventory, labor demands, and resources. Thomas and McClain (1993) differentiate the more detailed production scheduling, which focuses on the coupling of individual resources and products using smaller time units. Furthermore, the paradigm of hierarchical production planning is often additionally applied, which is defined as a narrowing of the details and aggregation level of the planning along the timeline. Higher planning levels focus on a longer planning horizon and provide lower planning levels with objectives and constraints, which are themselves focused on shorter planning horizons (McKay et al. 1995).

Applying a hierarchical production planning, it is useful to distinguish production planning by its timeframe into short-term, mid-term, and long-term planning. Hagemann (2022) includes in the short-term planning tasks such as scheduling, and in contrast, includes in the long-term planning the aspect of the manufacturing system planning. Since the research background of this doctoral thesis is in the planning of manufacturing systems, this aspect of production planning is focused.

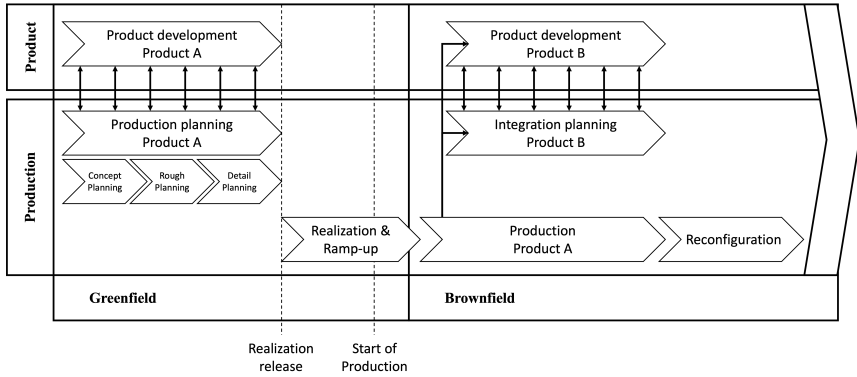
The planning of a manufacturing system is the first phase of a system lifecycle proposed by Schenk and Wirth (2004) followed subsequently by the construction and the ramp-up of the manufacturing system. The longest phase of a manufacturing system lifecycle is the operation of the system. At the end of the lifetime of the corresponding product, the deconstruction starts which may result in a recycling

or re-use of the manufacturing system or some of its parts. The system lifecycle applied for manufacturing system is illustrated in Figure 2.5.



**Figure 2.5:** System lifecycle of manufacturing systems based on Schenk and Wirth (2004).

In more detail and within the context of the superior manufacturing systems, Hagemann (2022) defines the production planning as the technical, functional, and economical planning of a production system. The system planning includes the process planning and the assembly planning and can again be separated into the production planning of a new production system for developed products and the integration planning where a production system is altered or adjusted so that a new product can be produced. Hagemann (2022) generalizes the production planning phases into three sequential phases of concept planning, rough planning, and detail planning. This follows the hierarchical production planning paradigm since more details and precision is added along the maturing planning phases. Using a visualization based on Hagemann (2022) and Kiefer et al. (2017), the production planning and product development phases are given in Figure 2.6.



**Figure 2.6:** Phases of production planning based on Hagemann (2022) and Kiefer et al. (2017).

The greenfield phase covers the planning activities in the case of a new developed manufacturing system, while the brownfield phase includes the planning activities regarding a manufacturing system already in use. All authors highlight the iterative nature of the production planning which results from the coordination with the product development (Hagemann 2022, Kiefer et al. 2017, Burr 2008). However, Kiefer et al. (2017) uses the term production engineering instead of production planning and just separates concept and detail planning. In contrast, Hagemann (2022) adds the rough planning and names the production planning in regard to the new development of a manufacturing system for a new product type or in regard to the integration into an already developed manufacturing system.

Using the definitions and description by Hagemann (2022), Walla (2015), and Burr (2008), the planning phases are defined for this doctoral thesis as the following:

1. **Concept Planning** begins directly after the start of product development which means that no detailed information about the product is available. However, first variants of the production system can be developed by using former product types. In addition, first estimations about the number of assembly processes, components, and materials can be drawn. This results in an estimation of costs and floor space required for the manufacturing system.

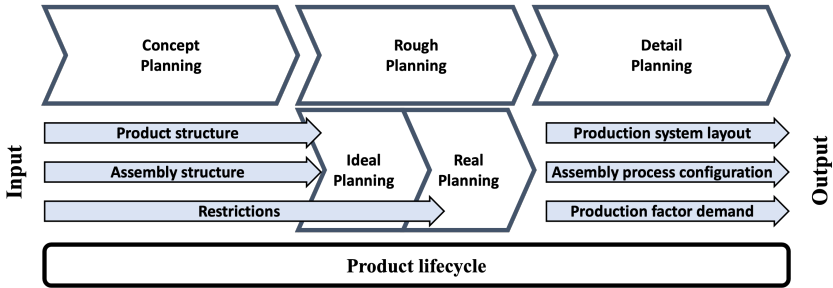
2. **Rough Planning** includes in addition the current preliminary product data from the product development. Therefore, the planning experts can utilize product model structures and geometry information. This results in first assembly structures considered during planning. However, since the product development is not finalized in this phase, changes might occur which require an adjustment of the planning. Nevertheless, planning experts are able to develop first production system layouts, assembly processes, and production system structures.
3. **Detail Planning** includes the planning of the final cell layouts, planning of production factors such as materials and labor, offline programming of resources such as robots, and the virtual safeguarding of the facility.

In addition, Hagemann (2022) further separates the rough planning phase into an ideal planning and real planning. While ideal planning covers the optimal manufacturing system layout, the real planning includes the restrictions, norms, and assumptions under the given facility, product, and budget. Hagemann (2022) notes that within the manufacturing industry only the concept and rough planning are conducted in-house, and the detail planning is contracted to external suppliers using an engineering-to-order or configure-to-order process. Thus, the rough planning is of great importance within planning scenarios in the manufacturing industry since it describes the first technical and economical feasible solution. Furthermore, the contracting procedure is based on this rough planning and thus might highly affect budgeting, economic forecasting, supplier selection, and supplier coordination.

Figure 2.7 gives an overview of the ideal and real planning as part of rough planning. The planning horizon is the whole lifecycle of the product, and the factory is built and operated for a specific product until it is discontinued. The ideal planning receives a product model structure, e.g., as prototype or 3D model, and an assembly structure to build a manufacturing system which enables the assembly of this specific product. In the real planning, the budget and product constraints and restrictions are added to the planning task so that an economic feasible manufacturing system can be designed. The output of the rough planning



includes a production system layout, the assembly process configuration, and the required demand for the fundamental factors of production, i.e., labor, materials, and resources. Using this information, a contracted supplier is enabled to construct the manufacturing system.



**Figure 2.7:** The scope of production planning of the production system, assembly process, and production factors within manufacturing.

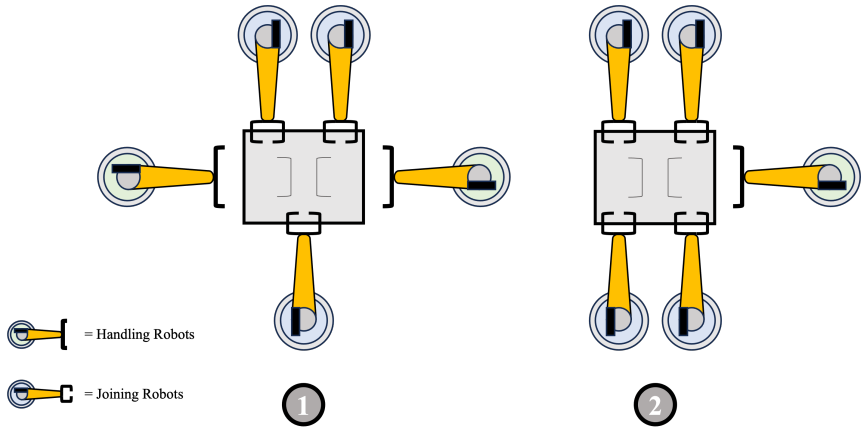
The output of the rough planning in most application is received in the form of an ontological network or can be converted into an ontological network. E.g., Kiefer et al. (2018) create a design graph which is capable of illustrating specific solution variants in order to design an assembly system for a specific product. These kinds of approaches require a graph-based design language as prerequisite within the production planning. However, ontological design languages are further adapted, and their utilization is seen as the current state-of-the-art in the manufacturing industry. Thus, the introduced support system must utilize the resulting rough planning outputs in form of ontological design networks or related concepts.

As important basis within the rough planning, Hagemann (2022) names the product, process, and resource (PPR) ontology applied within the Digital Factory. The concept of the Digital Factory was developed for the tasks of production planning in order to create an interface between all digital systems and simulations and thus, is a logical consequence of the ongoing digitalization and improved computational capabilities. This enabled an optimization of the product, process, and resource planning using visualization and modeling techniques to fulfill the

time-to-market requirements of modern manufacturing and also optimizes cost and product quality requirements (Schäfer et al. 2022a).

In more detail, the planning of the assembly system, an important part of the production system in manufacturing applications, is often discussed in the body of literature as the Assembly Line Balancing Problem (ALBP). The problem deals with the question of how to assign different tasks to workstations in order to optimize a given objective function, e.g., total costs of assembly or throughput, without violating restrictions such as precedence relationships and under consideration of process times (Boysen et al. 2007). With the introduction of robotic assembly methods, this task is today often adapted to the Robotic Assembly Line Balancing Problem (RALBP) which was first publicly introduced by Rubinovitz et al. (1993). In order to achieve more practical realism in the industrial production planning of assembly lines, Michels et al. (2018) introduce Robotic Assembly Line Design (RALD). In contrast to RALBP, RALD focuses on the capabilities of tools and thus takes parallel workstations, loading and unloading times, and equipment selection additionally into account. An assembly line is schematically visualized in Figure 2.8 using two sequential workstations with a different number of robots utilizing handling and joining processes which are responsible for picking, placing, or joining parts.

ALBP are in the body of literature further clustered in consideration of their specific use case. ALBP Type I targets the minimization of the number of workstations using a given cycle time. On the other hand, Type II problems aim at the scheduling of task to a set of predefined workstations in order to minimize the cycle time. Additionally, Type E problems aim at solving a multi-objective problem of optimizing the number of workstations and simultaneously cycle times. Type F problems are aiming at an efficient scheduling and examine whether a feasible solution exists given a certain set-up of workstations under a given cycle time. Chutima (2022) notes that most research is focused on Type II problems of task scheduling and the models are limited in their applicability by considering only a single objective. To meet industry standards, multi-criteria optimization problems should be targeted in the future.



**Figure 2.8:** Schematic visualization of a car body construction assembly line with two sequential workstations. The robots are equipped with handling and joining tools.

The result of an ALBP are therefore specific arrangements of resources or a job scheduling of product-related tasks. However, Hagemann and Stark (2020) note that the planning of a real production system additionally takes into account building geometry, robotic simulations, material flow simulations, buffers, and decoupling concepts. Thus, the resulting assembly lines by a RALD algorithm are tested and validated with at least a rough estimation by experts. It is concluded that the solutions of ALBP might yield overall useful results but still require manual corrections and further might not include all restrictions of a real production systems which currently hinders their wide-scale autonomous application.

Concluding this section, the relation between support systems and production planning is discussed. In general, support systems, often also called expert systems, decision support systems, or knowledge-based systems, aim at providing users with essential information, recommendations, and warnings during conducted planning activities. Support systems are therefore computer programs which use knowledge and deduction measures in order to solve problems which otherwise would require human expert interaction (Kumar 2019).

Support systems are responsible for increasing the degree of automatization of planning tasks with the long-term objective of an autonomous planning. Müller et al. (2021) name systematic process execution, adaptability to uncertainty, self-governance, and self-contentedness of the system as key characteristics of an industrial autonomous system. In contrast, an industrial automation system only utilizes a systematic process execution, and an intelligent automation system yields in addition a degree of adaptability (Müller et al. 2021). Thus, most practically applied systems in manufacturing categorize as industrial automation systems since task such as governance and external manual input by experts is still required.

Since a full autonomous production planning is currently not realized, one important task of support systems in production planning, regardless of whether the system is autonomous, automated, semi-automated, or manual, is the detection of faults in the rough planning stage of a production system, the validation of planning set-ups, and the recommendation of feasible solutions (Schmidt et al. 2014). Currently, this task is still conducted manually by production planning experts in many industry applications, as also mentioned by Hagemann and Stark (2020). Thus, an important aspect of support systems is the fault detection, recommendation, and validation within the rough production planning.

The scope of this doctoral thesis is further refined by focusing on the rough production planning using ontological designs applied in the Digital Factory of production systems. This is reasonable since within the application of the manufacturing industry itself and not its suppliers, an application in rough planning is the focus of the corresponding production planning departments. Thus, the introduced support system should cover the concept and rough planning phases. Using this bordering, the detail planning and the successional tasks of production scheduling or integration during operations as part of the production planning during operation is not the focus of this thesis.

## 2.2 The Paradigm of the Digital Factory

In order to develop an intelligent manufacturing, different paradigms to achieve this objective were developed. Within Europe, new concepts are often covered under the term Industry 4.0. This research field resulted in paradigms such as the Digital Factory and Virtual Factory, in particular within European research and industries initiatives, while in Chinese research and industry initiatives the paradigm of Cloud Manufacturing is focused. Both research paradigms focus on the development of the factory of the future but differ in some respects. Cloud manufacturing is derived from the concepts of cloud computing, the virtual factory is based on the environments of manufacturing (Salierno et al. 2019).

In addition, the Digital Factory is often described as successor of the concept of Computer Integrated Manufacturing (CIM). The CIM concept focuses on the integration for computer-aided subsystems into the organization of the factory (Bracht and Masurat 2005). This concept is regarded as a near-failure in its industrial application due to the inherit complexity of the system integration (Cagliano and Spina 2000, Bracht and Masurat 2005). However, CIM enabled improvements in production planning from an academic standpoint and nowadays have laid the foundation of the Digital Factory, which is focused on the interface of computer-aided tools for product development, production planning, and factory operation (Bracht and Masurat 2005).

This thesis focuses on an application of support systems within the framework of the Digital Factory as a commonly used paradigm within the European manufacturing industry, and its fundamental ideas behind digital planning activities are discussed in this section. However, different definitions exist for the Digital Factory in the body of literature. In particular within the context of the German manufacturing industry, the Verein Deutscher Ingenieure e.V. (VDI) is a leading organization in defining concepts, norms, and guidelines. Since the definition of the Digital Factory by the VDI is rather universally applicable, this manuscript uses the VDI definition for the Digital Factory. The definition is given as follows:

Digital factory is the generic term for a comprehensive network of digital models, methods and tools - including simulation and 3D visualisation - integrated by a continuous data management system.

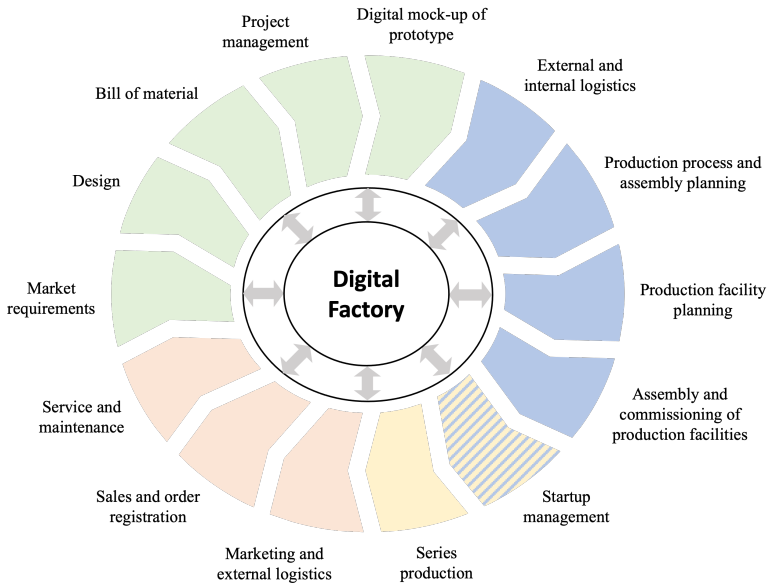
Its aim is the holistic planning, evaluation and ongoing improvement of all the main structures, processes and resources of the real factory in conjunction with the product.

VDI 4499 - Part 1 (2008)

From this definition follows that the scope of the Digital Factory comprises more than software-based solutions for manufacturing, it also includes the integrations of processes and methods (Bracht et al. 2018, chap. 1). Using the defined standard by the VDI, the Digital Factory comprises business tasks along the lifecycle of the whole business activities for a product. Using VDI 4499 - Part 1 (2008), the Digital Factory encompasses multiple fields of the production:

1. **Product development** which includes the definition of market requirements, the product design, the creation of a Bill of Material (BOM), project management tasks, and lastly the first 3D modeling of a product prototype.
2. **Production planning** as the main application area of the Digital Factory. This includes the planning of production processes, production systems, and industrial production plants. In addition, this also comprises the monitoring of their realization up to the start-up of production.
3. **Production start-up** which is the realization and start-up following the production planning and the transition from planning towards series production.
4. **Production operation** which is summarized as the operational performance process.
5. **Order processing** which is the control and monitoring tasks of production by production orders.

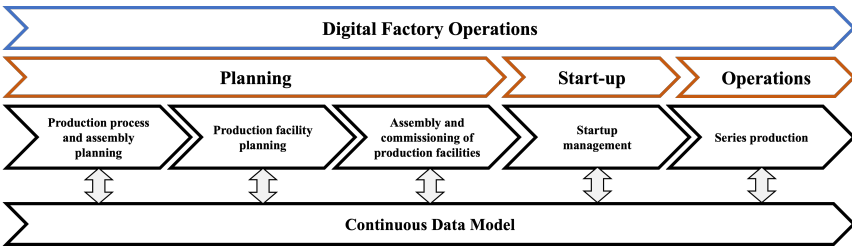
In more detail, a list of subsequent application areas of the Digital Factory across the lifecycle of a product is given in Figure 2.9 as defined by VDI 4499 - Part 1 (2008). The aims and target groups of the Digital Factory are derived directly from the listed application areas. However, the main aim of the Digital Factory is to accelerate and improve production planning (VDI 4499 - Part 1 2008).



**Figure 2.9:** Digital Factory in the lifecycle phases of a product across all business activities as defined by VDI 4499 - Part 1 (2008). The fields of the production are separated by color.

As an essential part of the lifecycle of a product, the Digital Factory comprises the Digital Factory Operations. While the Digital Factory also includes tasks such as sales, product design, or project management, the Digital Factory Operations includes the subsequent tasks of process and assembly planning, production facility planning, assembly and commissioning of production facilities, start-up management, and series production (VDI 4499 - Part 2 2011). The Digital Factory

Operations are the relevant branch for considerations within the field of production planning. The continuous data model is hereby from importance since it uses the planning results as input for operative IT systems. Thus, Digital Factory models can be used in order to improve the series production, if they are linked with real-world data (VDI 4499 - Part 2 2011). The continuous data model along the lifecycle phases of a production system for a Digital Factory Operations is given in Figure 2.10.



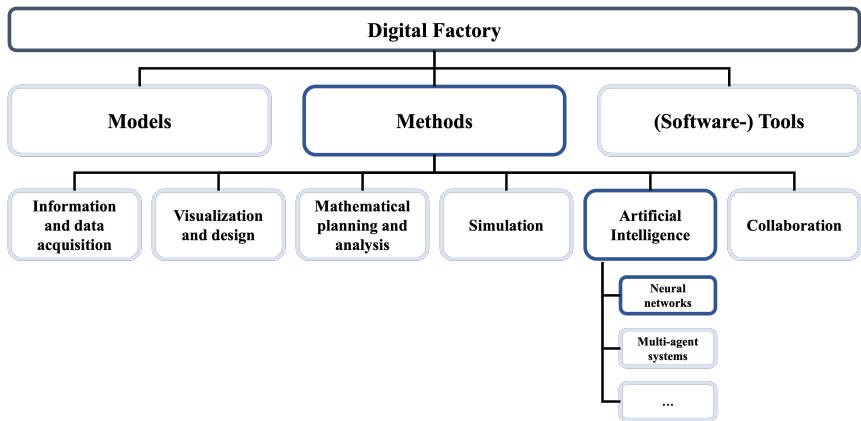
**Figure 2.10:** Digital Factory operations in the lifecycle phases of a production system as defined by VDI 4499 - Part 2 (2011).

As visible in comparison with Figure 2.7, the rough production planning focused here is covered within the Digital Factory Operations in the steps of the production process and assembly planning. Hagemann (2022) states that the integration of information technology, such as the Digital Factory, can improve efficiency in production planning and enable a continuous data transfer by applying tools from computer aided design (CAD) and computer aided planning (CAP). CAD and CAP software tools enable a planning within the structure of the PPR model. However, despite the targeting of a continuous data transfer of planning information in industry applications, a continuous planning using the PPR model is often not fully achieved and is often a challenge in industry applications (Hagemann et al. 2019). As a further field of interest in future research, Schäfer et al. (2022b) add that the relevant interfaces between the Digital Factory and the concept of Industry 4.0 and Building Information Modeling (BIM) should be focused on future factory planning and operations.



Under consideration of the listed challenges of the continuous data transfer of the PPR model within the Digital Factory, the integration of the PPR model in industry practice is of further interest for researchers. The AI support system introduced here is therefore of particular usefulness if it enables a continuous planning using the PPR model. As relevant focus of this doctoral thesis, a consistency check as proposed by Schmidt et al. (2014) of the rough planning models using the PPR model is targeted. This consistency check should include the validation of restrictions as highlighted in Figure 2.7 as well as enable an integration of the PPR model into following processes. Thus, the target is to validate the overall correctness of a PPR model within the rough planning phase.

Using the taxonomy of methods by Bracht et al. (2018, chap. 4), the AI support system proposed in this thesis as a solution for the validation of PPR models in the rough production planning phase can be placed in the category of neural networks in the branch of the AI methods of the Digital Factory. This is schematically visualized in Figure 2.11. While the usage of an AI is defined as a method, the whole introduced support system classifies as a software tool.



**Figure 2.11:** Method categories within the Digital Factory concept as defined by Bracht et al. (2018). The categorization of the method used in the introduced support system is highlighted.

Since the PPR model as an important method for the production planning emerges from this discussion, the following section focuses on the PPR model and the use of ontologies in manufacturing in general. This topic is further embedded in the concept of knowledge-based systems, also named expert systems, in which domain knowledge is added to an information system and thus enables decision-making capabilities (Kumar 2019). A Digital Factory tool which utilizes added domain knowledge and based on this knowledge enables an inference procedure is therefore a knowledge-based system since it can develop solutions for given production planning tasks and exceeds the capabilities of a pure database. Thus, by integrating ontological domain knowledge and inference tools (like the proposed AI support system), the core functionality of a Digital Factory tool is a knowledge-based system (Francalanza et al. 2017). However, the concept of the Digital Factory encompasses more aspects as presented in Figure 2.11.

Lastly, it is useful to distinguish the Digital Factory from the Digital Twin, Cyber-Physical Systems, Smart Manufacturing, and Intelligent Manufacturing since these terms are sometimes used interchangeable within the body of literature. However, a comprehensive discussion of these terminologies would exceed the scope of this thesis and thus, only a short overview is provided.

First, the Digital Twin is originally described by Grieves and Vickers (2017) as a virtual representation of a physical system which includes all obtainable information of the assets of the real physical system. Using this definition, the Digital Twin acts similar to a physical duplicate of the system but in a virtual environment. Thus, the virtual environment is often regarded as an implementation requirement (Xian et al. 2023). In contrast, Kritzinger et al. (2018) differentiate Digital Twins which utilize a bi-directional data flow from the physical object to a virtual representation and Digital Shadows with only a uni-directed data flow or Digital Models without an automated data flow. The task of a Digital Twin varies in the body of literature, but is often from the fields of data management, monitoring, simulation, control, or optimization (Spoor and Weber 2023). However, Spoor and Weber (2023) state that the definitions of Digital Twins in the body of literature are often varying, contradictory, or non-comprehensive from a

practitioner's perspective. In addition, Kober et al. (2022) note that the economic value of implementing a Digital Twin is commonly unclear and intangible.

Although the definition of a Digital Twin is debated, the Digital Factory can be distinguished since it encompasses not only the virtual modeling and representation but a comprehensive network of models, methods, and tools. Thus, the Digital Factory is an overarching concept which can comprise Digital Twins in addition to other applied methods. However, it should be noted that the excessive data requirements of Digital Twins will enable the use of support systems due to the larger knowledge base. Therefore, the here proposed AI support system can be enrolled for planning data derived from Digital Twins.

Second, Cyber-Physical Systems are defined by Lee (2008) as the integrations of computation with physical processes. Embedded computers and networks are utilized to monitor and control physical processes. This is accelerated by the increase in Internet of Things (IoT) applications and systems which enable a connection, data transfer, and comprehensive data collection of embedded systems. This control system includes feedback loops where not only the physical processes are controlled by the computation, but the conducted computations are also affected by the underlying physical processes. Thus, a Cyber-Physical System is a "system of systems" of multiple complex and heterogeneous subsystems interacting with each other (Khaitan and McCalley 2015). Digital Twins may hereby be subsystems of Cyber-Physical Systems where a physical object has its cyber part as its digital representation, which culminates in a Digital Twin (Barricelli et al. 2019).

The distinction of the Digital Factory and Cyber-Physical Systems is more nuanced. Francalanza et al. (2017) differentiate the Digital Factory as the holistic planning, evaluation and improvement of structures, processes, and resources of the physical factory in conjunction with the product and state that a Cyber-Physical System requires more effort in implementation and further research regarding the integration of systems, but its design can be driven by tools from the Digital Factory. Monostori et al. (2016) describe the mapping of processes into a virtual representation by the Digital Factory as a prerequisite of the development of control system capabilities of Cyber-Physical Systems. Thus, Cyber-Physical

Systems describe systems with a full holistic integration of subsystems compared to only a continuous data management system by the Digital Factory.

Third, Smart Manufacturing is commonly defined as integrated and collaborative manufacturing systems with the capabilities of responding in real-time to changed environment conditions, e.g., changes in supply networks or customer demands, to improve a company's capabilities to satisfy customer needs (Zheng et al. 2018). Cyber-Physical Systems, cloud computing, AI, and data science are hereby often referred to as enabling technologies (Kusiak 2018). However, Smart Manufacturing lacks a widely accepted definition (Zheng et al. 2018). In contrast to Smart Manufacturing, the Digital Factory lacks the call for a system integration. Thus, Cheng et al. (2018) sees the full integration of Cyber-Physical Systems as a development step from a Digital Factory towards a Smart Factory.

Fourth, the term Intelligent Manufacturing is described as the origin of the Smart Manufacturing and Cyber-Physical Systems research and itself is described as a research initiative with a focus on autonomy in manufacturing (Kusiak 2023). However, Wang et al. (2021) state critically that Smart Manufacturing and Intelligent Manufacturing share many similarities but yield slightly different focus areas, e.g., Smart Manufacturing is more focused on Big Data while Intelligent Manufacturing is more focused on AI. Wang et al. (2021) state further that the development of these two parallel paradigms is driven by national level strategies and a closer cooperation of both paradigms might be beneficial.

Despite the different focus of the listed research initiatives, all require as important enablers the set-up of a comprehensive data and knowledge base and virtual factory models. Thus, the development of methodologies for the implementation of Digital Factories becomes itself an enabler of Digital Twins, Cyber-Physical Systems, and Smart Manufacturing, as exemplarily demonstrated for Cyber-Physical Systems by Monostori et al. (2016).

To summarize the results by the previous short review of the used concepts in the digitalization of manufacturing, the commonly named distinctions between the concepts are highlighted in Figure 2.12.

	1 Digital Factory	2 Digital Twin	3 Cyber-Physical System	4 Smart Manufacturing
Scope	Planning, evaluation and improvement of structures, processes and resources	Virtual representation of a physical object	Integration of subsystems into a system of systems	Design of autonomous, real-time responsive, and integrated systems
Technologies	CAD, CAP, continuous data models	Virtual environment of the Digital Twin	Embedded computers and networks, IoT	AI, Big Data, Cloud Computing, Data Science
Enablers	Creation of a knowledge base and domain knowledge integration	Virtual factory and data models within the Digital Factory	Virtual factory and data models within the Digital Factory	Cyber-Physical Systems
Degree of Maturity	Applied in industry applications	First implementations and prototypes	First implementations and prototypes	No comprehensive application due to missing enabling technologies

**Figure 2.12:** Scope, technologies, enablers, and degree of maturity of the Digital Factory, Digital Twin, Cyber-Physical Systems, and Smart Manufacturing.

From the author's point of view, the current research lacks an explicit definition or distinction of the discussed terms. This is most likely the result of different national policies, different school of thoughts within engineering, and the competition between different research clusters in order to facilitate their terminology, which is also highlighted in the discussion by Salierno et al. (2019) and Wang et al. (2021). On a further critical note, the aspects of a full integration of systems by Digital Twins, Cyber-Physical Systems, or Smart Manufacturing yield similarities with the scope of CIM, which is posthumously often regarded as failure since the system integration proved itself an insurmountable barrier (Bracht and Masurat 2005) and the strategic and organizational complexity was not fully understood within industry implementations (Cagliano and Spina 2000). In comparison, Spoor and Weber (2023) name an overambitious scope of Digital Twins, organizational complexity, and opaque business cases as risks in implementation projects. While current advantages in the areas of AI, Big Data, and IoT might enable the development of fully integrated systems, the requirement for a comprehensive integration could still prove to be a significant implementation hurdle in industrial applications.

Within this thesis, the developed AI support system is defined as a tool of the Digital Factory which is also applicable for Digital Twins and Cyber-Physical Systems but does not require the extensive modeling or system integration associated with these concepts. Since to the author's best knowledge no universally

acknowledged distinction exists, it is advocated to standardize the terminologies in future research.

## 2.3 Application of Knowledge-based Systems

As discussed in the previous section, the development and set-up of data models and knowledge bases is an important requirement for the implementation of the Digital Factory. This assessment is also true for Industry 4.0 in general, since Capestro and Kinkel (2020) explicitly state that it is a knowledge-based approach with different sources providing the necessary knowledge. Thus, the management of new knowledge becomes a success factor for industrial applications of Industry 4.0. The call for knowledge transfer within an organization is also identified by Fakhar Manesh et al. (2021) as a common discussed topic in the context of Cyber-Physical System, Smart Manufacturing, and manufacturing in general. However, the integration of different necessary knowledge bases along the continuous data model of the Digital Factory operations and lifecycle phases creates hurdles for knowledge management. Klein et al. (2014) list a collaboration hurdle subset driven by:

1. Different syntax of the used database languages across data sources
2. Different visualization methods
3. Different procedural knowledge in handling and using the data
4. Different semantics of data labels and types

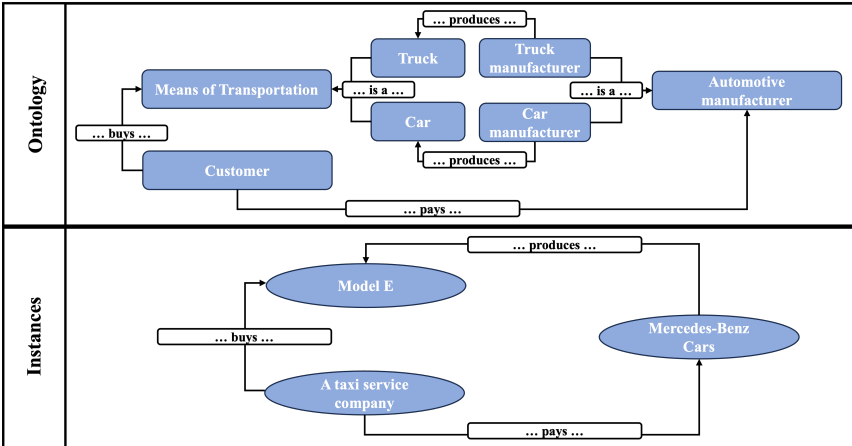
Using these hurdles, Klein et al. (2014) derives requirements for implementing industrial solutions of the Digital Factory. Regarding the hurdles of different procedural knowledge and visualization methods, Klein et al. (2014) name versioning, the storage of preprocessed data, and an enablement of different visualization and analysis methods as requirements. Regarding syntax and semantic, Klein et al. (2014) name the introduction of an extendable, application-specific semantic description beyond the data source as fundamental requirement.

Ontologies as methods to add semantic information to data are commonly seen as the solution for the required knowledge integration (Fakhar Manesh et al. 2021) and are named by Franke et al. (2016) as explicit solutions to the implementation of a Digital Factory. In addition, knowledge graphs, which are based on the development of ontologies (Ehrlinger and Wöß 2016), are an efficient way in order to enhance the learning ability of robots in industrial applications and additionally provide a more efficient data storage method using graph databases compared to relational databases due to their graph structure (Wang and Peng 2023, chap. 6).

A core functionality of ontologies is the expression of a formal logical model from multiple knowledge domains and to provide this information in a human-understandable and machine-readable manner (Feilmayr and Wöß 2016). The aspect of machine-readability is very important in order to embed the ontological model into the continuous data model of the Digital Factory. The ontology is often expressed by a standardized knowledge representation languages such as the Web Ontology Language (OWL). By providing domain knowledge, an ontology can enable context awareness (Dalzochio et al. 2020). A system which acquires and integrates information into an ontology to derive knowledge by reasoning is often described as knowledge graph (Buchgeher et al. 2021).

In general, an ontology uses concepts. Concepts are higher-ranking classes of objects which share specific attributes. For example, an ontological model could include the concept of "means of transport" which is a conceptualization of different kinds of methods for entities to travel distances. Instances are hereby individual entities or objects within this model. A specific entity is an object of a concept. In the presented example, a specific car model, e.g., "Model E", is an instance of the concept "car". Concepts might be in a relation to each other. A relation models the interaction between two concepts. For example, the concept of the "car manufacturer" is in the relation "produces" to the concept "car". In order to circumvent ambiguity of concepts across different domains, ontologies are mostly domain-specific. Any domain-specific ontology can be modeled by a network graph, whereby concepts are vertices and relations are edges. If a specific observation is embedded in this network structure, instances

are represented by vertices and the applied relations are represented by the edges. Using the given example, an incomplete ontology of the automotive industry and an implementation using specific instances is visualized in Figure 2.13.

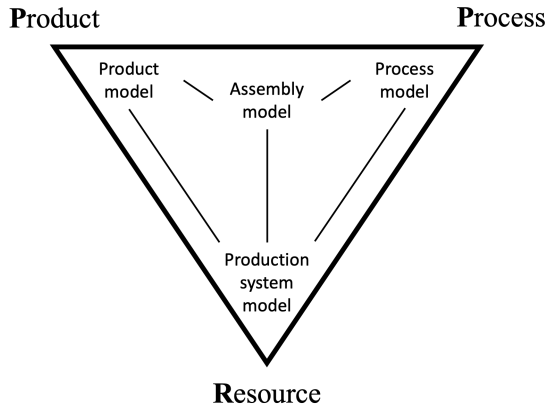


**Figure 2.13:** An exemplary (and incomplete) ontology of the automotive industry and an exemplary implementation for specific instances.

The example in Figure 2.13 shows that complex production planning concepts can also be modeled by an ontology. As important concepts within the PPR model, the processes, the products, and the resources are modeled. A specific process, e.g., a welding application, is therefore an instance of the concept process and is in a relation with the specific robot, which is an instance of the concept resource, conducting the process and the specific component, which is an instance of the concept product. As demonstrated in the exemplary ontology of Figure 2.13, a concept might be further subdivided into more specific concepts. Thus, the concept resource could be divided among others into the concept robot or conveyor belt. The rough production planning describes therefore a specific implementation of instances in relation to each other using the underlying ontology. As a common example using an ontology, a process planning can be modeled as a specific combination of ontological process types.



Hagemann (2022) discusses the individual information models within the rough production planning using the PPR model. The information models utilizing the PPR ontology within the rough planning phases are given in Figure 2.14.

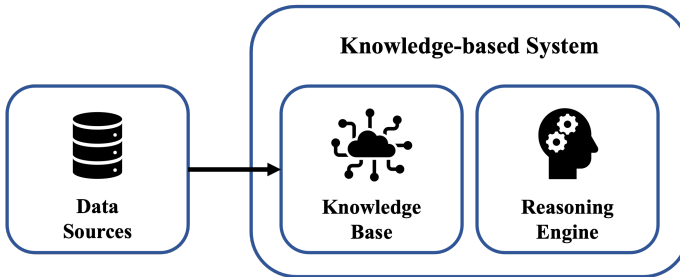


**Figure 2.14:** Information models utilizing the PPR ontology in the rough production planning phase based on Hagemann (2022).

As shown in Figure 2.14, the product model utilizes the concept of the product and its lower-level concepts to describe the structure of the product, which is produced on specific production system. Similar, the process model utilizes all sub-concepts from the process concept in order to describe the process planning and steps conducted on the underlying production system. Thus, the production system model, which is the ontological representation of the used resources, is referenced to the product model and the process model. In addition, the production system model is referenced to the assembly model which describes the assembly of the product structure using individual processes per assembly step. Each assembly step refers to specific substructures of the product structure and thus, the product model is referenced. Also, each assembly step is conducted by multiple process steps and thus the assembly model is referenced to the process model. In the industry application, Hagemann (2022) notes that the references are often not

continuously modeled, and data structures are not consistent across the different planning aspects.

An ontology itself is highly descriptive and allows a semantic modeling of knowledge but lacks the ability for inference. In order to use ontologies for inference applications, knowledge graphs are applied which utilize ontologies as a knowledge-base. Ehrlinger and Wöß (2016) define knowledge graphs hereby as knowledge-based systems which acquire and integrate information from various sources into an ontology and apply a reasoning engine to derive inference and new knowledge. In addition, Bellomarini et al. (2017) add Big Data capabilities as a requirement for knowledge graph management systems in order to distinguish these systems from knowledge base management systems. The architecture of a knowledge graph based on this definition is sketched in Figure 2.15.



**Figure 2.15:** Architecture of a knowledge graph as defined by Ehrlinger and Wöß (2016).

A defined ontology acts within a knowledge-based system as the knowledge-base and in order to create a reasoning engine, AI methods are commonly used to derive the new knowledge in the application (Ehrlinger and Wöß 2016). Ontologies are therefore not developed and applied for their own sake but enable this knowledge creation process. However, the usage of an ontology may steer the production planning process by its own by providing standards and a structure information transfer between users. This benefit is possible since ontologies enable a semantical conceptualization of domain knowledge of the production system in order for an integration of multiple different systems (Yahya et al. 2021).

Using the exemplary ontology in Figure 2.13, it is possible to demonstrate the application of knowledge graphs and the enhancement of reasoning it might provide to data models. In this simplified example, the car manufacturer "Mercedes-Benz" is an entity of the concept "Car manufacturer" and the truck manufacturer "MAN" is an entity of the concept "Truck manufacturer". A statement saying "I want to buy a Model E from MAN" can therefore be analyzed by a reasoning engine using a knowledge graph. The entities of this sentence are the "Model E" which has the relation "is a" to the concept "Car". In addition, the entity "MAN" has the relation "is a" to the concept "Truck manufacturer". Since the ontology in Figure 2.13 explicitly states that only car manufacturer produce cars and "MAN" is not a car manufacturer, the inference can be drawn that the person making this statement either wants to buy a car but confused "MAN" and "Mercedes" or wants to buy a truck but named the wrong model type. Thus, the reasoning engine can identify incorrect or misleading statements and propose targeted corrections. In addition, the knowledge graph can draw logical conclusions based on this sentence. Since the statement uses the relation "buy", it can be reasoned that the person making this statement is a customer. This simplified example highlights that knowledge graphs do not only enable checks of the syntax of a given statement (in this case the sentence is grammatically correct) but can also extract semantics from a statement. Thus, models using knowledge graphs can draw inferences and add machine-readable as well as human-understandable contextual information by using entities ("Model E"), relations ("is a"), and concepts ("Car").

In the case of production systems, ontologies might describe relation of, e.g., robots or tools to certain machinery types. If a certain task which requires a certain type of tool is conducted on a product, a knowledge graph can identify correct and incorrect tools using the ontological information and the reasoning engine. Therefore, ontologies and knowledge graphs are of great importance in manufacturing and are currently conceptualized, developed, and implemented in order to improve efficiency in production systems. However, Spoor et al. (2023a) list and discuss requirements with industry experts and show that the application of knowledge graphs today still has some implementation hurdles in practice, e.g., industry standards for the integration of heterogeneous data sources are still

under development and thus the development of reasoning engines is a secondary objective after the implementation of a fundamental knowledge-base. Overall, the integration of heterogeneous data sources is also a challenge in the practical application in multiple different domains besides manufacturing (Noy et al. 2019). As Hagemann (2022) states, ontologies are not consistently used and modeled across the whole production planning process and thus, a knowledge graph lacks the knowledge-base for a consistent utilization across different planning stages. Glawe et al. (2015) note that the used data within engineering is commonly not structured in an ontology but rather in hierarchical documents and exchange formats. Thus, Glawe et al. (2015) state that engineering data and ontologies which represent knowledge require a sufficient connection in practical applications.

Due to the recent accelerated development of Large Language Models (LLM), knowledge graphs gained popularity since a knowledge graph might be capable of enhancing and verifying the results by a LLM and *vice versa*, a LLM might be able to create large scale knowledge graphs or simplify the creation (Pan et al. 2023). The development of these solutions in an industrial context is still an open research question but might enable manufacturing companies to adopt knowledge graphs and ontologies faster.

One common data exchange format for knowledge-based information in automotive engineering is the Automation Markup Language (AML). Bigvand et al. (2016) show that this data format enables an iterative data exchange despite a common shared semantic description of the data sources. AML relations enable a representation of multiple relationships of data models transferred into this standard. Li et al. (2019) note that AML is useful for structure modeling but lacks a support of different semantics which requires a semantic mapping of AML and connected models. However, Hildebrandt et al. (2017) for example conduct a model implementation using AML as well as OWL. Despite these limitations, AML files are a useful data format to evaluate planning data from different tools and data models within a knowledge-based system. This means, however, that a support system in practice would not utilize the full semantically described domain ontology but standardized data models. One limitation of AML is that it is more specialized on the manufacturing process and is not optimized for the needs

of other related industries such as construction which are using markup languages for BIM models (Witte et al. 2018). Therefore, the integration of BIM models into the Digital Factory is a future field of interest in order to analyze plant facility information along the production system layout, too (Schäfer et al. 2022b).

## 2.4 Fault Detection in Production Systems

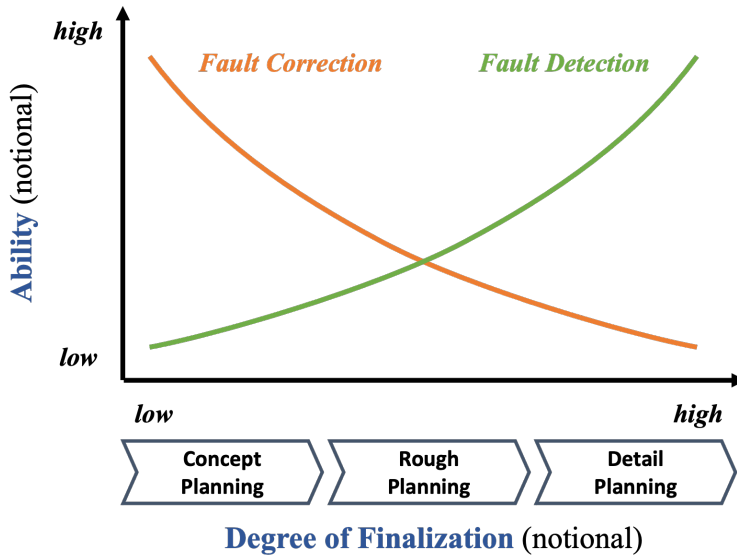
The task of the AI support system proposed here is a predictive detection and correction of faults in the rough production planning so that these faults are not later implemented during the realization of the production facility or that they must be discovered in a more costly manner during later stages in the overall planning procedure. For a definition of faults, it is useful to use the standard defined by the Deutsches Institut für Normung e.V. (DIN).

State of an item characterized by inability to perform a required function, excluding the inability during preventive maintenance or other planned actions, or due to lack of external resources.

DIN EN 13306:2018-02 (2018)

From this definition follows that a fault in the rough production planning might result in a functional inability during the later operations of the planned production system. Thus, it is from economic usefulness to detect these faults early. Park et al. (2020) describe the fault detection hereby as the first step of an implementation procedure to optimize a system. After the fault detection, the fault is secondly isolated, thirdly identified, and lastly a recovery of the process is conducted. The rough planning stage of the overall production planning procedure is a cost-efficient stage to detect the faults since most costs in planning a production system occur during a realization of the production system. Corrections during the operation of the production system can be even more costly. However, it is more difficult to detect fault in this early stage since the production system cannot be observed and analyzed based on its real physical implementation but only based on an abstraction using ontological information and former knowledge of

similar production systems. The relation between the degree of realization of the production system and the ability to detect and correct faults is sketched in Figure 2.16. A low ability to do corrections is equivalent to high costs for correction procedures.



**Figure 2.16:** Sketch of the ability to detect faults and to correct the faults during planning stages.

From this follows that it is most beneficial to detect fault in an early planning phase in order to correct the fault in a timely and cost-efficient manner. Thus, the proposed AI support system aims at an early fault detection in order to steer correction of the planned production system in time. This should prevent economic costs and increase the productivity of the production planning procedure. In addition, if faults are detected faster, the whole production planning can be fastened and thus, manufacturing companies can optimize the costs of the planning procedure. This also enables a faster reaction capability to changed business requirements by faster implementing and rolling-out new products. The rough planning phase is a

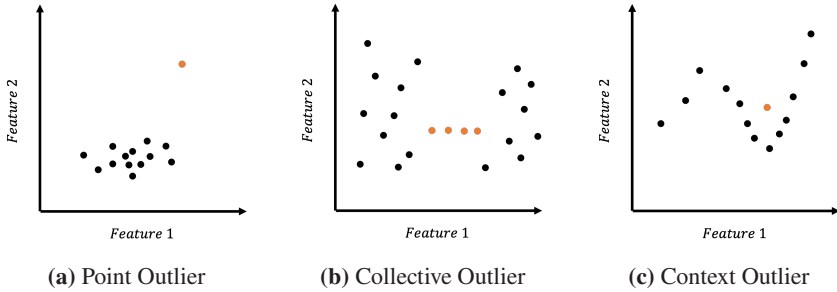
reasonable point during planning for a structured fault detection because it is the first phase where enough data and information is present to validate first ideas and to detect faults in a sufficient manner. On the other hand, because the planning is not too advanced, changes and adjustments of the planned production system do not become too costly at this point in time.

As 2.16 implies, an early fault detection is difficult to implement since the real production system cannot be observed. In the case of an observable production system, sensor signals measured on the production system can be analyzed. Abid et al. (2021) distinguish in their review sensor data from vibration, current, audio, and image signals. However, during the rough planning phases no sensor data and signals are available. Thus, the fault detection procedure must focus on available data such as the planned system layout, simulations, or ontological descriptions.

A concept related to fault detection is the field of anomaly detection, which is a method from the field of AI in order to find anomalous, conspicuous, or unusual data. Chandola et al. (2009) define anomaly detection as the task of finding patterns in a data set which do not conform to an expected behavior. The patterns resulting from this analysis are called anomalies. Very similar, Aggarwal (2017, chap. 1) defines anomalies as derivation from the normal model. In addition, Mehrotra et al. (2019, chap. 2) add that the deviation of an anomaly from the normal model should be substantial. Thus, a fault would be recognized in a data-driven analysis as an anomaly in the sense that the fault would differ in a substantial manner from the assumed norm of the production system. Anomaly detection is used as a method in multiple fields of research such as business science, medical science, or as discussed here in manufacturing for fault detection. Anomalies are hereby not always incorrect data samples but might yield an underlying relevant relation, e.g., Spoor (2023) shows that by an application in marketing important customers, so-called key accounts, can be identified using anomaly detection.

The terms anomaly and outlier are often used interchangeably and sometimes separated from the similar concept of novelties since novelties are incorporated into the normal model later (Chandola et al. 2009). Thus, Spoor et al. (2023c) separate anomalies which are true faults of incorrect or rare system behavior and

anomalies which are results from the limitations of the normal model. Spoor et al. (2022b) apply this separation in a system description and separate error terms using this distinction. While this concept is driven by a root cause analysis, anomalies are separated in a data-driven analysis by their appearance in the context of the whole underlying data set. Lindemann et al. (2021) separates point base outliers, contextual outliers, and collective outliers. Point outliers are single data points whose values are not in accordance with the other data points. In contrast, collective outliers are within a tolerance window regarding their values but the composition of a group of outliers is irregular compared to the normal model. Lastly, contextual outliers can only be detected in comparison to their direct neighboring data points. These anomaly types are sketched in Figure 2.17 to further highlight the differences.



**Figure 2.17:** Sketch of common outlier types as described by Lindemann et al. (2021).

Focusing on time series, Blázquez-García et al. (2021) state that also whole time series from a set of multiple time series can be outliers. Thus, it is possible to further distinguish local anomalies which are occurring within a set of data points and global anomalies where a whole set of data points is an outlier compared to other sets of data. To further distinguish anomalies, Foorthuis (2021) creates a typology which covers distinct anomalies across multivariate data sets, time series, and graph structures. The applied methods of anomaly detection also differ between these types of anomaly detection.



From particular interest are the anomaly types within graph structures since an ontological production planning can be modeled as a graph. Ma et al. (2021) separate the detection of anomalous vertices, edges, and the detection of anomalous subgraphs within a graph structure from the detection of anomalous graphs. If a production planning is modeled as a graph, anomalous vertices would be incorrect instances embedded in the production planning and anomalous edges would be incorrect relations between two instances. Using the example in Figure 2.13, an anomalous edge would be the relation of a truck manufacturer who is producing a car model. However, an anomalous vertex and edge are in the case of an ontology always related since incorrect selected instances would also invalidate the relation. These kinds of violations should be easily detected by observation of the ontology's restrictions. However, it is more difficult in the cases where contextual domain knowledge is required to detect the anomaly. E.g., the addition of salt and pepper when cooking a meal might be a valid relation in a recipe but it would be anomalous if it is a recipe for a dessert. Thus, a recipe modeled using an ontology might require context from other instances. Concluding, the anomaly detection of incorrect ontological set-ups during a rough production planning becomes in practice the detection of subgraphs and whole graphs which are incorrect combinations of instances and relations.

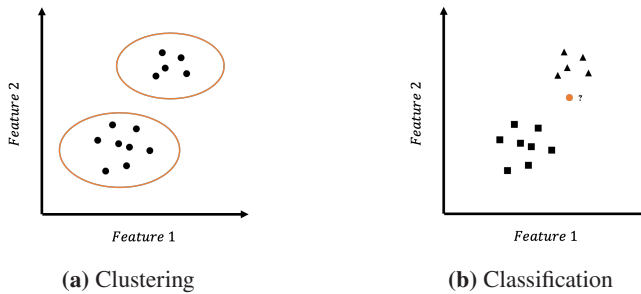
In a second case, when it is not required to detect a fault but to utilize knowledge from similar production system layouts, the task becomes the measurement of similarity of the new planned production system to former known production systems. Thus, two separate concepts must be applied. First, the former known production systems must be grouped into similar systems to extract structured knowledge from the specific type of group and second, the new planned production system must be classified as one specific group to use the knowledge from the similar production systems of the same group to draw inferences.

The task of grouping is also called clustering. Murphy (2012, chap. 25) defines clustering as the process of grouping similar objects. Fahrmeir et al. (1996, chap. 9) describe the objectives of a cluster analysis as providing various methods that assign a set of objects to smaller subsets, called groups or clusters. This assignment is then called grouping. The objects which are assigned to the same

subset should be as similar as possible with respect to their features and the subsets should differ as distinctly as possible with respect to the features of their assigned objects. The groups formed by a cluster analysis should be homogeneous with respect to the features of the objects within the group and heterogeneous with respect to the features of objects within the other groups. However, Fahrmeir et al. (1996, chap. 9) state that a grouping cannot be assessed as correct or incorrect, but the usefulness of the grouping created by the cluster analysis is determined by the consideration of the objective of the overall analysis. In the field of AI, clustering is commonly conducted by methods from unsupervised learning where no labeling of the individual objects within the data set is prior known (James et al. 2013, chap. 2).

The task of assigning the new planned production system is called classification. In a classification the analyzed objects are assigned labels of their predicted class assignments. Murphy (2012, chap. 1) distinguishes multi-class classification where objects are assigned to a class out of a set of different classes, and binary classification where there are only two distinct classes. By this definition, an anomaly detection is a binary classification since it assigns an object either to an anomaly class or a normal class. In addition, an object might belong to multiple classes so that the task becomes a multi-label classification. Classification is commonly conducted by methods from supervised learning where labeling is priorly known and used during a training phase of the AI model to enable a distinction between classes (James et al. 2013, chap. 2). The distinction between classification and clustering tasks is schematically visualized in Figure 2.18.

Relevant for anomaly detection, clustering, and classification is the selection of a metric which describes the similarity (or dissimilarity) between objects. Based on the similarity to a specific group, class, or outlier characteristic, a tested object is assigned the corresponding label. In the case of a production system, a similarity might be related to the probability that similar behavior or a similar fault might occur during the operation of the system (Spoor et al. 2022a). Thus, the similarity of a production system to another system, and thus the assignment to a class of similar production systems, might be important to draw structured inferences.



**Figure 2.18:** Sketch of clustering and classification tasks.

## 2.5 Summary

This section starts with a discussion of production planning, which is an essential dispositive production factor for companies from the manufacturing industry. The first phase of a system lifecycle in manufacturing is the development phase, which encompasses the production planning of the production system in a greenfield approach. The production planning can hereby be subdivided into the concept planning, the rough planning, and the detail planning. The rough planning receives as inputs the current product and assembly structure. First production system and assembly process layouts are developed during the ideal planning, the first sub-phase of the rough planning. These layouts are cross-checked with the underlying restrictions in order to create functional outputs of these layouts and production factor demands for the production system which is then finalized during the subsequent detail planning.

In order to gain economic advantages by utilizing an intelligent manufacturing system, the Digital Factory is introduced. The paradigm of the Digital Factory describes an applied network of methods and tools along a continuous data management system during the production planning, as well as in subsequent production tasks. To ensure this continuous data management and also to enable a structured planning procedure, ontologies are used to describe, among others, process, product, production system, and assembly models. Most commonly in the manufacturing industry, the PPR model is utilized. By enriching the ontology

with a reasoning engine, a knowledge-based system is created which enables an inference and creation of new knowledge using the ontology as knowledge base.

If the task of fault detection is focused on such a knowledge-based system, the rough planning stage is a useful phase to enroll and develop corresponding methods as the ability to detect a fault becomes economic feasible and the overall planning of the production system is still adaptable without costly changes. This is possible since a fault detection during the rough planning can identify problems before the final physical production system is fully constructed. As base for the fault detection, methods from the field of anomaly detection can be utilized. If the rough planning is conducted using an ontology, the task becomes a detection of anomalous graph and sub-graph structures.

The AI support system developed in this doctoral thesis should therefore enable the use case of a fault detection during the rough planning phase using ontological planning information as required input. In this context, the AI support system can be described as a reasoning engine in a knowledge-based system embedded into the paradigm of the Digital Factory.

### 3 State-of-the-Art

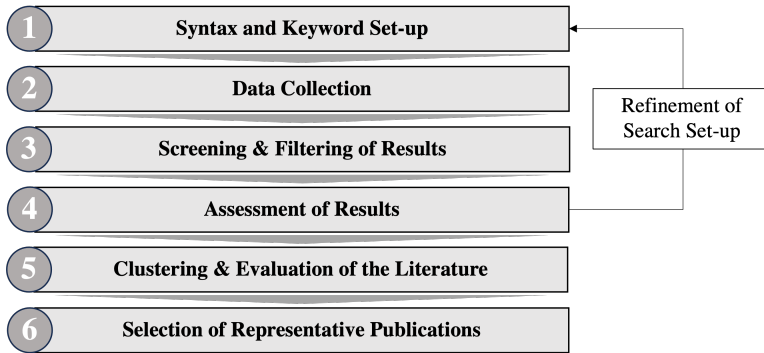
Since the importance of production planning for industry application is established in the previous section, an analysis of the state-of-the-art enables a structured detection and definition of current gaps in the literature. Among these are too few researched applications and use cases in production planning as well as methods which are not often utilized or not fully used to their potential. The detected gaps in the body of literature are used in order to flesh-out the objectives of this research by providing a set of open research questions.

In order to screen the body of literature, a structures methodology is developed and presented in the first subsection. This methodology enables an assessment of the literature, use cases, models, and application. Thus, the methodology results in an overview of the current state-of-the-art regrading support systems for production planning. However, the task of this section is not a comprehensive analysis of the literature but a focused assessment of gaps in the body of literature. Therefore, not all findings are discussed and presented but only a representative subset of the literature in order to highlight the most prominent use cases and models. An additional focus is on the introduction of the models. Thus, the models are also briefly described mathematically in order to archive an understanding which then enables a sufficient development of the proposed model of this thesis and a comparison regarding its design and mathematical implications.

Based on this introduced state-of-the-art, the methods for support systems are summarized, the gaps in the body of literature are derived, and the limitations in research and application are discussed in a subsequent subsection. Based on the gaps in the body of literature, the research questions of this thesis are deduced and presented in the last subsection.

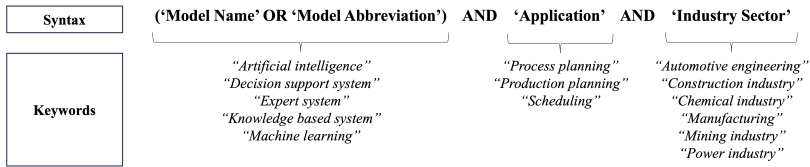
### 3.1 Methodology of Literature Review

In order to systematically grasp the body of literature, a methodology for the literature review is developed and fleshed-out in an iterative process. The overall structure of the methodology for the literature review is presented in Figure 3.1.



**Figure 3.1:** Schematic of the applied methodology in the literature review.

The methodology starts with the set-up of adequate search terms for application in an online search engine. In order to exclude different fields of research, three main terms are used, namely: first, the name of the applied model; second, the use case; and third, the industry sector. In order to not exclude methods which might be applicable in production planning for manufacturing but currently lack use cases in this specific field, similar industry sectors are included in the search since solutions developed there might be comparable and convertible for manufacturing. Using these main terms, keywords for each term are defined and composed to a syntax for the search term used in the online search engine. The initial selection of used keywords based on the IEEE Taxonomy (IEEE 2023) is given in Figure 3.2. As search engine, google scholar is selected since it comprises most relevant sources of literature.



**Figure 3.2:** Syntax and keywords applied as search terms in the online search engine.

As a second step, the data collection is conducted by entering each composed search term and extracting title and hyperlink for each publication. Since the list of resulting publications is still relatively long for conducting a manual screening in the case of most search term, only the first 100 results per search term are used. This builds the first list of literature. However, within the third step of screening and filtering of the results, review papers are separated from use cases. In addition, results outside the defined subject area are filtered.

Using this list of publications, the literature is reviewed and based on the findings, the keywords used in Figure 3.2 are adjusted. Instead of the generic term such as AI, particular model names are used which are extracted from priorly found review papers. In addition, terms indicating review papers are excluded in the search using the google syntax (-"Review" -"Survey") in order to find only models, use cases, and prototypes in this second search run. The literature search is iteratively repeated with adjusted keywords until a sufficient database of relevant use cases is created.

This final list of use cases is then clustered and reviewed. In order to structure the use cases along models as well as use cases, four use case categories are developed and applied in order to cluster the body of literature:

1. **Forecasting**, which includes the prediction of either customer demands, production factor demands, or process times of a certain process, group of processes, or the whole production process in order to steer the production planning. Thus, forecasting is the task of uncertainty quantification.
2. **Process Planning**, which focuses on the sequence of processes and the assignment of necessary parameters to a process or a set of processes

in order to assemble a given product model utilizing a given production system model. The process planning is therefore focused on the process and assembly model.

3. **Resource Planning**, which describes use cases where the geometrical location, sequence, and set-up of resources within a production system is planned. Objective of most resource planning tasks is the design of a valid workstation which handles a given assembly structure. Thus, resource planning is focused on the production system model.
4. **Scheduling**, which includes the task of assigning a certain product or process to a certain set or sequence of workstations to utilize their resources. Dynamic scheduling is hereby a subtask which considers changes and flexible assignments during the production system's operation.

The ALBP, including RALBP and analogously formulated disassembly problems, are viewed in this analysis as a subtask of resource planning or scheduling depending on their specific type. Type I problems aim at minimizing the number of stations and are therefore part of the resource planning tasks. Type II problems focus on the scheduling of task to a given set of different stations in order to minimize a given KPI and thus are considered scheduling problems. Type E problems are multi-objective problems which would cover resource planning as well as scheduling. Type F is analogously a validation task of a resource planning and scheduling use case.

The categories are build using a screening of the commonly used terms within the body of literature derived from keywords, titles, and the text body of publications and clustered by the author. In addition, the taxonomy by Gottschlich et al. (1994) is used to cross validate the categories. In contrast to the categorization proposed here, Gottschlich et al. (1994) list three main fields of interest: planning and integration of design & manufacturing; off-line planning under uncertainty; and on-line planning, execution & reaction. However, the applied categories align with the defined subgroups by Gottschlich et al. (1994). Concluding, it should be noted that these categories do not fully adhere to the mutually exclusive and



collectively exhaustive paradigm but are rather reflections on the most prominent research topics and used key word in the field of production planning.

In order to give the reader an understanding of the relevant applications, use cases, and applied models of support systems in the literature, only a selection of representative publications is presented in a tabular format in the following section. If multiple models are used, the model with the best performance as concluded by the authors is listed. The selection is based on showcasing interesting and diverse use cases, number of citations of the publication, recency of the publication, and the reputation of the peer-reviewed journal and publisher in order to prevent the listing of non-academic or predatory journals.

The result of this literature search and review procedure are presented in the following subsection along a brief introduction of the mathematical background of commonly applied models.

## 3.2 Review of Support System Models

In order to further structure the varying approaches, the different methodologies for setting up support systems in production planning are categorized along a similar differentiation as used by Hagemann and Stark (2018). These categories are broader in definition, so that the complexity of the methods, their applicability, and overall methodological approach might differ within the same category. However, the core idea and field of research are within the categories comparable and thus, it is possible to derive conclusions based on the conducted categorization. In this thesis, the available methods for support systems, which are proposing optimizations or detecting errors in the planning data, are categorized as follows:

1. **Rule-based Models**, which apply a set of conditional or fuzzy rules.
2. **Optimization Models**, which search a solution space of an optimization function under restrictions based on given parameters of the planning data in order to find a better parametrization.

3. **Artificial Intelligence Models**, which derive conclusions based on a mathematical model representing the logic of the problem in an abstract manner and which are parametrized using training data as input. Models with the capability to draw inference from a statistical and data-driven approach are often referred to as Machine Learning (ML) models and are considered a subclass of AI.

The different approaches and models of each category are described in the following separate subsections in more detail and a selection of representative implementations along the use cases categories are introduced as described in the methodology of the literature review.

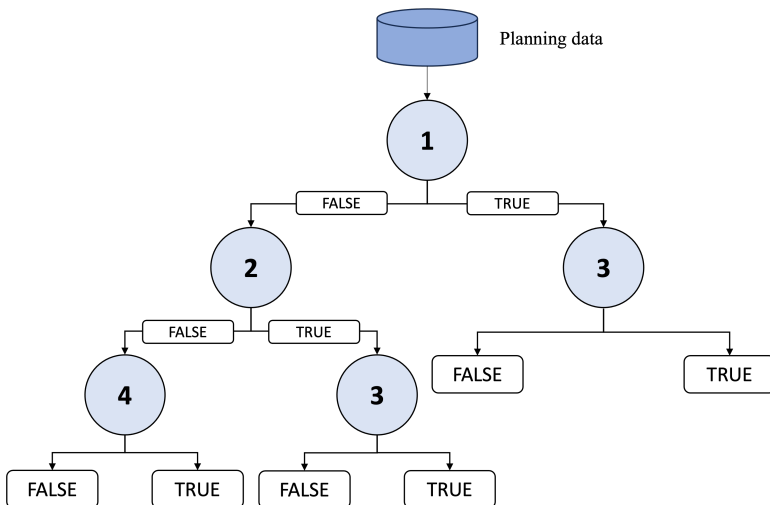
### 3.2.1 Rule-based Models

A core consideration of a rule-based model is a conditional logic applied to sets of features of a production system. In principle, these rules are defined using conditional logical operations of 'AND', 'OR', and 'IF', the mathematical operators, and relational operators. Thus, these rules using multiple operands and operators generate statements which are either 'TRUE' or 'FALSE'. By combining a multitude of rules, the conducted test can cover a wide variety of cases and incorporate a high complexity of the underlying use case. The operands are selected from either knowledge bases or are the parameterization of the current planning data of the production system. The arrangement of rules and operators is manually designed and programmed within the support tool. An exemplary conditional statement using an 'AND' operator and the mathematical operator "=" is given in Figure 3.3

IF ( 'Operand 1' = 5 ) AND ( 'Operand 2' = 10 ) = TRUE

**Figure 3.3:** An exemplary conditional statement using two operands.

The decisions itself can be visualized by a directed binary tree structure which starts at the highest hierarchical node and goes a path towards the branches till the last node is reached and the final assignment of the tested data structure is conducted. Each node represents a conditional test and the resulting assignments of 'TRUE' and 'FALSE' for the statement determine the selected edge. From the tree structure follows that the same rule will not be tested twice. However, complex directed graph structures of the rules are possible, where branches are interconnected (which could be transformed into a tree structure without connected branches again by duplicating the rules). This visualization is exemplary demonstrated in Figure 3.4, whereby each node is a conditional logic statement and the edges represent the assigned truth values of the corresponding higher-level node condition. In total, there are four different rules using the third rule twice. The set of operands is given by the entered data set.



**Figure 3.4:** Visualization of an exemplary rule-based decision process as a tree structure.

Using the tree structure, it is trivial to see that the number of rules tested is the number of computational operations conducted by the support system. The

worst-case computational complexity of testing planning data is a tree with only one branch and thus the computation time scales linearly in  $\mathcal{O}(n)$  with the total number of defined rules  $n$ . The average computational time scales with  $\mathcal{O}(\log n)$ .

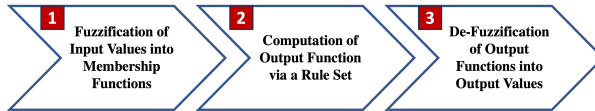
Since the conditional statements follow clear logical assignments, it is in addition trivial to see that the assignment of an evaluated set of data to a specific truth value can only change if either the input data or the rule set is changed. Thus, rule-based approaches are always deterministic.

In practice, the definition of these rules often requires ontological data with semantics to evaluate the rules. Thus, rules are defined using ontological concepts which are tested for correctness along defined relations. Lanz et al. (2008) develop an ontological model including product, process, and system ontologies in order to derive a reasoning based on rules derived by the ontological relations. Kim et al. (2006) demonstrate a methodology which uses ontologies that represent engineering, spatial, assembly, and joining relations of the assembly line and thus, the implicit constraints can be used for reasoning capabilities. Using the PPR-model's data structure, Mas et al. (2016) propose a knowledge-based assistant system for the production planning of an assembly line in the aviation industry. Mas et al. (2016) hereby emphasize that the support tool is dependent on expert knowledge to set-up the ontology and knowledge base. *Vice versa*, if only limited knowledge about the underlying problem is available, it might be not possible to model the ontology and corresponding rule set sufficiently. However, the use case by Mas et al. (2016) demonstrates that a time reduction in the planning of the assembly line can be achieved by the proposed support tool if the necessary knowledge is leveraged during an implementation of the ontology.

As discussed, the use of ontologies enables a human-understandable as well as machine-readable methodology (Feilmayr and Wöß 2016). Conditional rule sets can also be visualized and thus offer high explanatory value. Therefore, these approaches offer a computational time efficient and interpretable way of support systems for production planning.

In addition to rule sets using a conditional logic as illustrated in Figure 3.3, other approaches utilize a fuzzy logic. The core idea in fuzzy systems is that

continuous input variables are transformed into fuzzy sets, which are describing their membership assignment. This process is called fuzzification. Subsequently, logical operations are conducted on the different fuzzy variables in order to create an output function via a rule set. Concluding, the output function is again transformed into a crisp value during the de-fuzzification. This approach is illustrated in Figure 3.5 and is called Mamdani system. Mamdani systems are one commonly applied method in order to develop a fuzzy system, but other approaches exist as well.

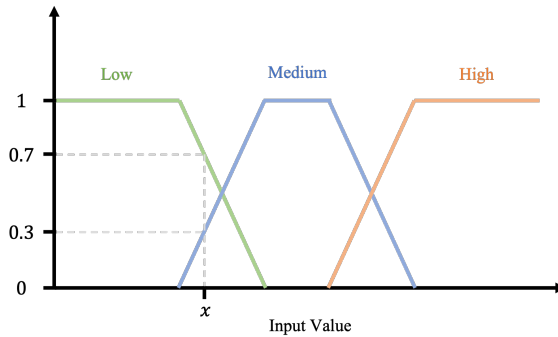


**Figure 3.5:** Approach of a rule evaluation by a fuzzy logic following the Mamdani system.

During the first step of the fuzzification, a variable is assigned a measure of vagueness which property it has. In conditional logic, statements are either true or false, and thus belong to the set  $\{0; 1\}$ . In fuzzy logic, statements are assigned to values between 0 and 1 for different memberships, often semantic relations such as, e.g., "low", "medium", or "high". A continuous input value can therefore be between "low" and "medium" and thus belong to both descriptions with a different level of vagueness. This is illustrated in Figure 3.6

After the fuzzification of all input variables, rules using operators similar to these in conditional logic are applied. However, the operators are applied differently in a fuzzy system, e.g., the logical operator 'x AND y' is transformed into the minimum operator  $\min(x, y)$ . These operators are then used to set up a fuzzy rule set which evaluates the input variables and creates an output function. Lastly, the output function is again transformed back during the de-fuzzification into a crisp variable, meaning a real value from a continuous interval or set.

Fuzzy logic proved useful in different applications, in particular in control theory, and also in use cases of production planning. However, while its capabilities and possible applications exceed these of normal rule sets using conditional logic, both



**Figure 3.6:** Exemplary fuzzification of a continuous input variable into a fuzzy set with a membership assignment "low" with a value of 0.7 and "medium" with a value of 0.3.

require an extensive rule definition using either conditional or fuzzy statements. Extensive expert knowledge is also required in the application of fuzzy logic for the design of the fuzzy rule set. Thus, fuzzy logic might be more nuanced and enables a better design of rules but also requires a manual knowledge creation process. In addition, experts do not have to assigned fixed thresholds or critical values to rules but can work with more plausible vague statements, e.g., in the case of scheduling: if the current load of a workstation is "low", then assign the next job to this workstation.

An application of rule sets in combination with non-rule-based methods are also commonly used, e.g., fuzzy system can be paired with neuronal networks from the field of AI models in order to create neuro fuzzy systems.

The computational effort for fuzzy rule sets is also mostly depending on the number of rules and possible membership assignments a variable can have. Thus, fuzzy rule sets also yield a sufficient computational complexity and are again deterministic since the rules are pre-programmed and do not change over time or after certain prior decisions.

An overview of a selection of prominent publications of rule-based approaches and support systems in production planning is given in Table 3.1.

**Table 3.1:** Selection of rule-based models for production planning.

Author	Use Case	Model
Mahdavi et al. (2009)	Scheduling	Fuzzy system
Mas et al. (2016)	Resource planning	Rule set & PPR model
Engel et al. (2018)	Process planning	Rule set
González Rodríguez et al. (2020)	Scheduling	Fuzzy system & Decision Trees
Widodo et al. (2021)	Forecasting	Neuro fuzzy system
Zhou et al. (2021)	Resource planning	Fuzzy system & Decision Trees
Okubo and Mitsuyuki (2022)	Process planning	Rule set & PPR model
Ren et al. (2022)	Scheduling	Rule set

Some publications, e.g., Guo et al. (2022), focus at the automated or standardized creation of a knowledge base as a first step in order to develop an AI-based solution using the knowledge base.

It should be also noted that rule sets are the state-of-the-art in practical applications in the manufacturing industry of support systems. Most standard software utilizes rule sets, which can be manually defined by the users, and are evaluating decisions based on these rules. From the author's experience, these rule-based solutions are currently favored by practitioners due to their accessibility and overall strong performance, if the rule set was designed with sufficient precision. They are particularly strong if paired with forecasting methods from the field of AI since this enables an uncertainty approximation as complement to a human-understandable and comprehensive rule set.

### 3.2.2 Optimization Models

Continuous optimization models utilize an optimization function  $f : \mathcal{A} \rightarrow \mathbb{R}$  which assigns a parameter vector  $\mathbf{x} \in \mathcal{A}$  from the set of possible solutions  $\mathcal{A}$  to a real number. This real number is the target of the optimization which is either minimized or maximized. Maximizations targets can be rearranged as a minimization target and *vice versa*. The function  $f$  can be a linear as well as nonlinear function. The resulting real value might in some applications be an optimization target with direct relation to an economical KPI such as costs, time, or profit but might in other cases do not have a direct economical interpretation. The optimization function is subject to a set of restrictions with a number of  $P$  inequality constraints  $g_i(\mathbf{x})$  with  $i = (1, \dots, P)$  and a number  $Q$  equality constraints  $h_j(\mathbf{x})$  with  $j = (1, \dots, Q)$ . Thus, the optimization problem can be generally written using a minimization target:

$$\begin{aligned} \min f(\mathbf{x}) \\ g_i(\mathbf{x}) &\leq 0, \quad i \in (1, \dots, P) \\ h_j(\mathbf{x}) &= 0, \quad j \in (1, \dots, Q) \\ P, Q &\geq 0 \end{aligned} \tag{3.1}$$

The models themselves are from different classes of problems. Convex optimization problems, where Linear Programming (LP) using a linear optimization function, inequality, and equality constraints is a subclass, are described by a convex optimization function and inequality constraints. Thus, all local optima of the optimization problem are also global optima. Nonlinear optimization functions and constraints are also possible and are called Nonlinear Programming models. In Integer Programming or Mixed Integer Programming (MIP), all, or rather some, variables are not from the set of real numbers  $\mathbb{R}$  but from the set of integers  $\mathbb{Z}$ . The cases of a set-up where the problem uses an identical description as in LP models but some variables are integers are called Mixed Integer Linear Problems (MILP). For large MILP models, heuristics must be applied in order to find solutions since these problems are NP-hard. MILP models are most common in

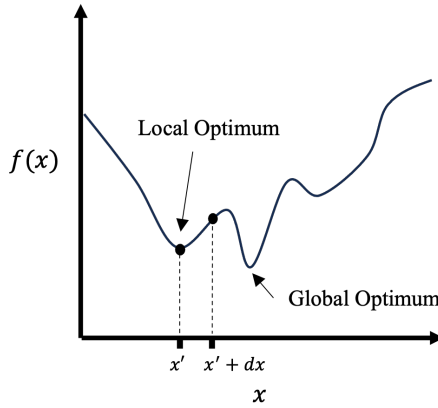


production planning use cases since, e.g., the number of workstations is an integer while the cycle time is not. In addition, combinatorial optimization searches for an optimal set from a finite set of objects, whereby the solutions and sets do not have a continuous representation.

In order to describe the optimization model, its mathematical classification is relevant for the process of finding suitable solutions. In practical application, the specific design of the optimization function is more important, in particular what KPIs are optimized and what constraints are considered. Thus, the model layout is more focused on the process of finding a solution but the core considerations for production planning are found in the design, not the model.

Optimization problems can be mathematically solved by finding the global optimum in specific cases, e.g., in the case of a linear optimization function and restrictions in LP models. However, in other cases, e.g., MIP models, it is not possible to analytically find the optimal solution. In these cases, a heuristic is used to approximate the global optimum. In addition, metaheuristics as higher-level procedures can be used to find the best heuristic for a given optimization problem. These classes of problems are often NP-hard and thus, it is assumed that it is not possible to solve them with a polynomial computational time complexity but instead with an exponential complexity.

In order to achieve an efficient computation time, heuristics might be aborted if the solution is close to an assumed optimum. In addition, often not the whole solution space  $\mathcal{A}$  is searched but only randomly selected values within intervals where the optimum is assumed. Within these intervals, the current best solution  $\mathbf{x}'$  is slightly adjusted in order to find gradual improvements. This however creates the risk of not finding a global optimum but only a local optimum, if the adjustment  $dx$  of the current best solution is not large enough to cross a local maximum between a local and global (or better local) optimum. However, most algorithms for these kinds of optimization problems circumvent this problem by multiple random initialization procedures and parallel testing of different intervals. An optimization trapped in a local optimum is schematically highlighted in Figure 3.7.



**Figure 3.7:** Visualization of an optimization problem trapped within a local optimum.

The random initializations and search within the solution space are the reasons why these models are non-deterministic. The same initial data set might result in multiple different solutions in multiple optimization procedures. However, by using the optimization function the different solutions can be compared for the best solution, which again does not guarantee a global optimum. By searching the solution space long enough (depending on the problem), it is reasonable in most applications to assume that a solution close to the global optimum is found. Thus, the quality of the results is often high if sufficient iterations of the optimization algorithm were conducted.

Applied heuristics in order to solve optimization problems are, among others, branch & bound methods, particle swarm optimization, simulated annealing, genetic algorithms, ant colony optimization algorithms, or tabu search. However, various different heuristics are developed. While all heuristics follow a different approach at searching the solution space  $\mathcal{A}$ , their specific design is not important in the review discussed here since they are all based on a given optimization function which require prior modeling and aim at finding solutions close to a global optimum in a sufficient computational time. Hence, these heuristics should not be confused as the underlying models. Instead, they are the solvers for

these models. As also discussed in the context of the models themselves, the core design decisions regarding the production planning approach are defined by specific arrangement of the optimization function.

A selection of representative optimization models, their applied mathematical model classification, if applicable, the applied heuristic as a solver (or most efficient solver in case of multiple solvers), and the use case within production planning are listed in Table 3.2.

**Table 3.2:** Selection of optimization models for production planning.

Author	Use Case	Model
Michalos et al. (2015)	Resource planning	MIP
Colledani et al. (2016)	Resource planning	MILP
Lopes et al. (2017)	Resource planning	MILP
Borba et al. (2018)	Resource planning	MIP (Branch and bound)
Michels et al. (2018)	Resource planning	MILP
Angizeh et al. (2020)	Scheduling	MILP
Hagemann and Stark (2020)	Resource planning	MIP (Branch and bound)
Liu et al. (2021)	Process planning	MILP
Zhang et al. (2022a)	Scheduling	MIP (Genetic algorithm)
Albus and Huber (2023)	Resource planning	MILP
Gelfgren et al. (2023)	Scheduling	MIP (Particle swarm optimization)
Stade et al. (2024)	Resource planning	MIP (Genetic algorithm)

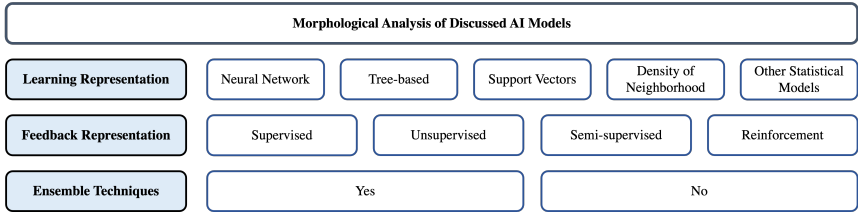
This table includes only a short exemplary list of research in this field. For a more nuanced review of ALBPs and the included KPIs, problem formulations, and solvers, the reviews by Hagemann (2022) and Chutima (2022) are recommended. However, this list provides a good overview of the high-level use cases tackled by optimization models.

### 3.2.3 Artificial Intelligence Models

There exists a wide variety of AI models in the body of literature for classification and anomaly detection task which are in principle applicable in manufacturing applications. As already discussed, AI models can help to solve NP-hard optimization problems under a limited time and data storage complexity. This section, however, discusses AI models which do not aim at solving a given optimization function but solve complex tasks without the explicit modeling of an optimization problem. Thus, these models evade a core limitation of optimization problem which is the difficult prior definition of an optimization function which encompasses multiple and sometimes contradicting KPIs such as costs, throughput, or cycle times. In addition, these models do not require a manual design of rule sets but enable an automated generation of rule grounded in a statistical analysis.

Fahle et al. (2020) name as applied methods in the literature of the field of manufacturing process planning Decision Trees, Random Forest, Neural Networks (NN), Support Vector Machines (SVM), Gradient Boosted Trees, logistic regression, Bayes inference, and Q-learning. However, it is proposed to use a more structured view of the varying applied models of AI in order to gain insights into the state-of-the art. Thus, a morphological analysis in Figure 3.8 is proposed to discuss the core principles and ideas in detail. It should be noted that this morphological analysis is no comprehensive review of AI models but a simplified structure in order to more efficiently discuss the different approaches within the context of this thesis.

The idea of this structure is to separate approaches along three dimensions:



**Figure 3.8:** Simplified morphological analysis of AI models in manufacturing applications.

1. **Learning Representation.** This describes the process in which the algorithm models the knowledge and the applied structure which is learned.
2. **Feedback Representation.** Feedback is the interaction of an AI model with the environment, thus in most cases the data on which a model is trained. Feedback can be the labeling of data into classes and providing the model with samples in order to drive the learning process.
3. **Ensemble Techniques.** Many approaches utilize ensemble learning which creates multiple different weaker models and adds a meta model which decides based on the results provided by the weaker models.

Of course, it is also possible to combine different learning and feedback representations within an ensemble. Thus, the morphological analysis in Figure 3.8 could have multiple different expression in the first to dimensions in the case of ensemble techniques.

Not all combinations of approaches might be applicable for all problems and *vice versa*, certain problems require a specific combination of approaches in the morphological analysis. The individual solutions are separately discussed in the following subsections.

### 3.2.3.1 Neural Networks

Since the AI support system introduced in this thesis falls into the category of NNs, the general layout and the core principles of NNs are discussed in this

section in more detail. This section follows the notation and structure presented by Aggarwal (2018) in order to present the core principles of NNs.

The first important learning structure is the Artificial Neural Network (ANN) layout. ANNs aim at imitating the process of neurons and synapses in biological brains. Despite ANNs being obvious simplifications of the underlying biological process, the principles of neuroscience proofed themselves useful in the design of ANN layouts (Aggarwal 2018, chap. 1). However, the basic process of synapses and neurons remains the same. A signal is sent via the synapses and if the signal strength exceeds a defined threshold value, the neuron is activated, which in turn sends a signal to neighboring neurons via synapses.

In a classification task, input values of an object, which are multivariate descriptions  $\mathbf{x}_i = (x_{i,1}, \dots, x_{i,j}, \dots, x_{i,J})$  of  $J$  features, are provided and the input values in turn lead to an output signal which provides information about the class membership of the object. The input values are a list of the active or inactive features  $x_{i,j}$  of the object, represented as input neurons. This is mathematically modeled by a signal which is transferred from the  $J$  input neurons to a connected neuron. Each binary neuron is hereby either active or inactive:

$$x_j = \begin{cases} 1, & \text{active neuron} \\ 0, & \text{inactive neuron} \end{cases} \quad (3.2)$$

If a neuron is active, it sends a signal of the strength  $w_j \in \mathbb{R}$  to the connected output neuron. This represents the biological function of synapses. The binary output signal  $\hat{y}$  is activated if the sum of the signals exceeds a predefined threshold. This neuron starts to be active again and thus, sends also a signal in a corresponding strength to neighboring neurons. Thus, it is important whether a neuron is activated by the incoming signals or not. Basically, this is mathematically modeled as a sum of all input signals. The input of a vector of the  $J$  input neuron is given as the  $\mathbf{x}$  feature vector of the object and the strength of the signal transfer is given

as a vector of weights  $\mathbf{w} = (w_1, \dots, w_j, \dots, w_J)$  connecting each input neuron with the output neuron. The signal is modeled as follows:

$$\sum_{j=1}^J w_j * x_j = \mathbf{w} * \mathbf{x} \quad (3.3)$$

The threshold decision whether the output neuron is activated or not is given as a function  $\Psi$  of the incoming signal strength.

$$\hat{y} = \Psi(\mathbf{w} * \mathbf{x}) \quad (3.4)$$

A commonly used activation function is the signum function.

$$\hat{y} = \text{sign}(\mathbf{w} * \mathbf{x}) \quad (3.5)$$

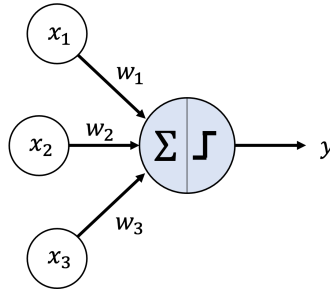
Additionally, it might be useful to incorporate a bias which is modeled like a neuron that always sends a signal in the strength  $b$ .

$$\hat{y} = \Psi(\mathbf{w} * \mathbf{x} + b) \quad (3.6)$$

Thus, the output neuron's value becomes 1 if the sum of signals is positive and zero if the sum of signals becomes negative. This output may then act, e.g., as a predictor of the class membership in a two-class classification task.

Figure 3.9 shows the Perceptron model initially proposed by Rosenblatt (1958) using 3 input features  $x_1$ ,  $x_2$ , and  $x_3$  which are summarized using the corresponding weights to create an outgoing signal  $y$  by a signum function.

The Perceptron is a very simple layout of a NN with only an input layer of neurons representing features and an output layer containing a single neuron with is either active or inactive to create a binary classifier. Despite its simplicity, it essentially enables, e.g., a regression classifier whether an object is above or below a curve dividing a data set.



**Figure 3.9:** An exemplary model of a Perceptron with three input features using a weighted summation and a sigmoid activation function to create an output.

To create a genuine predictor using this ANN layout, it is firstly important to generate weights which accurately represents the classification task. This is conducted in a training procedure of a number  $N$  objects in a data set  $\mathcal{D}$  with a known class membership  $y$ . Thus, this ANN layout corresponds to a supervised method due to the feedback representation.

In order to train the ANN, the quality of the class membership prediction  $\hat{y}$  is measured as the difference between the output of the ANN and the true class membership  $y_i$  of an object. This measurement of the accuracy of the prediction is very similar to the application in residuals of a regression analysis and of the Decision Tree. This function for measuring the prediction quality is referred to as the loss function  $L$ .

$$L = \sum_{\mathbf{x}_i, y_i \in \mathcal{D}} (y_i - \hat{y}_i)^2 \quad (3.7)$$

The weight vector  $\mathbf{w}$  is selected in order to minimize the loss function. Thus, this task is similar to the optimization in a linear regression analysis. The resulting gradient descent corresponds to the to the stochastic gradient descent of the method of least squares used in linear regression models.

$$\min_{\mathbf{w}} L = \min_{\mathbf{w}} \sum_{\mathbf{x}_i, y_i \in \mathcal{D}} (y_i - \hat{y}_i)^2 = \min_{\mathbf{w}} \sum_{\mathbf{x}_i, y_i \in \mathcal{D}} (y_i - \Psi(\mathbf{w} * \mathbf{x} + b))^2 \quad (3.8)$$



In order to optimize the loss function, the gradient is observed. The gradient is defined as the course of change of a variable in a particular direction. In the case of ANN, the gradient of the loss function depends on the respective weights and is defined as the partial derivative according to the respective weight. However, in the case of the ANN defined here, the gradient can be represented in simplified form as a smoothed function, which can be analyzed analogously:

$$\nabla L_{smooth} = \sum_{\mathbf{x}_i, y_i \in \mathcal{D}} (y_i - \hat{y}_i)^2 * \mathbf{x} \quad (3.9)$$

The weights can be iteratively adjusted using the known data set for training purposes. The loss function is minimized by computing the gradient using a parameter  $\alpha$  as the rate of learning applied to the weights for each object trained.

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha * (y_i - \hat{y}_i)^2 * \mathbf{x} \quad (3.10)$$

The weights are therefore recalculated and updated per each object in the training data set  $\mathcal{D}$ . Once all weights have been computed using the data set with the known true assignments  $y$  of objects to classes, new objects with as yet unknown assignments can be fed into the network as inputs in order to compute the predictions  $\hat{y}$  for the binary class assignment. This allows unknown objects to be classified in a two-class binary classification task.

The mini-batch method is an optimization method for computing the weights. In order to compute the weights in a mini-batch procedure, a subset  $\mathcal{S}$  from all objects of the entire data set is selected and the weight is updated for all objects within the subset at the same time. Thus, the weight update is not conducted individually.

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha * \sum_{\mathbf{x}_i, y_i \in \mathcal{S}} (y_i - \hat{y}_i)^2 * \mathbf{x} \quad (3.11)$$

The application of a mini-batch procedure is a trade-off between computing time and stability in determining the weights. Due to a computer's architecture, powers

of 2 are often chosen for parallelization of the computation task as the size of the subset  $\mathcal{S}$ , such as 32, 64, 128, and 256.

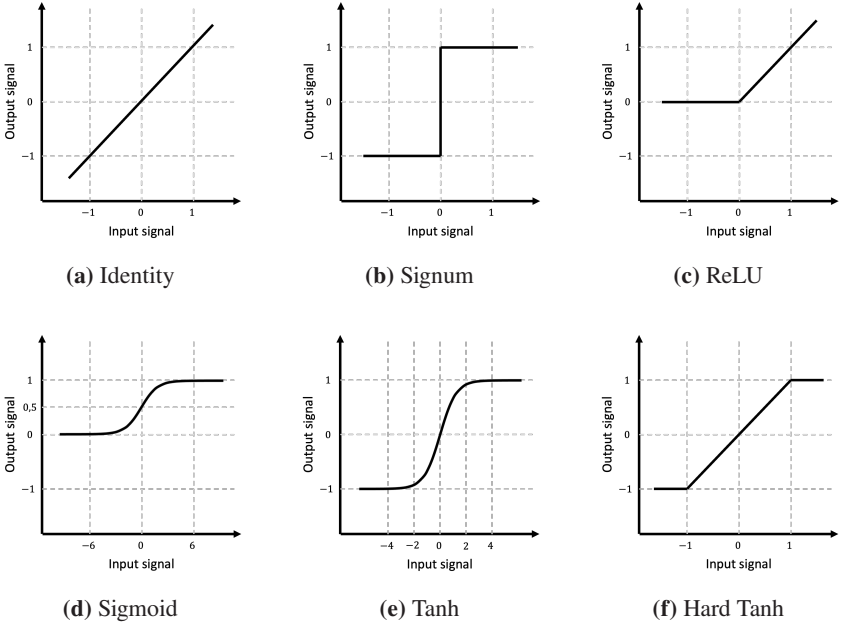
The selected function  $\Psi$  for activating the output neuron is an important influence on the performance of the NN. A neuron does not always have to generate binary outputs but can also accept normalized values between 0 and 1 or even any rational numbers, depending on the selected function, and then pass this value on to the downstream neurons with the respective signal strength. Therefore, the values of the input and output neurons are no longer strictly binary, as in equation (3.2), but can take on various values, whereby a normalization between 0 and 1 is often desired. In addition to the signum function in equation (3.5), multiple other functions can be utilized in an application of an NN. A list of commonly applied activation functions is given as follows:

$$\begin{aligned}
 \Psi(\mathbf{w} * \mathbf{x}) &= \max(0, \mathbf{w} * \mathbf{x}) && \text{ReLU} \\
 \Psi(\mathbf{w} * \mathbf{x}) &= \frac{1}{1 + e^{-\mathbf{w} * \mathbf{x}}} && \text{Sigmoid} \\
 \Psi(\mathbf{w} * \mathbf{x}) &= \frac{e^{2 * \mathbf{w} * \mathbf{x}} - 1}{e^{2 * \mathbf{w} * \mathbf{x}} + 1} && \text{Tanh} \\
 \Psi(\mathbf{w} * \mathbf{x}) &= \max(\min(\mathbf{w} * \mathbf{x}, 1), -1) && \text{Hard Tanh}
 \end{aligned} \tag{3.12}$$

The different activation functions of equation (3.12) are also schematically visualized in Figure 3.10 in order to highlight their specific properties regarding input and output signal of a neuron, whereby the different scaling of the input signal axis should be noted.

In order to use these more nuanced activation functions, the loss function and thus the gradient  $\nabla L$  is also adjusted since the partial derivate cannot be set-up identically as in the case of a signum activation. Thus, the weight updates in equation (3.10) and (3.11) are also adjusted since they are directly derived from the gradient. For a general case, the loss function is given as follows:

$$\mathbf{w} \Leftarrow \mathbf{w} + \alpha * \nabla L \tag{3.13}$$



**Figure 3.10:** Schematic diagram of the presented activation functions.

In addition to more nuanced activation functions, it is also possible to apply different loss functions. The loss function in equation (3.7) corresponds to a least-squares regression problem. However, a hinge loss function is also commonly applied, e.g., here displayed with an identity activation function as follows:

$$L = \max(0, 1 - y_i * \hat{y}_i) \quad (3.14)$$

In the case of binary classification targets, a loss function based on logistic regression can be applied using the identity activation function as follows:

$$L = \ln(1 + e^{-y_i * \hat{y}_i}) \quad (3.15)$$

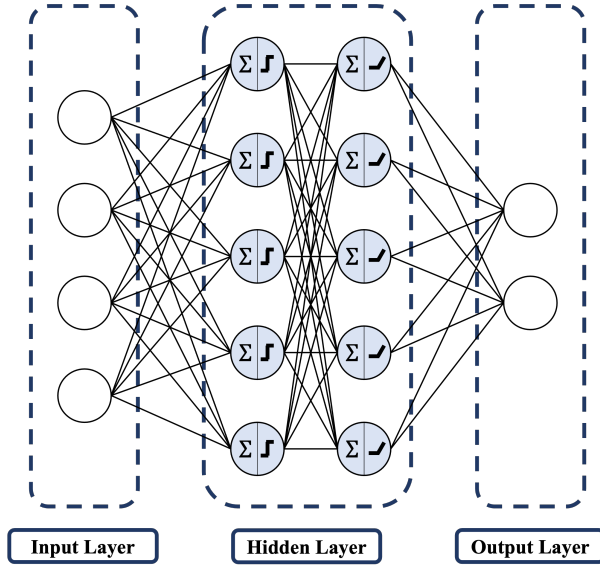
In this case, a learning method based on logistic regression is used. However, the loss function changes if other activation functions are applied. In addition, the loss and activation functions are selected depending on the underlying classification or regression objective, e.g., a multi-class classification task of categorical targets requires a different loss function compared to a binary classifier.

In addition to the selection of useful activation and loss functions, the structure of the NN can be extended from the very simple structure in Figure 3.9 by adding a number of  $P$  layers  $\mathbf{h}_p$  between the input layer and the output layer of the ANN or by using several outputs instead of just one. The resulting ANNs are called multilayer neural networks. In principle, different structures of this type of ANN are possible with different numbers of layers which represent the dimensionality of the ANN. Several bias neurons can also be added at different locations and layers. Also, different activation functions might be applied within the same ANN for different neurons or layers. In general, this type of ANN is also referred to as a feed-forward network since an input signal is passed from layer to layer until an output is produced at the end of the chain. An example is given in Figure 3.11.

Multilayer neural networks are often also called Multilayer Perceptrons (MLP), whereby MLPs are often considered as the special case of layout using linear activations functions, such as the signum function as initially proposed for the Perceptron, while multilayer neural networks use non-linear activation functions. However, both terms are in practice often used synonymously.

The processing of the signals per each layer depends on the selected activation functions of the corresponding neurons and the weights and inputs of the preceding neurons. The following relation applies to the neurons of each layer with a total dimensionality of  $P$  layers:

$$\begin{aligned}\mathbf{h}_1 &= \Psi(\mathbf{w}_1 * \mathbf{x}_i) \\ \mathbf{h}_{p+1} &= \Psi(\mathbf{w}_{p+1} * \mathbf{h}_p) \\ \hat{\mathbf{y}}_i &= \Psi(\mathbf{w}_{P+1} * \mathbf{h}_P)\end{aligned}\tag{3.16}$$



**Figure 3.11:** An exemplary model of a Multi-layer network with four input features using two hidden layers in order to create a two-dimensional output.

When determining the weights, each weight must then be adjusted backwards based on the prediction and the loss function. The determination of a weight is then carried out according to the partial derivation of the respective weight.

$$w_j \leftarrow w_j + \alpha * \frac{\partial L}{\partial w_j} \quad (3.17)$$

This type of back propagation leads to a more complex calculation of the weights via the partial derivation using the chain rules of the derivation along a single path  $\mathcal{P}$  of weights. A path is a single chain of how a signal is passed on from a single neuron within the input layer to a neuron in the output layer, whereby various paths naturally exist. For a single path, the following equation applies:

$$\frac{\partial L}{\partial w_j} = \frac{\partial L}{\partial y_i} \left( \frac{\partial y_i}{\partial h_P} \prod_{p=j}^{P-1} \frac{\partial h_{p+1}}{\partial h_p} \right) \frac{\partial h_j}{\partial w_j} \quad (3.18)$$

If multiple paths  $\mathcal{P}$  exist, the sum over all paths of the signal must be considered.

$$\frac{\partial L}{\partial w_j} = \frac{\partial L}{\partial y_i} \left( \sum_{h_p, y_i \in \mathcal{P}} \frac{\partial y_i}{\partial h_p} \prod_{p=j}^{P-1} \frac{\partial h_{p+1}}{\partial h_p} \right) \frac{\partial h_j}{\partial w_j} \quad (3.19)$$

The complexity of this computation can be sufficiently managed by using dynamical computation methods, the discussion of which exceeds the scope of this introduction. However, there exist various software solutions that have implemented this computation and enable a user-friendly parametrization and configuration of an ANN, such as Scikit-learn by Pedregosa et al. (2011) or PyOD by Zhao et al. (2019) which are used in this thesis in order to set-up NNs during benchmarking.

A problem which can occur when applying an ANN, alike to other models, e.g., in linear regression, is an overfitting. This occurs when too many output neurons in relation to the input neurons exist or the ANN has a layout that allows perfect linear combinations to be found. In these cases, the linear combinations do not follow a real relationship, analogous to the problem of overfitting in regression, but are simply a result of a too restrictive layout or too little or less diverse input data. Therefore, it is recommended to have a sufficient number  $N$  of objects in the data set  $\mathcal{D}$  to determine the weights. Alternatively, it is possible to add a penalty term  $\lambda > 0$  as a regularization technique in order to adjust the learning rule in equation (3.10).

$$w_j \Leftarrow w_j(1 - \alpha * \lambda) + \alpha * (y_i - \hat{y}_i)^2 * \mathbf{x} \quad (3.20)$$

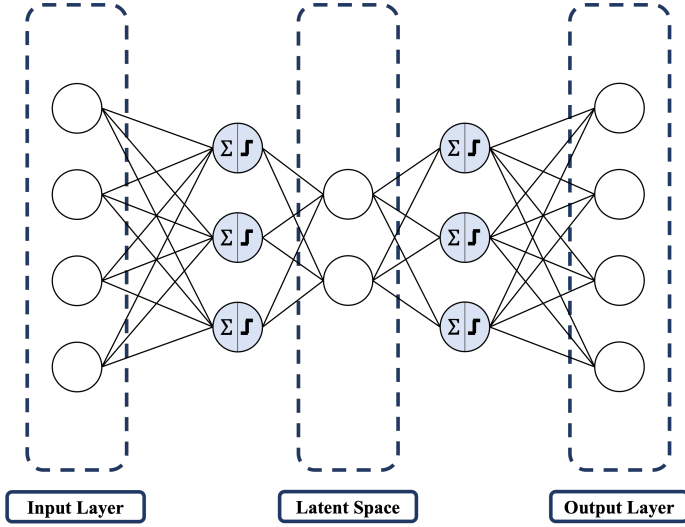
These types of learning rules can successfully avoid overfitting when calculating weights with a small amount of available data. The progress of regularization can be interpreted as a kind of forgetting of formerly trained weights. Furthermore, ANNs can use a technique called bagging. In bagging, a number  $B$  of data sets with  $N_b < N$  objects is randomly selected from the data set containing all  $N$  objects. This creates an ensemble of ANNs. The final classification for a tested

object can be determined by a majority voting of all ANNs within the ensemble which can also reduce the effect of overfitting.

Similar to the discussion in the section about the optimization models, the resulting optimization function of an ANN is also non-linear, which can lead to only local minima being found rather than the global minimum of the problem. This can be solved with a pre-training of the network by computing the weights for only a partial section of a single layer of the network.

In general, the layout of the network should be justified for the applied use case and a specific architecture should be chosen that supports the underlying problem. In addition, networks with more layers are less prone to overfitting than networks with many neurons per layer and overfitting can be avoided by adding depth to a network. However, very deep networks tend to have gradients of input neurons that are either very large or very small and the back-propagation procedure of the network becomes unstable. Very deep networks may therefore not converge to a solution fast enough, which is related to the problem of decreasing gradients for deep networks.

Worth mentioning as a special layout of ANNs are Autoencoders, which have fewer neurons in the hidden layers than the number of output and input neurons. The signal from the input layer is encoded and represented in a lower dimensional structure in the latent space and then decoded towards the output layer. The loss function is selected so that the difference between the original input and the first encoded and then decoded output is minimized using, e.g., the methods of least squares. By minimizing the difference between input and output, the Autoencoder trains the weights in order to create a representation of the inputs within a lower dimensional space which can be sufficiently decoded into a truthful representation of the inputs. Thus, Autoencoders can be used to extract essential information from data while reducing noise and enable a reduced representation of the input. Using the decoding and encoding procedure, it is possible to design Autoencoders for an anomaly detection which do not rely on labeled data as commonly used ANNs. An exemplary layout of an Autoencoder is presented in Figure 3.12.



**Figure 3.12:** An exemplary model of an Autoencoder with four input features and a latent space with only two dimensions. The decoding and encoding are conducted in one layer each.

Another interesting layout is the Radial basis function (RBF) network for regression tasks. These networks use a single hidden layer with an RBF as activation function and a single output neuron. The RBF is given as follows:

$$\Psi(\mathbf{x}) = \exp^{-\eta \|x - b_p\|^2} \quad (3.21)$$

By summarizing the resulting RBFs with a center at  $b_p$  of each neuron  $p$  in the hidden layer using weights  $w_p$  as in regular NN in the output neuron, a regressor is computed.

In summary, it should be mentioned that the computing capacity required for ANNs might be very large for certain use cases and therefore, either special hardware must be used or simplifications must be made in order to enable a computation of complex and deep ANNs in an acceptable time. In addition, a careful optimization of the hyper-parameters, such as the number of layers and



neurons per layer, the selection of the loss and activation function, the learning rate, or the regularization terms, is required when setting-up ANNs.

In addition to the conventional NN layouts, LeCun et al. (2015) name Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN) as the main direction of research in NN, which are often grouped under the term Deep Learning (DL) describing NNs with a complex structure and using multiple layers. DNNs utilize a high number of layers and yield a very high accuracy in prediction tasks. In addition, DNN can handle a high number of features and thus, a prior feature space reduction is not required. CNNs use the principles of DNN and are mostly applied for image processing. CNNs have convolutional layers and pooling layers. The convolutional layers map the input in a feature map while the pooling layer merges semantically similar features.

RNNs on the other hand are able to analyze and produce sequential inputs, such as text in the application in an LLM. RNNs process sequences one item at the time while storing in hidden units a representation of the history of the past items. One specific RNN layout is the long short-term memory (LSTM) network using one memory unit which accumulates memory and a second which can "forget" the memory if needed. LSTMs are often more effective than conventional RNNs (LeCun et al. 2015). Since RNNs and LSTMs are focused on the analysis of sequences, they can therefore be utilized in sequence planning tasks or forecasting where the next element for a given sequence is searched.

An unsupervised NN approach are Self Organizing Maps (SOM) proposed by Kohonen (1982) where the positioning of a neuron (and the positioning of its neighboring neurons) is iteratively updated in order to create a map representing the data set. Likewise, Hopfield neural networks are single layer NN with a fully interconnected layer of neurons which are iteratively activated, originally proposed by Hopfield (1984). Hopfield neural networks can memorize inputs by increasing the weight of a neuron connection if this pair of neurons is active together. Thus, an incomplete signal can be completed since neurons with a high weight to other neighboring neurons get also activated or *vice versa*, deactivated.

Lastly, it is possible to design agent-based systems with multiple agents using each an individual reward function. If an agent conducted a task correctly, the rewards function gives positive feedback in form of a reinforcement learning approach. Via a process called Q-learning, the agent's behavior is adjusted based on the received reward, the action conducted by the agent, and the measured environment factors. The feedback process is in this case regulated by the rewards function. Thus, multi-agent systems can be utilized for scheduling tasks. However, the design of the reward function is crucial for the learning progress since it should directly reflect the agent's objectives (Sutton and Barto 2020), similar to the design of an optimization function in optimization models.

Q-learning is per se not a NN approach. In simple cases, the Q-function is simply a table containing the set of states and the set of actions which are possible, and the table is updated using a learning rule. However, the Q-function is in practice often approximated by an ANN acting as a surrogate function (Aggarwal 2018). With respect to the initially proposed categorization in Figure 3.8, the Q-learning per se resembles the feedback representation while the learning (and predicting) of future states is conducted by a NN.

A list of approaches utilizing NN models in different use cases of production planning in manufacturing are provided in Table 3.3. This list gives a representative but not comprehensive overview of the different NN layouts applied across different tasks in production planning.

When searching the body of literature for the applications of NN approaches in production planning, it can be concluded that the field of (dynamic) job scheduling using reinforcement learning with Q-learning strategies is the most researched application. The examples provided in Table 3.3 act as placeholders for a large variety of different solutions for scheduling tasks using Q-learning.

Usuga Cadavid et al. (2020) find in their review that most AI approaches in the literature focus on smart planning and scheduling for self-organizing resources. The design of smart processes is on the other hand an under-researched field. However, approaches as provided by, e.g., Ming and Mak (2000) or Amaitik and Kiliç (2007), highlight the possibilities of NN layouts using supervised learning

**Table 3.3:** Selection of Neural Network models for production planning.

Author	Use Case	Neural Network Layout
Ming and Mak (2000)	Process planning	SOM & Hopfield Networks
Amaitik and Kiliç (2007)	Process planning	ANN & fuzzy rule set
Kutschenreiter-Praszkiewicz (2008)	Forecasting	RBF network
Tuncel et al. (2014)	Resource planning	Q-learning
Wang and Yan (2016)	Scheduling	Q-learning
Waschneck et al. (2018)	Scheduling	Q-learning
Wang et al. (2018)	Forecasting	RNN / LSTM
Kuhnle et al. (2019)	Scheduling	ANN
Lin et al. (2019)	Scheduling	Q-learning
Ma et al. (2020)	Forecasting	LSTM
Chen (2021)	Process planning	ANN
Mayer et al. (2021)	Scheduling	Q-learning
Moon et al. (2021)	Process planning	RNN / LSTM
Tan et al. (2021)	Forecasting	K-means & LSTM
Zhang et al. (2022b)	Scheduling	MLP

approaches for the planning of a production system layout with focus on process and assembly sequences. Based on this consideration, one identified gap in the body of literature, which this thesis tackles, is the under-researched field of process planning in production systems. Current promising solutions mostly utilize NN models for this field compared to other approaches.

### 3.2.3.2 Tree-Based Models

In general, tree-based approaches train a structure similar to the tree structure in Figure 3.4. However, in the case of rule-based methods the rules are defined manually by expert. Decision Trees, the core algorithm behind most tree-based approaches, build the tree structure by aiming at minimizing the within difference of all objects in a node by dividing a node in two sub-nodes with a maximum reduction of the within variance. All  $N$  objects start at the initial node of the tree. The decision tree then separates all objects in the nodes into two sub-nodes minimizing the sum of the within variance of the sub-nodes. This is iteratively repeated for all sub-nodes till only either a maximum depth of sub-nodes is reached or till every sub-node only contains objects of the same class.

This section follows the notation by James et al. (2013, chap. 8). The separation is realized by selecting a feature per node and defining a threshold criterium  $s$  for the feature per node. Thus, two new nodes are added to a tree with currently  $Z$  nodes using a decision process of:

$$\begin{aligned}\mathcal{R}_{Z+1} &= \{i \mid x_{i,j} < s_z\} \\ \mathcal{R}_{Z+2} &= \{i \mid x_{i,j} \geq s_z\}\end{aligned}\tag{3.22}$$

The decision threshold  $s_z$  is selected so that the sum of the within variance of the objects in the sub-node  $z$  is minimized. This variance is commonly measured by computing the difference in entropy of the initial node and the resulting sub-nodes. The entropy  $H_z$  of a data set within a node  $z$  is measured using the percentage  $p_{z,k}$  of objects in the data sets within node  $z$  which belongs to the class  $k$  out of a total of  $K$  classes.

$$H_z = - \sum_{k=1}^Z p_{z,k} \log p_{z,k}\tag{3.23}$$

Alternatively, the Gini impurity  $G_z$  can be applied.

$$G_z = - \sum_{k=1}^Z p_{z,k} (1 - p_{z,k}) \quad (3.24)$$

Basic Decision Trees use a supervised feedback system as shown in equation (3.23) and (3.24) which requires information of the class membership of the data during the training procedure. However, Isolation Forest by Liu et al. (2008) can use unlabeled data which creates an unsupervised tree-based approach separating data in two classes which is particularly useful in anomaly detection use cases where the true class memberships are unknown.

Decision Trees are deterministic and each training using the same data set will result in the same tree structure. In order to take overfitting into consideration and to improve the robustness of the method, Random Forest introduced by Breiman (2001) uses bagging and thus, an ensemble of trees is created. In addition, from the  $J$  features, only a random subset of  $J_b < J$  features can be selected for each tree. A class for a new tested object can be determined by a majority voting of all  $B$  trees.

Also, trees can use a technique called boosting. In this process, each tree uses information from the previous tree to adjust its prediction. The prediction of the previous tree multiplied by a shrinkage factor is added to each prediction of a new tree. The subsequent tree is then not built by using the multivariate description of the objects but by using the residuals. At the beginning, these are the differences of class memberships. The residuals are updated in the course of the training process as the previous residuals minus the prediction from the previous tree reduced by a shrinkage factor. The final prediction is then given as the sum of the individual predictions of the trees weighted by the shrinkage factor. Often a method called stochastic gradient boosting is applied, which combines aspects of randomness as used in bagging as well as the gradient of the learning progress (Friedman 2002).

From the listed algorithms in manufacturing process planning by Fahle et al. (2020), Decision Trees, Random Forest, and Gradient Boosted Trees fall into this category of learning representation. The main advantage of tree-based approaches is given by the tree-structure itself since the rules can be derived in a human-understandable manner similar to the rule-based approaches. Thus, these methods provide a high interpretability but have also the advantage of an automated rule creation without manual efforts. In addition, ensemble learning methods like Random Forest often yield very high accuracies and have an overall good performance in supervised classification tasks (Sagi and Rokach 2018). As an ensemble learning method, Random Forest can also deal with class unbalances where different classes are represented in different frequencies within a data set by balancing the subsets using an over- or undersampling (Batista et al. 2004).

Lastly, methods of hierarchical clustering use a special kind of tree, a dendrogram, which uses the distance between two nodes of the tree as the length of the edges. A data set is in these unsupervised methods clustered by splitting (or combining) the two nearest sets of objects. Thus, a tree structure emerges whereby more similar objects are longer within the same node and separated only in deeper laying nodes. The method used to define the distance measure between two sets of two nodes is called linkage. The most common linkage methods are average linkage which uses the average distance of all objects within the two nodes, Ward linkage which uses the within variance as measure, single linkage which uses the smallest distance of all possible pairs of objects with a pair consisting of one object from each node, and complete linkage which uses the largest distance of all respective pairs.

A list of approaches utilizing tree-based models in different use cases of production planning in manufacturing are provided in Table 3.4. This list gives a representative but not comprehensive overview of the different tree layouts applied across different tasks in production planning.

Isolation Forest is used more commonly in production operation as an anomaly detection model compared to planning tasks, e.g., in predictive maintenance application as investigated by Choi et al. (2022).

**Table 3.4:** Selection of Tree-based models for production planning.

Author	Use Case	Tree Layout
Schuh et al. (2017)	Process planning	Decision Trees
Lingitz et al. (2018)	Forecasting	Random Forest
Zheng et al. (2019)	Scheduling	Random Forest
Garre et al. (2020)	Forecasting	Gradient Boosted Tree
González Rodríguez et al. (2020)	Scheduling	Fuzzy system & Decision Trees
Zhou et al. (2021)	Resource planning	Fuzzy system & Decision Trees
Spoor et al. (2023b)	Forecasting	Random Forest

### 3.2.3.3 Support Vector Machines

The SVM is another popular method for classification and regression. SVMs were introduced by Cortes and Vapnik (1995) and are extensions of the concept of support vectors. This section follows the notation and structure presented by Hastie et al. (2009, chap. 12) in order to introduce the core principles of SVMs.

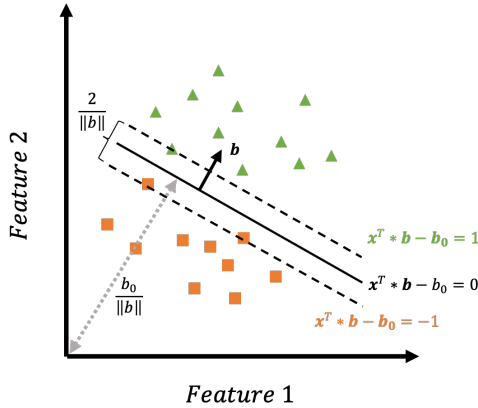
The core idea behind support vectors is that the different classes of objects in a data set are separable by a hyperplane. If a data set with  $J$  features is analyzed, it is possible to define one hyperplane which separates the instances of two classes. A hyperplane is given for an objects  $\mathbf{x}$  which lays directly on the plane as follows:

$$f(\mathbf{x}) = \mathbf{x}^T * \mathbf{b} - b_0 = 0 \quad (3.25)$$

The vector  $\mathbf{b}$  is the unit vector of the hyperplane with  $\|\mathbf{b}\| = 1$ . Since the hyperplane separates the two classes, a classification rule for the Support Vector Classifier (SVC) is derived as follows:

$$\hat{y}_i = \text{sign}(\mathbf{x}_i^T * \mathbf{b} - b_0) \quad (3.26)$$

The hyperplane is placed in the feature space in order to maximize the margin  $M$  between the objects on the one side of the plane and the objects on the other side. If the two classes are separable, the total margin becomes  $2M = \frac{2}{\|\mathbf{b}\|}$ . This is illustrated in Figure 3.13.



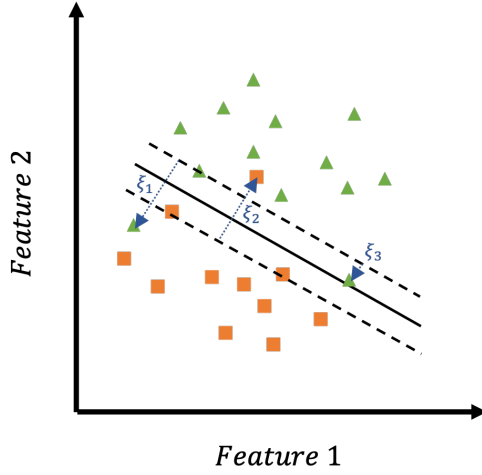
**Figure 3.13:** Schematic concept of a SVM using two separable classes without overlap.

In this separable case, the hyperplane can be found by an optimization problem using the correct class memberships  $y_i \in \{-1, 1\}$  of a number of  $N$  objects. The optimization problem is given as follows:

$$\begin{aligned} \max_{\mathbf{b}, b_0, \|\mathbf{b}\|=1} \quad & M \\ y_i(\mathbf{x}_i^T * \mathbf{b} - b_0) & \geq M, \quad i \in (1, \dots, N) \end{aligned} \quad (3.27)$$

The equation (3.27) denotes a convex optimization problem. However, in most cases a clean hyperplane with a clear-cut margin is not possible. Instead, the space of the two classes might overlap. Thus, each object is assigned a slack variable  $\xi_i$  which describes how much the condition  $y_i(\mathbf{x}_i^T * \mathbf{b} - b_0) \geq M$  is breached. This is shown in Figure 3.14.





**Figure 3.14:** Schematic concept of a SVM using two separable classes with three overlapping objects.

The slack variables can be defined as relative values (unlike Figure 3.14) in order to define another convex optimization problem. The sum of the slack variables is bound to a maximum  $C'$  in order to prevent too many overlapping objects.

$$\begin{aligned}
 & \max_{\mathbf{b}, b_0, \|\mathbf{b}\|=1} M \\
 & y_i(\mathbf{x}_i^T \mathbf{b} - b_0) \geq M(1 - \xi_i), \quad i \in (1, \dots, N) \\
 & \xi_i > 0, \quad i \in (1, \dots, N) \\
 & \sum_{i=1}^N \xi_i \leq C'
 \end{aligned} \tag{3.28}$$

Since this problem is a convex optimization problem, it can be solved and the support vectors  $\hat{\mathbf{b}}$  and  $\hat{b}_0$  can be computed based on the labeled objects  $(\mathbf{x}_i, y_i)$  in

the data set. In order to solve the optimization problem, the problem is rewritten using only the unit vector without the margin as follows:

$$\begin{aligned}
& \min_{\mathbf{b}, b_0} \|\mathbf{b}\| \\
& y_i(\mathbf{x}_i^T \mathbf{b} - b_0) \geq 1 - \xi_i, \quad i \in (1, \dots, N) \\
& \xi_i > 0, \quad i \in (1, \dots, N) \\
& \sum_{i=1}^N \xi_i \leq C'
\end{aligned} \tag{3.29}$$

In order to use a Lagrange formulation, the optimization problem is rewritten using an adjusted constant  $C$  as follows:

$$\begin{aligned}
& \min_{\mathbf{b}, b_0} \frac{1}{2} \|\mathbf{b}\|^2 + C \sum_{i=1}^N \xi_i \\
& y_i(\mathbf{x}_i^T \mathbf{b} - b_0) \geq 1 - \xi_i, \quad i \in (1, \dots, N) \\
& \xi_i > 0, \quad i \in (1, \dots, N)
\end{aligned} \tag{3.30}$$

Based on this optimization problem, the Lagrange primal function is derived:

$$L_P = \frac{1}{2} \|\mathbf{b}\|^2 + C \sum_{i=1}^N \xi_i - \sum_{i=1}^N \nu_i (y_i(\mathbf{x}_i^T \mathbf{b} - b_0) - (1 - \xi_i)) - \sum_{i=1}^N \mu_i \xi_i \tag{3.31}$$

The derivatives  $\frac{\partial L_p}{\partial \mathbf{b}}$ ,  $\frac{\partial L_p}{\partial b_0}$ , and  $\frac{\partial L_p}{\partial \xi_i}$  of the Lagrange function are computed and set to zero. From this results:

$$\begin{aligned}\mathbf{b} &= \sum_{i=1}^N \nu_i y_i \mathbf{x}_i \\ \sum_{i=1}^N \nu_i y_i &= 0 \\ \nu_i &= C - \mu_i\end{aligned}\tag{3.32}$$

By substituting these constrains in equation (3.31), the Lagrange dual objective function is constructed which results in a lower bound of the equation (3.31) for any feasible object.

$$L_D = \sum_{i=1}^N \nu_i - \frac{1}{2} \sum_{i=1}^N \sum_{i'=1}^N \nu_i \nu_{i'} y_i y_{i'} \mathbf{x}_i^T \mathbf{x}_{i'}\tag{3.33}$$

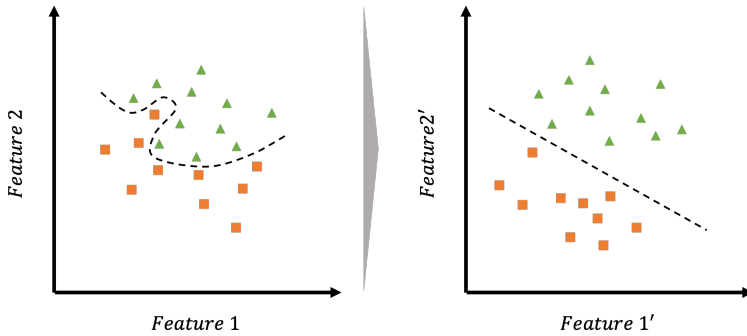
By maximizing this Lagrange function and using the Karush-Kuhn-Tucker conditions, it is possible to find a unique solution for the optimization problem. The resulting support vectors are given by the derivations of equation (3.32) and the estimators  $\hat{\nu}_i$  as follows:

$$\hat{\mathbf{b}} = \sum_{i=1}^N \hat{\nu}_i y_i \mathbf{x}_i\tag{3.34}$$

These support vectors are then used in the SVC in equation (3.26) in order to classify objects with an unknown class membership. The concept can also be adjusted to regression problems using a Support Vector Regressor (SVR).

The concept of SVC as discussed earlier only uses linearly separable boundaries. However, non-linear boundaries are common in practice and thus, the concept of the SVM is introduced.

The core idea of a SVMs is that two classes are not separable in the space of the  $J$  feature using for an object  $i$  the features  $x_{i1}, \dots, x_{ij}, \dots, x_{iJ}$  but might become separable in a higher space constructed using the features such as, e.g., a space with  $2J$  features with  $x_{i1}, \dots, x_{ij}, \dots, x_{iJ}, x_{i1}^2, \dots, x_{ij}^2, \dots, x_{iJ}^2$ . This is called the kernel trick since instead of an object  $\mathbf{x}_i$  the higher feature space representation  $h(\mathbf{x}_i)$  is used. The kernel trick is illustrated in Figure 3.15.



**Figure 3.15:** Schematic concept of the kernel trick transforming a data set using a higher dimensional representation in which the classes become linearly separable.

Since the Lagrange function in equation (3.33) uses the inner product  $\langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle$ , it can also be written as follows:

$$L_D = \sum_{i=1}^N \nu_i - \frac{1}{2} \sum_{i=1}^N \sum_{i'=1}^N \nu_i \nu_{i'} y_i y_{i'} \langle h(\mathbf{x}_i), h(\mathbf{x}_{i'}) \rangle \quad (3.35)$$

The function of the hyperplane in equation (3.25) becomes as follows using the support vectors of equation (3.34):

$$f(\mathbf{x}_i) = h(\mathbf{x}_i) * \mathbf{b} + b_0 = \sum_{i=1}^N \nu_i y_i \langle h(\mathbf{x}_i), h(\mathbf{x}_{i'}) \rangle + b_0 \quad (3.36)$$

Thus, the equations (3.35) and (3.36) only uses the transformation  $h$  of the features within the inner product and thus, only the kernel function  $K(x, x') = \langle h(x), h(x') \rangle$  is required. This kernel function can simply be substituted into the classifier build by the support vectors.

$$f(\mathbf{x}) = \sum_{i=1}^N \nu_i y_i K(x, x_i) + b_0 \quad (3.37)$$

Commonly applied kernel functions are as follows:

$$\begin{aligned} K(x, x_i) &= (1 + \langle x, x' \rangle)^d && \text{Polynomial of degree } d \\ K(x, x_i) &= \exp^{-\eta \|x - x'\|^2} && \text{RBF} \\ K(x, x_i) &= \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2) && \text{Tanh} \end{aligned} \quad (3.38)$$

SVMs are related to NNs. The hinge loss function in equation (3.14) is a realization of the learning approach by SVMs for Perceptrons. NNs also enable multi-class SVMs (Aggarwal 2018). In addition, SVMs are also related to logistic regression (Hastie et al. 2009).

Lastly, SVMs can be adjusted for anomaly detection tasks. Schölkopf et al. (2001) proposes the one-class SVM in order to separate a class of correct objects from outliers. Thus, SVMs can cover multi-, two, and one-class classification as well as regression tasks. Conventional SVMs are supervised approaches and one-class SVM enable in addition an unsupervised feedback representation.

As with the other AI approaches, SVMs are also utilized in different use cases of production planning in manufacturing. Table 3.5 provides a representative but not comprehensive overview of their application.

Since one-class SVMs are primarily used for outlier detection, their application is often in the field of predictive maintenance or quality control, e.g., discussed by Martí et al. (2015) and Abualsaud (2023).

**Table 3.5:** Selection of Support Vector Machines for production planning.

Author	Use Case	Model
Denkena et al. (2019)	Scheduling	SVM
Liu et al. (2005)	Scheduling	SVM
Denkena et al. (2021)	Process planning	SVM
Rožanec et al. (2021)	Forecasting	SVR
Doraid Dalalah and Marshall (2023)	Forecasting	SVM

### 3.2.3.4 Neighborhood and Density-based Models

While NNs and SVMs are related to each other and tree-based methods often utilize a within variance similar to the least square method which also acts as commonly applied basis for loss functions of NN, neighborhood and density-based models follow a comparably different approach by evaluating the objects in a close neighborhood and their respective distance towards each other.

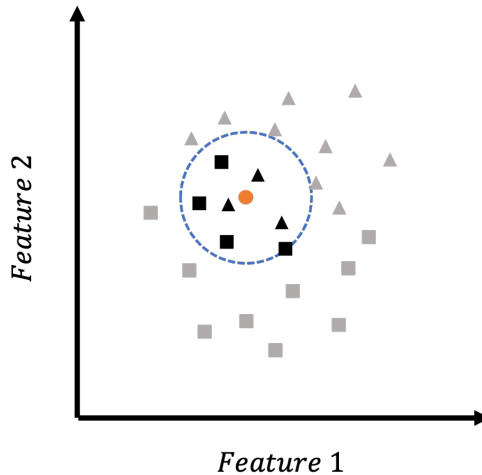
As a relevant representative of algorithms which use an object's neighborhood to derive inference, the  $k$ -nearest neighbors (KNN) algorithm is often applied in practice. The principle of KNN was first introduced by Fix and Hodges (1951) and later fully developed by Cover and Hart (1967). In short, a KNN selects a number of  $k$  closest neighbors of an evaluated object with an unknown class membership using a distance metric  $d(x, y)$  between the objects  $x$  and  $y$ . The selected neighboring objects are from a data set with known class memberships. Thus, KNN belongs to the class of supervised learning algorithm regarding feedback representation. The applied metric in order to measure a distance between the two objects  $x$  and  $y$  with  $J$  features is in most application a  $L^p$  metric defined as follows:

$$d(x, y) = \|x - y\|_p = \left( \sum_{j=1}^J |x_j - y_j|^p \right)^{\frac{1}{p}} \quad (3.39)$$

The set of the  $k$  closest objects using this distance become the set  $\mathcal{N}_k(x)$  of neighbors. The Euclidean distance is a commonly used special case in the  $L^2$  space with  $p = 2$  in applications of KNN.

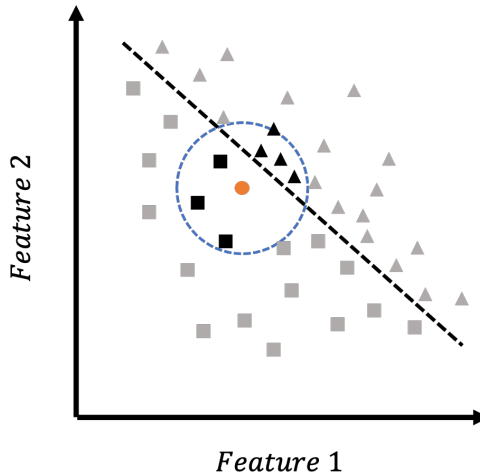
The class membership of these neighbors is then recorded in order to evaluate the class membership of the tested object by a voting system. The voting can either be conducted using a majority vote or a weighted voting procedure. The weighting is commonly based on the distance, but other weighting procedures are possible depending on the use case. In regression task, the predicted value is analogously assigned by an average (or weighted average) computed over all neighbors.

An illustrative exemplification of a KNN classifier is given in Figure 3.16 using the  $k = 7$  nearest neighbors and the Euclidean distance metric. If a majority vote is applied, the class would be determined as "square" since the majority of neighbors belong to this class. However, a distance-weighted vote might result in a classification as "triangle" (depending on the set-up of the weighting procedure) since the two closest objects belong to this class.



**Figure 3.16:** Illustrative KNN classifier using the  $k = 7$  nearest neighbor within Euclidean distance.

While the design and mechanism of KNN classifiers can be intuitively understood, the resulting classifications however lack sufficient causal explanation in certain edge cases. The objects within a data set might have a clear edge in the feature space between two classes, e.g., by a linear curve which in theory can separate the data, but the objects are differently dense per class. Thus, a majority of cases on the one side of the edge could overrule the objects on the other side of the edge in case of objects close to the edge. This is illustrated if the number  $k$  of evaluated neighbors becomes close to the total number of objects within a certain area of the feature space and thus, all tested objects will be determined as the majority class in this area. Figure 3.17 illustrates the described problem.



**Figure 3.17:** Exemplary miss-classification of linearly separable objects by a KNN classifier using the  $k = 7^{th}$  neighbor within Euclidean distance.

This consideration of miss-classification is in particular important if there exists a class imbalance in the data set. Therefore, the parametrization of  $k$  should be carefully analyzed and the density and balance of the data set taken into account. However, KNN's easy to understand principle makes it a commonly used algorithm in practice.



Next, the Local Outlier Factor (LOF) is a representative algorithm which uses the density of neighboring objects. LOF was introduced by Breunig et al. (2000) and is related to the clustering algorithm of Density Based Spatial Clustering of Applications with Noise (DBSCAN) by Ester et al. (1996) which is also formulated in more detail by Schubert et al. (2017).

In LOF, the set  $\mathcal{N}_k(x)$  containing the  $k$  nearest neighbors of an object  $x$  is created for each object in the data set. The  $k$ -distance  $d_k(x)$  denotes the distance to the  $k$ -nearest neighbors of an object  $x$ . In addition to a  $L^q$  distance, the reachability distance  $d_k^r(x, y)$  is defined as follows:

$$d_k^r(x, y) = \max(d_k(y), d(x, y)) \quad (3.40)$$

The reachability distance is therefore either defined as the distance  $d(x, y)$  within the  $L^q$  space, if object  $y$  is not in the neighborhood of object  $x$ , or as the distance of object  $y$  to its  $k$  neighbor, in the case that objects  $y$  is within the neighborhood of object  $x$ . Thus, the distance of an object  $x$  to an object  $y$ , is at least the distance of the most distanced object in the respective neighborhood of object  $y$  since  $d(x, y) < d_k(y)$  if object  $x$  is within the neighborhood  $\mathcal{N}_k(y)$ . However, the distance from object  $x$  to  $y$  might differ from the distance from object  $y$  to  $x$ . Thus, this defined distance measure is not symmetric<sup>1</sup>.

In order to compare the densities of the data points, a local reachability density, which is the inverse of the average reachability of an object to its  $k$  direct neighbors, is defined as follows:

$$\rho_k(x) = \frac{k}{\sum_{y \in \mathcal{N}_k(x)} d_k^r(x, y)} \quad (3.41)$$

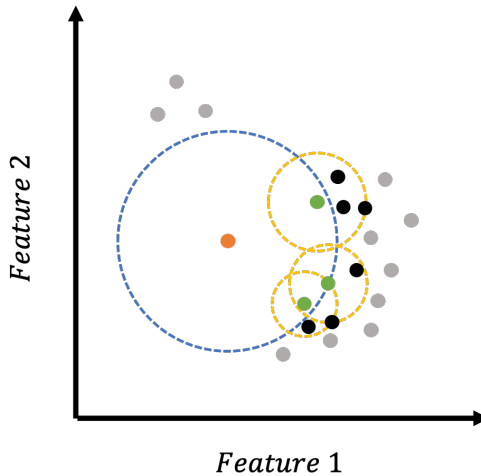
---

<sup>1</sup> The defined distance it therefore not a distance in the mathematical sense as later defined in more detail in section 4.4.3

The LOF score is then evaluated as follows:

$$LOF_k(x) = \frac{\sum_{y \in \mathcal{N}_k(x)} \frac{\rho_k(y)}{\rho_k(x)}}{k} \quad (3.42)$$

A LOF score close to 1 indicated that the ratio  $\frac{\rho_k(y)}{\rho_k(x)}$  of the local reachability density of an object compared to its neighborhood is very similar and thus, yields a similar density as its  $k$  nearest neighbors. An average ratio of  $\frac{\rho_k(y)}{\rho_k(x)}$  greater 1 indicates that the objects in the direct neighborhood have other objects than object  $x$  which are closer to them compared to the evaluated object  $x$ . This is the result as the non-symmetric equation (3.40) assigns higher distances to objects which are not in the direct neighborhood, in which all other objects have the same distance of the  $k$  nearest object, and thus, equation (3.41) results in a lower density compared to other objects. This relation is illustratively given in Figure 3.18 by an exemplary outlier detection task using LOF.



**Figure 3.18:** Illustrative outlier detection with LOF using  $k = 3$ . The neighbors (green) of the outlier (orange) have lower reachability distances and thus, higher local reachability densities.

LOF can be applied for every object within a data set as a method of anomaly detection. In contrast to KNN, LOF does not require labeling and thus, is an unsupervised algorithm regarding its feedback representation. This enables LOF to identify unusual or anomalous objects within a data set without labeling.

Despite KNN and LOF, other methods and variants of these algorithms exist. One further algorithm to mention is the Histogram-based Outlier Score (HBOS) by Goldstein and Dengel (2012) as an unsupervised outlier detection approach utilizing density expressed by histograms. HBOS compares the occurrence of objects within the histogram bins, and thus a discrete density measure, to derive an anomaly score.

Again, Table 3.6 gives a representative but not comprehensive overview of approaches using neighborhood and density-based models and their respective applications. In some cases, the base model as listed in Table 3.6 is adjusted for the respective use case.

**Table 3.6:** Selection of neighborhood and density-based models for production planning.

Author	Use Case	Model
Troncoso Lora et al. (2003)	Forecasting	KNN
Sheng et al. (2018)	Forecasting	LOF & Gaussian regression
Kück and Freitag (2021)	Forecasting	KNN
Deng et al. (2022)	Process planning	KNN
Öngelen and İnkaya (2023)	Forecasting	LOF & KNN

It should be noted that LOF is seldom applied in planning related tasks. The use cases by Sheng et al. (2018) and Öngelen and İnkaya (2023) utilize LOF instead as a weighting mechanism for other algorithms which are conducting the prediction. However, LOF is more commonly applied in monitoring and fault detection tasks during factory operations, e.g., by Ma et al. (2013).

**3.2.3.5 Statistical Approaches**

The field of statistical methods comprises a wide variety of supervised models, such as linear regression, logistic regression, Gaussian processes, entropy measures, or Bayes statistics. In addition to supervised models, unsupervised models also fall under this umbrella term such as K-means or Gaussian Mixture Models for clustering tasks. While these statistical approaches are sometimes not described as AI or ML models, they are clustered in this thesis in this category since they do not require the set-up of an optimization function by a domain expert.

Since an individual discussion of all statistical models would exceed the scope of this section, Table 3.7 shows a set of representative but not comprehensive statistical approaches in production planning in manufacturing.

**Table 3.7:** Selection of statistical approaches for production planning.

Author	Use Case	Model
Salehi et al. (2020)	Forecasting	Bayes statistics
Tan et al. (2021)	Forecasting	K-means & LSTM
Brasington et al. (2022)	Process planning	Gaussian process & Bayes statistics
Zhou et al. (2022)	Forecasting	Gaussian process
Santander et al. (2023)	Process planning	Bayes statistics









The tools for statistical analysis, e.g., linear regression, are built into many state-of-the-art planning tools and are therefore often utilized in practice. However, since this practical work yields very little novelty, these practical approaches are seldom published in recent scientific papers.

### 3.3 Evaluation of the State-of-the-Art

This section aims at answering two main questions, in order to derive the research question of this thesis in more detail.

1. What use cases are currently under-research in the body of literature? This includes the questions whether dominant models, which are primarily applied for the specific use cases, emerged and act as quasi standards for solving the use cases.
2. What are the theoretical limitations of the discussed models which hinder an application in the under-researched use cases? Thus, the mathematical implications and foundation of the models must be discussed within the context of the use cases in order to find the reasons why no model emerged as a dominant solution.

First, in order to further analyze the gaps within the body of literature, the prominence and maturity of the use cases and their respective solutions within the body of literature based on the findings of the conducted literature review are summarized in Figure 3.19.

	Use Case	Prominence of Research [nominal]	Maturity of Research [nominal]	Primarily Applied Models
1	Forecasting			<i>RNN, Random Forest, SVR, KNN</i>
2	Process Planning			—
3	Resource Planning			<i>Optimization Models</i>
4	Scheduling			<i>Q-learning</i>

 = low     = medium     = high     = very high

**Figure 3.19:** Evaluation of prominence, maturity, and primarily applied models for production planning use cases in the body of literature.

The body of literature focuses primarily on use cases of forecasting, scheduling, and resource planning, in particular using ALBP. Forecasting and scheduling use cases have a high maturity of the research methods. Scheduling is mostly conducted by agent-based systems using a reinforcement learning mechanism and Q-learning. This enables highly dynamic scheduling methods. Solutions differ between use cases since the design of the reward function is a major challenge in order to design such a system. This complex reward function design is a current limitation for generalization of the models. However, many authors show sufficient prototypes and use cases. Similarly, the resource planning is mostly conducted using optimization model where the main concern is the set-up of a useful optimization function. The solvers for the optimization function differ between authors and no dominant solver which performs leading in multiple use cases is available (which is another application of the no-free-lunch theorem).

One interesting aspect is that the conventional solution for scheduling problems was until recently to use optimization functions and respective solvers. However, in the current literature, Q-learning and reinforcement learning approaches are more dominant and might replace optimization models in the future. The rationale is that it is no longer necessary to develop a specific optimization function and constrain functions but to enroll a more generic agent-based system with a useful reward function. This, after a training procedure, does no longer require the computational complexity compared to the solving of a large-scale optimization problem in case of adjusted parameters since the underlying NN might already provide a good surrogate function to cover the solution space. A similar process might be applicable for resource planning models where the majority of research utilize optimization functions. Thus, a more data-driven and generic approach using easy-adjustable and enrolled AI methods might replace the use of optimization models in the future.

Forecasting is also a common use case in production planning, but the applied methods vary. In contrast to resource planning and scheduling, no dominant model has emerged in the literature. In contrast, ensemble methods seem to be a solution which prove themselves in some scenarios as more efficient. By applying an ensemble of models, forecasting yields a medium maturity: the solutions are

often precise, but no single model can create a causal prediction process which can simply be followed.

Two under-researched fields emerge however: first, process planning and second, resource planning without the use of optimization models. No primarily used models and strategies for process planning emerge and the use cases are currently limited by a lack of interpretability of the result created by, e.g., NNs. Rule-based models are still leading solutions in these applications in practice which require a high amount of expert knowledge and ontologies in order to derive a useful (fuzzy) rule set. Thus, these use cases can be identified as relevant gaps in the body of literature without a commonly applied data driven models and best practices. These use cases will therefore be investigated in more detail using the developed and introduced model in this thesis. In addition, models and approaches combining process planning as well as resource planning in a multi-criteria approach are rare and mostly focus on one use case primarily using a given solution from the other use case. This is consistent with the observation by Hagemann (2022) that the information models in Figure 2.14 are not continuously modeled and they do not sufficiently interact with each other in practical production planning applications.

An interesting aspect from the author's point of view is that it is quite easy to find a large body of prominent use cases, prototypes, and implementations of scheduling tasks in a literature review using "scheduling" as a keyword as described in Figure 3.2. In contrast, the majority of highly cited and prominent papers using keyword such as "production planning" or "process planning" are reviews or meta concepts about the integration of AI methods without specific showcases. While the field of scheduling proves the methods' applicability, the field of process planning is more focused on fundamental discussions about implementation frameworks, application scenarios, or strategies.

On a critical note, one could argue that the research community of process planning often talks about AI but seldom uses it. In addition, the discussed use cases often lack the necessary depth in evaluating and implementing the AI models, e.g., no or only a rough discussion of the (hyper-)parametrization, use of only a hold-out instead of a full k-fold or Monte Carlo cross-validation, and a limited discussion of

model intrinsic limitations. Instead of a fine-tuning or adjustments of models, out-of-the-box frameworks are often applied and benchmarked. This might indicate a lack of engagement by experts from the domain of manufacturing with the mathematical foundation of AI models which may hinder research progress in this area. These considerations are key aspects behind the rational of the conducted maturity assessment in Figure 3.19. As Garre et al. (2020) also note, a reason for the limited practically applied use cases might be the reduced interpretability of AI methods for production planning experts. Additionally, since many models are simply enrolled without fine-tuning or adjustment, the lack of novelty in these scenarios might restrict the amount of published use cases on real data and applications in production planning in manufacturing.

In addition to the gaps in the literature, the models' theoretical set-ups are in addition summarized and a conclusion for their respective applicability is discussed. The high-level descriptions, advantages, limitations, and use case of the different methodical approaches are listed in Figure 3.20.

First, rule-based models require the set-up of an algorithmic rule set to correct, update, or annotate planning data along the planning stages. While rules can follow the typical if-else logic, a more complex arrangement of logical test can also be modeled by combining rules. These solutions are developed for a certain problem using extensive domain knowledge with varying detail levels. Most noteworthy is that they require a manual update in case of changes in the environment, such as used technologies. These approaches are deterministic and deductive, but the quality of results might not be optimal since no true optimization is conducted. However, the computational effort is low and the algorithm themselves are highly human understandable and interpretable.

Second, optimization models use solvers or mathematical heuristics in order to find global optima of an optimization function. A given problem can be modeled using this optimization function and then a solution can be derived. In the case of support systems, a planning expert might input the given features of a planned production system and use the support system in order to optimize the selected parameters or identify restrictions which are not met. The set-up of the



	1 Rule-based Models	2 Optimization Models	3 Artificial Intelligence Models
<b>Quality of Results</b>	Highly dependent on the quality of the rule set, but often not optimal	In the case of good optimization function, close to an optimal solution	Highly dependent on the model, but rarely without any miss-classifications
<b>Implementation Effort</b>	Extensive domain knowledge is required for the rule generation	Extensive domain knowledge is required for the definition of an optimization function and restrictions	Effort is required for a collection and cleansing of input data, the model selection, and (hyper-) parametrization
<b>Computational Complexity</b>	Scales linearly with number of rules	NP-hard and exponential with the amount of features	P-hard and polynomial with the amount of features
<b>Deterministic</b>	Deterministic	Non-deterministic since random processes might be used to approximate optimum	Mostly non-deterministic in the case of permutations or changes of training data
<b>Interpretability</b>	Very high by a conductional rule set	High since solution can be verified and tested using the optimization function	Highly dependent on the model, but commonly low due to a black box characteristic of the models
<b>Advantages</b>	<ul style="list-style-type: none"> <li>Very high interpretability and acceptance of models by user</li> <li>Comparability low effort in setting up relatively good solutions</li> </ul>	<ul style="list-style-type: none"> <li>Solutions are close to optimal</li> <li>Very high interpretability by the use of an optimization function and restrictions</li> </ul>	<ul style="list-style-type: none"> <li>Solution considers unknown patterns which are not included in rule-based or optimization models</li> <li>Domain knowledge is only required for testing and data collection</li> <li>Given the availability of training data, the models can be set-up fast</li> </ul>
<b>Limitations</b>	<ul style="list-style-type: none"> <li>Solutions might be sub-optimal</li> <li>Very reliant on domain knowledge for the rule definition</li> </ul>	<ul style="list-style-type: none"> <li>Very high effort in defining the optimization function</li> <li>Very reliant on the quality of the optimization function</li> <li>High computational effort</li> <li>Risk of finding only local optima</li> </ul>	<ul style="list-style-type: none"> <li>Requirement of a high number of cleansed and labeled training data</li> <li>Low interpretability and black-box for user, thus often low acceptance by practitioner</li> <li>Solutions are often sub-optimal or sometimes even erroneous</li> </ul>
<b>Conclusion</b>	Useful for well-understood problems with extensive available domain knowledge and limited complexity	Useful for delimitable problems where precise solutions are required and an optimization function can be set-up	Useful for complex larger problems with limited available knowledge but available training data

**Figure 3.20:** Overview of the characteristics, advantages, limitations, and use cases of the different methodical approaches for support systems in production planning.

optimization function and restrictions is crucial in these methods since they build the target for all optimization procedures and missing restrictions or incorrect functions can highly distort the quality of the results. Thus, extensive domain knowledge is also required. The resulting solutions will be optimal, or close to optimal, based on the input data and optimization function. These models are not deterministic, since methods such as simulated annealing use randomly generated values in order to limit computational effort and efficiently test the search space. The closer the model should come to the true optimum, the more computational effort is required. Since most optimization functions are NP-hard, the computational complexity is assumed to raised exponentially with the number

of searched features. With knowledge about the optimization function, the results of the optimization can be manually tested and interpreted.

Third, AI models are quite diverse and serve different use cases and require different attention to different parametrization obstacles and have different limitations. AI models benefit from the data-driven approach since no optimization function or knowledge derived from domain experts is required in their set-up. However, they still require a testing and (hyper-)parametrization procedure and a solid understanding of their applicability. Most importantly, supervised AI models require a labeled data set which must be priorly created. In particular models with a strong performance such as DNNs require a large data set in order to produce sufficient results. Reinforcement models require a set-up of a reward function, which is similarly complex as the set-up of an optimization function and unsupervised learning is mostly relevant for outlier and anomaly detection. The biggest draw-back of AI models, in particular NNs, is their "black box" characteristic which hinders their acceptance by domain experts in practical use cases. While misclassifications will occur in most AI models, a bad rule set or optimization function in the other approaches might also result in suboptimal solutions. While the computational complexity of a NN as well as its accuracy for complex problems with many features scales with the number of layers, the training can be conducted in advance and the testing of new objects during operation of the system is considerably faster.

In conclusion, AI models seem on paper the most suitable for very complex problems within production planning since little effort in constructing a problem-specific knowledge base is required and since the existing data sets and knowledge bases can be utilized. Their low interpretability and perceived complexity however hinder a wide-scale application in practice. For smaller or easy to delimit problems with a little number of features to consider, rule-based approaches and optimization problems might still be beneficial.

## 3.4 Open Research Questions

The body of literature indicates that AI models in production planning are under-investigated, in particular in process and resource planning. However, AI models might enable intelligent process and resource planning if the right circumstances are given. In addition, there is a lack of models combining process and resource information models within a multi-criteria approach.

First, a sufficient amount of training data must be available. While manufacturing companies have an increasing amount of data available (Tao et al. 2018), faulty data is often deleted during the planning procedure and only correct planning data is available (Spoor et al. 2022a). In addition, contextual data must be leveraged to form an understanding of the problem and possible solutions (Friederich and Lazarova-Molnar 2021).

Second, the AI model must have an acceptance from the planning experts during operation since this is often not given in use cases (Garre et al. 2020). This acceptance can only be achieved by overcoming the "black box" characteristic of AI models without forfeiting too much predictive power of the model (Goldman et al. 2021).

Third, the use cases of process planning and resource planning should be focused since the maturity and prominence of research in these fields of production planning is relatively low compared to scheduling and forecasting. An introduction of an AI model in order to solve problems within these to use cases could enrich the current body of literature and add a significant contribution to the state-of-the-art in AI models for production planning.

Based on these considerations, a main research question is derived.

"How can an AI model be developed which addresses the use cases of process and resource planning using the available databases and knowledge within a manufacturing company without the need of contributing a significant effort in manually adding use-case specific additional domain knowledge during the procedure?"

Based on this research question, the main focus of the research of this thesis lies on the task of process and resource planning in a data-driven approach utilizing existing data which can be enrolled without prior fine-tuning of the knowledge base, data collection, or manual labeling of data sets.

This research question is accompanied by a second objective in order to ensure an applicability and acceptance by the domain experts in production planning.

"How can a highly accurate and data-driven AI model for process and resource planning be designed in order to be sufficiently interpretable by the domain experts and to also generate human-understandable knowledge which is returned to the domain experts and knowledge base of the manufacturing company?"

This second questions aims at reducing the "black box" characteristics of commonly applied AI models, in particular DNNs, and thus enable a structured data and knowledge management.

This thesis aims at providing a novel approach which sufficiently answers these questions and provides practitioners with a data-driven, accurate, and human-understandable model in order to solve complex tasks in production planning with a focus on process and resource planning. The resulting AI model which fulfills the demanded framework conditions can therefore enhance the state-of-the-art of support systems in production planning and close a major gap in the current body of literature.

## 3.5 Summary

First, this section provides a distinction between different solution approaches which are applied in support systems for production planning: rule-based models using manual constructed rule sets utilizing domain knowledge by experts, optimization models which require the careful definition of an optimization function, and AI models which solve the underlying problem in a data-driven manner using a

structured machine-readable knowledge base. In addition, the most important use cases in production planning are derived, namely: forecasting, process planning, resource planning, and scheduling.

By conducting a structured and comprehensive literature review, the body of literature is recorded and discussed. In addition, the concepts and core principles behind each method and model are discussed in more detail. In particular, NN, tree-based model, SVM, KNN, and LOF are introduced as state-of-the-art AI models applied for production planning in the literature. Using the results from the structured literature search, representative publications using a wide variety of models across all defined use cases within production planning are listed for each approach. The body of literature is then critically discussed with the result that the prominence and maturity of the uses cases of process and resource planning using AI models is comparably low and that these use cases are currently under-investigated. One reason for this research gap is the "black box" characteristic of AI models and the limited availability of labeled and structured data.

Using this research gap, two research questions are derived. In summary, this thesis aims at providing a data-driven, highly accurate, and human-understandable AI model which can in particular be applied in the use cases of process and resource planning. These tasks lay the foundation for the subsequent objective, conceptualization, and layout of the proposed novel AI model for support systems in production planning.



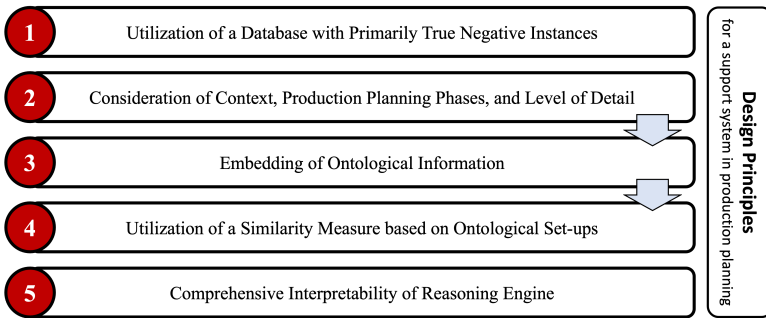
## 4 Methodology

As solution for the previous described limitations of the models in the body of literature and in practical applications, a support system is proposed using an AI model for a streamlined and explainable inference procedure which utilizes only as correct labeled configurations within a statistical approach. The objective of this model is a fault and inference procedure of a newly planned configuration or production layout during production planning. This inference should enable a detection of incorrect configurations, an identification of similar configurations, and an inference on possible errors, bottlenecks, or inefficiencies for the new configuration based on already known and implemented configurations.

First, the overall design principles to develop a support system which can be utilized within production planning are introduced and described according to the findings by Spoor et al. (2022a). The resulting objectives are derived in the subsequent second section. Based on these principles, the third section describes the overall sequence of the procedure and introduces the core functionalities of the support system. The proposed design is a set-up within production planning in manufacturing but discussed in a generic manner so that an adaptation to other domains is possible. The core of this introduced support system is a novel AI model. This AI model is based on the concept of a modified Hopfield neural network by Spoor and Weber (2024) and further introduced in the fourth section. Within this section, the AI model is analyzed regarding its network layout, its derived anomaly detection model and classification model are introduced, and its explainability capabilities are discussed. Concluding, the fifth section discusses how the introduced Hopfield neural network layout is embedded within the Digital Factory of manufacturing and how the AI model corresponds to the former proposed elements of the support system.

## 4.1 Design Principles

Analyzing support systems for fault detection, Spoor et al. (2022a) propose fundamental principles for support systems in production planning improving former considerations by Gelwer et al. (2020). Building on these formerly proposed principles and considerations, five relevant considerations for AI support systems in production planning are derived and highlighted in Figure 4.1.



**Figure 4.1:** Five applied design principles of an AI support system based on Spoor et al. (2022a).

The applied design principles in Figure 4.1 and their reasoning can be described in more detail as follows:

1. Although manufacturing companies are collecting increasingly larger amounts of production data and are utilizing Big Data capabilities (Tao et al. 2018), Arinez et al. (2020) state that obtaining data for the training of a system is difficult for researchers since a wide-scale data collection is often operationally prohibitive and data access and exchange is often restricted due to security concerns. Furthermore, faults and errors in applied production systems are very rare and thus, labeled incorrect data for training AI models are also rare. Since the resulting training data sets are either heavily imbalanced or even consisting only of true negatives, the detection of faults must be capable of a training using smaller data sizes and must be conducted based on a normal model as used in anomaly detection (Spoor et al. 2022a).



2. In addition to specific production system configurations, contextual information must be additionally used and embedded into a fault detection model. Context is capable of characterizing a situation and should encompass within manufacturing typical entities such as parts or products as well as planning information. This context is then able to improve decision making (Giustozzi et al. 2018). Furthermore, contextual data such as metadata about production assets or condition monitoring might be able to enrich a fault detection and might enable a causality analysis of detected faults in the production system (Friederich and Lazarova-Molnar 2021).
3. Any production system configuration must utilize an ontological structure to provide and express the required necessary context besides the quantity structure of the proposed configuration. The utilization of ontologies is important since the quantity structure does not disclose the relation of the included entities and do not describe similarities along an ontological hierarchy (Spoor et al. 2022a). Therefore, the requirement for the ontology is directly derived from the contextual data requirement since ontologies provide context awareness (Dalzochio et al. 2020). Ontologies enable a human-understandable as well as machine-readable way of expressing formal models from multiple knowledge domains. Thus, domain experts and the support system can utilize the same database and models. *Vice versa*, ontologies provide a user highly interpretable and logical structures which can be utilized by experts to gain further insight into the application (Feilmayr and Wöß 2016).
4. Based on the developed and used ontology, a similarity measure must be derived to compare configurations of production systems with each other. This measure must enable a comparison independently from the hierarchical positioning within the quantity structure and must also be compatible with configurations during, after, and at the start of the production planning. A quantifiable similarity measure can derive further context about a configuration by comparing it directly with similar instances and documented former lessons learned (Spoor et al. 2022a).

5. Any results, recommendations, and decisions taken must yield a high interpretability and the reasoning must include an explainability capability since domain experts must use and evaluate all results by the support system in order to implement changes and proposed countermeasures. Thus, users must trust and understand the reasoning engine implemented in the system. Goldman et al. (2021) claim that the usage of explainable systems can facilitate the adoption and implementation of AI models in manufacturing.

It should be noted that principles number 2 up to 4 are associated and successively build on each other.

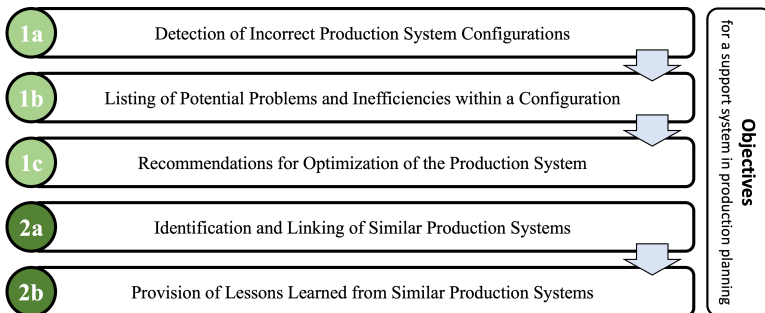
Under the consideration of these principles, it is necessary to develop a support system along an inference procedure for the evaluation of production configuration within different stages of production planning, macroscopic or microscopic planning scenarios, as well as within different level of details. It is proposed in this discussion to separate the user's view of the inference procedure from the components of the support system. The users of the support system should be capable of analyzing their production planning configuration along logical business process steps reflecting the planning approach. The support system is regulating this inference procedure by providing the necessary inputs and data.

Besides the database consisting of applied production data, planning information, and the ontology, the AI application is a major component of the support system. The AI system should fulfill the principles of using only true negatives during the training procedure and the application of an appropriate similarity measure to draw inferences. On the other hand, the overall manual configuration procedure by planning experts is not evaluated using statistical information but using human logic, currently implemented rule-based tools, and practical use cases and planning scenarios. It is therefore important that the users of the support system specify context and ontological information during configuration in order to be able to run a statistical inference analysis driven by the AI model. To enable the support system, the manual configurations are designed using the ontology and thus, are already machine-readable by the AI model.

## 4.2 Objectives

The core idea of the support system is to give users who have entered a newly planned configuration of a production system into the support system, recommendations about optimizations of the entered configuration, provide a list of possible errors within the configuration and accurate countermeasures to fix these, and to provide details about the lessons learned from former similar production system configurations. Lastly, the system must also include further context about possible considerations when dealing with similar configurations and utilize the documentation of experts who have worked on former planning projects. Thus, the system should first, include an anomaly detection to find errors within configurations and second, a classification to find similar configurations of production systems. All objectives must be enabled by a high explainability, and the system must provide human-understandable interpretations of its decisions.

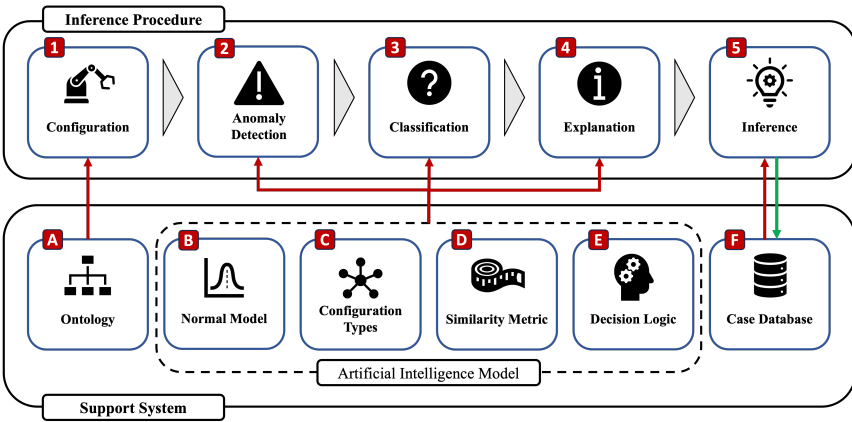
These objectives are directly derived from the discussed research questions and summarized in Figure 4.2 along the task of anomaly detection and classification. The principle of explainability is an enabler of the objectives, in particular for the descriptive task, but not a separate technical task in itself.



**Figure 4.2:** Objectives of an AI support system separating the associated tasks related to (1) anomaly detection and (2) classification.

### 4.3 Design of the Support System

Using the derived design principles, the sequential steps to fulfill the discussed objectives of the proposed inference procedure during production planning tasks are schematically visualized in Figure 4.3. In addition, the core components of the AI support system as inputs along the inference procedure are also given.



**Figure 4.3:** Schematic overview of the sequential inference procedure and of the elements of the introduced support system.

The support system yields five steps along the inference procedure which are enabled by six core functions within the backend of the support system. The five steps of the inference procedure are further explained as follows:

1. **Configuration.** This is the new layout, structure, parametrization, and set-up of the analyzed production cell or whole production line proposed by the experts from production planning. The configuration is a new but similar approach for a production system layout.

2. **Anomaly Detection.** The defined configuration in this step is analyzed for errors or incorrect configured set-ups. This is therefore the first validity check whether the newly planned production system is set-up correctly.
3. **Classification.** If the proposed configuration is valid, it can be classified within the known types of different production layouts. This step is important if multiple approaches or types of set-ups exist with each type yielding different characteristics such as specific errors. By classification, the lessons learned from former configuration can be applied for the newly planned configuration.
4. **Explanation.** If the configuration is labeled as incorrect during the anomaly detection, an explanation why the system derived this assessment is given in this step. Also, the reasoning behind its classification is provided and the specifics of this particular configuration and differences to the existing classes are displayed.
5. **Inference.** In case of incorrect configurations, a proposal is provided on how to correct this configuration. In the case of a classification, the known errors, specifics, bottlenecks, costs, and lessons learned from this class of configurations are displayed and first countermeasures proposed. Similar existing layouts and configurations are also listed.

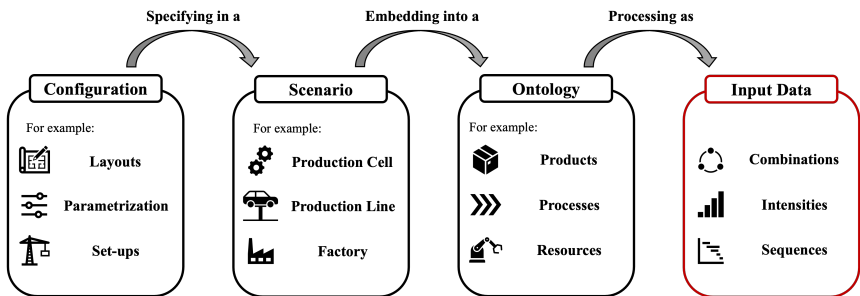
While anomaly detection and classification are displayed in Figure 4.3 as consecutive steps, it is possible to utilize only one or both functions within an applied procedure depending on the target and context of the analysis and thus focusing on either objective group (1) or group (2) of Figure 4.2. The explanation step is also from the perspective of an end user considered optional since further context on the recommendation by the system is not in all cases required by the users of the support system. In addition, the explanation step can highly vary regarding the scope of the analysis depending on the requested context from the user.

The steps of the inference procedure are enabled by the following six core functionalities of the support system, whereby the functionalities B up to E are encompassed by the applied AI model.

- A **Ontology.** Base for any configuration is the development, implementation, and utilization of a valid ontology. Any configuration is described by an ontological layout and thus, this ontological assessment is the initial point of interest of any support system. Thus, the ontology is an input for any planned configuration and all former configurations are included in its design.
- B **Normal Model.** To detect a combination which is anomalous, the baseline of this assessment is the definition on what a normal configuration within the context of the applied production planning task is. Normal in this context is stated as the absence of incorrect, rare, or novel layouts and set-ups of the configuration.
- C **Configuration Types.** These are essentially the possible layouts used in the context of the specific applied planning tasks. The core idea is that multiple possible approaches exist and any new configuration can be classified as belonging to one (or multiple) of these approaches.
- D **Similarity Metric.** To enable a comparison with a normal model or any layout type, it is necessary to define a metric which describes the similarity or dissimilarity to the normal model or to other classes. This metric should enable a direct comparison regarding the similarity of the newly planned configuration and existing configuration(-types).
- E **Decision Logic.** The AI model must encompass a structured reasoning why certain assessments were conducted and provide an explainable mathematical and statistical approach behind recommendations or decisions.
- F **Case Database.** This database includes all lessons learned from former configurations. Using the defined similarity metric, this enables a direct evaluation of errors occurring in former configurations and possible solutions on how a newly planned configuration might prevent these errors. Results from newly planned configurations are fed-back into this database.

The foundations of this evaluation are the configurations developed by production planning experts. These configurations might include detailed layouts of robotic facilities, parameterizations of the used components, or whole set-up descriptions.

To analyze configurations in a structured manner, a data preprocessing of these configurations is proposed. First, the support system is extracting the context and particular scenario of the planned configuration. This can be granular use cases of a single production cell but also configurations of whole production lines or factories. For each particular scenario, the configuration is embedded into a corresponding ontology, e.g., the PPR model, using for the applied scenario an appropriate ontological hierarchy level. The resulting input data of the model are abstractions of the configuration using the applied combination of instances within the configuration (meaning there exists a relation between the two ontological instances), the intensity of applied relations along these ontological instances, and sequential information of the order of the instances. The data preprocessing is visualized in Figure 4.4.



**Figure 4.4:** Data preprocessing of a configuration within a specific scenario by embedding the configuration into an ontology to convert the configuration into the applied input data format.

Using this approach, each production planning set-up or layout is displayed as a combination of, e.g., processes, resources, and products, which together build the PPR model as used by, e.g., Agyapong-Kodua et al. (2014) or Ferrer et al. (2016), or additionally other entities from ontologies such as sensors, location,

or situation used by Giustozzi et al. (2018). Thus, each production system uses a specific combination of these ontological entities.

For simplicity and generalizability in other use cases, the ontological instances are in the following called objects. An object is defined in the introduced methodology as a single instance from the used ontology on a corresponding hierarchical level of the analyzed scenario. An object is called active if it is within an applied combination. Since each object belongs to a group based on its ontological hierarchy, the combinatorial description can be adjusted for a higher-level view of object groups using a specific hierarchy level. Furthermore, restrictions to the combinability of the applied combinations of objects or object groups apply in a production planning task. Since the number of combinations might be too large to individually check for restrictions, a statistical approach is necessary and rule-based approaches cannot be applied. Furthermore, combinations of objects with themselves must be considered and restriction might also include rules for combining objects with themselves or objects from groups with other objects from the same group if a higher-level view is used in the analysis.

The proposed support system adheres to the formulated principles in Figure 4.1 by utilizing a normal model and classes trained on only true negative data, by pre-processing its data using an ontology and context-based scenarios taking the steps in Figure 4.4, and by applying a similarity metric which enables a direct comparison of planned configurations with former similar configurations. Thus, an AI model must also be developed which is capable of fulfilling these defined requirements. Its core functionalities directly enable the support system to fulfill the listed objectives in Figure 4.2.

## 4.4 Modified Hopfield Neural Networks

As a feasible AI model which is able to fulfill the prior defined design requirements for a support system and inferences procedure in production planning, the neural network proposed by Spoor and Weber (2024) is presented in this section for the



tasks of production planning. The model is introduced in a generalized manner for applications in multiple use cases.

First, the conceptualization of the algorithm and the layout of the proposed neural network is described. The context to the research questions of production planning is derived and it is described how production planning data is converted into a connection matrix which acts as basis for the proposed algorithm. This set-up of the connection matrix is generalized so that different applications and use cases from production planning but also use cases from other domains can be analyzed using the proposed modified Hopfield nets. Next, the energy measurement for this specific network layout is defined. Afterwards, the training procedure of the network is derived in more detail. In addition, adjustments to the network layout for multi-dimensional ontologies and sequential data are presented.

Second, the anomaly detection model is introduced using the priorly defined layout of the network. This anomaly detection model is presented in a general manner so that use cases of other domains also apply to the proposed model. The anomaly detection model follows the proposed methodology by Spoor and Weber (2024). Concluding, a recommendation phase is discussed which enables users to adjust inputs which are classified as anomalous in order to correct them.

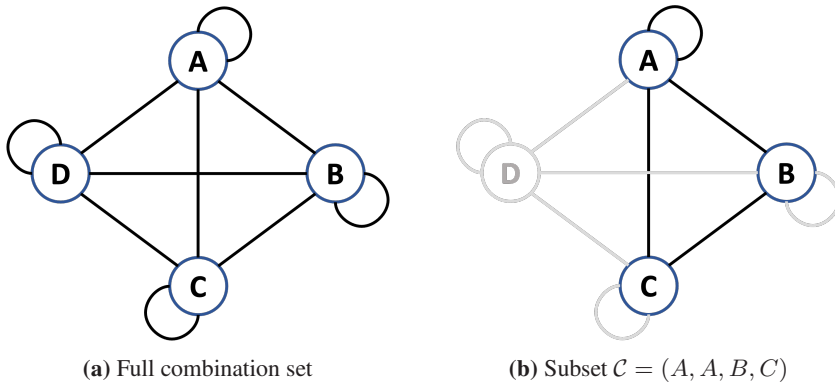
Third, building on the layout of the neural network and the anomaly detection model, a classification model is introduced. In addition to the classification capabilities, the model enables a similarity measure between different classes.

Fourth, the explainability capabilities of the anomaly detection as well as the classification model are discussed in more detail.

#### **4.4.1 Neural Network Layout**

The core idea of the proposed modified Hopfield neural network is that early planning data are displayed as combinations of objects described in an ontology. In the case of a problem complexity using a number of  $V$  possible objects which can be used when building a combination, all combinations of objects can be

modeled using a network of a number of  $V$  vertices and  $E$  edges representing the combinability. The set of objects of one specific combination is called  $\mathcal{C}$ . If two objects are active in a combination  $\mathcal{C}$ , the resulting graph must have an edge between the two vertices representing the objects and if the same object is applied twice in the combination  $\mathcal{C}$ , thus combined with itself, the graph has a loop in the corresponding vertex. Therefore, the combination is represented using an undirected simple graph permitting loops. The maximum network layout if all objects are active and also all loops are applied yields a number of  $E_{max} = \sum_{k=1}^V k = \frac{V(V+1)}{2}$  in the case of a full combination set including all objects and combinations of objects with themselves. All possible combinations are subsets of the full combination set. This network is schematically illustrated in Figure 4.5.



**Figure 4.5:** Schematic display of combinations as a network using  $V = 4$  objects and a maximum of  $E_{max} = 10$  edges. Unused edges and vertices are grayed out.

Using this network structure, it is also possible to derive the maximum number of possible unique combinations  $\mathcal{C}$  excluding impossible combinations, e.g., when the subsets  $\{A, B\}$  and  $\{A, C\}$  are within a combination set, the subset  $\{B, C\}$  is also included. Each object  $i$  can be within a combination once, twice (and thus a loop exists), or not even once. Thus, there are 3 distinct cases for objects within a combination. Each of the 3 cases of object  $i$  can be combined with the 3 cases

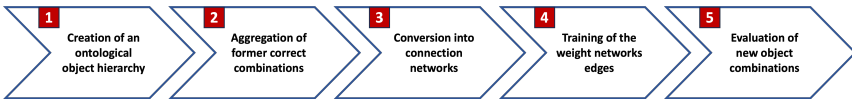
of another object  $j$  and thus, there are  $3 * 3 = 3^2$  distinct cases for both objects. Using a derivation of the rule by induction, it can be reasoned that for  $V$  objects, there are  $3^V$  unique combinations. However, to calculate the number of possible unique networks, this number of combinations must be reduced by  $V$  since in the cases of only one and no object within set  $\mathcal{C}$ , there are no active edges of the network. Thus, the number of unique networks when considering the edges is given as  $3^V - V$ . Since the number of networks grows exponentially, it is reasoned that rule-based approaches for individual combinations or specific combination constraints are not feasible for larger numbers of objects and a statistical approach must be considered.

The first objective of the proposed methodology is to assign a weight to each edge of the full combination network, resulting in a weight network which gives an estimation of the likelihood of a combination of objects. This weight network layout is similar to the full combination network since the vertices represent the objects and the edges represent the combinations. Therefore, the weight network has  $E_{max}$  edges and  $V$  vertices and the resulting weight network is given as an undirected multigraph with loops, also called pseudograph. This weight network should then be capable of computing an anomaly score of new combinations tested using the given weight network. This anomaly score is used as likelihood whether the tested combination is incorrect, novel, or generally speaking anomalous.

As a second objective, the proposed model should result in a recommendation on how to adjust the tested network to create a correct and valid combination. Additionally, the resulting anomalies scores and weight networks' recommendations must yield a good interpretability in order to discuss results with domain experts and evaluate why and how specific results are derived. The results by the model should be understandable for experts with mainly engineering expertise since a safe application of this methodology must include the considerations of the engineering domain. Production planners must understand the reasoning behind an anomaly or recommendation to alter the production planning appropriately. The methodology should also create awareness of errors or mistakes during planning

and highlight these problems within any production planning layout. To summarize, the model must have a high explainability in order to be enrolled in a practical application.

The proposed procedure is displayed in Figure 4.6. First, the objects are ordered and classified using an ontological hierarchy. The objects are analyzed along one specific level of the ontological hierarchy. Thus, also groups of objects can be analyzed as one object by selecting a higher hierarchy level. Second, the former correct combinations are aggregated as database for the training of the proposed modified Hopfield neural network. Third, all former correct combinations are converted into the described network structures. Fourth, using the network structures the edges of the weight network which represent the likelihood of combinations are trained. Fifth, new combinations of objects are evaluated using the weight network which was trained using the former correct object combinations. Thus, new object combinations can be classified as anomalous and errors within the tested combinations are derived. In addition, a recommendation is formed on how to adjust the object combination to create a less anomalous or even valid object combination.

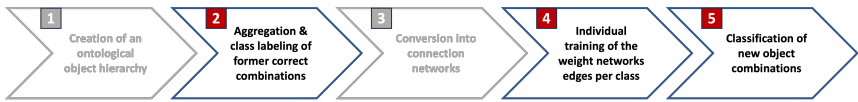


**Figure 4.6:** Sketch of the proposed procedure in five phases for the anomaly detection and evaluation of combinations using modified Hopfield nets.

In conclusion, the conceptualization and the resulting use cases can be generalized as the task of anomaly detection of unusual, wrong, and novel graph structures and networks using a set of true negative networks for the training of the model. By computing an anomaly score of a tested network, the likelihood whether an observed combination is correct can be evaluated with a high explainability.

A third objective is derived from the anomaly detection model for the specific case of different classes which utilize the same set of objects for combinations but are viewed as separated classes. Each scenario uses different typical combinations

so that each resulting weight network differs after an individual training for each scenario using only combinations from this specific scenario. Therefore, new combinations which are not yet grouped into a scenario are classified according to which scenario they most likely belong. Thus, this task can be generalized as the task of a classification of networks into different network classes which were priorly trained using known labeled networks. The adjusted proposed procedure for classification task is given in Figure 4.7.



**Figure 4.7:** Sketch of the proposed procedure in five phases for the classification of combinations using modified Hopfield nets.

The overall sequence and logic of the procedure of an anomaly detection task stays unchanged for the procedure of classification tasks. However, in the case of a classification task a labeling is added in phase 2 so that a weight network is computed for each class in phase 4. In phase 5, new objects are not evaluated for anomalous combinations but for the most likely class to which the analyzed object should be assigned to.

#### 4.4.1.1 Set-up of a Connection Matrix

For any combination set  $\mathcal{C}$  using a maximum number of  $V$  different objects, each combination is given by a  $V \times V$  connection matrix  $\mathbf{M}$ . This connection matrix shows if two vertices, which are the objects, are active together. If objects are combined with themselves, the connection between a vertex and itself is also analyzed. Active edges of this network represent the existence of a combination. Thus, an active edge shows whether the two corresponding vertices are both active and thus an edge is marked as active, if both vertices are currently active. This means in the application that both objects are currently within the applied

combination set. For each element  $m_{ij}$  of the connection matrix  $\mathbf{M}$  the following condition is evaluated:

$$m_{ij} = \begin{cases} +1, & \text{active edge} \\ -1, & \text{inactive edge} \end{cases} \quad (4.1)$$

The resulting connection matrix is symmetrical so that  $m_{ij} = m_{ji}$  since the order of objects is not analyzed in the connection condition.

Equation 4.1 is applied for Boolean values of the active vertices. However, if the activation is not given using Boolean values but in terms of intensity values of each vertex, the equation must be adjusted using an intensity threshold  $t_i$  per vertex for the intensity  $c_i$  of vertex  $i$ . To transform the intensity values into a Boolean activation, it can simply be checked if the intensity exceeds the predefined threshold. If both vertices of a combination exceed their respective threshold, the edge is active. The connection condition is adjusted as follows:

$$m_{ij} = \begin{cases} +1, & c_i \geq t_i \wedge c_j \geq t_j \\ -1, & c_i < t_i \vee c_j < t_j \end{cases} \quad (4.2)$$

Combinations of vertices using Boolean values and intensity can be computed by setting an edge as active if the vertex using an intensity exceeds a Boolean intensity threshold and the vertex using Boolean values is simultaneously active.

In some cases, the sum of both vertices' intensity might be relevant and high intensity values of one vertex might compensate lower intensities of the other vertex so that a connection is still applied despite one vertex yielding a lower intensity. In these cases, a combination is given by both intensities fulfilling a sum condition of  $b_{ij}c_i + c_j = t_{ij}$  and the connectivity condition is adjusted as follows:

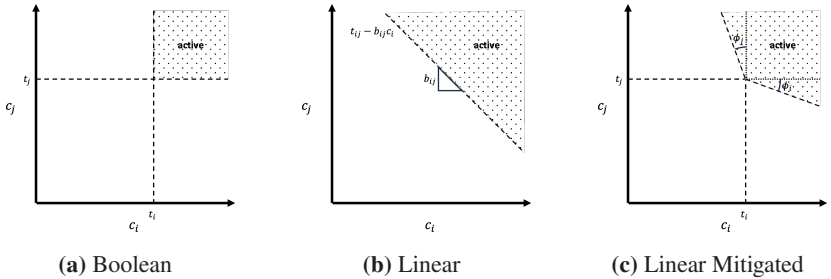
$$m_{ij} = \begin{cases} +1, & b_{ij}c_i + c_j \geq t_{ij} \\ -1, & b_{ij}c_i + c_j < t_{ij} \end{cases} \quad (4.3)$$

A disadvantage of a linear condition is that high intensity values of one vertex can result in active connections despite very low value for the corresponding other vertex. Therefore, the linear condition should be mitigated so that both vertices yield high intensities. This is achieved by adjusting the connection condition to apply a linear mitigated threshold using a slope  $\Phi_i$  for the threshold of vertex  $i$ . This slope can be translated into a slope of linear function using  $b_i = \frac{1}{\tan(\Phi_i)}$ . The condition for the connectivity is adjusted as follows:

$$m_{ij} = \begin{cases} +1, & c_i \geq t_i + b_i(t_j - c_j) \wedge c_j \geq t_j + b_j(t_i - c_i) \\ -1, & c_i < t_i + b_i(t_j - c_j) \vee c_j < t_j + b_j(t_i - c_i) \end{cases} \quad (4.4)$$

It should be noted that when using intensities, the combinations of vertices with themselves are either excluded or should be interpreted as simply high values of an object and not as a combination. The resulting connection matrices for intensity values are also symmetrical.

The proposed thresholds for intensity values are illustrated as a sketch in 4.8. It should be noted that the intensity values might use different scales or units and have a different order of magnitude. Thus, the parametrization should consider each vertex, i.e., the intensity of each object, carefully. The thresholds should be evaluated using domain experts and should have a technical significance.



**Figure 4.8:** Sketch of proposed intensity thresholds for combinations with objects using intensities.

It is assumed that in most applications a Boolean threshold is most useful due to lower parametrization efforts. However, the selection of the intensity threshold type should also consider the application and domain knowledge about the underlying objects.

#### 4.4.1.2 Energy Measurement

Each edge's combinability between vertex  $i$  and vertex  $j$  is represented by a weight  $w_{ij}$  which describes the strength of the connection. These weights are displayed as a  $V \times V$  weight matrix  $\mathbf{W}$ . The individual weights are within the range of  $-1$  and  $1$  such that following statement is applicable:

$$-1 < w_{ij} < +1 \quad \forall i, j \quad (4.5)$$

The weight matrix is for the analysis of combination symmetrical so that  $w_{ij} = w_{ji}$  is applied. A negative weight indicates a negative correlation of two vertices, i.e., both objects are more likely to be not combined. On the other hand, a positive weight shows that this combination of vertices is usually applied and both vertices are more likely to be active together.

To evaluate whether a combination is an anomaly or not, the network using the weight matrix  $\mathbf{W}$  gets stimulated using a specific connection matrix  $\mathbf{M}$  which represents the analyzed combination of objects. The energy of this stimulated state is measured using an adjusted energy measurement of Hopfield networks (Hopfield 1984) or of the Lenz-Ising model (Brush 1967), respectively. For this adjusted energy measurement, the edges are used for calculation instead of the vertices and thus loop requirements, i.e., combinations of vertices with themselves, are enabled. Thus, the applied requirement of  $w_{ii} = 0$  of the Hopfield model is circumvented. In addition, the energy level is corrected for higher and lower numbers of combinations using a correction value  $\theta$  and the discrete unit sample function  $\delta[n]$ , which is a special case of a two-dimensional Kronecker delta



function with  $\delta[n] \equiv \delta_{n0} \equiv \delta_{0n}$  and counts the active vertices. If the weight network is stimulated, the energy is measured as follows:

$$H_{stim} = - \sum_{i=0}^V \sum_{j=i}^V w_{ij} m_{ij} - \theta * \delta[m_{ij} - 1] \quad (4.6)$$

It should be noted that the energy is summed over just one half of the symmetrical weight and connection matrix including the complete diagonal. Due to the symmetry each combination is counted only once and thus the symmetrical counterparts are not additionally added in the energy calculation.

Since the correction value should balance the different number of active combinations, it is computed as the average effect on the system energy when any combination is active. Thus, a negative correction value indicates that within a normal combination only a small number of combinations are active. A positive correction value indicates that it is more likely for combinations to be active than inactive. The weight  $w_{ij}$  is used as an estimation of the probability if the connection, i.e., the edge, between vertex  $i$  and  $j$  is active. For the correction value applies therefore accordingly  $-1 < \theta < +1$  as in the case of the individual weights  $w_{ji}$  of the weight matrix  $\mathbf{W}$ .

$$\theta = \frac{2}{V(V+1)} \sum_{i=0}^V \sum_{j=i}^V w_{ij} \quad (4.7)$$

Regarding the correction value, it is assumed that in some cases the average weights do not sufficiently balance the frequency of active and inactive combinations. To take infrequently active combinations into consideration, the usage of a biased correction value is recommended for some applications. In general, a biased correction value is constructed by applying a weighted average of the connection matrix entries instead of the average as initially suggested. This weighted average is able to compensate an imbalanced number of sets of combinations in the training data set with highly frequently active combinations. The calculation of the bias

is adjusted using weightings  $a_{ij}$  when averaging the weight matrix entries. If the weightings are restricted by  $\sum_{i=0}^V \sum_{j=i}^V a_{ij} = 1$ , a general equation for a biased correction value is given as follows:

$$\theta_{biased} = \sum_{i=0}^V \sum_{j=i}^V a_{ij} w_{ij} \quad (4.8)$$

If diagonal entries, which are corresponding to loops and connections of objects to themselves, are more common in the connection matrix, Spoor and Weber (2024) use a biased correction value which weights the diagonal entries less by counting the non-diagonal entries twice. The application of this biased correction is useful, if diagonal entries have significantly higher weight values compared to other entries and thus are falsely computing active connections as more likely.

$$\theta_{biased} = \frac{1}{V^2} \sum_{i=0}^V \sum_{j=0}^V w_{ij} \quad (4.9)$$

To compare the stimulated energy, an equilibrium energy of the network without a specific connection matrix as an input is analyzed. This equilibrium is an equivalent to the Lenz-Ising model without any stimulation applied. The weight  $w_{ij}$  is again used as the estimation of the probability that a connection is active. Thus, no connection matrix  $\mathbf{M}$  is applied. The equilibrium energy is defined as follows:

$$H_{eq} = - \sum_{i=0}^V \sum_{j=i}^V w_{ij}^2 \quad (4.10)$$

The equilibrium is set-up in an equivalent manner to the Lenz-Ising model without applied external forces.

#### 4.4.1.3 Training Procedure

The objective of the training is that a weight matrix is created which indicates, when stimulated using a new combination, if this combination is memorized or unknown. This objective is achieved by minimizing the absolute difference between the energy in the equilibrium and the stimulated state in order to average out the impact of more unusual and more usual but overall correct combinations within a training data set of combinations.

$$\begin{aligned}
 \Delta H &= H_{stim} - H_{eq} \\
 &= - \sum_{i=0}^V \sum_{j=i}^V w_{ij} m_{ij} - w_{ij}^2 - \theta * \delta[m_{ij} - 1] \\
 &= - \sum_{i=0}^V \sum_{j=i}^V w_{ij} (m_{ij} - w_{ij}) - \theta * \delta[m_{ij} - 1]
 \end{aligned} \tag{4.11}$$

The network will be stimulated for each combination of objects within the training data set  $\mathcal{T}$  and an optimization function is set up to reduce the energy difference by adjusting the weights so that the simulated state's energy changes in direction to the equilibrium energy (minus the correction value). Since the objective of the learning procedure is that the difference  $\Delta H$  between stimulated energy and equilibrium energy becomes zero, a gradient descent  $d^t$  per iterative step  $t$  using a learning rate  $\alpha \in (0, 1)$  for the step size must be computed so that following update procedure of the weight applies:

$$w_{ij}^{t+1} = w_{ij}^t + \alpha * d^t \tag{4.12}$$

It is trivial that the energy difference  $\Delta H$  is optimized by the sum of the terms  $(m_{ij} - w_{ij})$  becoming zero. However, the optimization target can be formally

transformed into a minimization objective by using each terms' squared value since all terms are linearly separable.

$$\min f(\mathbf{w}) = \sum_{i=0}^V \sum_{j=i}^V (m_{ij} - w_{ij})^2 \quad (4.13)$$

This representation highlights a benefit of the optimization method: a minimization objective can be solved by a gradient descent method using  $d^t = -\nabla f(\mathbf{w}^t)$ . Each element of the resulting vector of length  $E$  is given as follows:

$$\frac{\partial}{\partial w_{ij}} f(\mathbf{w}) = -2 * (m_{ij} - w_{ij}) \quad (4.14)$$

The Hessian matrix is given as follows:

$$\nabla^2 f(\mathbf{w}) = \text{diag}(2, \dots, 2) \quad (4.15)$$

It is possible to set a gradient descent  $d^t$  per iterative step  $t$  for the optimization of  $f(\mathbf{w})$  so that the following weight update using a learning rate  $\alpha \in (0, 1)$  for the step size applies:

$$w_{ij}^{t+1} = w_{ij}^t + \alpha (m_{ij} - w_{ij}^t) \quad (4.16)$$

The initial weight matrix entries are given as  $w_{ij}^{t=0} = 0 \forall i, j$ . The scaling factor of 2 is shortened since the direction of the descent is independent of this scaling factor. Formally, the proposed iterative weight update process also follows the gradient descent of the optimization function  $f(\mathbf{w})$  using Newton's method with a descent of  $d^t = -(\nabla^2 f(\mathbf{w}^t))^{-1} \nabla f(\mathbf{w}^t)$ .

This applied learning rule also follows the optimization of a squared error  $e_{ij} = (m_{ij} - w_{ij})^2$  which is describing the difference between the desired value and the current values. Thus, the learning rule corresponds to the stochastic gradient descent of the method of least squares used in linear regression models. This is

also visible in the optimization function which is the sum of all squared errors  $f(\mathbf{w}) = \sum_{i=0}^V \sum_{j=i}^V e_{ij}$  and thus similar to the optimization function in linear regression models. Furthermore, the applied weight update is similar to the weight update rules for Autoencoder using the Widrow-Hoff rule (Abdi et al. 1996).

In order to not penalize inactive combinations, it is possible to only update weights which have active combinations in their corresponding row or column. Thus, the weight update is only applied for a weight between vertex  $i$  and  $j$  if the following condition is met:

$$\sum_{j=0}^V m_{ij} > -V \vee \sum_{i=0}^V m_{ij} > -V \quad (4.17)$$

It should be noted that if the outlined condition is applied, combinations including a single object once will not affect the training process. If this feature is required, the condition can be mitigated by only updating weights corresponding to combinations of two objects where at least one of the two objects occurs at least once in the underlying combination set.

In commonly applied Hopfield neural networks, the network's vertices are activated iteratively after another whether the summarized weights of the predecessor vertices exceed a defined activation threshold. In this proposed modification of the Hopfield neural network layout, for each applied network in the training data, the corresponding values of the edges, in other word the active combinations, are imprinted in the weight network. This approach distances the proposed modified Hopfield model from the conventional design of neurons in neural networks.

Furthermore, it is possible to add two commonly used regularization techniques as adjustments to the weight updates:

1. A reduction of the training rate  $\beta \in [0, 1]$  after each iteration for a number of training data  $T$  so that the learning rate of the weight updates is linearly reduced in later iterations.

2. A decay  $\gamma \in [0, 1)$  towards zero since in the case when only a limited amount of training data is available for a specific edge, these rare weight updates for this edge are not overestimated and slowly corrected.

It is quite common to use decay during the training of neural networks to prevent an overfitting. The usage of a reduced training rate is also common to enable a higher learning rate  $\alpha$  at the beginning of a training procedure and thus preventing the optimization function becoming trapped within a local optimum. After multiple iterations, the step size of the learning rate is then reduced to stabilize the solution. Thus, the reduced learning rate enables a sampling of the area around a possible optimum of weights by using smaller increments of weight changes. Both regularization techniques can be applied for the proposed Hopfield neural network using an adjusted weight update as follows:

$$w_{ij}^{t+1} = w_{ij}^t + \alpha \left( 1 - \beta \frac{t}{T} \right) (m_{ij} - w_{ij}^t) - \gamma * w_{ij}^t \quad (4.18)$$

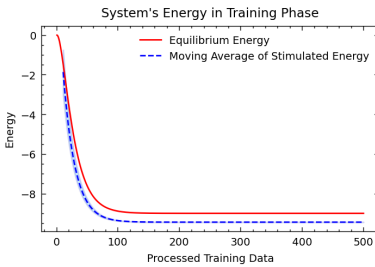
For the evaluation of a conducted training procedure, it should be noted that each individual weight correlates with the probability that this specific edge is active within a combination. Since only correct combinations are available during training, all combinations should arrange themselves closely around an average energy level, and thus the energy of a common or usual combination of objects should be within a certain range of the average energy of the training data. However, rare or unusual combinations should yield a higher energy in average than the average energy of the combinations in the training data.

During the training of the weight network, the equilibrium and average stimulated energy of correct states will converge towards a similar energy level minus the correction value. In case of just identical combinations used during training, the equilibrium energy converges towards the stimulated states of correct combinations minus the correction value times the number of active combinations. However, since the connection values  $m_{ij}$  are given as Boolean data of  $-1$  and  $1$ , the correct combinations will yield different energies as the equilibrium at the

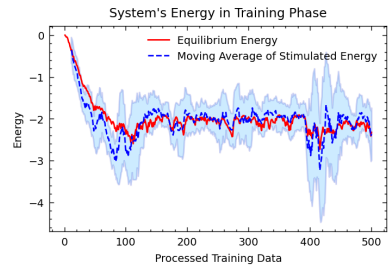
beginning of the training, even when the correction value is not considered, since it uses weights  $-1 < w_{ij} < 1$  smaller (or higher) than the connection value  $m_{ij}$ .

If multiple different combinations are used in the training data set, the weights  $w_{ij}$  will always be different compared to the correct combination's connection value  $m_{ij}$  and thus result in a higher equilibrium and average stimulated energy. Nevertheless, the equilibrium and average stimulated energy of the training data should also converge towards a constant energy level minus the correction value.

The training procedure is illustrated using the exemplary network set-up of the example in Figure 4.5. The network is trained using (a) only the exemplary combination set of  $\mathcal{C}_1 = (A, A, B, C)$  and (b) in addition a random order of the equally often occurring combinations sets of  $\mathcal{C}_1$ ,  $\mathcal{C}_2 = (A, C, C)$ ,  $\mathcal{C}_3 = (A, A, C, D)$ , and  $\mathcal{C}_4 = (B, C)$ . The results of the training procedure are given in Figure 4.9. The equilibrium energy, the moving average stimulated energy over 10 iterations, and the moving average's interval within 1-standard deviation are displayed in the evaluation of the exemplary network in Figure 4.5.



(a) One combination



(b) Four combinations in random order

**Figure 4.9:** Training procedure of two exemplary cases using (a) only one identical combination and (b) four equally often but randomly arranged combinations.

As clearly visible in Figure 4.9, in the case of only one combination the equilibrium and average stimulated energy converge towards a stable energy level which is identical if the correction value is deducted. Since the correction value is negative, the stimulated energy is slightly below the equilibrium energy. The already very

low standard deviation of the moving average also declines during the training since the weight matrix also converges towards a final weight matrix. The entries of this final weight matrix reach values through a high amount of training data which come close to the input matrix of the one combination set. There is only the exception of matrix entries where no rows or columns were affected during the training so that these values stayed at zero.

The second case in Figure 4.9 shows a higher equilibrium and average stimulated energy level since multiple different combinations are applied. Since the combination sets also highly differ in their combination setups and number of active edges, the standard deviation is higher. However, the equilibrium and average stimulated energy converge towards an energy level which is higher than the energy level in case of only one trained combination. The correction value in this case is also slightly negative but since the stimulated energy levels are fluctuating, the difference is not visible.

The analysis of the energy level progress is very important to evaluate the quality of the training. First, it can be checked if the energy levels converge towards a stable energy level. If this is not the case, more training data are required. Second, the standard deviation and fluctuations give a good first indication of the necessary thresholds which are used to detect anomalies: the average stimulated energy level plus the standard deviation can often be used as a rough estimate. Third, if the fluctuations of the average energy levels are too high, there might be a hidden variable involved which separates certain types of combinations in distinct classes. In these cases, the training data should be checked, and the different classes should be differentiated prior to the analysis since the applied combinations during training should all be applied within a distinct scenario.

For calculating the computational complexity, a total of  $T$  iterations per available combination in the training data set is considered. Per iterative step, a weight update of a  $V \times V$  connection matrix is calculated. Considering the symmetrical properties of the connection and weight matrix, a total of  $\frac{V(V+1)}{2}$  updates are necessary per iterative step  $t$ . Thus, the highest order of the variables for computation of the complexity are  $V^2$  and  $T$  and the computational complexity of the



training procedure is given as  $\mathcal{O}(TV^2)$ . Hence, the complexity of the training of the proposed modified Hopfield neural network mostly depends on the number of possible objects considered when setting up combinations. Therefore, higher ontological hierarchy levels which are grouping multiple objects into one cluster might be preferable since these will reduce the number of vertices  $V$  from the number of objects to the smaller number of clusters. In addition, if a new object must be considered, the model stays unchanged as long as the new object can be grouped within an already existing and analyzed cluster.

In order to decrease the necessary computational time of the training, the equation (4.18) of the proposed Hopfield neural network can also be adjusted using a mini-batch procedure as described in equation (3.11) in the case of an ANN.

#### 4.4.1.4 Adjustments for Sequential Data

A common application within production planning is the sequence planning of processes and finding a sequence which is optimizing KPIs such as costs, cycle times, waste, and produced units. The introduced methodology is not able to create such sequences but is capable of checking existing or proposed sequences for correctness. Thus, the use case shifts from an analysis of combinations to an evaluation of sequences whether they are correct or incorrect.

In the case of sequences, the combinatorial information over the whole set-up is less important and the focus is on which objects are predecessor or successors of other objects. Therefore, the connection matrix must be constructed using this data as input. The connection matrix is adjusted as follows:

$$m_{ij} = \begin{cases} +1, & \text{object } i \text{ is successors of object } j \\ -1, & \text{else} \end{cases} \quad (4.19)$$

An object can be its own successors and diagonal entries of the connection matrix are then set as  $m_{ii} = 1$ . This is the case if, e.g., the same process is conducted consecutively within a sequence planning of a production task.

Furthermore, it is possible that multiple different objects are successors of the same objects and *vice versa*, it is possible that a single object is successor of multiple different objects. The connection condition applies unchanged and the connection matrix entry  $m_{ij}$  is set as +1 for each successor object of an object  $j$  or for a number of multiple objects.

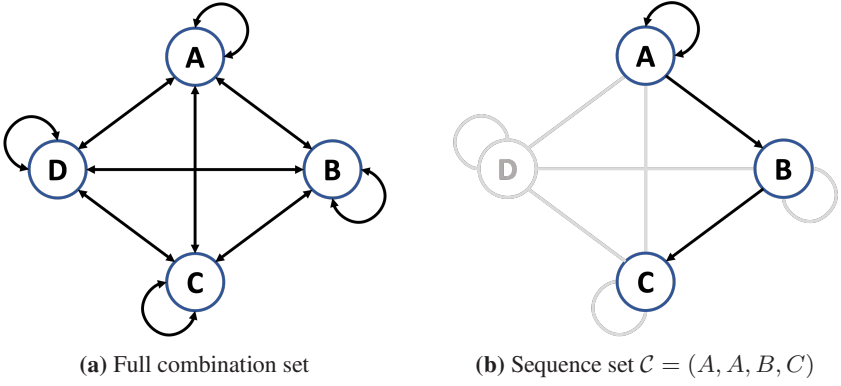
Instead of analyzing the successors sequences, this connection condition can be adjusted by swapping the indices which represent the objects in order to analyze the predecessor sequences. If the successors sequences result in a connection matrix  $\mathbf{M}$ , the predecessor sequences will result in the transpose of the connection matrix  $\mathbf{M}^T$ . Thus, the selection of a successors sequences the predecessor sequences condition does not affect the overall set-up of the neural network.

By using sequence data, the connection matrix is no longer symmetrical and can yield different entries for the values  $m_{ij}$  and  $m_{ji}$ . From this follows that the weight matrix  $\mathbf{W}$  is also no longer symmetrical and the individual weight matrix entries  $w_{ij}$  and  $w_{ji}$  might be different. Furthermore, the maximum number of edges of the networks representing the sequences is given as  $E_{max} = V^2$ .

In the case of sequential data, the network layout illustrated in Figure 4.5 can be adjusted to a directed multigraph permitting loops. Edges are now directed and thus, twice the number of edges which are not loops are required (one for each direction) but the same number of loops are applied. As a result, the connection and weight matrix are no longer symmetrical and they yield information about the sequence's order. The new visualization of the network layout for analyzing sequential data is given in Figure 4.10.

As a result, the stimulated energy computation as per equation (4.6) is also affected since all entries of a matrix must be summarized.

$$H_{stim} = - \sum_{i=0}^V \sum_{j=0}^V w_{ij} m_{ij} - \theta * \delta[m_{ij} - 1] \quad (4.20)$$



**Figure 4.10:** Schematic display of sequences as a network using  $V = 4$  objects and a maximum of  $E_{max} = 16$  directed edges. Unused edges and vertices are grayed out.

The same adjustment is necessary for the equilibrium energy of equation (4.10).

$$H_{eq} = - \sum_{i=0}^V \sum_{j=0}^V w_{ij}^2 \quad (4.21)$$

Lastly, the correction value in equation (4.7) must also be adjusted and becomes identical to the former biased correction value as used by Spoor and Weber (2024) of equation (4.9).

$$\theta = \frac{1}{V^2} \sum_{i=0}^V \sum_{j=0}^V w_{ij} \quad (4.22)$$

The regular biased correction values of equation (4.8) can still be calculated analogously using the changed weight condition of  $\sum_{i=0}^V \sum_{j=0}^V a_{ij} = 1$ .

The overall training procedure stays the same with the only difference being that the weight updates of the weights  $w_{ij}$  and  $w_{ji}$  are different. Thus, sequential data can be analyzed in the same manner as combinatorial data. Using the common conventions for the notation of computational complexity, the computational complexity does not change and is still measured as  $\mathcal{O}(TV^2)$ . However, the actual runtime in a practical application is slightly higher since the full matrix

of weights using  $V^2$  entries must be considered and not, as in the case without sequences, the reduced matrix with  $\frac{V(V+1)}{2}$  entries due to the connection and weight matrices' symmetry.

It should be noted that in certain use cases of production planning, subsequences might be conducted multiple times, e.g., process  $A$  in the sequence in the example in Figure 4.11, but the resulting graph structures do not differentiate sequences with multiple repetitions of subsequences. Thus, the sequence  $\mathcal{C} = (A, A, B, D, D)$  and  $\mathcal{C} = (A, A, A, B, D, D)$  do result in the same network. This might limit the applicability of the model in certain use cases, if repetitions of subsequences are a characteristic of anomalous process sequences. However, this limitation applies to all analyses and anomaly detection methods using only the graph structure. Therefore, a separate anomaly detection model to find conspicuous repetitions of subsequences can be used in combination with a modified Hopfield neural network in order to fully validate the sequence data.

The model for anomaly detection and classification described in the following sections can also be applied for sequential data. It will be noted whether adjustments for sequential data are necessary.

#### 4.4.1.5 Adjustments for Ontologies with multiple Concepts

Within an application in production planning, there might be multiple different ontological concepts and hierarchies of sub-concepts within a concept applied for a specific use case. Thus, objects might come from different concepts within an ontology, e.g., if the PPR model is applied, objects might be products, processes, or resources and a planning set-up includes objects from all three concepts. However, the model also applies for application where multiple different ontological concepts are used in an analyzed combination. This is discussed for a general case using  $G$  distinct ontological concepts.

In the case that objects belong to one of a number  $G$  different ontological concepts, the total number of possible used objects within any combination is given as  $V_i$  for the objects within concept  $g$ . These numbers might differ. Thus, the total number

of possible objects included within a combination is given as  $V = \sum_{i=1}^G V_i$ . The connection matrix  $\mathbf{M}$  is then still a  $V \times V$  matrix. However, the connection matrix consists of multiple matrices which are of interest.

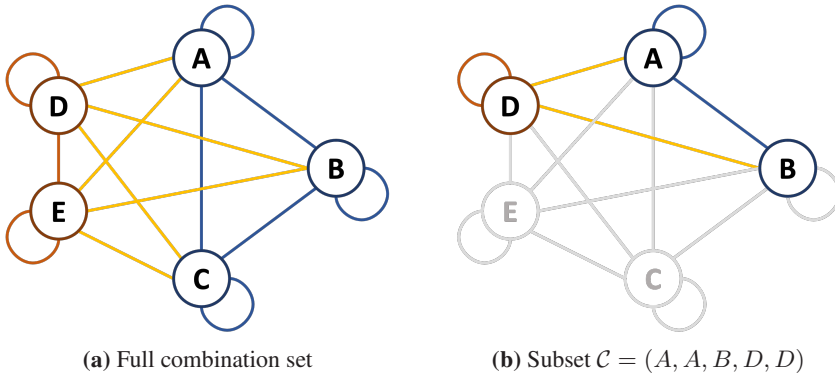
$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_{11} & \dots & \mathbf{M}_{1i} & \dots & \mathbf{M}_{1G} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{M}_{i1} & \dots & \mathbf{M}_{ii} & \dots & \mathbf{M}_{iG} \\ \dots & \dots & \dots & \dots & \dots \\ \mathbf{M}_{G1} & \dots & \mathbf{M}_{Gi} & \dots & \mathbf{M}_{GG} \end{pmatrix} \quad (4.23)$$

The connection matrix  $\mathbf{M}$  contains the  $V_i \times V_i$  matrices  $\mathbf{M}_{ii}$  which are the combinations of objects with objects from the same ontological concept. In addition, the connection matrix contains the  $V_i \times V_j$  matrices  $\mathbf{M}_{ij}$  which are the combinations of objects from an ontological concept  $i$  with the objects from an ontological concept  $j$ . Within production planning, these might be the combinations of, e.g., products, processes, and resources in a case of  $G = 3$  and the matrices  $\mathbf{M}_{ij}$  describe how these different ontological concepts are combined within a single applied combination.

The resulting connection matrix  $\mathbf{M}$  is still symmetrical. For each combination matrix applies  $\mathbf{M}_{ij} = \mathbf{M}_{ij}^T$  since they share the same combinatorial data. In addition, the matrices  $\mathbf{M}_{ii}$  are also symmetrical since the same conditions as discussed in the previous sections also apply. The number of unique edges for a connection matrix  $\mathbf{M}_{ii}$  is given as  $E_{ii_{max}} = \frac{V_i(V_i+1)}{2}$  and the number of unique edges of a connection matrix  $\mathbf{M}_{ij}$  are given as  $E_{ij_{max}} = V_i * V_j$ . The total number of unique edges can still be computed as formerly described and equals the sum of all edges of the unique matrices.

In the case of planning scenarios using multiple concepts from an ontology, the network layout visualization in Figure 4.5 can be improved by highlighting the different matrices  $\mathbf{M}_{ij}$  and objects from the different ontological concepts. Thus, an updated exemplary visualization of the network layout for analyzing sequential data is given in Figure 4.11 using color codes to differentiate objects from different

concepts and the different connection matrices  $\mathbf{M}_{ij}$  building the whole connection matrix  $\mathbf{M}$ .



**Figure 4.11:** Schematic display of ontologies with multiple concepts as a network. The resulting sub-connection matrices are differently colored, unused edges or vertices are grayed out.

In the exemplary case in Figure 4.11, two ontological concepts are analyzed. The first concept contains object  $A$ ,  $B$ , and  $C$  colored in blue, and the second contains object  $D$ ,  $E$  colored in red. Thus, the number of objects per concept is given as  $V_1 = 3$  and  $V_2 = 2$  and the connection matrix  $\mathbf{M}$  includes in total  $V = V_1 + V_2 = 5$  objects. The edges corresponding to the connection matrices  $\mathbf{M}_{11}$  using a maximum of  $E_{11_{max}} = 6$  edges and  $\mathbf{M}_{22}$  using a maximum of  $E_{22_{max}} = 3$  edges are colored using gradients of their ontological concepts. The edges corresponding to the connection matrix  $\mathbf{M}_{12}$  using a maximum of  $E_{12_{max}} = 6$  edges are colored in yellow. Thus, the total number of maximum edges is given as  $E_{max} = 15$ .

Since the problem using multiple ontological concepts can be transferred into the same set-up as already discussed, the layout of the neural network and the training procedure does not change. Therefore, different concepts from an ontology and objects from these concepts can be analyzed using the introduced network layout. However, since the model's computational complexity scales over-proportionally

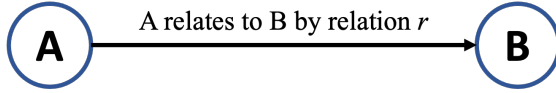
with the number of vertices  $V$ , increasing the number of used concepts might negatively affect the runtime of the training procedure. Due to the over-proportional computational complexity increase, it might be beneficial in practical application to not divide concepts into too many fragmented sub-concepts. It is rather beneficial to use higher-ranking concepts which combine objects.

On the other hand, it is possible to individually train the weight matrices  $\mathbf{W}_{ij}$  corresponding to the connection matrices  $\mathbf{M}_{ij}$  of two concepts and only later combine them into a model covering all ontological concepts. Additionally, it is possible to add a new ontological concept to an already trained model by only training the affected matrices without the need for re-training the other matrices. *Vice versa*, it is possible to use only a specific combination of ontological concepts from a trained weight matrix  $\mathbf{W}$  containing multiple concepts. However, this requires that the condition of equation (4.17) is not used or mitigated as described in the corresponding paragraph.

If there are specific types of ontological product planning structures which should be analyzed separately and which utilize distinct applied concepts and combinations, the task becomes an anomaly detection where anomalous combinations do not belong to any class out of a number of different classes. This case is discussed in the following section about the classification model.

#### 4.4.1.6 Adjustments for Ontologies with multiple Relations

In some use cases, two objects might not only relate to each other by one single relation described by one edge but by multiple directed logical relations. In these cases, an edge describes a certain relation  $r$  from a finite set containing all possible relations between two objects  $\mathcal{R}$ . The number of all possible relations between two objects is given by  $R$ . Therefore, it is important to model the kind of relation between two objects. It is possible that two objects yield multiple, one, or no relations between each other. A sketch of an exemplary annotated directed edge for a relation  $r$  is given in Figure 4.12.



**Figure 4.12:** Sketch of an annotated edge using a certain logical relation  $r \in \mathcal{R}$  between two vertices representing the objects.

Analogously to the network layout in Figure 4.11, each data set is represented by a connection network which is a subset of the fully connected network of all vertices using all relations. However, since the type of relation adds more information to the network, the connection is not modeled by a connection matrix but a connection tensor  $\mathbf{M}^{(3)} = V \times V \times R$  in order to represent the full complexity of the relations. This connection tensor includes the connections between all objects for each relation  $r$ . Thus, a connection is given as follows:

$$m_{ijr} = \begin{cases} +1, & \text{object } i \text{ relates to object } j \text{ by } r \\ -1, & \text{else} \end{cases} \quad (4.24)$$

In the case of multiple possible relations between objects, the weight matrix therefore becomes analogously a weight tensor  $\mathbf{W}^{(3)} = V \times V \times R$ . Each weight entry  $w_{ijr}$  of the weight tensor describes the weight between two objects  $i$  and  $j$  for a specific relation  $r$ . In case of directed logical relations, the resulting connection tensor is not symmetrical. In the case of undirected logical relations, the tensor will yield a symmetry within the  $V \times V$  matrix of a specific relation. However, the directed case is more common and thus, the following equations are based on the equations of the prior adjustment for sequential data.

Since the new model must also consider the different relations, the stimulated energy computation as per equation (4.21) summarizes over all relations in addition. The resulting stimulated energy is given as follows:

$$H_{stim} = - \sum_{r=0}^R \sum_{i=0}^V \sum_{j=0}^V w_{ijr} m_{ijr} - \theta * \delta[m_{ijr} - 1] \quad (4.25)$$



The equilibrium energy of equation (4.22) also changes accordingly to the new energy definition which includes multiple relations.

$$H_{eq} = - \sum_{r=0}^R \sum_{i=0}^V \sum_{j=0}^V w_{ijr}^2 \quad (4.26)$$

Concluding, the correction value in equation (4.23) is adjusted.

$$\theta = \frac{1}{R * V^2} \sum_{r=0}^R \sum_{i=0}^V \sum_{j=0}^V w_{ijr} \quad (4.27)$$

In the case of logically undirected relations, the equations (4.6), (4.8), (4.10) are accordingly adjusted by adding a summarization over all relations.

Since the number of connections and weights is increased due to the dimensionality of the tensor by the number of relations  $R$ , the complexity of the training procedure is also increased by the number of modeled relation and thus, the new computational complexity is given as  $\mathcal{O}(RTV^2)$ . The complexity scales linearly with the number of different relations, which can enable an efficient computation of use cases utilizing ontologies with multiple modeled relations. In addition, a model can be trained for an individual relation and only later combined with other models using different relations but the same ontological concepts. Thus, an existing model can be enhanced without re-training if an additional relation is added. *Vice versa*, a model consisting of multiple relations can also be evaluated only for a subset of the included relations.

The following sections will use modified Hopfield neural networks with only a single relation and the case of annotated edges is not discussed in further detail. However, it is possible to adjust the described models for anomaly detection and classification by considering the summarization over all relations and defining a suitable tensor norm in order to measure distances between weight tensors.

## 4.4.2 Anomaly Detection Model

The presented anomaly detection model follows the approach proposed by Spoor and Weber (2024) and is more generalized in the following section to enable a broader field of use cases. First, the anomaly detection procedure is described. Second, a recommendation procedure is introduced which enables user to adjusted anomalous combinations to correct them.

### 4.4.2.1 Anomaly Detection

The core idea behind the anomaly detection model is that stimulated energies of common combinations are lower than stimulated energies of more uncommon or rare combinations. Thus, the probability that a combination is anomalous is given by the height of the computed stimulated energy. High energies are more anomalous than low energies. If multiple combinations are analyzed by stimulating the modified Hopfield network, the combinations with lower energies are more normal or usual occurrences of combinations while the combinations with higher stimulated energies are more rare, uncommon, or anomalous. The specific energy level depends on the weight matrix and thus, the energies must be evaluated in comparison to the given weight matrix and the set of combinations. For a given combination  $\mathcal{C}$ , a connection matrix  $\mathbf{M}$  can be computed. This connection matrix is used as an input to the stimulated energy function as described in the previous section. Thus, for a given combination which is tested whether it is anomalous or not, the stimulated energy  $H(\mathcal{C})$  is computed. This energy acts as a benchmark on how likely a combination is anomalous.

As discussed in the training procedure, it is useful to compare this stimulated energy with the average stimulated energy of the known correct combinations used during training. The used training set  $\mathcal{T}$  has a number of  $\|\mathcal{T}\| = T$  combinations

used for the conducted weight updates. Therefore, the average stimulated energy of this training data is computed as follows:

$$\bar{H} = \frac{1}{T} \sum_{l \in \mathcal{T}} H(\mathcal{C}_l) \quad (4.28)$$

Since the energy fluctuates between the correct combinations, it is useful to calculate the standard deviation  $\sigma$  of the stimulated energy of correct combinations.

$$\sigma = \sqrt{\frac{1}{T} \sum_{l \in \mathcal{T}} (H(\mathcal{C}_l) - \bar{H})^2} \quad (4.29)$$

Using the average stimulated energy and the standard deviation, it is possible to define a critical energy  $H_{crit}$ . Combinations with stimulated energies above this critical energy are then considered anomalous. Thus, the critical energy should be selected so that  $H_{crit} > \bar{H} + c * \sigma$  given a selected confidence interval factor  $c$ . It should be noted that only a one-sided evaluation of this confidence interval is necessary since low energy states are assumed to be correct or common.

For any selected critical energy, the assignment of a combination  $m$  is given by  $\hat{z}_m \in \{0, 1\}$  where 0 denoted outliers and 1 denotes common or correct combinations. The assignment is computed as follows:

$$\hat{z}_m = \begin{cases} 1, & H(\mathcal{C}_m) < H_{crit} \\ 0, & H(\mathcal{C}_m) \geq H_{crit} \end{cases} \quad (4.30)$$

By adjusting the critical energy value  $H_{crit}$  it is possible to detect anomalies using a different true positive rate and corresponding false positive rate. Thus, a receiver operating characteristic (ROC) can be computed. Using the ROC, a critical energy value  $H_{crit}$  can be selected which optimizes true and false positive rates for any given use case.

To conduct a test for a single combination, the stimulated energy must be computed using the entries of the weight matrix. Therefore, each computation of an energy must summarize over the  $\frac{V(V+1)}{2}$  weight entries. This results in a computational complexity of  $\mathcal{O}(V^2)$  for testing whether a combination is anomalous or not.

It should be noted that a sufficient anomaly detection can only be conducted if the combinations within the training data show a sufficient similar behavior or combinatorial patterns. If the combinations within the training data highly vary, it might be impossible to detect incorrect combinations since the network cannot detect any combinatorial patterns of common combinations of objects or objects which cannot be combined. This case should be visible by a high standard deviation of the stimulated energy and a high fluctuation of the moving average stimulated energy during the training procedure.

If there exist multiple types of combinations whose stimulated energies are within the same type less fluctuating, these combinations can be assigned to a specific class. Combinatorial patterns might emerge within the separated classes but not within the whole data set. In this case, the method should be adjusted to a classification task with an additional outlier class. This method is described in detail in the following section about the classification model.

#### 4.4.2.2 Recommendation Procedure

It is possible to directly evaluate why a certain combination  $\mathcal{C}$  results in a specific stimulated energy evaluation  $H(\mathcal{C})$  by directly analyzing the impact of an object  $i$  within the combinational set  $\mathcal{C}$ . If an object is within the analyzed combination set so that  $i \in \mathcal{C}$ , the impact on the energy evaluation is measured as follows:

$$I_i = \sum_{j=0}^V w_{ij} + \left( \sum_{j=0}^V w_{ij} m_{ij} - \theta * \delta[m_{ij} - 1] \right) \quad (4.31)$$

The first term of the equation describes the energy as if all combinations of objects with object  $i$  were given as  $m_{ij} = -1$  and the second term describes the

resulting energy of the current connection applied in the connection matrix. Both terms are subtracted so that the difference of the energy between a combination  $\mathcal{C}$  which includes object  $i$  and a combination  $\mathcal{C}^*$  without object  $i$  is measured. If  $I_i > 0$  applies, the energy is increased and the combination is less common due to the impact of object  $i$  within the combination  $\mathcal{C}$ . Thus, this specific combination is evaluated as less common or potentially incorrect within the analyzed combinations. *Vice versa*, if  $I_i < 0$  applies for the impact, the energy is decreased and object  $i$  is a reason why the combination is more common. The higher the impact, the higher the effect of this specific object within the combination on its evaluated correctness or incorrectness.

If only the combination of an object with itself within a combination set (a loop in the combination network) is analyzed but not the overall impact of the object on the combinations with other objects, the impact changes to:

$$I_i = 2 * w_{ii} - \theta \quad (4.32)$$

This analysis is equivalent to reducing the number of object  $i$  within the combination set  $\mathcal{C}$  to one. The impact is essentially the change of the energy by the weight  $w_{ii}$ . As before, a positive impact increases the energy indicating a less likely occurrence of this loop and a negative impact reduces the energy indicating that this loop is common within the analyzed combinations.

To summarize, the impact can be used to evaluate if an object  $i$  or the multiple occurrences of object  $i$  is seen as more correct or incorrect for any given combination.

In addition, an anomalous combination can be evaluated on how it should be changed, by adding or removing objects from the set. The idea is to iteratively add or remove objects from the analyzed sets of objects within the combination and to reduce the stimulated energy. For each object of the set of all of objects, it is computed whether adding or removing it from the set  $\mathcal{C}$  is reducing the energy. The object with the largest energy reduction is then selected and added or removed from the set  $\mathcal{C}$ . Thus, a more common process combination  $\mathcal{C}$  is

created using the trained weight matrix as baseline for evaluating combinations as correct or incorrect. Objects are added or removed iteratively until the stimulated energy of the adjusted combination has decreased below the selected critical energy level  $H_{crit}$  and thus, is assumed to be a correct combination. This results in a recommendation procedure on how to correct an as incorrect classified combination. The conducted iterative procedure is similar to the state changes of commonly used Hopfield neural networks. Concluding, two cases are evaluated for all  $V$  objects per iteration when conducting a recommendation procedure.

1. Removing one object  $i$  from the set  $\mathcal{C}^* = \mathcal{C} - \{i\}$
2. Adding one object  $i$  to the set  $\mathcal{C}^* = \mathcal{C} + \{i\}$

It should be noted that in case that two object  $i$  are within the set  $\mathcal{C}$ , removing one object  $i$  only deletes the combination with itself and thus only the loop connection but does not affect all other connections. Also, if an object is currently in the set of combination only once, adding another same object creates a loop connection and thus not affect the other connections.

In both cases, a new connection matrix  $\mathbf{M}^*$  is created. However, in case of only removing an object, the impact computed by the equations (4.31) and (4.32) can be used and no new connection matrix must necessarily be created. In general, the energy difference can be evaluated as follows:

$$I_i = H(\mathcal{C}^*) - H(\mathcal{C}) \quad (4.33)$$

The object with the highest negative stimulated energy impact is selected and removed or added. This recommendation procedure is iteratively reevaluated using the optimized set  $\mathcal{C}^*$  as new starting point until the stimulated energy of the new set  $\mathcal{C}^*$  falls below the critical energy. The progress of the set  $\mathcal{C}^*$  should be tracked to later evaluate the steps and thus, this creates a recommendation procedure where combinations are iteratively corrected.

### 4.4.3 Classification Model

Building on the anomaly detection model as proposed by Spoor and Weber (2024), the Hopfield neural network also enables a classification model. In the field of production planning, new combinations of ontological planning data can be assigned to known types of planning set-ups. Nevertheless, the proposed classification model has use cases in multiple domains and is thus introduced in a generalized manner and in later section the applications for production planning are derived.

First, the classification procedure is introduced which utilizes the already introduced modified Hopfield neural network layout and training procedure. Second, a distance measure is derived to compare different classes.

#### 4.4.3.1 Classification

In the cases that a set of classes  $\mathcal{K}$  exists where each class is distinct from each other, it is possible to classify a combination  $\mathcal{C}_l$  and assign a class to this specific combination.

For each class  $k$ , a weight matrix is trained using a training data set  $\mathcal{T}_k$  with a number of included combinations of  $\|\mathcal{T}_k\| = T_k$ . This training set only contains combinations which are known to be assigned to this specific class. The weight matrix resulting from the training procedure is given as  $\mathbf{W}_k$  and the corresponding correction value computed from the weight matrix as  $\theta_k$ . Each class has an individual weight matrix and correction value. The training follows the procedure described before in the previous sections. It should be noted that each training procedure for each analyzed class should converge during training towards a stable energy level.

Using the weight matrix, the stimulated energy of each combination of the training data set is evaluated. For each combination  $l$  which belongs to the training data set

of combinations of the class  $k$ , the stimulated energy is calculated and an average energy is computed as follows:

$$\bar{H}_k = \frac{1}{T_k} \sum_{l \in \mathcal{T}_k} H_k(\mathcal{C}_l) \quad (4.34)$$

In addition, the standard deviation of the stimulated energy of all combinations within the training set is computed as follows:

$$\sigma_k = \sqrt{\frac{1}{T_k} \sum_{l \in \mathcal{T}_k} (H_k(\mathcal{C}_l) - \bar{H}_k)^2} \quad (4.35)$$

For any combination  $m$  with no prior known class, it is possible to calculate a score on how likely this combination is assigned to a specific class  $k$  by comparing the difference between the stimulated energy of combination  $m$  using the weight matrix of class  $k$  with the average stimulated energy  $\bar{H}_k$  of combinations from this class. Thus, a score  $s_{mk}$  for the likelihood of the assignment of combination  $m$  to class  $k$  is given as follows:

$$s_{mk} = \frac{H_k(\mathcal{C}_m) - \bar{H}_k}{\sigma_k} \quad (4.36)$$

The lower score  $s_{mk}$ , the higher is the probability of a combination  $m$  belonging to class  $k$ . In case of multiple classes, the class with the lowest score is selected as assigned class. A class assignment  $\hat{z}_m$  is computed as follows:

$$\hat{z}_m = \arg \min_{k \in \mathcal{K}} s_{mk} = \{k \mid s_{mk} = \min_{k \in \mathcal{K}} \frac{H_k(\mathcal{C}_m) - \bar{H}_k}{\sigma_k}\} \quad (4.37)$$

If multiple combinations are tested, this class assignment results in a partition of all combinations  $\mathcal{D}$  using the following partition definition:

$$\bigcup_{k \in \mathcal{K}} \mathcal{K}_k = \mathcal{D}, \quad \mathcal{K}_k \cap \mathcal{K}_l = \emptyset \quad \forall k, l \in \mathcal{K} \quad (4.38)$$



In the case that a classification as well as an outlier class is required, combinations with scores higher than a critical value  $s_{crit}$  can be assigned to the outlier class. Thus, anomalous combinations can be detected which do not match with any of the available classes. In addition, the problems of high fluctuating stimulated energies when combining multiple classes in one data set during an anomaly detection procedure is prevented. The resulting classification is due to the outlier set no longer a partition and enables a conduction of the anomaly detection task in the case of combinations from multiple different classes. The class assignment using outliers is computed as follows:

$$\hat{z}_m = \{k \mid s_{mk} = \min_{k \in \mathcal{K}} \frac{H_k(\mathcal{C}_m) - \bar{H}_k}{\sigma_k} \wedge s_{mk} < s_{crit}\} \quad (4.39)$$

The classification procedure also works if a combination is assigned to multiple classes. In this case, a labeled combination is included in all training data sets of the classes to which the combination was assigned. Instead of selecting the minimum score for a class assignment, the procedure is adjusted using a critical value similar to the anomaly detection procedure. If the calculated score is smaller for a tested class than the predefined critical value, this class is assigned. The adjusted class assignment is given as follows:

$$\hat{z}_m = \{k \mid s_{mk} < s_{crit}\} \quad (4.40)$$

This method of class assignment will also result in an outlier class which cannot be assigned to any of the classes in set  $\mathcal{K}$ . If no outlier class is requested, all outliers can be assigned to the class with the smallest score  $s_{mk}$  as given in the partitioning single class assignment procedure.

Since the training procedure must be conducted for each class, the computational complexity is increased. For a number of  $K$  different classes using each a number of  $T$  training data per class, the computational complexity derived in the previous section is adjusted to  $\mathcal{O}(KTV^2)$ . Main complexity driver is still the number  $V$  of possible objects within combinations but the training complexity

additionally scales linearly with the number of classes. The classification of a single combination is given by the number of classes and the scale of the weight matrix to compute the energy. Thus, a classification yields a computational complexity of  $\mathcal{O}(KV^2)$  which is again similar to the complexity of anomaly detection times the number of classes which are separately evaluated.

It should be noted that a sensitive class assignment is only possible if the corresponding classes are sufficiently distinct. Thus, classes should significantly differ regarding the common combinations. Therefore, it is important to priorly test the classes regarding their similarity.

#### 4.4.3.2 Distance Measure between Classes

To compare different classes of combinations and evaluate similarity or dissimilarity between classes, the trained weight matrices can be used. Each class  $k$  yields a specific weight matrix  $\mathbf{W}_k$  which was computed by training only with combination or sequence data from this class. The difference between two classes  $k$  and  $l$  can be compared by the individual entries of the weight matrices and squaring the results. Thus, the delta weight matrix  $\Delta\mathbf{W}_{kl}$  between class  $k$  and  $l$  using the Hadamard product is computed as follows:

$$\Delta\mathbf{W}_{kl} = (\mathbf{W}_k - \mathbf{W}_l) \odot (\mathbf{W}_k - \mathbf{W}_l) \quad (4.41)$$

Each element of the delta weight matrix is calculated as follows:

$$w_{ijkl} = (w_{ijk} - w_{ijl})^2, w_{ijkl} \geq 0 \quad (4.42)$$

Values close to zero indicate that these combinations between object  $i$  and  $j$  are as common (or uncommon) in both classes. Values larger zero, in particular values larger 1, will indicate combinations which are very likely in one class but highly unlikely in the other. Thus, these are the combinatorial possibilities which separate the classes. If two classes yield entries in their weight matrices which

are very close, they will be more difficult or even impossible to separate during an anomaly detection as discussed in the previous subsections. On the other hand, classes with high delta weight values  $w_{ijkl}$  for all combinations will be easily separable by the anomaly detection method. Thus, classes with overall high delta weight values are more different to each other and classes with low high delta weight values are more similar to each other.

If the direction of the relation between class  $k$  and  $l$  is relevant, a score  $q_{ijkl} \in (-1, 1)$  using a percentage ratio for each weight matrix entry can be calculated. A score of 1 indicates that a combination between object  $i$  and  $j$  is always active in class  $k$  but never in class  $l$ , while a score of  $-1$  would indicate that this combination is always active in class  $l$  and never in class  $k$ . Zero acts as a neutral value of combinations which are equally common in both classes. This score is computed as follows:

$$q_{ijkl} = \frac{w_{ijk} - w_{ijl}}{2} \quad (4.43)$$

It should be noted that only the absolute value of the score  $q_{ijkl}$  is relevant for a distinction between two classes and the sign of the score is only a measure of the "direction" of the relationship of a combination between two classes.

To estimate the similarity or dissimilarity between two matrices, the matrices can either be compared using a matrix norm or a distance. A distance  $d(\mathbf{A}, \mathbf{B})$  between two matrices  $\mathbf{A}$  and  $\mathbf{B}$  is measured using a metric which must fulfill the following requirements:

1. The distance between a matrix and itself is zero. (Nonnegativity)
2. The distance between matrices is positive. (Positivity)
3. The distance from matrix  $\mathbf{A}$  to  $\mathbf{B}$  is equal to the distance from  $\mathbf{B}$  to  $\mathbf{A}$ . (Symmetry)
4. The triangle equation applies.

A matrix norm is different from a metric since it measures the size of a matrix but not the distance between two matrices. In addition, a norm must fulfill the

requirement of homogeneity: If a matrix is scaled by a scalar factor  $a$  the resulting matrix norm also scales by the absolute scalar value  $|a|$ . The Nonnegativity requirement becomes the requirement of definiteness: The norm of a matrix becomes zero, only if it is the zero matrix  $0_{m,n}$ . In addition, the requirements of subadditivity and submultiplicativity are often required. Therefore, it is possible to derive a metric from a norm; and thus, also the distance can be computed using a matrix norm. *Vice versa*, if a metric fulfills the norm requirements, the distance  $d(\mathbf{A}, 0)$  is a norm for measuring matrix  $\mathbf{A}$  (Horn and Johnson 1985, chap. 5).

Since the similarity between two weight matrices is computable by using a norm, it is possible to derive a suitable norm from the commonly used Frobenius norm  $\|\bullet\|_F$ . The Frobenius norm adheres to the requirements of homogeneity, definiteness, subadditivity, submultiplicativity, and the triangle equation applies. The Frobenius norm of a  $M \times N$  matrix  $\mathbf{A}$  with the matrix entries  $a_{ij}$  is defined as follows:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=0}^M \sum_{j=0}^N |a_{ij}|^2} \quad (4.44)$$

Since the weight matrix is in case of combinations symmetrical, the applied norm should only summarize the unique entries. Thus, the matrix norm is adjusted for the purpose described here to the norm  $\|\bullet\|_{F^*}$ . In addition, the delta weight values  $w_{ijkl}$  are already squared and non-complex scalars. Therefore, the applied norm becomes as follows:

$$\begin{aligned} d(\mathbf{W}_k, \mathbf{W}_l) &= \|\mathbf{W}_k - \mathbf{W}_l\|_{F^*} \\ &= \sqrt{\sum_{i=0}^V \sum_{j=i}^V (w_{ijk} - w_{ijl})^2} \\ &= \sqrt{\sum_{i=0}^V \sum_{j=i}^V w_{ijkl}} \end{aligned} \quad (4.45)$$

The derived norm  $\|\bullet\|_{F^*}$  adheres to all listed norm requirements only in case of symmetrical matrices and thus, is in this defined case applicable as norm to compute a similarity. In the case of sequential data, the Frobenius norm  $\|\mathbf{W}_k - \mathbf{W}_l\|_F$  can be applied unchanged since it is here required to summarize over all entries. In the case of combinations, the norm  $\|\mathbf{W}_k - \mathbf{W}_l\|_{F^*}$  induces a metric to compare two weight matrices  $k$  and  $l$  and thus, a distance between these two classes is defined.

If there exists a number of  $K$  classes, a  $K \times K$  distance matrix  $\mathbf{D}$  for the dissimilarity between all classes can be created. The distance matrix yields at position  $k$  and  $l$  the corresponding entries  $d(k, l) = \|\mathbf{W}_k - \mathbf{W}_l\|_{F^*}$ . Using this distance matrix, the classes can be formally compared and evaluated, e.g., by applying a hierarchical clustering algorithm. Results could be displayed by a dendrogram. Classes within the same subtrees are more similar to each other while classes on subtrees which become separated early are more different and easier to distinguish when applying an anomaly detection. Furthermore, if the moving averaged stimulated energy after a training, which utilizes multiple classes for an anomaly detection task, fluctuates too much, specific classes can be excluded by analyzing the distance matrix first. In addition, classes which are highly different to all other classes can be excluded. These outlier classes can be detected, e.g., by applying a LOF as proposed by Breunig et al. (2000) using the distance matrix as input for the anomaly detection model.

It should be noted that the proposed metric induced by the Frobenius norm is just one useful example of norms to compute the distance matrix. Other entry-wise  $L_{p,p}$  matrix norms might also be applicable. A  $L_{p,p}$  norm is defined as follows:

$$\|\mathbf{A}\|_{p,p} = \left( \sum_{i=0}^M \sum_{j=0}^N |a_{ij}|^p \right)^{\frac{1}{p}} \quad (4.46)$$

It is assumed that  $L_{p,p}$  norms are preferable over a  $L_{p,q}$  norm since columns and rows should be handled the same since the arrangement of objects is in most cases arbitrary or without any domain-specific technical considerations.

In addition, further norms, distances, and approaches exist to compare matrices, e.g., the Kullback-Leibler divergence or a comparison using the eigenvalues, since a metric can in principle be freely defined and selected. However, the Frobenius norm proves itself most useful in this application since it directly relates to the delta weight matrix analysis of equation (4.42) and is thus not freely selected but directly derived from the model's definition and layout. Therefore, the Frobenius norm yields in this case a stronger interpretability compared to other methods, which is favorable for an application in manufacturing.

### 4.4.4 Explainability

While the recommendation procedure in case of the anomaly detection model and the distance metric in case of the classification model are already introduced tools for modified Hopfield neural networks to gain insights into the decision logic of an applied model, a further discussion about explainability is useful since the topic of Explainable Artificial Intelligence (XAI) is of great interest in research and application. In addition, planning experts confronted with such system also demand high levels of explainability for their production planning tasks since they need to adjust and alter their current layouts and process planning based on the recommendations proposed by the introduced system. This demands trust in the system as well as an understanding of the decisions and recommendations of the system in human comprehensible terms. Therefore, it should be evaluated if the introduced modified Hopfield neural networks encompass sufficient functionality to be classified as an explainable model or at least provide some explainability in their decision and recommendation procedure.

Confalonieri et al. (2021) describe the origins of XAI in the development of knowledge-based expert systems and it is nowadays revived as a research field with the objective to convey safety and trust to users by adding further information about the "how" and "why" into an automated decision-making for AI models applied within different domains and applications.

A commonly assumed hypothesis (cf. Došilović et al. (2018) and Barredo Arrieta et al. (2020)) within the body of literature is that there exists a trade-off between explainability and performance of a model, e.g., DL models provide a high accuracy in multiple domains and applications but lack the necessary explainability. On the other hand, decision trees have a very high explainability but limited accuracy. Decision trees can be expanded into a Random Forest, an approach from ensemble learning, and thus, they lose explainability but gain accuracy. Therefore, Angelov et al. (2021) state that the objective is not to create any XAI solution but an XAI solution with an accuracy comparable to DL models.

Angelov et al. (2021) distinguish in the context of XAI for ML and DL models the terms of transparency, interpretability, and explainability. Gilpin et al. (2018) use completeness as an additional term for XAI systems. Derived from these definitions, the terms are in this thesis defined as follows:

1. **Transparency.** A model can potentially be understood without a consultation using further tools or models and thus is the opposite of a "black box" which does not disclose its internal design as defined by Adadi and Berrada (2018).
2. **Interpretability.** A model provides the capability to explain or to present using human-understandable terms as defined by Gilpin et al. (2018).
3. **Completeness.** A model can be described in an accurate way and thus, the system's behavior can be anticipated by users in multiple situations (Gilpin et al. 2018).
4. **Explainability.** A model provides an interface between itself and human users and is accurate as well as comprehensible to any human user. Thus, the model encompasses interpretability and completeness whereby explanations allow a trade-off between both for users to select. This can be archived by either explaining the processing or representation of the data inside the model or by creating system architectures which are designed to deliver interpretations of their behavior (Gilpin et al. 2018).

Angelov et al. (2021) note that these very similar terms regarding their meanings still confer to different levels of AI accepted by human users. Gilpin et al. (2018) note that explainable systems should not hide complexity but allow for more detailed descriptions with more completeness at the cost of interpretability.

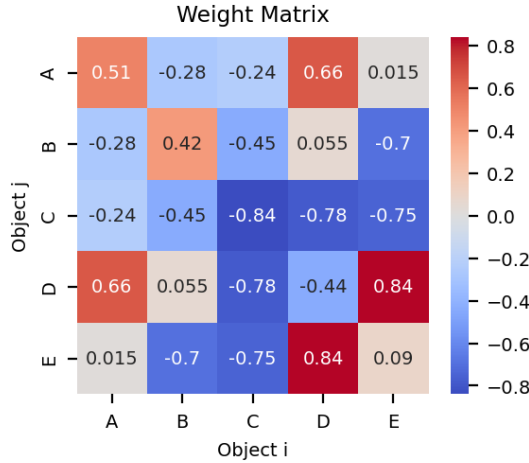
A further discussion of XAI and developed taxonomies for these models would go beyond the scope of this work. In the following discussion, modified Hopfield neural networks are evaluated in regard to the previously discussed tasks and concepts of XAI.

First, the analysis of the weight matrix  $\mathbf{W}$  is of importance to understand the decisions of the anomaly detection or classification model. As already discussed, the entries of the weight matrix are directly interpretable as the probability of a combination of two corresponding objects. The weight entries are bounded by  $w_{ij} \in (-1, 1)$  and thus, a weight of  $+1$  can be interpreted that this specific combination of objects must be present at all times within correct or common combinations. Using the same idea, weight entries close to one indicate combinations of objects which are most of the time within a combination  $\mathcal{C}$  and their absence would indicate some sort of irregularity. If multiple common combinations with values close to  $+1$  are not within the set  $\mathcal{C}$ , the set is most likely anomalous. *Vice versa*, if weight entries are  $-1$ , these combinations are never present within correct combination sets  $\mathcal{C}$  and values close to  $-1$  indicate very unlikely combinations. Thus, if many combinations are active with weight entries close to  $-1$ , the combination is most likely anomalous. This is directly measured by the impact  $I_i$  of an object within a combination as given in equation (4.31).

Using the formulated idea, the weight matrix computed in an anomaly detection model or for a single analyzed class in a classification model can be visualized using a heatmap plot in the case of a low number of possible objects. An exemplary weight matrix with  $V = 5$  possible objects is given in Figure 4.13.

In the example in Figure 4.13, combinations of object  $D$  and  $E$  are very likely. Thus, most sets  $\mathcal{C}$  will contain  $\{D, E\}$  as a subset. On the other hand, connections of object  $C$  with itself are very uncommon and sets will most likely not contain subsets of  $\{C, C\}$ . Overall, all combinations including  $C$  are more uncommon.





**Figure 4.13:** Exemplary weight matrix visualized by a heatmap plot. Highly common combinations are colored red, anomalous combinations are colored blue.

Furthermore, by observing the highest values of this weight matrix, very common types of combinations can be extracted. In case of the weight matrix in Figure 4.13 these common combinations are  $\{B, B\}$ ,  $\{D, E\}$ , and  $\{A, A, D\}$  as well as its resulting subsets. The neutral values of the weight matrix close to zero for combinations of  $\{A, E\}$ ,  $\{B, D\}$ , and  $\{E, E\}$  indicate that these combinations are sometimes within the analyzed sets. Thus, an exemplary combination of  $\{A, A, D, E, E\}$  would be most likely a correct combination since it contains multiple highly common subsets. Therefore, it is possible to derive common and uncommon combinations by directly analyzing the weight matrix. In case of large object number  $V$ , the most likely and most unlikely combinations can be extracted to reduce the matrix to its most relevant values in a manual analysis.

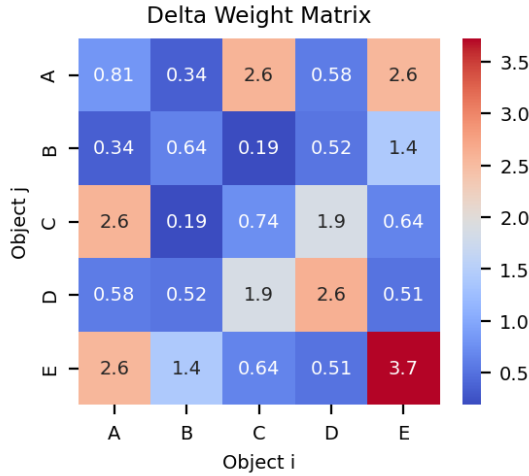
By analyzing the highest and lowest weight matrix entries, a user of a modified Hopfield neural network can directly evaluate possible combinations and create valid interpretations of why some combinations are anomalous. It is further beneficial that the weight matrix entries are  $w_{ij} \in (-1, 1)$  and thus, a metric to evaluate single combinations is directly derived from the model. The impact

$I_i$  can furthermore evaluate single objects within a combination and also gives a direct interpretation by computing whether the impact is negative or positive.

In case of the comparison of two distinct classes  $k$  and  $l$  within a classification model, the introduced weight matrix  $\Delta \mathbf{W}_{kl}$  can be analyzed. Each entry of this matrix yields a value of  $w_{ijkl} \in (0, 4)$  and can be analyzed in a similar manner as the single weight matrices. Values close to zero are indicating combinations of objects  $i$  and  $j$  which are common in both classes. *Vice versa*, values close to 4 are indicating combinations of object which are very common in one class but uncommon in the other and thus, are the main distinguishing factors between the two analyzed classes. If a combination with a high value of  $w_{ijkl}$  is present within a combination set, this will highly influence the decision of the classification model. However, combinations with values of  $w_{ijkl}$  close to zero will have a smaller influence on the classification decision. This can also be analyzed by using the score  $q_{ijkl}$  as given in equation (4.43). This score has the additional advantage of showing the "direction" of the effect and whether a combination is more likely in class  $k$  or  $l$ . An analysis using the score  $q_{ijkl}$  is conducted in a similar manner as using the delta weight matrix, whereby the delta weight matrix is more focused on the absolute difference between two classes and the score is more focused whether a combination is more common in one class or another.

From this follows that classes with very distinct weight matrices (thus multiple high values are present within  $\Delta \mathbf{W}_{kl}$ ) are easier to separate within a classification task. This is directly measured by the computed distance matrix  $\mathbf{D}$  using the matrix norm applied in equation (4.45). If this distance matrix is visualized using a dendrogram, classes which are easier to separate are in distinct trees of the dendrogram and *vice versa*, classes which are harder to separate are within the same subtrees. Using classes in a classification model which are harder to distinct, will result in a reduced accuracy of the model due to mistaken classification in particular between the similar classes.

An exemplary delta weight matrix is visualized for a classification model using also a heatmap plot for the case of a low number of possible objects of  $V = 5$  in Figure 4.14.



**Figure 4.14:** Exemplary delta weight matrix between two classes visualized by a heatmap plot. Shared common combinations are colored blue, distinguishing combinations are colored red.

In this case, combinations of  $\{E, E\}$  are very likely to occur within one class but mostly never present within the other analyzed class. Thus, if the combination  $\{E, E\}$  is present, this will highly affect the outcome of the classification. The combinations  $\{A, C\}$ ,  $\{A, E\}$ , and  $\{D, D\}$  are also very common in one class but uncommon in the other class. Thus, a combination such as  $\{A, C, D, D, E, E\}$  will most likely result in an unambiguous classification. On the other hand, combinations between objects with small values of  $w_{ijkl}$  such as  $\{B, C\}$  or  $\{A, B\}$  will have little relevance during the classification regardless of whether they are common or uncommon in both classes.

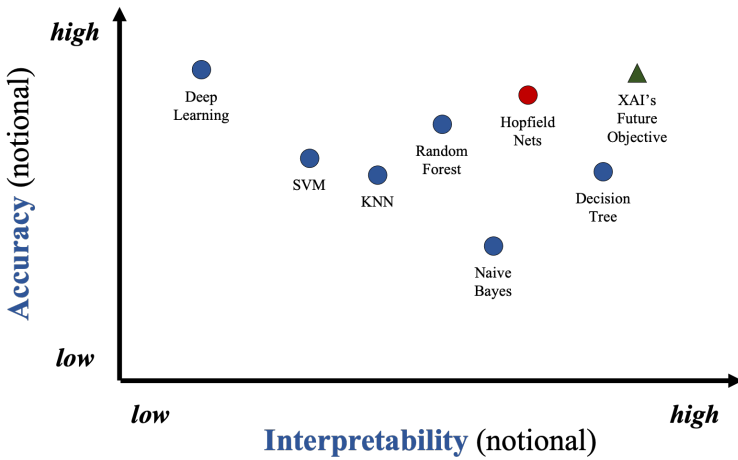
Similar to the anomaly detection model, it is possible to derive common and uncommon combinations for each class by directly analyzing the delta weight matrix and thus, create an interpretation whose combinations are typical for a specific class. Furthermore, the analysis of all classes using the distance matrix **D** and a dendrogram gives additional context regarding the similarity of classes. Again, in case of a large object number  $V$ , the most distinguishing combinations can be extracted from the delta weight matrix to reduce the matrix to its most

relevant values in a manual analysis. It is also beneficial that the delta weight matrix entries are within a bound of  $w_{ijkl} \in (0, 4)$  and the values can be directly interpreted.

To summarize, the introduced modified Hopfield neural network is highly transparent due to the direct computation of the weight matrices  $\mathbf{W}$ , the delta weight matrices  $\Delta \mathbf{W}_{kl}$  as per equation (4.41), and the impact  $I_i$  as per equation (4.31). The decisions and recommendations of the model can be understood using only the already computed parameters and evaluation metrics of the model and no further analyses or tools are required. The bounded values of the weight matrix entries  $w_{ij}$  and delta weight matrix entries  $w_{ijkl}$  can be used to directly derive an interpretation of combinations and classes as well as the evaluation whether the impact is positive or negative. The dendrogram using the distance matrix  $\mathbf{D}$  computed by the matrix norm in equation (4.45) is a further capability of the model to directly interpret result, recommendations, classes, and combinations. These interpretations are given in human-understandable visualizations and values. Therefore, it is concluded that the model is transparent as well as interpretable as summarized by Angelov et al. (2021). Since the weight matrix also encompasses the decision process of the model, an interpretation using the weight matrix and impact also adheres to the objective of completeness as described by Gilpin et al. (2018) since the used description is an accurate representation of the model's decision process.

The performance of modified Hopfield neural networks very high is in certain use cases and can excel the performance of One-class SVM (Schölkopf et al. 2001) and of Isolation Forest (Liu et al. 2008). This is shown by Spoor and Weber (2024) in the case of an anomaly detection task where only true negative cases are available during the training procedure. The model's performance is also further analyzed and benchmarked in later sections about the validation of the methodology. However, using the already analyzed performance, modified Hopfield neural networks can be placed within the accuracy vs. interpretability evaluation as given by Angelov et al. (2021) in the context of the use case of anomaly detection of combinations using true negative training data. Since accuracy as well as interpretability of modified Hopfield neural networks are very

high in the use cases which they were designed for, the scoring in the evaluation reflects its competitive edge for these applications: its accuracy exceeds those of SVMs or decision tree-based approaches and its interpretability is more straightforward compared to those of Random Forests. The accuracy vs. interpretability assessment including modified Hopfield neural networks is given in Figure 4.15.



**Figure 4.15:** Accuracy and interpretability assessment for different ML models by Angelov et al. (2021) including modified Hopfield nets for the evaluation of combinations.

To evaluate whether the model is explainable using the taxonomy by Angelov et al. (2021), it is more important how to design the HMI within an application of the model and how to document, aggregate, and display data. The model has the capability to show the specific objects which are most relevant for the evaluation in an anomaly detection task for any combinations and can provide recommendations on how to adjust a tested combination. The classification is also capable to provide the same reasoning. Therefore, the model is in principle capable of being explainable. However, this term highly depends on the implementation of the HMI in particular, which is domain and application specific. Thus, the implementation should consider the trade-off between completeness of an explanation

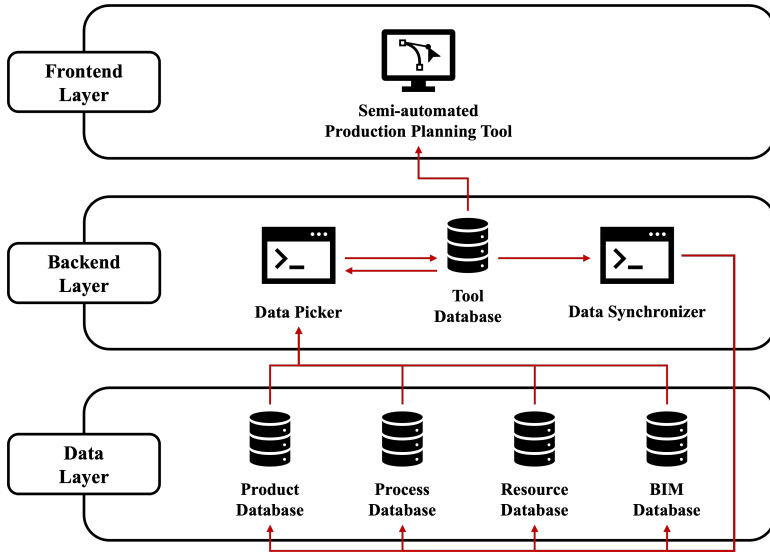
and interpretability. Completeness within an explanation is archived by using the full weight matrix and computations steps. An easy interpretability for users is archived by renouncing to display too many details or precise mathematical computations and rather highlight the domain-specific object and combinatorial relations in order for users to better understand and fully trust the system. However, users should be capable of selecting their required level of interpretability and completeness to prevent any concealment of complexity.

Within the field of production planning analyzed in this thesis, the proposed design and extend of the HMI is further discussed in the following section describing the embedding of modified Hopfield neural networks within a support system in production planning. Therefore, the following section will show how to implement an explainable support system for production planning in practice using modified Hopfield neural networks.

## 4.5 Implementation in Production Planning

This section aims at a discussion of the relevant interfaces of the methodology to an applied production planning process within the manufacturing industry. Firstly, the implementation might highly differ between individual industrial applications, manufacturing systems, and companies. Thus, this section does not propose a detailed implementation plan but provides context and guidance for any implementation. As main field of interest in a practical implementation, the information system architecture, the data management and governance, which can be summarized to the overall data quality, and the usability of a planning software which incorporates the methodology, often also referred to as user experience (UX), is discussed in more detail.

First, the information system architecture is discussed with a focus on the data flow and data management of the support system and the corresponding ontological information. Figure 4.16 proposes a generic view of an implementation of the production planning tool into a manufacturer's IT architecture.



**Figure 4.16:** Schematic design of the IT system architecture for the management of the required data transfer from different subsystems into the product planning tool.

Most notably in Figure 4.16, a variety of databases must be utilized in order to create a consolidated database for the production planning. The knowledge domains within manufacturing often utilize different systems and an integration of multiple structured or unstructured systems using heterogeneous data sources is a key requirement. An integration might require data from domains such as factory, building, product, process, resources, systems, strategy, or management (Agyapong-Kodua et al. 2014). From a system perspective, this may include, among others, a product, components, and auxiliary products database, a process library, a resource database, and a BIM database, including energy or consumable material such as pressurized air requirements. Within larger organizations, these databases will be managed and owned by different interorganizational stakeholders, e.g., the product data might be managed and primarily owned by the product development department, while the resource data might be managed by the product planning department (Bazaz et al. 2020). Thus, ownership over the master data will be distributed over the organizations across departments. In some use

cases, the integration of supplier systems might be required (Galkin et al. 2017). This integration of data sources from heterogeneous systems is a commonly discussed hurdle when designing knowledge-based systems, in particular discussed for knowledge graphs by Noy et al. (2019). Therefore, a data picker module within a backend must create and manage a tool-specific database in order to establish a suitable consolidated database acting as a central point of truth for the production planning tool. The data picker conducts in addition a data preprocessing and a data cleansing procedure in order to circumvent incorrect or redundant data since duplications are a common problem in knowledge-based systems such as knowledge graphs (Galkin et al. 2016). The resulting database includes the full ontological PPR model data in a suitable data format and in a sufficient data quality to enable the production planning tasks. In order to utilize the ontological data, a graph database might be superior to a relational database in this application (Wang et al. 2021, chap. 6).

Since the different stakeholders of the databases might result in deviating or redundant master data, it is proposed to additionally utilize a data synchronizer which documents and corrects contradictory information across the different databases. This enables a continuous updating and improvement of the ontological models within an organization. This described data synchronizer acts as an automated solution for data governance. The task of data governance includes data security, data consistency, a strategy to enable a profitability of data usage, and also an organizational concept for the accountability regarding the data quality (Al-Ruithe et al. 2019). However, an automated solution still requires organization and company guidelines.

Lastly, the production planning tool is located in the frontend using the created tool database by the data picker within the backend. Users of the system interact via an HMI with the planning tool in the frontend layer.

It should be noted that the specific layout of the IT system architecture will differ between companies. However, the core elements of the proposed schematic design in Figure 4.16 are essential in order to manage the complexity of a PPR model and the distribution of data sources within a manufacturing company.



As the second field of interest in a practical implementation, the organizational data ownership and governance is discussed. This topic has already been touched on in the discussion about the databases, data picker, and data synchronizer. However, these technical solutions might in short-term mitigate a problem which should be tackled in the long-term by an organizational approach of handling planning data. Bazaz et al. (2020) note in the context of the Digital Twin development that future research should aim at providing solutions for issue of data ownership. The requirements are summarized as data quality which should fulfill five principles based on the prior considerations by Wang and Strong (1996), Jarke and Vassiliou (1997), and Reuter and Brambring (2016):

1. **Data Consistency.** The data gathered from different data sources do not result in contradicting information, this requires in particular a concept for long-term use after updating of the data which might result in time inconsistencies.
2. **Data Correctness.** The gathered data reflect the underlying real circumstances correctly and errors are systematically documented and corrected.
3. **Data Completeness.** The data were gathered without missing instances and are fully accessible, in case of the ontological data this requires a complete data modeling along the PPR model.
4. **Data Relevancy.** The data are relevant for the user of the system, redundancies are avoided, and users are enabled to draw conclusions without a consultation of additional data sources.
5. **Data Representation.** The data are documented in an intelligible and clear manner, e.g., in the use case discussed here using a machine-readable as well as human-interpretable ontology.

Organizations address data quality in traditional data governance concepts with the roles of Data Stewards and Data Custodians. Data Stewards focus on content, context and business rules while Data Custodians focus on safeguarding, storage and transfer, and application of business rules (Kasrin et al. 2018). Since the support system proposed here heavily relies on data quality, it is recommended

that in addition to a data quality focused IT system architecture, an organizational concept for data management must be developed in order to safeguard a long-term usability of the support system. This is particularly important if lessons learned from similar production systems are documented and utilized as part of the inference procedure to derive knowledge about new planned production systems as targeted in the objectives of the support system in Figure 4.2. The roles of Data Stewards and Data Custodians might be applicable solutions but to the author's best knowledge no standardized concept within the manufacturing industry exists in practice which solves the data quality requirements. Thus, it is proposed that the organizational aspects of data quality must be individually addressed within any practical implementation of the support system as part of the implementation and development process. In particular the ontology and case database of the support system highlighted in Figure 4.3 should be a focus of the considerations regarding data quality.

As the third field of interest, the task of UX design is discussed as a central aspect when implementing new software solutions in the manufacturing industry since business and user requirements are becoming increasingly important success factors (Pokorni et al. 2020). In addition, UX design is a core aspect in implementing the XAI principles for users of an AI support system. Thus, the task becomes the visualization of the anomaly detection, classification, and recommendation procedure in a manner which satisfies UX as well as XAI requirements.

Regarding the interface of usability and XAI principles, it is important that the system does not hide any complexity but enables a more detailed and complete explanation based on user preferences (Gilpin et al. 2018). Thus, it is proposed to visualize the resulting weight matrix, in particular the weight of combinations which might explain a detected anomaly. A simplistic approach is the visualization of the planned production system as a graph network. However, the software should enable detailed analysis of subgraphs within a larger network and also enable a view of different ontological levels. A user should be able to analyze the connection along the PPR model but also a deeper view of used tools, sub-processes, or product components should be enabled.

In case of the visualization of the results of the support system, it should be noted that in practice no uniform systematic methodology is applied but tools are situationally selected which are determined as appropriate by experts for this specific application (Parsons 2022). However, as a basis for future implementations, a concept for a visualization method of the support system's results is in the following proposed using the inherited graph-based structure of the data.

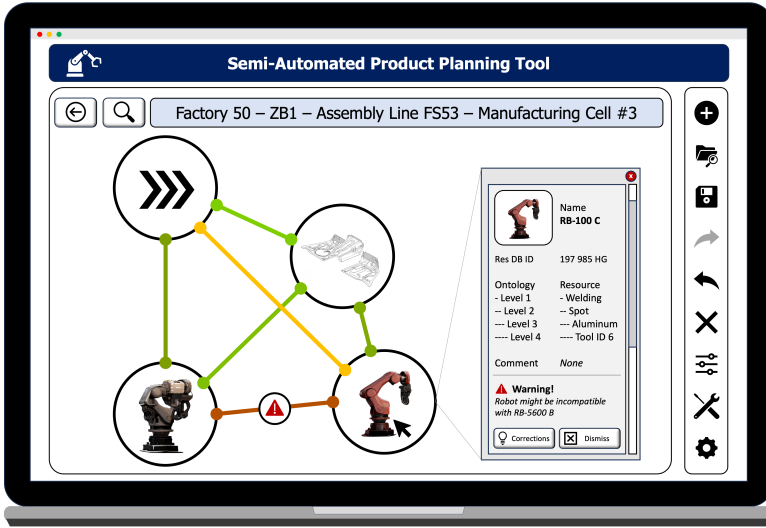
As an exemplary mock-up of a planning software using the anomaly detection model which utilizes a graph-based visualization in order to represent the weight matrix is given in Figure 4.17<sup>1</sup>. Weight entries with values close to 1, i.e., very common and likely combinations, are colored in green, weight entries with values close to  $-1$ , i.e., unlikely and anomalous combinations, are colored in red. Values between 1 and  $-1$  are colored using color gradings between green and red. This enables an interpretable view of the manufacturing cell planning layout utilizing the anomaly detection model. Users should also be able to handle the proposed recommendation system. Based on the manufacturing cell in Figure 4.17, an exemplary mock-up of the recommendation procedure is given in Figure 4.18<sup>2</sup>.

As proposed, the recommendation procedure selects the objects with the highest negative stimulated energy impact and proposes this as an addition to the current manufacturing line. In Figure 4.18, objects are only added. However, the recommendation might add and remove objects. Since a user might not always automatically want to select the object with the highest impact on the energy measure, a list of possible objects should be proposed. The order in which the proposed objects are listed within the software should reflect their relative impact compared to the computed best option. This can be displayed in percent in order to hide the full energy computation while also providing a detailed and complete explanation but aggregated explanation of the recommendation. This approach is only semi-automated since the final manual decision is made by the users of

---

<sup>1</sup> The robot icons were made using the AI image generation platform Midjourney. The CAD model shows a car body component in the underbody of the vehicle and was made available by the Mercedes-Benz AG.

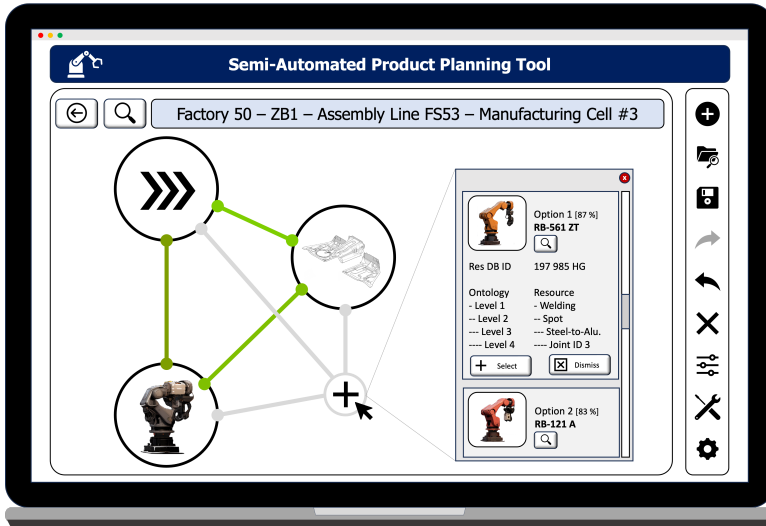
<sup>2</sup> See footnote 1



**Figure 4.17:** Schematic mock-up of a software using the graph-based visualization of the weight matrix of a manufacturing cell for an explainability of a detected anomaly.

the production planning software. However, the proposal of useful alternatives might steer and fasten the manual correction of an automatically created assembly design by optimization methods, as discussed by Hagemann and Stark (2020).

It should be noted that Figure 4.17 and 4.18 are stylized suggestions in order to highlight a possible design which utilizes the explainability provided by the weight matrix. Genuine solutions within a software implementation might differ since more production planning aspects such as documentation, data management, or KPIs might also become relevant which impact meaningful and practical UX designs. Additionally, the trend in automotive manufacturing, in particular in production planning, is to introduce collaborative work environments. This trend is driven by an increasing international collaboration in planning new manufacturing systems where experts from different company locations and international supplier networks are working collaboratively. On aspect hereby is the development of Digital Twins as enablers of an Industrial Metaverse (Yao et al. 2022) which is a form of an immersive collaborative virtual environment. Usability is



**Figure 4.18:** Schematic mock-up of a software for the recommendation procedure in order to correct the manufacturing cell of Figure 4.17.

a main factor for the acceptance of immersive collaborative virtual environments (Mayer et al. 2023) and thus, the development of refined UX concepts is required.

As a main take-away of this section, it is concluded that the aspects of IT system architecture, data quality, and UX design are core requirements to safeguard an implementation in practice. The applicable solutions to these challenges are case-specific and might differ between different manufacturing companies implementing the proposed support system (or similar support systems in general). However, the discussed aspects should not be handled as afterthoughts but as key success factors in a practical application of the proposed AI-based support system. Thus, adequate measure must be taken regarding these fields of interest in any implementation of the model.

## 4.6 Summary

This section provides a comprehensive methodology for the fault detection during the rough production planning. The AI support system follows the principles of a utilization of only true negative data during training, the consideration of context, the embedding of ontological information, the derivation of a similarity measure between ontological set-ups, and the securing of interpretability of results. The first objective of this AI support system is the detection of anomalous configurations, the showcasing of problems, and the recommendation of alternatives. This is enabled by an anomaly detection model. The second objective is the identification of similar production systems and the provisions of lessons learned from former set-ups. This objective is solved by a classification model. Both objectives require a high explainability and result in a structured inference.

In order to fulfill the listed principles and resolve the objectives, a one-layer NN which follows the idea of Hopfield neural networks is introduced. Ontological planning information are mathematically modeled as a graph network. Instances of the ontology are modeled as vertices and the relations between instances as edges. Each configuration is transferred into a connection matrix. In order to describe which edges are common or uncommon, a weight matrix is derived measuring how likely it is whether the specific edge between the instances is active or inactive. Using this model, a stimulated energy is defined and the model can be trained so that likely ontological configurations yield low stimulated energies and anomalous configurations yield high energies. The training procedure is conducted iteratively by weight updates. It is shown that the training can be broken down into an optimization method using least means squares in order to optimize a defined equilibrium energy. By evaluating the stimulated energy level, an anomaly detection is enabled. In order to define a classification model, a specific weight matrix is trained for each class. Configurations can be assigned to a class by comparing the resulting stimulated energy levels using the corresponding weight matrix of each class.

This model yields a high explainability and interpretability by analyzing the resulting weight matrices. In addition, specific recommendations can be created to optimize configurations based on the weights for specific edges. In addition, the model can be used for configurations using sequences, different types of relations between instances, and ontologies using multiple concepts. Concluding, conceptualizations of the IT system architecture and mock-ups are provided which show how the model can be embedded into production planning software. Thus, the introduced AI support system using the modified Hopfield neural networks can be utilized in the rough production planning as a tool for an early fault detection.





## 5 Validation of Methodology

In order to validate the developed methodology, three use cases are presented in the first subsection. Two use cases are from the field of manufacturing and are direct implementation of the proposed support system and AI model in different subfields of production planning. The third use case is an excursus within the field of oncology in order to demonstrate the application of modified Hopfield neural networks in a more general manner. By analyzing the use cases, the application of the methodology is demonstrated in practice, the performance of the underlying AI model is benchmarked, and the method is validated by applying the support system as a proof-of-concept in the specified use cases.

Concluding, the found benefits, limitations of the AI model and support system, relevant details or conditions for an application, and occurring challenges in the presented use cases are discussed. This discussion complements the validation of the methodology by providing practitioners with the relevant information whether a specific use case is a possible field of application and which prior considerations need to be accounted for in an application. In addition, the discussion provides researchers with potential fields of action for a further development or an adjustment of this AI support system in order to enable more use cases and applications across multiple domains. Furthermore, the AI model using modified Hopfield neural networks is discussed as the core functionality of the developed support system in production planning in order to generalize its application, encourage an application in domains outside of production planning, and highlight modified Hopfield neural network as original and independent AI models for multiple kinds of data analysis.

# 5.1 Use Cases

In the following section, three use cases are presented in order to demonstrate and validate the develop methodology. The first two use cases are from the domain of manufacturing, in particular, the first use case is from the field of automotive Body-In-White assembly, and both use cases are direct implementations of the proposed AI support system using the modified Hopfield neural networks. In addition, the third use case showcases the capabilities of modified Hopfield neural network within another domain as part of an excursus in order to prove the versatility of the AI model for anomaly detection and classification tasks.

The use cases differ in their domain, in the underlying type of objects which are analyzed in the connection matrix, in how object values induce a combination, in what kind of training data is available, and in the task of the use case. The overview of the demonstrated use cases of this section is presented in Table 5.1.

**Table 5.1:** Overview of the basic information of the analyzed use cases.

Use Case	5.1.1. Process Planning	5.1.2. Simulation with Synthetic Data	5.1.3. Leukemia Diagnosis
Domain	Automotive Manufacturing	Manufacturing	Oncology
Objects	Ontological Process Groups	Ontological Resource Groups	Gene Expressions
Object Values	Boolean	Boolean	Intensities
Training Data	Correct Combinations	Correct Combinations & Sequences	Labeled Subtypes of Leukemia
Task	Anomaly Detection	Anomaly Detection	Classification

A major difference between the use cases are the different object values. In the two use cases of process and resource planning, the connection matrix is computed using a Boolean condition for the active edges between two objects which are in the analyzed combination set  $\mathcal{C}$ . In these use cases, the connection condition of equation (4.1) applies. In the case of gene expressions, all genes are always present in a patient but yield different intensity values. Thus, a combination is assumed in cases of bilateral high intensity values of the measured gene expressions as visualized in Figure 4.8. The second difference is the overall task and the used underlying training data. The first use case describes an anomaly detection and utilizes only correct data in order to identify incorrect data as postulated in the design principles of a support system in Figure 4.1. The third use case describes a classification where the data set is labeled according to the corresponding class.

The presented use cases encompass the relevant design principles and objectives of the AI support system and are therefore capable of providing application examples, a demonstration of an analysis using the support system and modified Hopfield neural networks, and they validate the methodology by showcasing the performance and application of the AI model and support system.

### **5.1.1 Process Planning in the Body-In-White Assembly**

This use case was first presented by Spoor and Weber (2024) and is discussed in this subsection within the context of the proposed AI support system. First, the use case is described in more detail and the context and relation to the introduced AI support system is discussed, in particular to the design and functionalities of the system. Next, the training procedure and hyperparameter optimization is presented. Based on the result of the parametrization, an anomaly detection is conducted. The results are analyzed regarding performance and benchmarked with other commonly used AI models. Concluding, a recommendation procedure as embedded tool of the Digital Factory is described and the explainability of the model is demonstrated, which enables a transparent support system in adherence to the XAI principles.

### 5.1.1.1 Use Case Description

The presented use case comprises the automation and the assistance of the early planning phases of production planning of the Body-In-White assembly in manufacturing. Spoor and Weber (2024) demonstrate a fault detection for production systems during the early process planning. In the case of a process planning task for production systems, which could be conducted automated, semi-automated, or manually, a validation whether a specific process combination is a correct set-up of a production system is necessary. In the practical application, this is often still a manual approach and all planned process combinations are evaluated by domain experts. As a semi-automated solution and to increase the planning speed, Spoor and Weber (2024) propose a validation procedure of process combinations including a recommendation score of the likelihood whether a new planned system configuration is incorrect or correct. Thus, the objective of the support system is a highlighting of the suspicious process combinations while accepting commonly used process combinations. This is an improvement of the overall planning procedure since the manual validation efforts can be concentrated on suspicious cases. Thus, the presented use case follows the objective 1a – 1c of Figure 4.2. These objectives correspond to the explainable anomaly detection functionality of the support system as given in Figure 4.3 and the use case can be conducted along the methodology of training and evaluation the modified Hopfield neural network as summarized in Figure 4.6.

As presented by Spoor et al. (2022a), state-of-the-art methods of anomaly detection face common limitations, in particular incorrect process combinations are commonly disregarded during production planning and are neither documented nor archived. Thus, this use cases can only utilize correct combinations for the training procedure. The large amount and variety of processes results in a difficult implementation of rule-based approaches as first state-of-the-art methods of the Digital Factory in Process Planning. In addition, the analyzed database of implemented and correct process combinations is limited by the number of the current and former running production systems of the manufacturer. The second type of state-of-the-art methods of mathematical optimization approaches using linear or

nonlinear programming are also not feasible since a modeling of the restriction functions would as in the case of the rule-based approaches require a high initial effort. Also, the definition of an optimization function is not feasible since the objectives are difficult to quantify, in particular objectives such as operability, safety, or processing quality. Also, the trade-off between costs and cycle times is also hard to quantify beforehand. Therefore, Spoor and Weber (2024) apply a modified Hopfield neural network for this task in order to identify anomalous process combinations based on a training procedure conducted with only correct process combinations.

The data used by Spoor and Weber (2024) contain 8674 different process combinations which are former and current implementations of production systems of six years between the year 2017 and 2022 in the Body-in-White assembly from two different sites of the European automotive manufacturer Mercedes-Benz AG. The processes analyzed by Spoor and Weber (2024) cover the construction of the body frame, including assembly of the front, substructure, and rear carriage, the assembly of the side panels, and the assembly of doors and hatches of a premium car series. Therefore, the applied real-world data set covers the complete range of operations of the different manufacturing lines of the Body-In-White assembly.

The 8674 different process combinations are set up using 586 different individual processes. However, prior to the analysis, the data were grouped along their ontological data into 12 process clusters containing processes on the same ontological hierarchy level and thus, they cover similar tasks, applications, and uses from an ontological perspective. The ontological analysis was done by domain experts. In case of a newly developed process, this process would be assigned to the corresponding ontological group and could be analyzed in a similar manner. The data set used by Spoor and Weber (2024) is summarized in Table 5.2.

Each process combination is based on a combination from the set of the 586 unique processes. Processes can be combined with themselves. Spoor and Weber (2024) conducted the analysis based on the 12 ontological process groups. Thus, each process combination is a combination of objects from the set of the process groups, whereby process groups can be combined with themselves

**Table 5.2:** Description of the applied data set of process combinations.

Instance	Quantity	Comment
Process combinations	8674	from 2017 to 2022 full Body-In-White assembly
Unique processes	586	
Unique process groups	12	
Years	6	
Plants	2	

which follows from the combination of processes with themselves or with other processes from the same ontological group. In addition, certain restrictions may apply resulting in processes and process groups which cannot be combined together in one production system. Also, it is possible that some processes or process groups can only be combined in one production system if in addition one or multiple specific other processes are also within the combination set. Thus, a complex set of restrictions applies to the use case of process combinations for the production systems in the Body-In-White assembly and these restrictions must be modeled and furthermore be explainable during the interpretation of the results of the anomaly detection task.

In this application analyzed by Spoor and Weber (2024), the 8674 process combination are transformed into process group combinations using the ontological grouping. This process is highlighted schematically by Figure 4.4. Thus, the resulting complete network is an undirected multigraph permitting loops with  $V = 12$  vertices and  $E_{max} = 78$  edges corresponding to the exemplary network layout in 4.5. Each individual combination is a subgraph of the full combination network and thus, is also modeled as an undirected multigraph permitting loops. No sequential information was given in the analyzed data set.

The 12 process groups results in  $3^{12} - 12 = 531,429$  different possible process combinations in the production planning. This number includes combinations of process groups with themselves and is corrected for impossible combinations, e.g., if a process combination of group (A, B) and group (B, C) is applied, the

combination (A, C) must also be applied. Since there are duplicates within the list of process combinations, the number of unique combinations in the analyzed data set is  $< 1.7\%$  of the total number of possible combinations. The frequent occurrence of duplicates follows in particular from the use of process groups since different combination of processes may anyway result in similar or the same process group combinations. However, duplicates are not removed from the data set since they indicate often occurring combination which should be weighted accordingly.

Considering the exemplary number of combinations, it is trivial to see that rule-based approaches are limited in their applicability. Furthermore, in a practical application of production planning mostly correct instances are recorded but incorrect instances are deleted or insufficiently documented during the planning of former production systems. Thus, the available combinations only include validated correct combinations of objects. If an anomaly detection from the field of AI models is applied instead of rule-based approaches, only true negative instances are available for the training of a supervised anomaly detection algorithm.

The applied network structure does not change when additional processes are added to the ontology. It is assumed that new individual processes are most likely added to an existing ontological group. Spoor and Weber (2024) assume in this use case that in the case of a distributive technological change, a new ontological group is added and the underlying network structure changes. If a novel ontological group is added, another vertex is added to the network including the corresponding edges. This ontological group is then a new process group in addition to the existing 12 groups analyzed. Thus, it would be necessary to redo the analysis. However, in case of a distributive technological change, all previous knowledge would require a reassessment and also other AI models, optimization models, or rule-based approaches would require a reevaluation.

### 5.1.1.2 Training Procedure

For the analysis, the data sets are first transformed into symmetrical connection matrices  $\mathbf{M}$  using equation (4.1). Thereafter, a training procedure is conducted to iteratively adjust the weights of a  $12 \times 12$  weight matrix  $\mathbf{W}$  using equation (4.18). For the purpose of the evaluation and since no incorrect combinations are within the analyzed data set, Spoor and Weber (2024) created 30 incorrect combinations for this evaluation in cooperation with domain experts.

First, the relevant parameters of equation (4.18) must be selected. Thus, a hyperparameter optimization for selecting the values of the training rate  $\alpha$ , reduction of training rate  $\beta$ , and decay  $\gamma$  is necessary. The hyperparameter optimization is conducted by Spoor and Weber (2024) using a grid search minimizing the False Positive Rate (FPR) and using a fixed True Positive Rate (TPR) of 90%. True Positives (TP) are defined as process combinations which are incorrect and are also classified by the applied anomaly detection model as incorrect. *Vice versa*, True Negatives (TN) are defined as process combinations which are correct and also accurately classified as correct. False Positives (FP) are defined as correct process combinations which are falsely classified by the applied anomaly detection model as incorrect combinations and False Negatives (FN) are defined as incorrect combinations falsely classified as correct. Using these definitions, the evaluation criterions FPR and TPR are computed as follows:

$$TPR = \frac{\text{Number of TP}}{\text{Number of FN} + \text{Number of TP}} \quad (5.1)$$

$$FPR = \frac{\text{Number of FP}}{\text{Number of FP} + \text{Number of TN}} \quad (5.2)$$

TPR is in the body of literature often also called sensitivity or recall. It should be noted that the relation between FPR and TPR is not linearly and may not follow a specific trend. However, if the TPR is increased by a selected parametrization, the FPR does not decrease and instead may either stay stable or also increases.



*Vice versa*, a decrease of the TPR either results in a decrease of FPR or the FPR stays unchanged.

This classification into FP or TP combinations is conducted using equation (4.30) from the anomaly detection model. If the energy evaluation exceeds the critical energy value  $H_{crit}$ , the combinations are denoted as anomalous (and thus incorrect) combination. The critical energy value  $H_{crit}$  is hereby set so that the selected TPR of 90% is realized and the resulting FPR is computed. As discussed before, there might be more favorable combinations of TPR and FPR from a domain perspective but in order to compare the parametrization, the TPR is fixed.

The resulting FPR are computed for 30 Monte Carlo cross-validations. 90% of combinations from the data set described in Table 5.2 are used for the training procedure and the remaining 10% as well as the 30 incorrect combinations are used for the evaluation of the model. In addition, the biased correction value as specified in equation (4.9) is used in the evaluation by Spoor and Weber (2024). The biased correction value is used since Spoor and Weber (2024) state that diagonal entries in the connection matrices  $\mathbf{M}$  of the combinations are more common, which means that processes are combined more commonly with processes from their own process group. Thus, the biased correction value might improve the results by accounting for this imbalance in the data set. The resulting FPR for each tested parametrization is listed in Table 5.3. The lower the FPR, the less combinations are falsely classified and therefore it is supposed that the anomaly detection model is better.

First, from Table 5.3 results that the parametrization is rather stable. Small changes in the applied parameters do not distort the resulting anomaly detection in a significant manner. This is a benefit of the modified Hopfield neural networks since they enable a rather easy and fast parametrization and since users of the anomaly detection model do not need to extensively adjust all parameters. Only in case of higher values for training rate, decay, and reduction of training rate, the model loses in performance regarding FPR.

Second, the error of the 30 Monte Carlo cross-validations is rather small and thus, the order of the training data as well as the split between training and test data

**Table 5.3:** Effect of parametrization evaluated by the FPR using a given 90.0% TPR.

$\alpha$	$\gamma$	$\beta$			
		0	0.1	0.25	0.5
0.01	0	(1.7 $\pm$ 0.4)%	(1.6 $\pm$ 0.5)%	(1.4 $\pm$ 0.4)%	(1.5 $\pm$ 0.4)%
	0.02	(1.6 $\pm$ 0.5)%	(1.5 $\pm$ 0.5)%	(1.6 $\pm$ 0.4)%	(1.4 $\pm$ 0.4)%
	0.04	(1.8 $\pm$ 0.6)%	(1.6 $\pm$ 0.7)%	(1.5 $\pm$ 0.5)%	(1.3 $\pm$ 0.5)%
	0.1	(1.7 $\pm$ 0.6)%	(1.6 $\pm$ 0.5)%	(1.8 $\pm$ 0.8)%	(1.7 $\pm$ 0.8)%
0.05	0	(1.3 $\pm$ 0.5)%	(1.4 $\pm$ 0.5)%	(1.5 $\pm$ 0.5)%	(1.4 $\pm$ 0.4)%
	0.02	(1.5 $\pm$ 0.5)%	(1.7 $\pm$ 0.6)%	(1.8 $\pm$ 0.6)%	(1.4 $\pm$ 0.5)%
	0.04	(1.6 $\pm$ 0.6)%	(1.5 $\pm$ 0.5)%	(1.4 $\pm$ 0.6)%	(1.6 $\pm$ 0.6)%
	0.1	(1.9 $\pm$ 1.0)%	(1.8 $\pm$ 1.1)%	(1.8 $\pm$ 1.0)%	(1.5 $\pm$ 0.7)%
0.1	0	(1.4 $\pm$ 0.5)%	(1.5 $\pm$ 0.6)%	(1.4 $\pm$ 0.5)%	(1.5 $\pm$ 0.6)%
	0.02	(1.8 $\pm$ 0.8)%	(1.7 $\pm$ 0.7)%	(1.4 $\pm$ 0.6)%	(1.6 $\pm$ 0.6)%
	0.04	(1.8 $\pm$ 0.7)%	(1.6 $\pm$ 0.8)%	(1.7 $\pm$ 0.8)%	(1.4 $\pm$ 0.5)%
	0.1	(2.1 $\pm$ 1.1)%	(2.0 $\pm$ 1.1)%	(1.9 $\pm$ 1.1)%	(1.6 $\pm$ 0.7)%
0.25	0	(2.7 $\pm$ 1.5)%	(1.8 $\pm$ 1.3)%	(1.9 $\pm$ 0.8)%	(1.9 $\pm$ 1.0)%
	0.02	(2.0 $\pm$ 1.0)%	(2.1 $\pm$ 1.2)%	(2.1 $\pm$ 1.1)%	(1.7 $\pm$ 0.9)%
	0.04	(2.4 $\pm$ 1.3)%	(2.0 $\pm$ 1.0)%	(1.9 $\pm$ 0.9)%	(1.6 $\pm$ 0.7)%
	0.1	(2.9 $\pm$ 1.5)%	(2.6 $\pm$ 1.7)%	(2.6 $\pm$ 1.9)%	(2.1 $\pm$ 0.9)%

\*Results by Spoor and Weber (2024)

also does not strongly distort the analysis. Thus, the model can be safely applied, and the selection of training data is capable of representing the complexity of the process combination task.

Third, the grid search enables the parametrization of the model. While the grid search might not have found the global optimum of the parameter optimization, the small occurring changes of FPR when changing the parameters indicate that any further improvement of the global optimum might be rather low. In addition, it is assumed that the standard deviation of the cross evaluations is larger than the possible improvement of FPR in the optimum. Thus, the parameters can be quite freely selected in the range of  $0.01 \leq \alpha \leq 0.1$ ,  $0 \leq \beta \leq 0.5$ , and

$0 \leq \gamma \leq 0.04$  since no significant changes of the FPR are computed for this range of parametrization. It should be noted that changing the parametrization within this range might affect the resulting equilibrium and stimulated energy level but will still enable in all cases a selection of useful critical energy value  $H_{crit}$  which separates incorrect and correct process combinations efficiently.

Since the effects of the reduced training rate  $\beta$  and decay  $\gamma$  are not significant, Spoor and Weber (2024) propose to simplify the training procedure by setting  $\beta = 0$  and  $\gamma = 0$  and selecting a positive training rate  $\alpha \leq 0.1$ . Thus, the iterative adjustments of weights in the training procedure simplifies to equation (4.16). However, other data sets or use cases might benefit from these parameters and it should in each case be evaluated in advance if the iterative weight updates are simplified.

Next, Spoor and Weber (2024) apply a training rate  $\alpha = 0.05$  and test the effect of different splits of the training size during the evaluation. Thus, different splits between the training data size and test data size are evaluated using 30 Monte Carlo cross-validations. As prior, an TPR of 90% is selected and the results are tested whether they minimize the FPR. The biased correction value is again applied in this evaluation. The results are given in Table 5.4.

**Table 5.4:** Effect of different training sizes evaluated by the FPR using a given 90.0% TPR.

	Training size to data set size ratio				
	50 %	60 %	70 %	80 %	90 %
$T$	4337	5205	6072	6940	7807
FPR	$(1.5 \pm 0.5)\%$	$(1.6 \pm 0.4)\%$	$(1.6 \pm 0.5)\%$	$(1.5 \pm 0.5)\%$	$(1.3 \pm 0.5)\%$

\*Results by Spoor and Weber (2024)

As visible in Table 5.4, the training size does not affect the results of the anomaly detection in a significant manner. Since all tested training data sizes do not significantly differ in their results, it is assumed that the performance is rather independent regarding this parametrization. Furthermore, it is possible to conduct

a useful anomaly detection using only 4337 combinations as training data and therefore, the iterative training will begin to stabilize prior to reaching this number. Thus, the number of currently used combinations in this evaluation is sufficient for conducting the analysis. Based on this result, Spoor and Weber (2024) select a training size of 90%.

The last relevant parametrization is the selection of the correction value. In the prior analysis, the biased correction value of in equation (4.9) is applied. Using a training rate  $\alpha = 0.05$  with a simplified weight update as per equation (4.16), a training size of 90%, and 30 Monte Carlo cross-validations, the resulting FPR is given in Table 5.4 as  $(1.3 \pm 0.5)\%$  for the biased correction value. If the regular correction value as per equation (4.7) is applied, the FPR is computed as  $(1.9 \pm 0.6)\%$ . Thus, Spoor and Weber (2024) conclude that the usage of the biased correction value increases the performance of an anomaly detection in this use case. It is assumed that the imbalance of common process combinations with other processes from the same group is adequately addressed by the biased correction value.

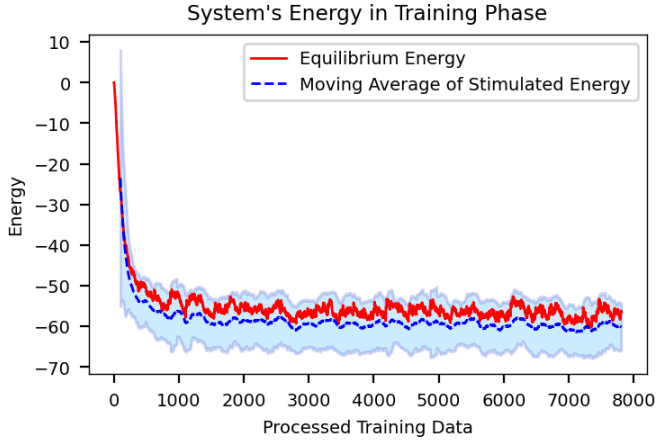
It should be noted that nonlinear effects between all analyzed parameters might exist, and thus, the parameters might be further optimizable by an analysis of the effect between different training sizes, weight update parameters, and bias or unbiased correction values. However, since the FPR is already rather low and the effects of the parametrization are also low, Spoor and Weber (2024) assume no significant effect of a different parametrization. Also, more advanced methods of the parameter optimization such as pattern search could potentially be applied but are assumed to not significantly change the outcome. While the parameters might be further optimizable, the overall anomaly detection is assumed to be rather stable and significant effects are not assumed, which enables a set-up of an anomaly detection model without an extensive hyperparameter optimization procedure. However, other use cases might benefit from a more in-depth parametrization. The final applied parameters as selected by Spoor and Weber (2024) are summarized in Table 5.5 and the described anomaly detection model uses this parametrization as well as the biased correction value.

**Table 5.5:** Applied parameters for the training procedure.

Parameter	Symbol	Value
Training data size	$T$	7,807
Training rate	$\alpha$	0.05
Reduced training rate	$\beta$	0.0
Decay	$\gamma$	0.0

Using the selected parametrization, the iterative training procedure can be visualized by plotting the progression of the equilibrium energy during training as well as the stimulated energy, which can be averaged over the last 100 iterations. Using the moving average, the corresponding 3-standard derivation indicates how much the applied combinations during training differ in their stimulated energy level. Thus, the training procedure can be analyzed as exemplarily demonstrated in Figure 4.9. If the training procedure results in a stable energy level of the averaged stimulated energy and a small standard deviation, the training successfully converges towards a final weight matrix and the anomaly detection model can be applied. The resulting progression of the energy level is given in Figure 5.1.

First, it can be seen in Figure 5.1 that the training procedure stabilizes after around 1000 processed training data during the weight updates. Thus, the amount of training data is sufficient to conduct the analysis and additional training data are not necessary in the case that they correctly represent the underlying use case in its full complexity. This is also shown during the parameter optimization of the training data size where the smaller training sizes do not result in significantly different FPR. Second, the moving average does not fluctuate too much and the 3-standard derivation area is also quite narrow. This indicates that the used process combinations do all share a similar combination logic and no set-up of different classes is necessary in the analysis. The upper edge of the 3-standard derivation area can also indicate a first useful critical energy value  $H_{crit}$  for the resulting anomaly detection.



**Figure 5.1:** Training procedure using the moving average of the stimulated energy over 100 iterations, its 3-standard deviation area, and the equilibrium energy.

In conclusion, the training procedure successfully results in a weight matrix  $\mathbf{W}$  and a biased correction value  $\theta_{biased}$ . These can then be used to evaluate the energy level of new combinations and conduct the following anomaly detection as in equation (4.30).

### 5.1.1.3 Anomaly Detection Results

The anomaly detection conducted here utilized the same parametrization and evaluation procedure as Spoor and Weber (2024) but the weight matrix and the corresponding anomaly detection is re-computed. Thus, the results slightly differ but the following analysis shows that in both analyses the results are within the same order of magnitude and both are showing results which are significant.

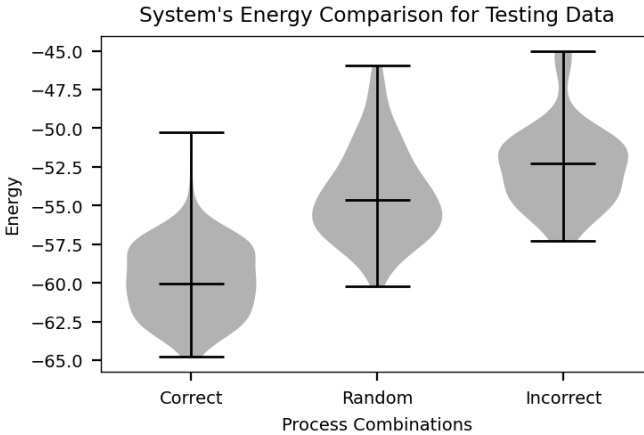
First, the correction value is analyzed and interpreted. The biased correction value is computed using the final weight matrix after the training procedure as  $\theta_{biased} = -0.79$ . A negative correction value indicates that combinations between multiple different process groups are uncommon, *vice versa* a positive

correction value would indicate that combinations between many different process groups are very common. Since this computed value is close to  $-1$ , most process combinations are only between a limited number of process groups and most entries  $m_{ij}$  of the connection matrix are negative. This holds true and most combinations in the analyzed data set have only between one and five process groups applied. Since most weights of the weight matrix are negative, combinations with a large number of connections are more penalized during the energy measurement, which the correction value slightly compensates to create a better comparability between combinations with more and less applied process groups. However, combinations with a large number of connections will still be penalized in a manner which increases the stimulated energy of this combinations and therefore makes them less likely when evaluated using the anomaly detection model. Despite the penalization, it is still possible that process combinations with many different process groups achieve low energy levels in the case that the corresponding individual process combinations mostly occur between two process groups with a positive weight matrix entry  $w_{ij}$  and thus precisely between process groups which are more commonly combined together. *Vice versa*, if multiple rare process group combinations are within a combination, the stimulated energy will be higher and are more likely classified as anomalous.

Next, the performance of the anomaly detection model using modified Hopfield neural networks is analyzed. This is realized by evaluating the energy level of the 867 process combinations, equal to around 10% of the total number of process combinations in the analyzed data set, which are not used during the training procedure. These 867 process combinations should all be classified as correct combinations by the anomaly detection model and thus, should yield a low stimulated energy value. In addition, the 30 incorrect combinations by Spoor and Weber (2024) which were created by domain experts are also evaluated and should be classified by the anomaly detection model as anomalous. These incorrect combinations should yield higher stimulated energy levels compared to the correct combinations. In addition, 867 random process combinations are evaluated. These random process combinations are created by adding a number between two or five different process groups to a combination. If a process

group is added twice, a combination between a process group and itself applies. These random process combinations should yield in average a higher stimulated energy level compared to the correct combinations. However, it is possible that the random combinations simply create a valid process group combination by change and thus, these combinations might be classified as correct and yield a low stimulated energy level.

The correct, random, and incorrect combinations are all evaluated by applying the weight matrix, which resulted from the training procedure using only correct process combinations, and the computed correction value during the energy measurement. The resulting energy levels of each process combination computed using equation (4.6) are displayed as a violin plot in Figure 5.2 which utilizes a kernel density estimation to compute a distribution of the energy levels. The highest and lowest values as well as the median are highlighted in Figure 5.2 in addition to the estimated distribution by the violin plot.



**Figure 5.2:** Distribution of the energy level of correct, random, and incorrect process combinations displayed as violin plot.



As visible in Figure 5.2, the correct process combinations yield lower stimulated energy levels compared to the random and incorrect process combinations. In particular, the estimated distributions of the violin plots of incorrect and correct process combinations differ significantly so that both distributions only slightly overlap at the corresponding edges. Thus, it can be assumed that a useful critical energy level  $H_{crit}$  can be defined to differentiate incorrect and correct process combinations efficiently. The distribution of the stimulated energy of random process combination is also visibly above the distribution of correct process combinations and its median is closer to the incorrect combinations. However, the distributions of the random and correct process combinations overlap and thus, a critical energy level  $H_{crit}$  will not achieve a comparable good separation as between incorrect and correct processes. As earlier discussed, this is the result as random combinations might be correct or at least inconspicuous by change. In addition, by limiting the number of processes within a combination to a maximum of five, unlikely combinations between a multitude of different process groups, which are rarely combined together, are not within the set of random process combinations.

The average stimulated energy level of the correct combinations is computed as  $\overline{H}_{correct} = -59.9 \pm 2.2$  and does differ over  $3\sigma$  from the average of the incorrect combinations which is computed as  $\overline{H}_{incorrect} = -52.5 \pm 2.6$ . Both distributions overlap at around  $1.5\sigma$  of their corresponding standard deviation. This also indicates that the incorrect and correct combinations differ significantly regarding their stimulated energy level. The average stimulated energy level of the correct combination differs from the average energy level of the random combinations computed as  $\overline{H}_{random} = -54.3 \pm 2.8$  only by over  $2\sigma$  and their distributions overlap at around  $1.1\sigma$  of their respective standard deviation. The comparison of the average energy levels confirms the visual interpretation of the violin plot in Figure 5.2.

Since the distributions of the correct and incorrect process combinations differs significantly, it is possible to define a critical energy level which should be capable of separating correct and incorrect process combinations efficiently. As already computed during the parametrization prior to the training procedure, it is possible

to select  $H_{crit} = -55.4$  in order to receive a TPR of 90.0% and a corresponding FPR of 1.3% (cf. Table 5.4).

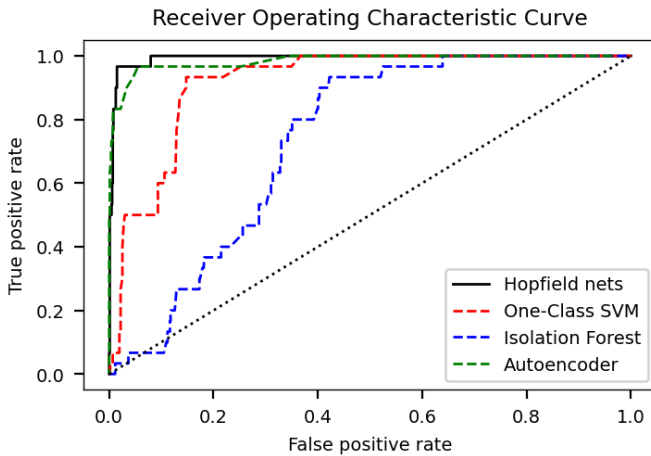
If a higher TPR is preferred, it is possible to select a critical energy value of  $H_{crit} = -56.4$  to achieve a TPR of 96.7% and a corresponding FPR of 1.5%, which is only slightly increased compared to the FPR applied earlier. Depending on the objective of the analysis, whether FP or FN are seen as more severe and should be avoided, the critical energy level can be set accordingly. Independent of the applied critical energy, both resulting TPR and FPR indicate that the modified Hopfield neural network is an efficient anomaly detection model for identifying incorrect process combinations as part of an AI support system for production planning.

In addition, the anomaly detection model of modified Hopfield neural networks is benchmarked with a one-class SVM (Schölkopf et al. 2001), an Isolation Forest (Liu et al. 2008) implemented by sklearn (Pedregosa et al. 2011), and an Autoencoder implemented by Zhao et al. (2019) as commonly applied state-of-the-art AI models for anomaly detection. These AI model can be utilized for the benchmarking since they enable an anomaly detection using data sets without prior labeled classifications and can be trained for use cases with little or no contamination of anomalies within the training data set. As necessary pre-processing, the applied  $12 \times 12$  connection matrices  $\mathbf{M}$  must be converted into Boolean vectors which describe all 78 individual process group combinations. This might limit these models since only Boolean values are used during training. However, it is possible to train all models using the same training data set as used for the modified Hopfield neural network.

The Autoencoder by Zhao et al. (2019) is parametrized using four layers with a number of 64, 32, 32, and 64 neurons of the corresponding layer. The hidden layers use a ReLU activation, output layers use a sigmoid activation. The Autoencoder uses 100 epochs and a batch size of 32. In addition, a contamination rate of 1% is assumed. Other parameters are set according to the proposes default values by Zhao et al. (2019). The one-class SVM uses an RBF kernel, a gamma of  $\frac{1}{78\sigma^2}$ , and for the other parameters the default setting by sklearn. In case of the Isolation

Forest, the contamination is automatically computed as proposed in the model and the other parameters are also set as per default by sklearn. The parametrization for all models was prior to the analysis tested, evaluated, and roughly optimized using a grid search. The results using this described parametrization show the best and most stable performance from all tested models with different parameters.

The benchmarking using a receiver operating characteristic (ROC) curve is given in Figure 5.3 showing the individual curves for the modified Hopfield neural network, the one-class SVM, the Isolation Forest, and the applied Autoencoder. The highlighted diagonal line shows the performance as if the classification of a process combination as an anomaly had been made at random.



**Figure 5.3:** Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection task.

Figure 5.3 shows that the applied modified Hopfield neural network performs the best compared to the other models. However, the Autoencoder shows an overall comparable performance but yields a slightly lower TPR for an applied FPR between 1% and 5%. Thus, the modified Hopfield neural network performs best in particular in the most relevant region of the ROC curve of this use case.

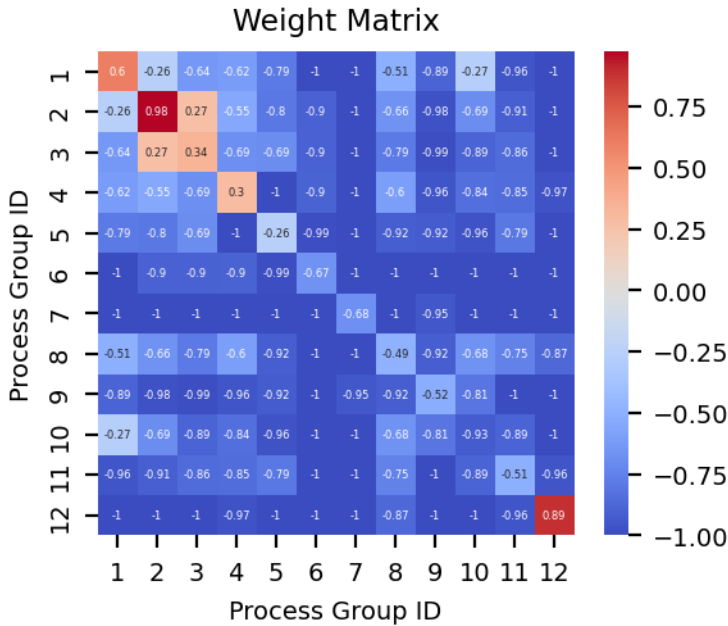
The performance of the one-class SVM and Isolation Forest is also significantly exceeded by the modified Hopfield neural network. One reason for the comparably bad performance by the one-class SVM and the Isolation Forest might be the use of the Boolean vector containing the combinations and the fact that no anomalous combinations are provided within the training data. However, overcoming these limitations is of particular importance in the use case of production planning and this shows that the described modified Hopfield neural network improves the current state-of-the-art for these specific use cases.

While the Autoencoder performs only slightly worse, these types of models are often described as "black box" approaches and are unfortunately only difficult to interpret for domain experts using them. Thus, a modified Hopfield neural network provides domain experts with an explainability which an Autoencoder cannot provide, which is another benefit setting the proposed AI model apart from the state-of-the-art models. The explainability as well as the recommendation procedure is described in detail in the following section and further highlights the benefits of modified Hopfield neural networks.

#### **5.1.1.4 Recommendations and Explainability of Results**

As discussed priorly, the anomaly detection model of a modified Hopfield neural network benefits from a high explainability and derived recommendation based on the interpretation of the weight matrix. First, it is possible to analyze the weight matrix and conclude based on the weight entries  $w_{ij}$  what process group combinations in particular result in high (or low) stimulated energy levels. Thus, the weight matrix can be used to evaluate individual stimulated energy measurements. The weight matrix  $\mathbf{M}$  which resulted from the training procedure of this analysis is visualized as heatmap plot in Figure 5.4.

By using the equations (4.31) and (4.32), it is possible to compute the impact of any combination within a process combination set  $\mathcal{C}$ . The positive values close to +1 in the weight matrix (highlighted in reddish colors in the heatmap of Figure 5.4) indicate very likely and correct process combinations while the



**Figure 5.4:** Resulting weight matrix after the training procedure visualized as a heatmap plot.

negative weight values close to  $-1$  (highlighted in bluish colors) indicate very unlikely combination which result in higher stimulated energy levels. If multiple combinations between negative weight values are within a combination, this combination is most likely anomalous and classified as incorrect by the anomaly detection model.

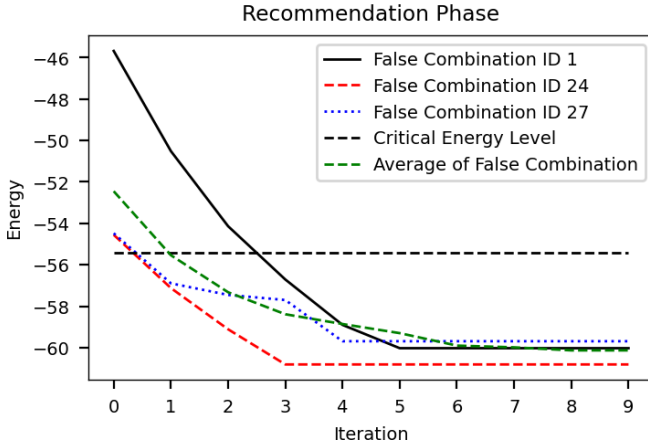
Using this approach, Figure 5.4 shows that in particular combinations of process groups with themselves, e.g., process group 1, 2, 3, 4, and 12, are very likely and will result in lower stimulated energy levels. Combinations between process groups 2 and 3 are also very likely. Some process groups are very uncommon and indicate rarely used processes, e.g., group 6 and 7, and thus, these process groups are most likely paired only with themselves. In particular group 7 is in most cases only paired with itself and sometimes with process group 9. On the other hand,

the very high values of the combination of group 2 and group 12 with themselves indicated that these process groups mostly have to be paired with themselves and combinations with only one process from these groups will receive higher stimulated energy levels and thus, will be evaluated as more anomalous. Since the values are overall more often low than high, the correction value is negative (as priorly calculated) and combination sets with less different process groups are more common than sets with multiple different groups.

In summary, the weight matrix can be applied for each individual analyzed process combination  $\mathcal{C}$  and the impact of a combination within the set (or not within the set) can be computed. Thus, the energy level of any set can be manually computed easily and the reasons for a high or low stimulated energy evaluation can be directly derived from this computation.

In addition, the described approach for a recommendation procedure can be utilized for the analyzed 30 incorrect process combinations to create correct process combinations. For any process combination set  $\mathcal{C}$ , each possibility of either removing or adding a process group is tested in each iteration. The new resulting set  $\mathcal{C}^*$  with either one additional process group or one removed process group is selected which reduced the energy level the most as per equation (4.33). Then, the next iteration tests again the removing or adding of another process group. The energy level of the analyzed combination set  $\mathcal{C}$  is reduced till a combination with a stimulated energy level lower than the critical energy threshold  $H_{crit}$  is reached. Using this approach, all 30 process combinations are evaluated and the resulting stimulated energy level changes per iterative step are visualized in Figure 5.5 for three exemplary incorrect combinations and the mean energy of all 30 processes using a critical energy level of  $H_{crit} = -55.4$  as applied in the prior analysis to achieve a TPR of 90%.

Figure 5.5 shows that after only a few iterative adjustments, most process combinations are below the applied critical energy level  $H_{crit}$  and thus, are no longer classified as anomalies. Most incorrect combinations can be corrected by a single adjustment which is consistent with the assessment of the domain experts who created these incorrect process combinations especially for the testing procedure.



**Figure 5.5:** Energy levels after multiple iterations of recommendation with single activations or deactivations of processes per iterative step.

However, the specific adjustments depend on the underlying process combination and thus, highly differ regarding added or removed process groups and changes of energy level. In addition, the lowest possible energy level after all adjustments differs between the three exemplary process combinations. Thus, the resulting recommendations differ for each analyzed process combination  $\mathcal{C}$  since only one process group is changed per iterative step and the final recommendations are not necessarily global minima of the stimulated energy level but local minima depending on the starting configuration.

The individual adjustments per iterative step of Figure 5.5 of the corresponding process combination set  $\mathcal{C}$  for the three exemplary incorrect combinations are listed in more detail in Table 5.6. The critical energy level of  $H_{crit} = -55.4$  is used as threshold whether the combinations are classified as correct or incorrect.

Since the correction value  $\theta$  is negative, process combinations  $\mathcal{C}$  with many active connections between process groups will most likely result in recommendations to reduce the number of process groups within a combination. Thus, the stimulated energy level of most process combinations will be optimized by decreasing the

**Table 5.6:** Recommended process group adjustments per iterative step.

Iteration	Process Combinations		
	ID 1	ID 24	ID 27
0	6, 7, 10, 11, 12*	1, 3, 12*	6, 8, 10*
1	6, 10, 11, 12*	1, 3, 12, 12	8, 10
2	10, 11, 12*	1, 1, 3, 12, 12	10
3	10, 11, 12, 12	1, 1, 12, 12	1, 10
4	11, 12, 12	1, 1, 12, 12	1, 1, 10
5	12, 12	1, 1, 12, 12	1, 1, 10
...	12, 12	1, 1, 12, 12	1, 1, 10

\*The energy of the combination is above the critical value.

number of individual active connections  $m_{ij}$  since most weights  $w_{ij}$  as highlighted in Figure 5.4 are also negative. Exceptions might occur in cases of the loop connections, e.g., of process group 2 and 12, where adding another process might result in lower stimulated energy levels if the number of other process groups included in the analyzed process combination is already small. This results from loop connections yielding high weight values  $w_{ii}$  for most process groups. Thus, in most iterative recommendation steps listed in Table 5.6, process groups are removed from the combination sets  $\mathcal{C}$ . Exceptions occur in Table 5.6 in cases where an additional loop connection is created since these connections yield high weight values  $w_{ii}$  for most process groups.

The exemplary incorrect process combination ID 1 yields at the beginning of the analysis a notably high stimulated energy level. The process combination set  $\mathcal{C}_1$  contains the process groups 6, 7, 10, 11, and 12. By analyzing the weight matrix visualized in Figure 5.4, it is directly visible that in particular the inclusion of process group 6 and 7 affect the stimulated energy level since both process groups are very uncommonly combined with other process groups. Thus, the first two steps of the recommendation are the exclusion of these two process groups which is also reasoned by the direct analysis of the weight matrix. Next, the process group 12 is in most cases combined with itself and thus, the next iterative



adjustment is the addition of another process from process group 12 creating a loop in the underlying network. Since the weight matrix entry  $w_{12,12}$  is close to +1, this combination highly affects the energy measurement. Hence, the adjusted process combination yields a stimulated energy level below the critical energy level  $H_{crit}$  and the resulting process combination set is classified as correct.

The process combinations ID 24 and 27 follow a similar rationale. In case of process combination ID 24, the adding of process group 12 creates a loop and thus, the weight entry  $w_{12,12}$  improves the energy assessment so that the recommended process combination is classified as correct. In case of process combination ID 27, the removal of process group 6 reduces the stimulated energy level so that the recommended combination is also classified as correct.

In all three cases it is possible to further adjust the energy level even after the reaching the critical energy level. As seen in Figure 5.5, the resulting optimal combinations based on the initial starting combination differ and thus, the resulting energy levels also differ.

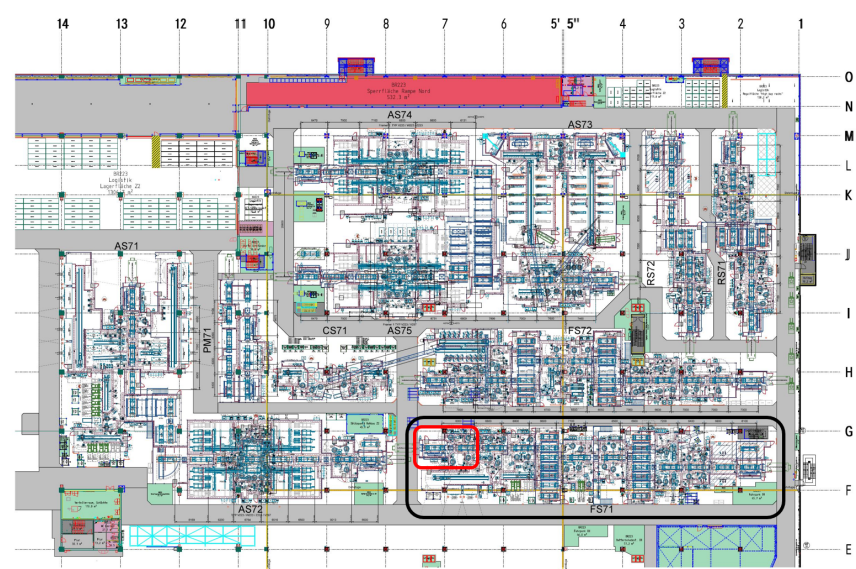
#### **5.1.1.5 Analysis of an Exemplary Process Combination**

Spoor and Weber (2024) demonstrate the capabilities of the modified Hopfield neural networks in a practical example of one correct process combination, which is currently used in an automotive manufacturing plant. Spoor and Weber (2024) select as an example a more unique process combination in order to show the resulting energy evaluation. This process combination is within the production line FS71<sup>1</sup> and applied in the assembly process for mounting side panels. The process combination analyzed by Spoor and Weber (2024) is within the first manufacturing cell of the production line FS71 through which the assembled body frame of a car passes. The manufacturing line FS71 is selected due to its more unique layout and since it encompasses different applications as well as special welding, riveting, gluing processes, and multiple tool exchanges. This

---

<sup>1</sup> FS71 is the internal identifier of the production line by the Mercedes-Benz AG.

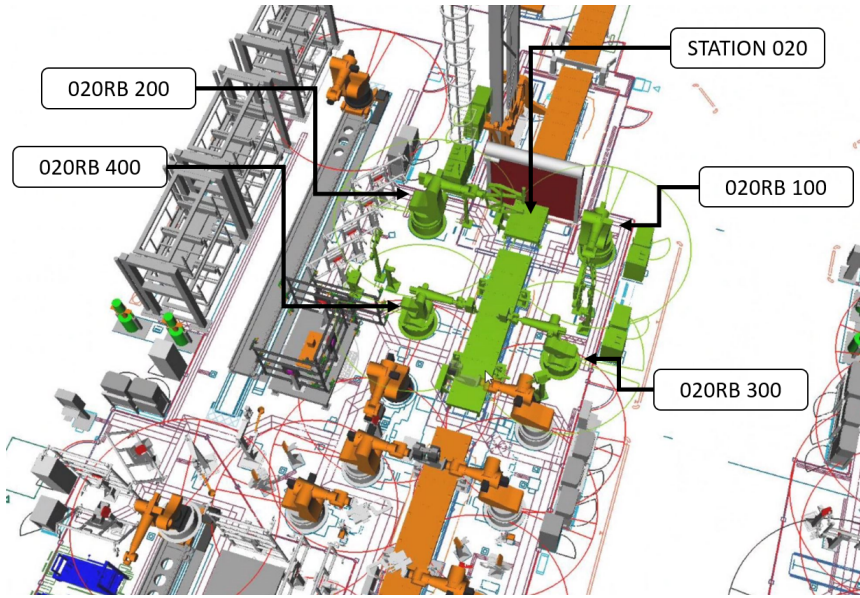
specific manufacturing line is therefore often used as internal benchmark and testing ground of new ideas and concepts of Mercedes-Benz AG’s Digital Factory research because of its higher complexity and differing processes. Figure 5.6 shows the positioning of the manufacturing line FS71 within the whole car side panel assembly plant. FS71 is colored in red.



**Figure 5.6:** Layout of a side panel assembly plant. FS71 is located in the lower right of the map. The position of the exemplary process is marked in red (Spoor and Weber 2024).

To further analyze the process combination, the corresponding manufacturing is visualized using a digital plant layout in Figure 5.7. In particular, the manufacturing cell includes four robots, which are named 020RB 100 up to 020RB 400, and a conveyor belt, which is named Station 020. The four robots and the conveyor belt are the resources of this first manufacturing cell of the production line FS71.

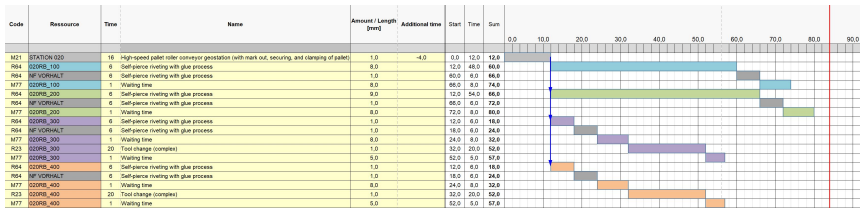
The analyzed process combination running in the manufacturing line FS71 within this first manufacturing cell is visualized using a process chart in Figure 5.8. As introduced by Spoor and Weber (2024), the process begins with a delivery via the



**Figure 5.7:** Layout of the first cell of the manufacturing line FS71. The resources are marked in green. Robots and conveyor belt are named with arrows (Spoor and Weber 2024).

conveyor belt of a corresponding car body. Next, the process chart encompasses a self-pierce riveting process, which in addition includes a gluing application by all robots. While the riveting process conducted by robots 020RB 100 and 020RB 200 is still ongoing, the robots 020RB 300 and 020RB 400 already finish this riveting process. Thus, these two resources come to a prior halt in order to start a tool exchange, which is described by planning experts as a more complex and non-standard tool exchange when compared to other processes. The tool exchange is conducted in order to enable a subsequent process. After this tool exchange is completed, 020RB 300 and 020RB 400 have a waiting time until 020RB 100 and 020RB 200 terminate the before described self-pierce riveting process. The process ends with another waiting time.

The discussed process combination includes four different process groups as set-up by domain experts. The corresponding process codes are named M21, R64,



**Figure 5.8:** Analyzed process chart with 020RB 100 marked in blue, 020RB 200 marked in green, 020RB 300 marked in violet, and 020RB 400 marked in orange (Spoor and Weber 2024).

M77, and R23. Process M21 is included in the combination once and comprises a delivery of a car body using the conveyor belt. Process M21 is part of process group 11, which encompasses the applications without the use of robots but with handling of ramping and clamping procedures. Next, process R64 is the conducted self-pierce riveting and is included within the process combination eight times. This process is part of process group 3. Group 3 encompasses so-called technical robot main processes, which are mainly welding applications. The process M77 is the conducted waiting times and included six times. M77 is part of group 8 which includes preprocessing tasks, in particular all preprocessing without any actions by robots or a handling of components. Lastly, process R23 is the discussed special tool exchange of the robots 020RB 300 and 020RB 400. This particular interesting process R23 is included twice. R23 belongs to group 2. Group 2 includes the processes which are running on robots but do not include a processing of the corresponding component.

For a further analysis, the process combination is set-up in a manner which allows an analysis using the described methodology. Since it is only important if a process is included once, twice or more, or is not included, the resulting process combination is written as  $\mathcal{C} = \{2, 2, 3, 3, 8, 8, 11\}$ . This process combination could also be visualized by a network graph similar to the example in Figure 4.5. In a network visualization, the edges (2, 3), (2, 8), (2, 11), (3, 8), (3, 11), and (8, 11) would be included as well as the loops (2, 2), (3, 3), and (8, 8). For the processing of this network, a connection matrix  $\mathbf{M}$  can be created. This connection matrix yields positive values of +1 for the entries  $m_{2,3}, m_{2,8}, m_{2,11},$

$m_{3,8}$ ,  $m_{3,11}$ ,  $m_{8,11}$ ,  $m_{2,2}$ ,  $m_{3,3}$ , and  $m_{8,8}$  as well as the symmetrical counterpart entries of the matrix. All other entries yield the negative value  $-1$ .

This connection matrix  $\mathbf{M}$  is applied to the from the training procedure resulting weight matrix  $\mathbf{W}$  in Figure 5.4 and the corresponding computed biased correction value is used in order to measure the stimulated energy level as per equation (4.6). The resulting stimulated energy is given as  $H(\mathcal{C}) = -58.8$ . This stimulated energy is lower than the prior used critical energy values of  $H_{crit} = -55.4$  and  $H_{crit} = -56.4$  and thus, the applied process combination would be truly classified as correct and is a true negative evaluation. However, the stimulated energy level is slightly above the mean energy level of the correct processes during the training of  $\overline{H}_{correct} = -59.9 \pm 2.2$  and also in a higher segment of the distribution of the violin plot of Figure 5.2. Since the FS71 is a more unique production line, a slightly higher stimulated energy compared to the mean value of correct combinations is even expected. Thus, the stimulated energy level also provides a rough estimation of the complexity or rarity of a process combination.

Using the weight matrix  $\mathbf{W}$  in Figure 5.4, the reasons for the energy evaluation can be discussed in more detail. This evaluation can be quantified by the computed impact of equation (4.31), (4.32), and (4.33) but also roughly estimated by a visual analysis of the weight matrix. First, the inclusion of the combinations network edges (2, 2), (2, 3), and (3, 3) are lowering the energy level since the weight matrix has positive entries for the weights  $w_{2,2}$ ,  $w_{2,3}$ ,  $w_{3,3}$ . However, the inclusion of the process combinations (2, 11) and (3, 11) are very rare as indicated by the negative weight matrix entries  $w_{2,11}$  and  $w_{3,11}$ . Thus, these sub-combinations of the analyzed process combination are increasing the energy level. In a similar manner, the combinations (2, 8), (3, 8), and (8, 11) are also increasing the energy level. In addition, the loop of (8, 8) is rare and yields a negative weight entry  $w_{8,8}$ . Thus, the energy level could be further reduced by, e.g., removing the one process M21 of process group 11 or all eight instances of process M77 of process group 8 from the combination set  $\mathcal{C}$ .

However, not only the included combinations are affecting the energy levels but also combinations not included. Since most other combinations, e.g., (2, 12) or

(4, 10), are inactive and are not included in the set  $\mathcal{C}$ , the corresponding mostly negative weight entries are reducing the energy level. *Vice versa*, since the loop connection (1,1), (4,4), and (12,12) which all yield positive weight entries are not within the combination set  $\mathcal{C}$ , these not included combinations will increase the stimulated energy level. However, the effect of this loop combinations is reduced since, e.g., the inclusion of loop (12,12) might at first glance decrease the energy level but would also result, among other new edges, in the active edge (2,12) which in turn increases the energy level. Thus, the summation of equation (4.33) is important to analyze the full impact of excluding or including an edge.

To summarize, the rarer aspects of this analyzed process combination result in an increase of the stimulated energy but the overall assessment indicates correctly a true negative process combination. The weight matrix can be used to interpret the results in more detail and the equations (4.31), (4.32), and (4.33) can in addition quantify this analysis.

#### 5.1.1.6 Embedding into the Digital Factory

Lastly it is important to analyze how this use case relates to the introduced methodology. Thus, the use case is discussed analogously to the described objectives, design, and embedding of the methodology introduced here.

First, the overall objectives of this use case are the detection of incorrect process combination and the recommendation of adjustments in a transparent and interpretable manner so that the planning experts can manually steer the analysis. The anomaly detection corresponds to objective (1a) and the recommendation corresponds to objective (1c) as listed in Figure 4.2. Since this recommendation process should also enable manual adjustments and evaluations by domain experts, the objective (1b) in Figure 4.2 is covered by the potential detailed analysis of the weight matrix by planning experts which are adjusting a process combination or reconstructing the recommendation. Since the task is an anomaly detection of incorrect process combination, the classification tasks of objective (2a) and (2b) are not covered by this use case.

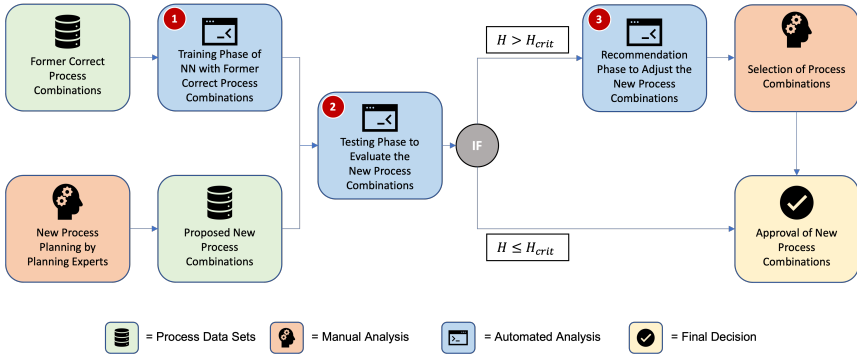
In more detail, the use case is an implementation of inference procedure step (1), (2), (3), (4), and (5) of the support system design in Figure 4.3. The step (3) classification of the inference procedure is not included since the use case focuses on the anomaly detection of incorrect process combinations, as discussed for the objectives. The process groups are developed using an existing ontology of the Mercedes-Benz AG and the connection matrices are set-up by an analogous procedure as in Figure 4.4.

Regarding the core functionalities of the support system in Figure 4.3, the used ontology is the first functionality (A) of the support system. The computed weight matrix corresponds to the functionality of the (B) normal model since the weight matrix encompasses a quantification of the normally applied process combinations. This is the result of the training using only correct process combinations. The (D) similarity metric is the application of the stimulated energy evaluation as in equation (4.6) since this is a direct measure of the correctness based on the weight matrix. The (E) decision logic is the selection, evaluation, and application of the critical energy level to separate correct and incorrect process combination. Lastly, the (F) case database is the recommendation phase based on the known processes as applied in the weight matrix. The core functionality (C) is more relevant for classification tasks and thus not included. The functionality (F) is also more detailed in classification tasks since this allows the model to compare specific configurations and not only abstractions using a weight matrix.

In addition to the analysis of correct and incorrect process combinations, Spoor and Weber (2024) also provide a recommendation on how the process planning evaluation can be embedded into the production planning as a business process. This proposal is given in Figure 5.9.

First step of the process planning evaluation in Figure 5.9 by Spoor and Weber (2024) is the training of the modified Hopfield neural network using the former correct processes as input data. This corresponds to step (4) of the procedure for an anomaly detection in Figure 4.6. The data preprocessing of step (1) to (3) in 4.6 relates to the visualized input data in Figure 5.9 which are used in the training procedure. Step (5) in Figure 4.6 corresponds to the testing procedure





**Figure 5.9:** Proposal for an embedding of the process planning use case into the business processes of production planning by Spoor and Weber (2024).

in second step of Figure 5.9. This testing uses as inputs the trained model, i.e., the weight matrix, and the applied process combination which is developed by planning experts and is also transformed into a connection matrix. The third step of the recommendation phase exceeds a pure anomaly detection and is therefore not covered by Figure 4.6. The recommendation phase is only applied if a process combination is anomalous and exceeds the critical energy level. In this case, the recommendations are provided and the planning expert is capable of adjusting the process combination supported by the quantified impact of equations (4.31), (4.32), and (4.33) and the provided recommended process.

Overall, this highlights that the presented use case by Spoor and Weber (2024) can be implemented according to the methodology introduced here. While the use case of Spoor and Weber (2024) is a proof-of-concept, the accordance of the use case and the methodology of this doctoral thesis proves that an embedding into the tool and software landscape can be analogously achieved as described in the methodology section.



## 5.1.2 Simulation of Machinery Sequencing with Synthetic Data

The second use case addresses the field of simulation of machinery sequences in manufacturing. In particular, the task of generating highly realistic synthetic data for new sequences based on real-world data is analyzed and a new methodology based on the introduced modified Hopfield Neural Network is proposed. This methodology is introduced by Spoor et al. (2024) and capable of improving simulations by increasing the quantity of test data while also maintaining a high quality and degree of realism in the synthetic data set. A former approach for data generation by Matthes et al. (2023) is enhanced by implementing firstly, a filtering of the generated synthetic data using the anomaly detection capabilities of modified Hopfield Neural Network and secondly, a validation and comparison of the synthetic data based on the energy distribution of the stimulated energy of the synthetic data set. This following section is based on the work by Spoor et al. (2024) and adds additional context and details to the analysis.

### 5.1.2.1 Use Cases Description

The operations and maintenance of modern production systems require the ability to run simulations since they are cost-efficient, yield low execution times, can provide insightful results, and do not add additional risks to the operation of the system, compared to testing procedures using the real manufacturing system (Mourtzis 2020). Simulations became therefore an integral part of the planning, commissioning, operation, and modernization of production systems. However, simulations require a large amount of real-world data records as input in order to generate realistic results. The data amount becomes in particular important for the development of data-driven methods in manufacturing such as methods from the field of AI and ML.

As already discussed in section 4.1, the amount of available data today has significantly increased but a distributed data ownership or prohibitive data collection

result in a limited capability to run simulations using a large amount of data. Arinez et al. (2020) and Wuennenberg et al. (2023) describe the difficulties for researchers to obtain large amounts of data due to an operational prohibitive wide-scale data collection and restrictions of data access and exchange due to security concerns. While open-access libraries of data sets suitable for simulation purposes exist, which are offering sufficiently comprehensive data sets and are validated for correctness, these data sets are often solely available for certain use cases and only yield a specific set of features with a specific distribution of values. For example, using a different technology during a very similar production process might result in different features and makes an open-access data set less suitable for the comparison with the underlying use case a practitioner is analyzing. As a result, practitioners in production planning can only in limited cases utilize public open-access data sets for their analyses but must rather conduct their own data set collection, which can be costly and time-consuming.

The generation of synthetic data however circumvents a costly and sometimes impossible collection of a real-world data set and can additionally be tailored to the analyzed use cases' specific requirements, in particular to the distribution of values and utilized features. Furthermore, synthetic data sets are also comprehensive and are not limited in the scope or amount of data (Völker et al. 2001). Synthetic data sets are also useful to tailor and improve the usability of a collected real-world data set by two mechanisms: firstly, they can improve a real-world data set's quantity by generating more instances to enlarge available data sets and secondly, they can improve the quality of the collected data by generating more relevant cases, e.g., rare error events which are important during simulation runs, and thus enhancing an available data set. Further benefits of the usage of synthetic data are privacy regulations compliance, an ability to share data across institutions which would otherwise be restricted, and a potential sped-up of development cycles by mitigating prior time-consuming data collections. Libes et al. (2017) summarizes the use cases of synthetic data as the training of ML models, the verification of computer models and their mathematical equations and solutions, the validation of simulations and how accurately a simulation can represent real-world instances, improvement of real-world instances by knowledge

generated with synthetic data, and the augmentation of real-world data sets, e.g., by replacing missing data entries within a collected data set. However, synthetic data sets are only particularly valuable if the generated synthetic data are capable of reflecting the real-world properties of a production system accurately, e.g., machinery sequences in the presented use case. If a synthetic data set meets these criteria, it can be highly valuable for the validation of production planning approaches through sophisticated simulations.

Despite the overall usefulness of realistic generated data to enhance or enlarge real-world data sets, the capability by a data generator to accurately reflect the complexity of a real-world data set, and thus also the complexity of real-world production systems, is still a challenge in the field of production planning. Production systems yield multiple interdependencies and patterns inherent in their environments which must be accurately reflected so that a synthetic data set becomes useful. Production data might yield a wide range of attributes and relationships specific to the manufacturing process, i.e., machine sequences, production schedules, and material requirements. Thus, commonly applied tools such as simple data anonymization libraries cannot capture potential nuanced interdependencies of production systems. Data generators must therefore leverage algorithms which can replicate these complex correlations, patterns, and relation between the multiple features of a real-world production system. Thus, authors such as Fernandes et al. (2020) and Adolphy et al. (2015) explicitly focus on the creation of realistic, synthetic copies of real-world production data. Mannino and Abouzied (2019) use state-of-the-art approaches from ML and statistical approaches considering domain knowledge to generate synthetic data sets.

In order to ensure high quality results, a synthetic data set must also be validated for its realism in a final step after the data generation procedure. This field of research receives limited attention and very little research in the development of synthetic data validation methods is conducted, whether a data set is an accurate representation of the real-world instances. As with the generation itself, also the validation of the degree of realism of synthetic data sets is difficult since generated data sets are often large in size and thus only aggregated KPIs, which are describing the distribution of one or multiple features, can be efficiently

analyzed. However, a KPI describing an individual distribution or a mixture of distributions might lack the insights into the complex interdependencies, patterns, and correlations of production system data and thus, KPIs might be misleading and cannot guarantee plausibility. Krockert et al. (2021) therefore state that data generators often produce data which represent real-world data sets regarding their KPIs defining their structure, but on the other hand may lack the necessary degree of realism. Despite this limitation, Krockert et al. (2021) prove that machinery sequences can be sufficiently simulated by using a data generator. On the other hand, a detailed evaluation without an aggregation procedure might be too time-consuming and might require additional manual input.

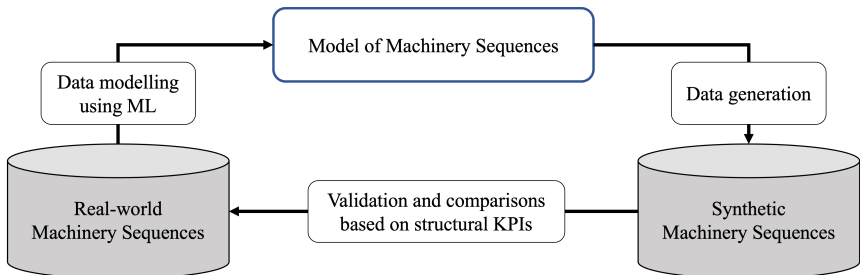
In a related use case of machinery sequences, Matthes et al. (2023) aim at an evaluation of authenticity of machinery sequences in production planning using aggregated statistical KPIs, i.e., the average length of a sequence computed via the arithmetic mean and the relative frequency of occurring subsequences, which can be compared to a probability distribution function. This aims at the assessment of specific patterns within the data sets and their distribution and frequency. Other statistical approaches utilize the Jaccard similarity (Jaccard 1901) measuring the overlap, the Levenshtein distance measuring similarity between sequences, and the K-gram indexing, which aims at the identification of common subsequences. Overall, similarities measures can be used to quantify the differences of synthetic data and real data and to create insights into differences between both. These statistical methods are most commonly used in fields such as text processing but can be adjusted for the here described validation purposes of machinery sequences in production systems.

As discussed by Spoor et al. (2024) and to the author's best knowledge, developing validation procedures which are testing whether a synthetic data set is a suitable representation of the underlying real-world use case, is an under-researched field in the area of simulation theory and applications. As a solution, Spoor et al. (2024) propose a fast and comprehensive data validation without the use of aggregated KPIs. Spoor et al. (2024) aim for a validation method which is agnostic of the used data generator model and can be integrated into all data generation approaches in order to test a synthetic data set's plausibility based on a limited amount of

real-world data as benchmark. Additionally, Spoor et al. (2024) add a correction procedure to detect and filter non-plausible data records. The methodology by Spoor et al. (2024) is based on the data generation approach by Matthes et al. (2023) and utilizes an embedding of modified Hopfield neural networks. These are used in this methodology as the filter mechanism of non-plausible data records and to validate different data generation approaches with each other.

### 5.1.2.2 Proposed Methodology of Use Case

The method by Spoor et al. (2024) is based on a data generation methodology by Matthes et al. (2023). In this approach, Matthes et al. (2023) generate synthetic data sets using a Bayesian network, a transformer network from the field of DL architectures, and two statistical approaches. For the validation and comparison of the generated data sets, Matthes et al. (2023) utilize aggregated KPIs, which are limited by their degree of realism and granularity. As a result, the methodology by Matthes et al. (2023) is limited by not evaluating all interdependencies, patterns, and correlation of data records in detail and thus, a comprehensive validation whether all machinery sequences of the synthetic data set are accurate representation of the real-world production system considering quality and quantity is not possible. The initial methodology by Matthes et al. (2023) is given in Figure 5.10.



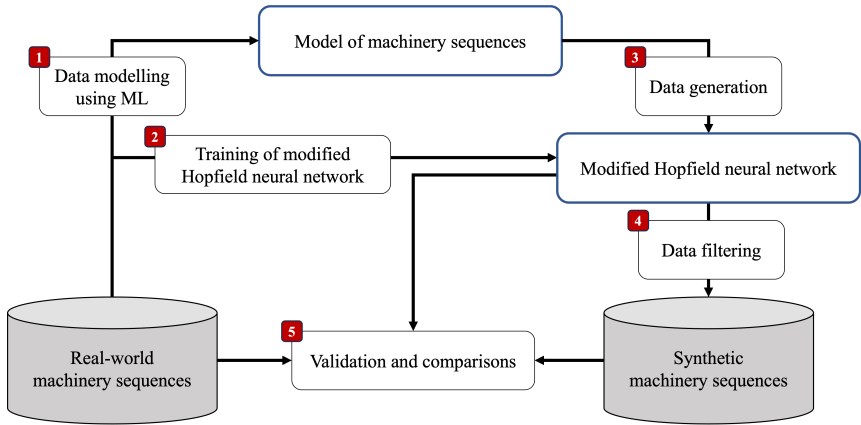
**Figure 5.10:** Schematic methodology to generate, validate, and compare synthetic data as described by Matthes et al. (2023).

Spoor et al. (2024) build upon this approach by adding the capabilities of modified Hopfield neural networks as proposed by Spoor and Weber (2024). The introduction of this model can improve the methodology by Matthes et al. (2023) in two ways: firstly, modified Hopfield neural networks can be utilized with their anomaly detection capability in order to filter the generated data and exclude data records which are not realistic and secondly, by using the classification capabilities, the model can be utilized for the validation and comparison of multiple data sets, i.e., the data generation methods by Matthes et al. (2023). Machinery sequences are suitable for an analysis with modified Hopfield neural networks since sequential data can be modeled as directed multigraphs with loops. Compared to the aggregated KPIs, the usage of modified Hopfield neural networks enables an in-depth analysis of each data record. Lastly, the same real-world data set, which is used to set-up the synthetic data sets, can also be applied for the training procedure of the modified Hopfield neural network. A further advantage of modified Hopfield neural networks to other outlier and anomaly detection model in this use case is that the synthetic data generators as well as the modified Hopfield neural networks only require correct data as input. Thus, no additional labeled incorrect data records or outliers must be created to train the model. The proposed methodology by Spoor et al. (2024) is given in Figure 5.11.

The new methodology can be described by five steps marked in Figure 5.11. In step (1), the real-world data set is used for the set-up of the synthetic data generator models, which are using ML methods and statistical approaches. This data modeling of machinery sequences is the base for the later generation of synthetic sequences by data generators.

In the second step (2), the same data set is utilized in the training of the modified Hopfield neural network. Since only true negative instances are required, the same data set used for the model of sequences as base for the data generator containing only validated real-world data records, i.e., correct real-world data without incorrect examples, is sufficient for this purpose.

In the third step (3), the data generators use the model of machinery sequences to generate synthetic data sets. The mechanism behind the generation process (and



**Figure 5.11:** Schematic methodology to generate, validate, and compare synthetic data utilizing a modified Hopfield neural network for validation and filtering by Spoor et al. (2024).

as well as behind the modeling of the sequences) are briefly introduced in the subsequent section.

The fourth step (4) contains the filtering approach using the anomaly detection model of the trained modified Hopfield neural network. Each generated synthetic data set is individually filtered and corrected by excluding unlikely, uncommon, or anomalous data records.

As the fifth step (5), the validation procedure takes place. The objective of this step is to compare which generated data set comes closest to the underlying real-world data set and thus, is the most accurate representative of the production system. Based on this comparison, the most suitable synthetic data set can be selected for the subsequent simulation tasks. For this step, the classification model of the modified Hopfield neural network is utilized.

In summary, the proposed methodology by Spoor et al. (2024) filters and recommends the most suitable data set which, compared to the other synthetic data sets, represents the real-world data set most accurately. This comparison can be conducted between different data generator models but also for the same data generator model with a different parametrization. Individual less-realistic data records

are excluded from the comparison and can be filtered. Thus, the methodology focuses on structural realism of the data, including interdependencies, patterns, and correlations, as well as on the realism of individual data records.

### 5.1.2.3 Synthetic Data Generation Methodology

For the simulation of machinery sequences in production planning, transition matrices, which can also be displayed via graph networks, are a common model for an abstract machinery sequence description (Schuh 2007). A transition matrix yields the probability of a transition from a machine  $i$  to a machine  $j$  within a sequence as its elements  $\pi_{ij}$ . If there exists a number of  $V$  different machines which are simulated, the transition matrix is a  $V \times V$  square matrix. An exemplary transition matrix for 3 machines is given in Figure 5.12.

		Machine j		
		M1	M2	M3
Machine i	M1	0.21	0.54	0.25
	M2	0.31	0.33	0.36
	M3	0.56	0.21	0.23

**Figure 5.12:** Exemplary simple transition matrix (ST) of 3 machines for the simulation of sequences.

The application of these transition matrices results in a stochastic process called Markov chain. Transitions in a Markov chain are characterized by a stochastic process where the outcome of the next state of a sequence is only dependent on the current state but not on any prior states, i.e., the probability of the next machine in a generated sequence is only dependent on the current machine as seen in Figure 5.12. For the purpose of synthetic data generation, Markov chains



were until recently a state-of-the-art benchmark and also widely used in practice but are lately superseded by neural network-based approaches (Bauer et al. 2024).

The transition probabilities  $\pi_{ij}$  are computed based on a given real-world data set. The generation of sequences is achieved by sampling over the transition matrix using a priorly defined sequence length, which is also sampled based on the distribution of sequence lengths from the real-world data set used for setting up the data generator. In this section, the data generator using a simple transition matrix is abbreviated as ST.

In order to circumvent the selection of the sequence length beforehand and to integrate this relation into the transition matrix, a source and sink condition is added. This approach is called transition matrix with added source and sink conditions (abbreviated as TASS). The sampling process starts at the source row (Q) and stops when it reaches the sink row (S). The sequence length is therefore incorporated into the matrix itself by the probabilities for reaching the sink. An exemplary transition matrix is given in Figure 5.13.

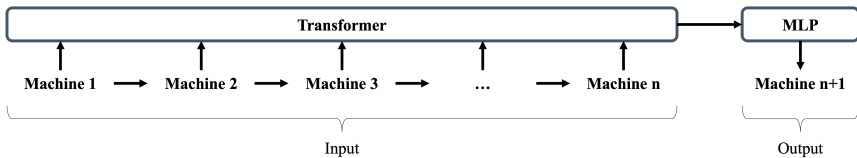
		Machine j			
		M1	M2	M3	S
Machine i	Q	0.11	0.72	0.17	0.00
	M1	0.12	0.78	0.05	0.05
	M2	0.43	0.23	0.27	0.07
	M3	0.51	0.20	0.15	0.14

**Figure 5.13:** Exemplary transition matrix with added source and sink condition (TASS) of 3 machines for the simulation of sequences.

The TASS approach might result in sequence lengths not within the real-world data sets since due to sampling statistics, a sink could be reached quite late. The ST approach, however, will always generate sequence lengths within the range of

the given real-world data set. In addition, Matthes et al. (2023) name the inability to adequately represent recurring sub-sequences of machines as a short-coming of the ST and TASS approach since the transition of a machine  $i$  to machine  $j$  depends only on the row  $i$  and prior sub-sequences are not considered. Thus, both approaches do not account for patterns within the sequences of machines.

The second data generator applied by Matthes et al. (2023) for setting up a synthetic data set of machinery sequences is a transformer network with an autoregressive architecture. The architecture of transformers was initially proposed by Vaswani et al. (2017) and has proven itself as the state-of-the-art method for LLM in the field of natural language processing (NLP) and similar applications. The autoregressive architecture was developed by Radford et al. (2019) and Brown et al. (2020) and describes a neural network that receives sequences data as input and models the probability distribution for the next elements of this sequence. Hereby, the previous elements  $x_1$  to  $x_N$  are used as context to predict the subsequent element  $x_{N+1}$ . This modeling of a sequence of length  $N$  as inputs and the prediction of the subsequent element as output is schematically presented in Figure 5.14.



**Figure 5.14:** Simplified schematic transformer architecture.

The transformer model uses a self-attention process whereby the weight relations between positions of elements of an input sequence are considered. The transformer acts as a function to model a conditional probability  $P$  using as function arguments the input sequence. Thus, the context of all preceding elements is utilized in the prediction.

$$P(x_{N+1} \mid x_1, \dots, x_n, \dots, x_N) = \text{Transformer}(x_1, \dots, x_n, \dots, x_N, \mathbf{w}) \quad (5.3)$$

In order to model the probability distribution of the subsequent element  $x_{N+1}$ , the output vector of the transformer is used in a classification model by a MLP using the weights  $\mathbf{w}$ . As with the common layouts of neural networks, the weights are iteratively adjusted during a training procedure to optimize the transformers predictive performance using a training data set. Thus, a transformer network is capable of modeling patterns of sequences and uses in contrast to the statistical approaches of the ST and TASS not only the previous element of a sequence but the whole sequence for the prediction of the following element. Transformer networks are therefore capable of predicting a following element in a sequence based on multiple preceding elements.

The fourth model used by Matthes et al. (2023) is a Bayesian network. Bayesian networks also utilize graph structures. Hereby, nodes represent random variables, which may yield different states from a set of all possible states, and edges represent direct (causal) dependencies between the random variables. The parent nodes of each node may yield different combinations of states. Thus, each node is connected with a probability distribution of the parent's state combinations and therefore, each node state's likelihood is derived by the states of the parent nodes. The Bayesian network is set-up using a training procedure with a training set of correct examples.

The architecture of Bayesian networks enables use cases in the fields of forecasting under uncertainty. However, most importantly for the here presented use case is the capability to draw samples from a Bayesian network which statistically represents the training data sets and thus, Bayesian networks can be used to sample synthetic data sets which represent the real-world data set also used for the training of the Bayesian network. This sampling procedure of the Bayesian network is used by Matthes et al. (2023) to generate synthetic machinery sequences based on data records from a real production system. Matthes et al. (2023) model the nodes of the Bayesian network as the positions of work operations and each position may yield different states representing the distinct machine at this certain position from the number of  $V$  different machines. Thus, the Bayesian network is capable of modeling sequences since every node is assigned a specific machine during the sampling procedure. Additionally, the encoding of these conditional probabilities

enables the modeling of the described patterns and complex correlations between different states of a sequence. Therefore, Bayesian networks are also capable of modeling complex machinery sequences of production systems. A sampling process therefore results in representative sub-sequences within the synthetic data set representing the real-world data set.

Matthes et al. (2023) use as a learning approach a score-based hill-climbing (Tyugu 2007) and the Bayesian information criterion (BIC) in order to measure the quality of the model (Koller and Friedman 2009). Additionally, Matthes et al. (2023) employ a forward sampling approach of the Bayesian network (Koller and Friedman 2009) to achieve a subsequent generation of machinery sequences.

#### 5.1.2.4 Synthetic Data Validation Methodology

The validation methodology uses the modified Hopfield neural networks as described in the methodology section of this thesis. However, for the analysis of combinations of machinery sequences the methodology as introduced by Spoor and Weber (2024) is applied but in order to analyze machinery sequences, the adjusted equations (4.19), (4.20), (4.21), and (4.22) are used.

First, for the filtering of anomalous values from a generated synthetic data set, the anomaly detection capabilities are applied by comparing the stimulated energy  $H(\mathcal{C})$  of a sequence  $\mathcal{C}$ . This utilizes the concept that common combinations or sequences will yield lower stimulated energies than uncommon combinations or sequences. Thus, combinations or sequences with lower energies are more common occurrences and *vice versa*, combinations or sequences with higher stimulated energies are either uncommon or incorrect. Hence, the energy distribution of real-world machinery combinations and sequences are compared with the generated synthetic instances. The network is therefore trained using a set of real-world instances with a separate test set for benchmarking with the generated data sets. Based on the real-world data, the average stimulated energy of the real-world training data set, its standard deviation  $\sigma$ , and a critical energy  $H_{crit}$  are derived. Synthetic combinations or sequences with stimulated energies above

the defined threshold are considered anomalous. The critical energy level can be selected as proposed in the context of equation (4.30). However, it is also possible to select the maximum energy of the real-world data set and thus, a FPR of 0%. This is reasonable in this specific use case since synthetic data are per definition not real and are often also designed to increase the number of rare cases. A stricter filtering would therefore exclude more interesting and rarer cases and therefore would not suite the underlying use case.

Second, the different methods of data generation must be compared in order to find the most suitable one. For this task, the classification capabilities of modified Hopfield neural networks are utilized. Each data set, including all synthetic data sets and the real-world data set, are used as separated classes and the network is trained individually per data set. This results in one weight matrix per data set. As described in the classification model of modified Hopfield neural networks, the different classes with individual weight matrices can be compared for similarity by using the matrix norm  $\|\bullet\|_{F^*}$  of equation (4.45) based on the Frobenius norm. In the case of sequential data, the regular Frobenius norm  $\|\bullet\|_F$  is applied. As discussed, other matrix norms are also mathematically feasible, but the Frobenius norm is preferably chosen due to its high interpretability in regards to the weight matrices. For a number of  $K$  analyzed data sets, this results in a  $K \times K$  distance matrix  $\mathbf{D}$  for the dissimilarity between the different data sets. Most important is the distance of the classes of data generation methods to the class of the real-world data sets. However, the data generation methods can additionally be compared regarding similarity to each other.

As a second approach for the similarity between a real-world and generated data set, the distribution of the stimulated energy levels can also be used as a measure. The method uses the network with the weight matrix trained by real-world data as a baseline. In order to derive the distance between two distributions of energies, the Wasserstein metric  $W$  between the resulting energy distributions is evaluated. The Wasserstein metric is a commonly applied measure in image processing but also in anomaly detection use cases (c.f. Spoor et al. (2023)). The first-order Wasserstein metric  $W_1$  can be visualized as the Earth-Movers-Distance (EMD) where one bar chart is viewed as piles of earth which must be transported to holes

of the other bar chart. The distance is the least amount of work required to fill the holes, while the work is defined by the amount of sand times the transported ground distance of the sand (Rubner et al. 2000). The EMD is the solution which minimizes this transport problem. The ground distance between bins of the distribution is evaluated using the  $L_1$  metric. Therefore, the distribution must be discretized in advance.

Since the ground distance between two bins is evaluated using the  $L_1$  metric, the ground distance between the two bins  $i$  and  $j$  of the distribution is given as:

$$C_{i,j} = \|i - j\| \quad (5.4)$$

In the EMD, one bar chart  $x_i$  acts as supply  $\alpha_i$  of sand and the other bar chart  $y_j$  as the demand  $\beta_j$  for sand. The bar chart distributions are normalized so that  $\sum_{i=1}^N \alpha_i = \sum_{j=1}^N \beta_j = 1$  with all values of  $\alpha_i$  and  $\beta_j$  being positive values. Thus, two measures are derived whereby  $\delta_x$  denotes the Dirac delta distribution:

$$\begin{aligned} \nu &= \sum_i \alpha_i \delta_{x_i} \\ v &= \sum_j \beta_j \delta_{y_j} \end{aligned} \quad (5.5)$$

As described in the EMD, one bar chart acts as sources of entries flowing towards the second bar chart which is a sink. From this, the source and sink conditions of the optimization problem are derived:

$$\begin{aligned} \sum_j Q_{i,j} &= \alpha_i \\ \sum_i Q_{i,j} &= \beta_j \end{aligned} \quad (5.6)$$

The Wasserstein  $L_1$  metric is the optimal solution of the corresponding transport problem:

$$W_1(\nu, \nu) = \min \sum_{i,j} Q_{i,j} C_{i,j} \quad (5.7)$$

The Wasserstein metric is evaluated for each pair of energy distributions from a total number of  $K$  analyzed data sets. Thus, this results also in a symmetric distance matrix  $\mathbf{D}$  which is used to evaluate the dissimilarity of the energy distribution of two data sets. The resulting distance matrix shows which data sets are more similar in regard to the energy distribution. If the synthetic data sets cover the same range of machinery sequences, the distance between the energy distributions should be rather small.

Both methods for computing the distance have different advantages and disadvantages. The Frobenius norm between the weight matrices directly measures the sequence likelihood over the whole data set. Thus, changes in the likelihood of sequences are directly measured by this evaluation. However, this requires the training of all classes to converge and thus, each data set must have enough data points available. On the other hand, the Wasserstein metric between the energy distributions measures the similarity of the complexity of the data. A synthetic data set might have an overrepresentation of common cases which individually fit exactly with the weights but result in energy distribution differences. In this case, the distance from the Frobenius norm between the weight matrices would be small, but the distributions would look quite different and might result in a large distance using the Wasserstein metric. *Vice versa*, the real-world data could yield some rare but correct cases which are represented in the resulting weight matrix but are so seldom that they are not reflected as modes of the energy distribution. In this case, the distance between the energy distributions is very small, but the distance between the weight matrices is larger.

While both similarity measures result in distance matrices, the distance values itself lack a direct interpretability. There exists no distance threshold where the real-world and synthetic data set are perceived as too different or the data generator as insufficient. The evaluation only compares different generators and

selects the most suitable one from a collection but does not evaluate if any generator is suitable. Thus, the distance can only be interpreted in the context of other generators. However, optically close energy distributions, i.e., a small distance between the energy distributions using the Wasserstein metric, are a good indicator that the generated data is very similar. The decision whether the similarity is sufficient must still be conducted on a case-by-case basis.

This limitation can be mitigated if prior to the comparison of real-world and generated data, the average distance between multiple subsets randomly drawn from the real-world data set is investigated. This enables the evaluation of a threshold distance. Data sets within this threshold distance are then considered to be originating from the same data set. Thus, it is possible to estimate a threshold under which synthetic data is considered as realistic. However, this threshold is use case dependent and not generalizable.

### **5.1.2.5 Data Filtering and Validation Results**

Spoor et al. (2024) use as data generators a ST, TASS, transformer network, and a Bayesian network. The sequences generated by these networks are then filtered to exclude anomalous machinery sequences and the realism of these generator models is compared in order to derive the most suitable network for a synthetic data generation approach for the specific underlying use case in production planning.

For the validation of the model, the machinery sequences of a manufacturing company are analyzed. This data set contains 170,787 sequences collected between 2022 and 2023 of all real-world production processes of one production facility. The number of different machinery types deployed in this factory is  $V = 73$ . Each production process runs through a different number of workstations, with each workstation having one specific machine assigned to out of the set of all machinery types. The number of workstations of a production process is between 1, meaning only a single machine is deployed during this manufacturing process, and 77 different workstations with each workstation yielding a specific machine. Thus, the same machinery type can be deployed during a manufacturing process



multiple times. This data set is utilized for the training of the selected data generators. An overview of the data set is given in Table 5.7.

**Table 5.7:** Description of the applied data set of machinery sequences.

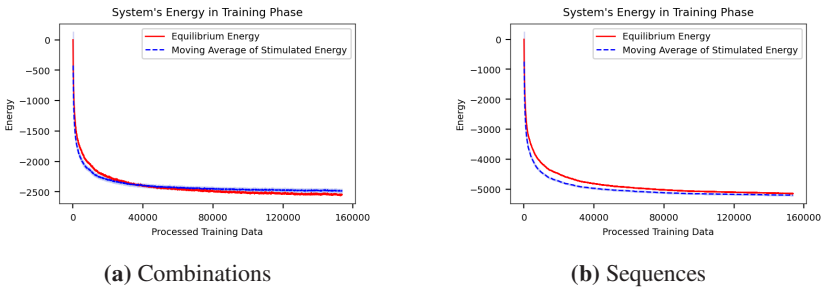
Instance	Quantity	Comment
Machinery sequences	170,787	
Unique machineries	73	
Minimum sequence length	1	
Maximum sequence length	77	
Years	2	
		from 2022 to 2023

From a domain-perspective, two attributes of the synthetic data set of machinery sequences are important:

- (a) the capability of the synthetic data set to correctly identify combinations of machines over the whole manufacturing process, e.g., if two certain machinery types are never combined within the same process or, *vice versa*, if multiple machinery types must be applied within the same process together independent of their position in the process.
- (b) the capability of representing predecessor and successor relationships across the process correctly for all pairs of machines within the same processes and thus, also validating the set of process pairs within a sequence.

The first evaluation tests whether there are incorrect combinations in a synthetic data record but neglects their arrangement. The second evaluation tests correct successor and predecessor relations of single workstations but does only validate combination of immediately succeeding workstations. To ensure that a synthetic data set adheres to a high degree of realism, both relations must be tested. As a pre-processing step, each data record of a machinery sequence is transformed into two  $73 \times 73$  connection matrices, a first using the combination relation of equation (4.1) and a second using the sequence relation of equation (4.19).

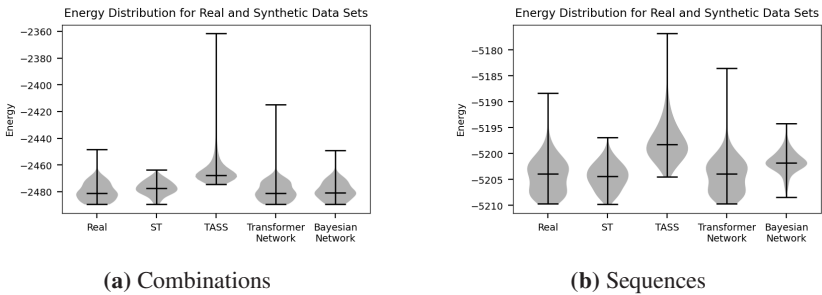
For the training procedure, the real-world data set is randomly split into a training set and a test set using a hold-out cross-validation method. However, a full parametrization using a more nuanced cross-validation method is not conducted since modified Hopfield neural networks prove themselves rather stable regarding their parametrization, as shown in the first use case in Section 5.1.1 and validated during this procedure. Thus, 90% of the real-world data are used for the training of the modified Hopfield neural network, while the complete real-world data set is used to set-up the synthetic data generators. The test set, containing 10% of the real-world data set, is later applied to the trained weight matrices, which are set-up for each data generator model and the training set, to compute the stimulated energies. Subsequently, the stimulated energy distributions are compared with each other. The training is conducted using equation (4.16) with a training rate of  $\alpha = 0.05$  and no additional regularization terms are applied. As with the pre-processing, the training is conducted separately for (a) the connection matrices of combinations and (b) the matrices of sequences. The results of both training procedures are visualized in Figure 5.15.



**Figure 5.15:** Training procedure using the moving average of the stimulated energy over 100 iterations, its 3-standard derivation area, and the equilibrium energy (Spoor et al. 2024).

Both training procedures start to converge at around 20,000 iterations and reaches a stable energy level at around 40,000 iterations of the training process. Therefore, the used amount of training data is sufficient to setup the modified Hopfield neural networks and to reach a stable equilibrium energy.

Next, the stimulated energy levels are evaluated by applying the test data set, derived from the real-world data set, to the trained weight matrices from the four synthetic data sets created by different generator models and the training data set. The distributions of the energy levels, which result from stimulating the trained weight network with the tested connection matrices, can be analyzed using a violin plot. Again, the procedure must be separated for the evaluation of (a) combinations and (b) sequences. The results are given in Figure 5.16.



**Figure 5.16:** Distribution of the real-world and synthetic data sets' energy levels displayed as violin plots (Spoor et al. 2024).

The evaluation of the stimulated energies enables the anomaly detection model of the modified Hopfield neural network and thus, describes the filtering step of the proposed methodology by Spoor et al. (2024). For both evaluations, only a small amount of the synthetic data records has stimulated energies above the maximum energy level of the test data set. Since first, the real-world sequences are always correct by definition of the use case, and second, it is a useful property of synthetic data sets to contain rare instances, i.e., rare and uncommon but realistic machinery sequences in order to also simulate edge case events, it is most useful to apply a FPR of 0% and use the maximum stimulated energy level of the real-world test data set as threshold  $H_{crit}$  for an anomaly detection, separately set for the evaluation of combinations and sequences. This approach might result in an inclusion of anomalous machinery sequences and a lower TPR but ensures that the synthetic data set also includes relevant rare cases.

With a FPR of 0%, there are no anomalies in neither the Bayesian network nor the ST approach for both evaluations. The transformer network yields a low ratio of anomalies of 0.004% for the evaluation of combinations and 0.005% for the evaluation of sequences. The transformer model therefore generates only in very rare cases anomalous data records. In contrast, TASS yield the highest number of anomalies for all synthetic data sets in both evaluations with 5.13% anomalous data records for combinations and 1.54% for sequences, respectively. This result is also an indication that TASS might generate less realistic synthetic data. However, the filtering step improves the realism of the TASS data set by excluding all anomalous data records.

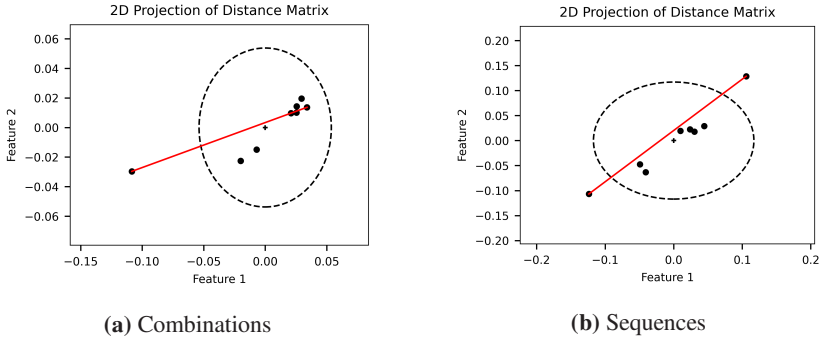
As the second step of the methodology by Spoor et al. (2024), the synthetic data sets are compared regarding their degree of realism and the most suitable data set is selected. As the first evaluation method for realism, the distributions in Figure 5.16 are compared with the distribution of the real-world data set acting as the benchmark for the degree of realism. From a simple visual analysis, it can be derived that the TASS approach is likely to generate rare data within the upper end of the real-world data set's energy distribution, which indicates that the TASS approach generates more uncommon, rare, or sometimes incorrect data records. This is expressed by the higher median of the distribution of TASS compared with the other distributions and also since the energy levels of most data records generated by TASS exceed the median of the real-world data set. Therefore, the data set is not similarly balanced as the real-world data set and yields a deviating ratio of common and rare machinery sequences, which decreases its realism and usefulness during simulations. Regarding the other synthetic data sets, the Bayesian network yields a deviating distribution with a higher median for the sequence evaluation than the real-world data set but behaves quite similar for the evaluation of combinations. The ST approach has a close median to the real-world data set, but the distribution differs from the real-world data set in both evaluations since the distribution has a shape similar to a bell curve and lacks the distinctive modes of the real-world data set's distribution. The transformer model's distribution however behaves very similar to the real-world data set from a visual comparison for combinations as well as sequences. Thus, despite the small

number of outliers in the upper box plot area, this indicates that the transformer model might perform best.

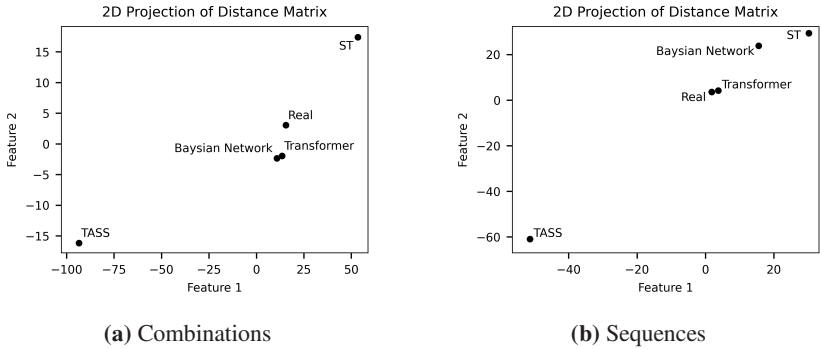
This visual analysis can be quantified by the application of the Wasserstein metric and the computation of a distance matrix between each energy distribution. First, a threshold within a data set is considered indistinguishable from a real-world data set is derived. Since the training procedure starts to converge towards a stable energy level at around 20,000 iterations, the training data set, which is derived from the real-world data set, is randomly split into eight subsets, for which an individual training procedure is then applied. For each weight matrix trained by a subset, the test set is applied to measure the stimulated energy level and to create an energy distribution. These distributions are then measured regarding their similarity using the first-order Wasserstein metric in order to derive an average and maximum distance between the real-world subsets. The average and maximum distances can then be applied as thresholds within a synthetic data set which is indistinguishable from a real-world data set.

The stimulated energy distributions are discretized using 200 bins. The first-order Wasserstein metric is computed using the python library POT by Flamary et al. (2021). The resulting distance matrix is transformed using a 2D projection in order to visualize the result as a plot. The 2D projection is conducted using Multi-dimensional Scaling (MDS) implemented by the sklearn library by Pedregosa et al. (2011). Please note that the distances in the 2D projection by MDS might be slightly distorted since MDS only preserves the distances as well as possible for a given data set. The resulting 2D projection of the distance matrix between the eight subsets of the training set stimulated with the test set is displayed in Figure 5.17. The maximum distance between the eight subsets is marked in red and the average distance is displayed by the radius of the centroid of the data points.

The resulting distances from Figure 5.17 can then be used as benchmark for the degree of realism of the synthetic data sets. The distance between the distributions of Figure 5.16 are computed by the first-order Wasserstein metric and the resulting distance matrix is also visualized using a projection by MDS. The 2D projection is displayed in Figure 5.18.



**Figure 5.17:** 2D projection of the distance matrix between the energy distributions by a test set using randomly drawn subsets of the real-world data set for training of the weight matrix.



**Figure 5.18:** 2D projection of the distance matrix between the energy distributions in Figure 5.16 using the Wasserstein metric (Spoor et al. 2024).

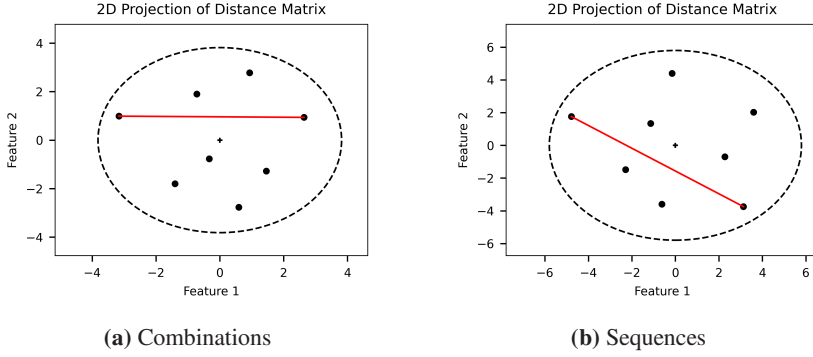
Figure 5.18 shows that the distances between the synthetic data set and the real-world data set are a magnitude greater than the distances between the subsets from the real-world data set. Thus, the average and maximum distance between the subsets from the real-world data set are not added in Figure 5.18. Concluding, the threshold for the degree of realism is not met in this analysis for the evaluation based on combinations and sequences since the Wasserstein metric is rather strict in its assessment. However, the proximity of the synthetic data sets to the real-world data set can still be used to select the most realistic synthetic data sets.

First, in the case of the analysis of combination, the Bayesian network and the transformer are closest to the real-world data set with both synthetic data sets being very similar to each other. However, the transformer yields with a distance of  $d = 5.28$  the smallest distance overall. The TASS approach in contrast yields with  $d = 110.97$  a comparably high distance. TASS comparably worse performance can also be derived solely based on the visual analysis of Figure 5.16. Also, the ST approach yields a rather high distance, which again can be seen in Figure 5.16 due to its bell curve-like shape. All synthetic data sets do not meet the threshold of the maximum distance between the subsets of the real-world data set of  $d_{crit} = 0.15$  and therefore are distinguishable from the real-world data set.

Second, for the analysis of sequence information, the transformer performs the best with a distance of  $d = 1.61$  comparably close to the real-world data set. However, compared with the maximum distance between the subsets of the real-world data set of  $d_{crit} = 0.33$ , the data records generated by the transformer network are still distinguishable from real-world data records. The Bayesian network comes second in distance and the TASS approach again places last regarding similarity to the real-world data set.

In conclusion, the transformer model performs best for all tested synthetic data generation approaches using the first-order Wasserstein metric between the energy distributions. Most notably, the ranking in similarity of the data generators to the real-world data set based on increasing distances stays the same for the evaluation based on combinations and sequences. In summary, the second closest generator is the Bayesian network, third is the ST approach, and the TASS approach performs worst in both the evaluation of combinations and sequences.

As a second metric to compare the synthetic data sets, the Frobenius norm, or rather the adjusted Frobenius norm of equation (4.45) in the case of an evaluation of combinations, is applied. Again, eight randomly drawn subsets from the real-world data set are used to compute a benchmark for the degree of realism. The resulting distance matrix is displayed as 2D projection using MDS in Figure 5.19.

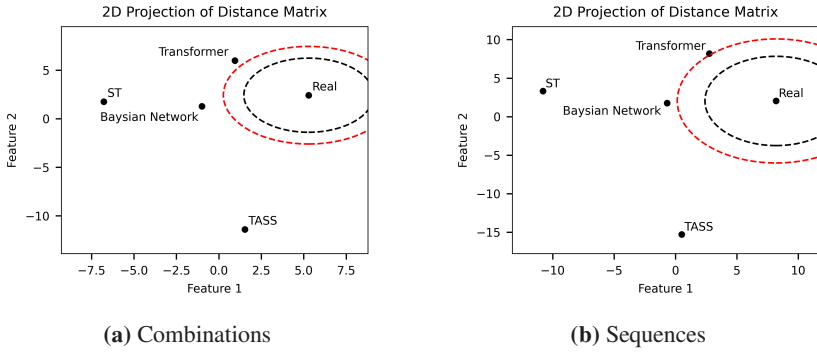


**Figure 5.19:** 2D projection of the distance matrix between the trained weight matrices of randomly drawn subsets of the real-world data set.

The maximum and average distances of the distance matrix between the eight subsets of the real-world data set are again used as threshold to compare the similarity of the synthetic data sets to the real-world data set. The distance matrix based on the trained weight networks of the synthetic data sets and the real-world data set is computed using the Frobenius norm, or rather the adjusted Frobenius norm. The 2D projection of the resulting distance matrix for an evaluation of combinations and sequences is given in Figure 5.20. The average (black circle) and the maximum (red circle) distance between the randomly drawn subsets of the real-world data set are added in Figure 5.20.

The Bayesian network performs best for the evaluation of combinations with a distance of  $d = 6.55$  and a threshold using the maximum distance of the sampled subsets  $d_{crit} = 5.02$ . The transformer also performs close to the threshold with a distance of  $d = 6.66$ . Therefore, both approaches yield a high degree of realism and are, compared to the prior analysis using the Wasserstein metric, quite close to the threshold within a synthetic data set becomes indistinguishable from a real-world data set. Similar to the results of the analysis using the Wasserstein metric, the ST approach comes third and the TASS approach last with a distance of  $d = 15.05$ .





**Figure 5.20:** 2D projection of the distance matrix between the weight matrices of the real-world data set and the synthetic data sets using the Frobenius norm.

In the evaluation of sequences, the Bayesian network again performs best with a distance of  $d = 9.29$  and a threshold of the maximum subset distance of  $d_{crit} = 8.05$ . The transformer network also performs close to the threshold with a distance of  $d = 9.57$ . As in the case of sequences, the Bayesian network and transformer network are close to being indistinguishable from the real-world data set based on the trained weight matrices. Please note that the 2D distances computed by MDS in Figure 5.20 are slightly distorted when comparing the Bayesian network and transformer network distances to the real-world data set. Again, the ST approach performs third and the TASS approach last with a distance of  $d = 19.92$ .

In summary, the Bayesian network and the transformer network are both suitable for the data generation approach. While the respective performance of both generators is nearly on-par in the case of the Frobenius norm based on the trained weight matrices, the analysis using the Wasserstein metric suggests an advantage of the transformer network in the case of an evaluation of sequences. Therefore, the usage of a transformer network as a generator model for synthetic data can be recommended in this certain use case. The Bayesian network is a suitable alternative. Since the order of similarity does not change when evaluating machinery combination or sequences for the same metric, the method is robust regarding this evaluation. In addition, since the Bayesian network and transformer model are

comparably close to the real-world data set in the analyses (besides the evaluation of sequences using the Wasserstein metric), it is reasoned that the method is also rather robust regarding the selection of the metric, i.e., the use of the Wasserstein metric between the stimulated energy distributions and the use of the Frobenius norm between the weight matrices of the different trained modified Hopfield neural networks. The results confirm the KPI-based comparison by Matthes et al. (2023). However, the method by Spoor et al. (2024) presented here adds further detail and a more nuanced analysis including the benchmarking using the subset of the real-world data set to evaluate the degree of realism.

#### **5.1.2.6 Advantages and Limitations of Methodology**

In this subsection, solely the proposed data generation and validation methodology is discussed. A discussion on the general performance, mathematical foundation, and application of modified Hopfield neural networks based on the presented use cases is conducted in the following Section 5.2.

The data generation and validation methodology by Spoor et al. (2024) yield advantages in a practical application. Firstly, as with the application of modified Hopfield neural networks, the training procedure and anomaly detection do not require labeled training data including true positives, i.e., incorrect records as examples for the training procedure. Since in a synthetic data generation approach only correct real-world data are also used, the proposed method does not require any additional data collection or labeling compared to other state-of-the-art anomaly detection models. Thus, the data requirements of the filtering and validation capabilities do not exceed the requirements set by the data generators.

Secondly, since the training already converges at around 40,000 data records, the amount of data required is also comparably low and thus, the modified Hopfield neural networks can be embedded in most data generator approaches even when only a small amount of real-world data is available.

Thirdly, the methodology by Spoor et al. (2024) can evaluate synthetic data sets without the use of structural KPIs but tests every data record separately

for anomalous behavior, focused on either combinations or sequences, during the filtering step. The validation step also incorporates the real-world data set's structural information about all available data records via the training of the weight matrices of the network and the comparison using the Frobenius norm.

Fourthly, as also shown in the first use case, the differences in the synthetic data sets can be directly discussed and analyzed in more depth using the weight matrices. Thus, the method adds interpretability to its decision due to the inherent interpretability of modified Hopfield neural networks. In addition, the set-up of weight matrices for the synthetic data sets can add explanatory value to the generator models, i.e., the results by a "black box" generator such as a transformer network can be discussed and analyzed in more detail using the weight matrix and the Frobenius norm as similarity metric between the real-world and synthetic data sets.

There are two major limitations of the methodology by Spoor et al. (2024). Firstly, the computed distances are only comparable within the context of the specific use case and the analyzed data sets but do not yield any real-world interpretation. However, this limitation is mitigated for the comparison by computing a threshold using subsets of the real-world data set. It should be noted that the threshold is also use case specific and an evaluation whether a synthetic data set is sufficiently realistic must be conducted carefully on a case-by-case basis.

Secondly, the evaluation must be conducted twice for combinations and sequences and only one criterium can be tested per distance matrix. Thus, a final decision on the usefulness of the synthetic data set must be conducted by a weighting of both evaluations. In this discussed use case, the results are similar independent of the selected evaluation. However, this might not always be the cases and thus, a case-by-case decision might be required in other use cases. This should be addressed by adding domain knowledge to the analysis whether sequence information or the combinations of machineries is more important in order to generate plausible synthetic data sets.

Thirdly, the results for both applied metrics, the Wasserstein metric between the energy distributions and the Frobenius norm between the trained weight matrices,

might differ. In this use case, the ranking of the Bayesian network and transformer network is swapped in both metrics. However, this also should be mitigated by carefully assessing the domain requirements. If an over- or underrepresentation of certain complexity levels of data records, and thus different modes within the energy distribution, are less important but the combination or sequence logic must be preserved, the Frobenius norm might be more suitable. If the exact sequence and combination logic is less important but all data records should yield a similar complexity to the real-world data, the Wasserstein metric is a better choice.

In summary, the methodology by Spoor et al. (2024) can add the required filtering and validation procedure to synthetic data generators, whereby the capabilities of modified Hopfield neural networks can be leveraged to create impactful results and to exceed the possibilities of state-of-the-art methods. All limitations can be mitigated by a diligent incorporation of domain knowledge. The advantages of the method are underlying the capabilities of modified Hopfield neural networks, i.e., the usage of only correct data, the high interpretability and explainability, and the anomaly and classification capabilities for combinations as well as sequences.

### **5.1.3 *Excursus:* Classification of Leukemia Subtypes**

The developed anomaly detection and classification model using modified Hopfield neural networks yields applications beyond the discussed use cases in production planning and is, as an AI model, versatile enough for applications in multiple domains. As an exemplary application of modified Hopfield neural networks without the embedding in production planning but in a use case from the field of oncology for the classification and clustering of gene expressions of Leukemia subtypes, an excursus is given in this section.

### 5.1.3.1 Use Case Description

The data analyzed here are gene expression from The Cancer Genome Atlas Research Network (2013) downloaded from the Broad GDAC Firehose platform<sup>2</sup>. The following used data set contains  $T = 166$  patients with  $V = 386$  gene expressions from different subtypes of acute myeloid leukemia (AML). The number of patients per subtype using the French-American-British (FAB) classification by Bennett et al. (1976) is listed in Table 5.8.

**Table 5.8:** Number of patients per AML FAB subtype.

Subtype	FAB subtype description	Quantity [#]	Share [%]
m0	undifferentiated AML	16	9.6%
m1	AML with minimal maturation	42	25.3%
m2	AML with maturation	39	23.5%
m3	acute promyelocytic leukemia (APL)	16	9.6%
m4	acute myelomonocytic leukemia	35	21.1%
m5	acute monocytic leukemia	18	10.8%

Most relevant in this use case is the distinction between the APL subtype and the other subtypes, e.g., as analyzed by Thrun et al. (2022). Thus, the modified Hopfield neural network is applied to distinct combinations of gene expressions between these subtypes. This is a classification task where the explainability and distance metric between classes is further analyzed.

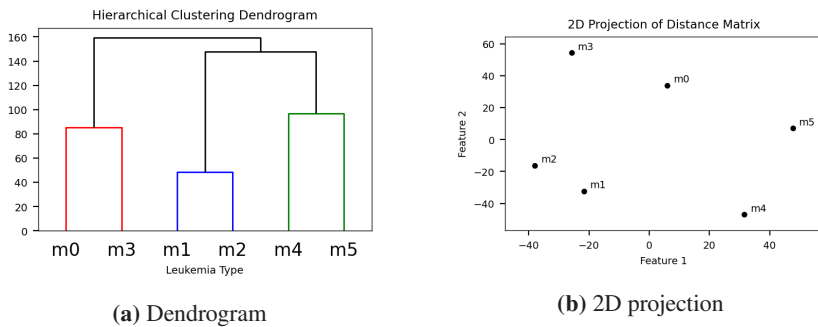
First, the gene expressions must be preprocessed to be converted into a connection matrix  $\mathbf{M}$ . The expression intensities  $g_i$  for each gene  $i$  are logarithmized using  $g_i^* = \log_{10} g_i$ . For the activation procedure building the connection matrix, a Boolean intensity threshold as per equation (4.2) is used for the activation of edges. The threshold of  $t_i = -4.6$  for all genes  $i$  is selected based on a grid

<sup>2</sup> [http://firebrowse.org/?cohort=LAML&download\\_dialog=true](http://firebrowse.org/?cohort=LAML&download_dialog=true); accessed 26.10.2022

search to optimize results and keep the parametrization simple. This Boolean threshold corresponds to an intensity of  $g_i \approx 2.5 * 10^{-5}$ . Thus, each patient's gene expressions are processed into a  $386 \times 386$  connection matrix  $\mathbf{M}$ .

### 5.1.3.2 Similarity Measure between FAB Subtypes

For each subtype, the corresponding weight matrix is trained using the procedure as described in the section introducing the classification model. Thus, a weight matrix  $\mathbf{W}_k$  is created for each FAB subtype  $k$ . It should be noted that the available training data are very limited for the subtypes. Thus, the results are potentially limited in their explanatory power, in particular for the APL subtype since only 16 patients' gene expressions are available. However, the model results in weight matrices which can be analyzed using the adjusted Frobenius norm  $\|\bullet\|_{F^*}$  of equation (4.40). This adjusted Frobenius norm is used to build a  $6 \times 6$  distance matrix  $\mathbf{D}$  between the  $K = 6$  different FAB subtypes. This distance matrix can then be analyzed using a dendrogram which clusters and describes similar FAB subtypes using a hierarchical agglomerative clustering with Ward linkage. In addition, the distances between the subtypes can be displayed as a 2D projection computed via MDS to highlight the distances between subtypes in a visual way. The results are given in Figure 5.21.



**Figure 5.21:** Dendrogram and 2D projection of the distance matrix for the different analyzed FAB subtypes of AML.

Most notably in Figure 5.21, the resulting weight matrices of the FAB subtypes m1 and m2 are very similar and grouped together in the clusters of the Dendrogram. This is an expected result since both subtypes are AML with different levels of maturation. Using the 2D projection of the distance matrix, it can be seen that the relevant subtype m3 of APL is the most different from the subtype m4 of acute myelomonocytic leukemia.

In order to differentiate the APL subtype m3 from the other subtypes, it is possible to first analyze its differences from subtype m4 since these two subtypes should be easy to differentiate using a classification model. This is analyzed using the classification model. Second, an analysis can be conducted to separate subtype m3 from m1 and m2. This task might be more difficult since these subtypes are closer to each other as visible in the 2D projection of Figure 5.21. Due to the small number of patients with a diagnosis of FAB subtype m0, m3, and m5, no differentiation of these three subtypes is conducted since the small amount of available data highly limits results and model capabilities.

### 5.1.3.3 Classification of FAB Subtypes m3 and m4

To distinguish FAB subtype m3 and m4, a binary classification model using the modified Hopfield neural network is set-up. The learning rate is set as  $\alpha = 0.05$  and no further regularization techniques are used. Thus, equation (4.16) applies for the training procedure. The unbiased correction value is computed using equation (4.7). The connection matrix is set up as described in the previous subsection. It should be noted that the energy levels during training do not converge since only a limited amount of patient data per class is available, in particular in the case of subtype m3. Therefore, the classification results might be improved by generating and analyzing more training data of gene expressions from patients with a diagnosis of subtype m3.

In addition, the performance is benchmarked with the classification capabilities of a SVM (Cortes and Vapnik 1995), KNN, Decision Tree, Random Forest (Breiman 2001), and MLP implemented by sklearn (Pedregosa et al. 2011). The SVM uses

an RBF kernel and a gamma of  $\frac{1}{\sqrt{V\sigma^2}}$ . The KNN is implemented using 5 neighbors. Random Forest is using 100 trees and  $\sqrt{386} \approx 20$  features per split. Decision tree and Random Forest are using Gini impurity as the information loss criterion. The MLP is configured using a single layer with 100 neurons and a learning rate of 0.01. The AI models use the full gene intensities  $g_i$  and not the reduced connection matrix. The results are given in Table 5.9 for 30 Monte Carlo cross-validations of a 60% training data and 40% test data split.

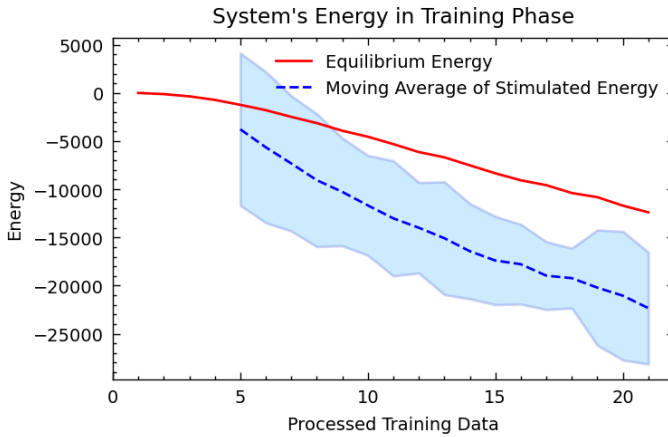
**Table 5.9:** Accuracy of AI models for FAB subtype m3 and m4 classification.

Model	Accuracy
Hopfield net	$(96.9 \pm 3.1)\%$
SVM	$(96.6 \pm 3.0)\%$
KNN	$(97.5 \pm 2.4)\%$
Decision Tree	$(91.1 \pm 7.3)\%$
Random Forest	$(94.8 \pm 5.0)\%$
MLP	$(95.9 \pm 2.7)\%$

As expected by the large distance between both subtypes using the 2D projection of the distance matrix, the accuracy of the modified Hopfield neural network classification model for subtypes m3 and m4 is very good and in most cases all patients' gene combinations are classified into the correct subtype diagnosis. All models yield an overall good performance and considering the standard deviation over the 30 cross-validations, there are no significant differences in performance for all tested AI models. Therefore, the modified Hopfield neural network proves itself efficient in this use case and it is concluded that the classification model is feasible for binary classification tasks in gene expressions.

This task can additionally be analyzed using an anomaly detection model by training the network using only subtype m4 and evaluating whether gene expressions of patients with a diagnosis of m3 are detected as anomalous when tested. First, the energy levels during the training procedure are visualized in Figure 5.22.



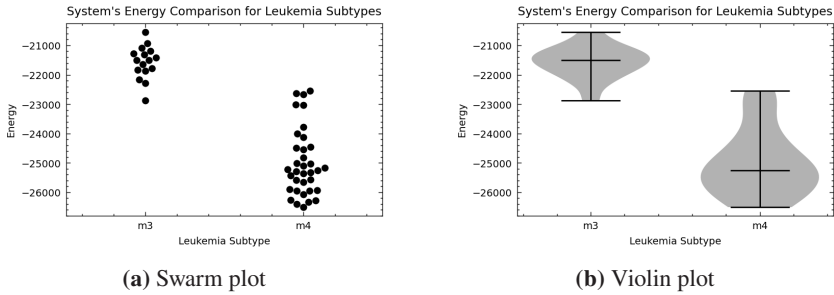


**Figure 5.22:** Training procedure of FAB subtype m4 using the moving average of the stimulated energy over 5 iterations, its 3-standard derivation area, and the equilibrium energy.

As visible, the moving average of the stimulated energy of the training data does not converge and thus, the amount of training data is lower than necessary to conduct a precise anomaly detection. Nevertheless, it is possible to conduct the anomaly detection by evaluating the stimulated energy levels of the test data and the gene combinations of patients diagnosed with the AML FAB m3 subtype. The resulting energy levels of subtypes m3 and m4 when the network is trained with instances from subtype m4 using a holdout cross-validation of a 60% training data and 40% test data split is given in Figure 5.23.

Figure 5.23 shows that the subtypes can be efficiently distinguished using an anomaly detection model. This can be reasoned since the swarm plots of the stimulated energy levels of FAB subtypes m3 and m4 are only overlapping for one patient's gene expressions diagnosed with subtype m3. All other patients' gene expressions are visually separable by their energy level. Therefore, a suitable critical energy threshold value can be selected to efficiently distinct both subtypes.

The anomaly detection capabilities can be further highlighted by a ROC curve and a benchmarking using as in the prior use cases a one-class SVM (Schölkopf et al. 2001), an Isolation Forest (Liu et al. 2008) implemented by sklearn (Pedregosa



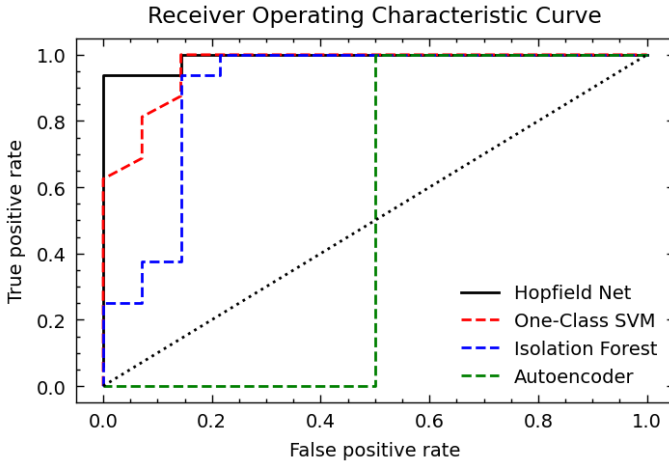
**Figure 5.23:** Energy levels of the gene combinations of all patients with FAB subtypes m3 and m4 for a weight matrix trained on subtype m4 using a 60% training data size.

et al. 2011), and an Autoencoder implemented by Zhao et al. (2019). The one-class SVM uses a polynomial kernel and a gamma of  $\frac{1}{V\sigma^2}$ . For the Isolation Forest as well as the Autoencoder a small contamination of 1% is assumed. The Autoencoder utilizes six layers in the following ordered sizes of 128, 64, 32, 32, 64, and 128 neurons, 1000 epochs, and a batch size of 16. As in the classification task, the one-class SVM, Isolation Forest, and Autoencoder are trained and evaluated using the full data set of all gene expressions  $g_i$  and not the preprocessed connection matrix. The resulting evaluation is given in Figure 5.24.

Figure 5.24 shows that the modified Hopfield network is outperforming the other benchmarked models despite their use of the full gene expression data  $g_i$  instead of the simplified pre-processed connection matrix. The relatively bad performance of the Autoencoder is most likely a result of the limited amount of available training data. As demonstrated using the binary classification model, the modified Hopfield network shows a strong performance in analyzing gene expressions for AML FAB subtypes m3 and m4.

#### 5.1.3.4 Classification of FAB Subtypes m1, m2, and m3

In addition to the differentiation of FAB subtypes m3 and m4, the differentiation between subtypes m3 from subtypes m1 and m2 is analyzed in a multi-class

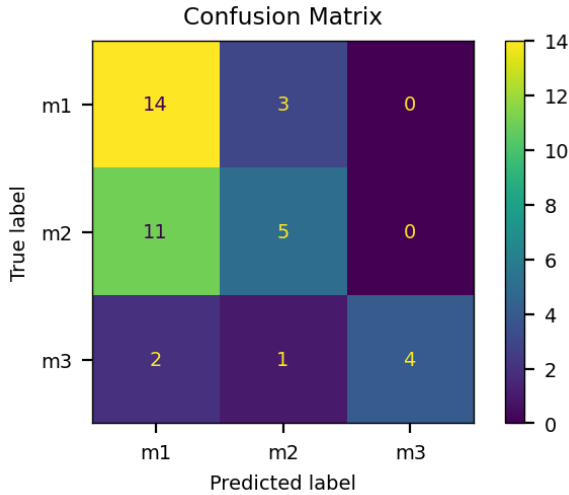


**Figure 5.24:** Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection separating FAB subtype m3 from m4.

classification. For the following analysis, the same parametrization as in the prior conducted differentiation of subtypes m3 and m4 for all AI models is applied. The classification model's results for a single evaluation can be displayed by a confusion matrix. Again, the energy levels during training do not converge, in particular for subtype m3, since a limited amount of training data is available. The confusion matrix for the classification using the modified Hopfield neural networks is given in Figure 5.25.

As expected, the FAB subtypes m1 and m2 cannot be separated by the model in an efficient manner. This can be reasoned since both weight matrices are very similar as measured in Figure 5.21 using the adjusted Frobenius norm as similarity measure between the subtypes and the model loses accuracy mostly by wrong classifications between subtypes m1 and m2. However, the subtype m1 and m2 can be quite well separated from the subtypes m3 but patients with subtype m3 are often misclassified as subtype m1 or m2.

The model's performance is again benchmarked using SVM, KNN, Decision Trees, Random Forest, and MLPs. The parametrization of the models is the same



**Figure 5.25:** Confusion matrix of modified Hopfield neural networks for a classification of FAB subtype m1, m2, and m3.

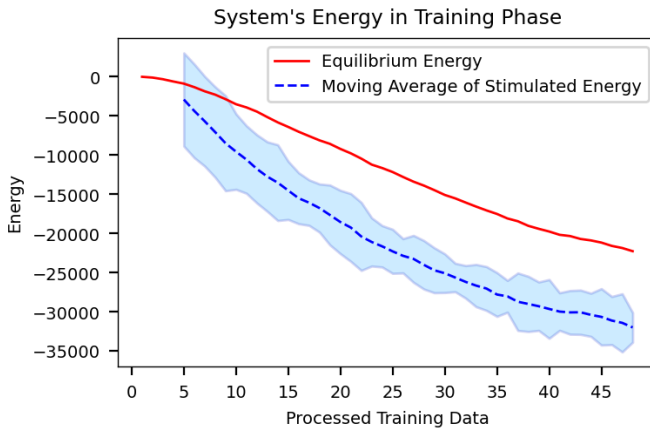
as priorly given. The full accuracy over all classes is computed and in addition, the accuracy for only separating m3 from m1 and m2 is also computed. The results are given in Table 5.10.

**Table 5.10:** Accuracy of AI models for FAB subtype m3 and m4 classification.

Model	Full accuracy	Accuracy only m3
Hopfield net	$(48.4 \pm 7.5)\%$	$(88.3 \pm 3.8)\%$
SVM	$(53.6 \pm 9.5)\%$	$(88.0 \pm 4.8)\%$
KNN	$(61.1 \pm 8.6)\%$	$(97.1 \pm 2.0)\%$
Decision Tree	$(55.1 \pm 7.9)\%$	$(91.9 \pm 4.2)\%$
Random Forest	$(62.6 \pm 5.3)\%$	$(98.9 \pm 1.9)\%$
MLP	$(51.2 \pm 9.1)\%$	$(88.4 \pm 6.5)\%$

The analyzed modified Hopfield neural network is underperforming in this analysis, in particular compared to the good performance by the Random Forest and KNN algorithms. This underperformance is the result of the class imbalance of very limited available data of FAB subtype m3 since the stimulated energy during training of this subtype does not converge towards a stable energy level.

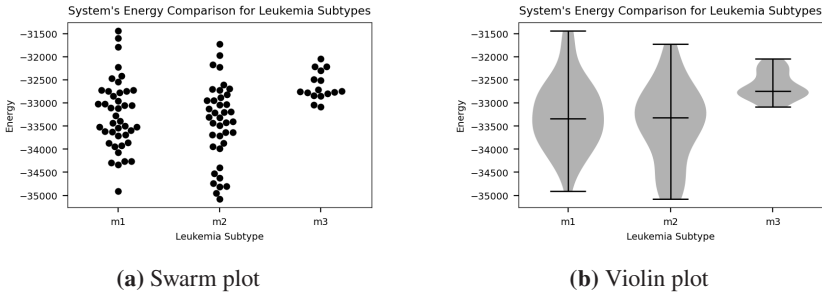
However, using subtypes m1 and m2 for the training in an anomaly detection task might improve the performance. In addition, the training size is increased due to the larger number of m1 and m2 subtypes and since the small number of patients with a diagnosis of m3 must not be considered when selecting a relative training size. Thus, the anomaly detection model is trained using classes m1 and m2 and is also tested for class m3. This is one important feature of modified Hopfield networks since they do not require as anomalous labeled data during training. The training procedure is given in Figure 5.26.



**Figure 5.26:** Training procedure of FAB subtype m1 and m2 using the moving average of the stimulated energy over 5 iterations, its 3-standard derivation area, and the equilibrium energy.

Despite the larger amount of training data, the network still does not start to converge against a stable energy level. However, it should be possible to conduct

a useful anomaly detection. The same parametrization is selected as in the prior analysis of subtypes m3 and m4. The resulting energy levels of subtypes m1, m2 and m3 when the network is trained with instances from subtypes m1 and m2 using a holdout cross-validation of a 60% training data and 40% test data split is given in Figure 5.27.

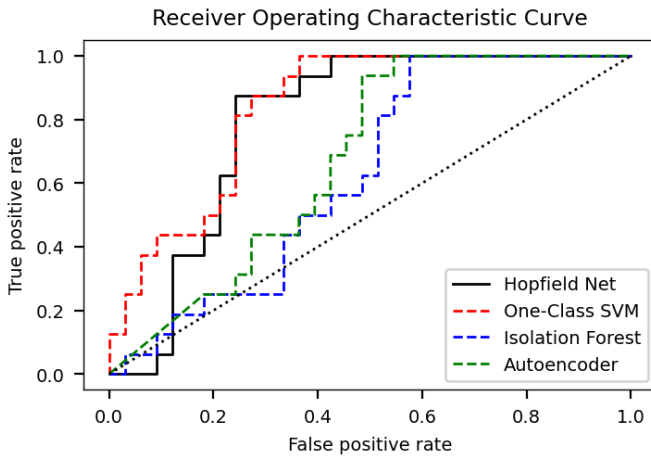


**Figure 5.27:** Energy levels of the gene combinations of all patients with FAB subtypes m1, m2 and m3 for a weight matrix trained on subtype m1 and m2 using a 80% training data size.

Figure 5.27 shows that subtypes m1 and m2 have many patients with gene expressions resulting in very low energy levels. However, there are gene combinations resulting in higher energy levels comparable to those of the subtype m3. Thus, an anomaly detection is not capable of separating these classes in a very accurate manner. This might result from subtypes m1 and m2 containing a larger variety of gene combinations which are not all sufficiently represented within the data set. In addition, Figure 5.21 also shows that these subtypes are more similar and thus, also harder to separate.

In conclusion, the performance is benchmarked against one-class SVMs, Isolation Forest, and Autoencoders. The parametrization of this evaluation is the same as in the former analysis. The results are given in Figure 5.28.

The applied one-class SVM performs best during this benchmarking. However, the modified Hopfield neural networks are also capable of a good performance exceeding the performance of the Isolation Forest and Autoencoder.



**Figure 5.28:** Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection separating FAB subtype m1 and m2 from m3.

To summarize, it is concluded that modified Hopfield neural networks are capable of a classification and an anomaly detection for gene expressions. While some models yield in certain cases similar or sometimes better performances, the modified Hopfield neural networks are overall also useful. It should be noted that the use case of gene expression is not highlighting the main advantages of modified Hopfield neural networks since firstly the training data are fully labeled and not only true negative instances are available, and secondly the gene expressions are not only depending on a domain perspective of combinations but different intensity values.

### 5.1.3.5 Explainability of the Subtype Classification Model

While the modified Hopfield neural networks might not always be the best model from a pure performance perspective as seen in the prior section for the differentiation of FAB subtypes m1, m2, and m3, it adds high interpretability which might be useful for domain experts in certain use cases. Since authors in the field of

XAI often assume a trade-off between performance and interpretability (Barredo Arrieta et al. 2020), modified Hopfield neural networks are very useful in these use cases where performance should be exchanged for interpretability. This can be shown by further analyzing the weight matrices.

A first useful interpretability is given by the distance matrix  $\mathbf{D}$  between subtypes and the already discussed Figure 5.21. This analysis is a unique feature of modified Hopfield neural networks to increase the interpretability of subtype differences. As second analysis, the differences can be analyzed in more detail by calculating the delta weight matrices  $\Delta\mathbf{W}$  and highlighting the gene combinations which are very likely in one subtype, but not in the other compared subtypes. Since the number of analyzed genes  $V = 386$  is too high to visualize the  $V \times V$  weight matrices in a useful manner, the gene combinations with the highest difference can be extracted and listed. The five gene combinations which are more common in the FAB subtypes m1, m2, m4 compared to m3 and the gene combinations which are more common in subtype m3 but less likely in subtypes m1, m2, and m4 are given in Table 5.11. The absolute value of the score  $q_{ijkl}$  as per equation (4.43) is in addition computed and listed.

From the Table 5.11 is reasoned that combinations of, e.g., gene CD114 and CD300E are more common in FAB subtype m1 compared to subtype m3. However, if gene CD300E is combined with CD138, this combination is more common in subtype m3 than m1. Thus, high gene expressions of CD300E must be analyzed in the context whether gene CD114 or CD138 yield a higher intensity. It should be noted that high values of gene CD114 or CD138 alone are not predictive for the diagnosis since the diagonal entries of CD114 and CD138 yield only a score of 42% and 13.5% and thus, the combinations between genes of high intensity are more significant for the classification than the individual intensities per gene. The gene combinations of the other subtypes can be interpreted analogously. The calculated scores are also direct measures for computing the classification or anomaly detection and are capable of fully explaining the model's decision logic, which is an important requirement for XAI models. In addition, the requirement of completeness as described by Gilpin et al. (2018) is fulfilled in this case since the depth of the analysis can be adjusted by any user of the model by either analyzing



**Table 5.11:** Gene combinations more or less common in different FAB subtypes.

More common in m1 / m2 / m4 compared to m3								
m1			m2			m4		
Gene 1	Gene 2	Score	Gene 1	Gene 2	Score	Gene 1	Gene 2	Score
CD300E	CD114	61.6 %	CD1D	CD49f	63.2 %	CD158D	CD48	69.3 %
CD59	CD114	61.3 %	CD52	CD49f	59.3 %	CD107b	CD48	67.4 %
CD51	CD114	61.3 %	CD321	CD49f	59.3 %	CD158B2	CD48	67.4 %
CD319	CD114	61.3 %	CD300E	CD48	58.2 %	CD6	CD48	67.4 %
CD107b	CD114	61.3 %	CD113	CD114	57.9 %	CD85e	CD48	67.4 %

More common in m3 compared to m1 / m2 / m4								
m1			m2			m4		
Gene 1	Gene 2	Score	Gene 1	Gene 2	Score	Gene 1	Gene 2	Score
CD26	CD138	63.6 %	CD327	CD258	63.5 %	CD327	CD258	69.7 %
CD66a	CD138	62.1 %	CD186	CD85c	61.7 %	CD66e	CD258	69.7 %
CD66e	CD138	61.6 %	CD326	CD258	61.0 %	CD11b	CD258	68.8 %
CD300E	CD138	61.2 %	CD186	CD199	60.0 %	CD85e	CD258	64.7 %
CD49e	CD138	60.8 %	CD49e	CD186	59.6 %	CD85d	CD258	64.7 %

the full weight matrices, the scores of the most relevant combinations, or just the dendrogram and 2D projection of the distance matrix.

This information about common and uncommon combinations might provide domain experts with knowledge about relevant gene expressions and synergies of gene expressions resulting in an AML and APL diagnosis. Thus, domain experts can analyze the biological effects of these genes in more detail and focus attention to more relevant gene expressions. Table 5.11 add interpretability to the results of any conducted classification, and analyzing the weight matrices in more detail provides further insights for researchers in the field of AML. Therefore, modified Hopfield neural networks can be used as powerful XAI models in biological research helping to explain the effect of combinations of gene expressions for the risk of the development of certain types of cancer.

Concluding, this excursus proves that modified Hopfield neural networks have use cases beyond production planning in manufacturing and are versatile XAI models with high performances. Modified Hopfield neural networks prove themselves in particular useful for anomaly detection tasks with only few true positives labeled data available and can be applied in multiple domains.

## 5.2 Discussion

Firstly, the result from the use cases regarding the predictive performance of the modified Hopfield neural network is discussed. Here, the model outmatches the state-of-the-art AI models of Autoencoders, one-class SVM, and Isolation Forest during the benchmarking in an anomaly detection task. While Autoencoders and one-class SVMs perform on par in certain cases, the performance of the modified Hopfield neural network is leading independently of the use case. In addition, the modified Hopfield neural network also enables an explainability which cannot be achieved by the Autoencoder or the one-class SVM due to their "black box" design without a direct interpretable functionality. Simply based on the performance and its XAI capabilities, the proposed modified Hopfield neural networks are improving the state-of-the-art in anomaly detection models. Additional to the good predictive performance, it is possible to train the network with only true negative instances and a small amount of training data. Even more important for applications in production planning might be the recommendation system which enables a human-understandable correction process that other state-of-the-art models cannot provide. However, it is required that the analyzed objects of the modified Hopfield neural network are setup using an ontological model of the production system. In general, the model is designed to analyze data which can be structured by a graph network.

Similar to the anomaly detection tasks, the modified Hopfield neural networks perform very good in two-class classification tasks. However, the model loses its advantage over the state-of-the-art methods in multi-class classification tasks. In these scenarios, other models perform better and yield a higher predictive

accuracy. Using confusion matrices, e.g., Figure 5.25, it is shown that the model struggles to distinguish certain similar classes. In these applications, Random Forest shows a good performance compared to the other benchmarked models. However, the capability to extract a similarity measure between classes directly from the model is quite useful for a clustering and comparison in a multi-class scenario.

An additional advantage is the very robust parametrization of the modified Hopfield neural network. The (hyper-)parametrization procedure in the first use case shows very stable performances across different learning rates as well as reduced learning rates and decay values. This indicates that the model can easily be adapted to new use cases if embedded in a structured programming framework. Other state-of-the-art models are more difficult to parametrize, e.g., the performance of SVMs highly differs based on the selected kernel function. This makes the modified Hopfield neural network very practitioner-friendly since users can enroll the model without an extensive prior review of the underlying mathematical foundations, which is more important for, e.g., SVMs.

Secondly, the model's mathematical foundation can be compared to the layout of conventional NN designs. The main difference is the activation of the edges not vertices, the neurons in NNs. This is obvious when comparing the activation function of equation (4.1) of a modified Hopfield neural network and equation (3.2) of conventional NNs. This is also the main difference of modified Hopfield neural networks with the original proposal of Hopfield networks by Hopfield (1984). Additionally, the edges get activated between  $-1$  and  $1$  and not  $0$  and  $1$  as in conventional Hopfield networks. Another difference is that conventional Hopfield networks use an iterative activation of neurons in order to recreate patterns of active and inactive neurons from a given input signal based on memorized trained patterns. In the proposed modified Hopfield neural network, no activation is conducted but instead the energy measurement of equation (4.6) is evaluated and compared. Similar, the evaluation of the class membership is also different since the modified Hopfield neural network does not evaluate an output neuron but the energy level using a set-up similar to a hypothesis test.

Despite these differences, in both cases former input signals are imprinted into the network by a training procedure. The modified Hopfield neural network use the Widrow-Hoff rule of weight updates in equation (4.16), which is similar to the approach by conventional NNs in equation (3.13). The gradient of the loss function  $\nabla L$  corresponds to the applied gradient descent of the energy difference  $\Delta H$  via the optimization function in equation (4.13). Hence, the core learning procedure does not change and thus, the proposed modified Hopfield neural network is classified as a NN based on the aspect of the learning representation as defined in Figure 3.10 but is unsupervised regarding the feedback representation since no prior labeling is necessary in the case of anomaly detection tasks. In classification tasks, the modified Hopfield neural networks become a supervised approach since the class membership must be priorly labeled in the training data.

The recommendation procedure is also more similar to convectional Hopfield networks since it utilizes an activation of a neuron, in the derivation of this thesis also called vertex, in equation (4.42) similar to a perceptron as described in equation (3.3). In both cases is the weight times the input value responsible for the strength of the output. However, in modified Hopfield neural network only the strongest signal of all neurons, or rather vertices, is activated and not all neurons which exceed a threshold. The recommendation procedure is then again iterative as in conventional Hopfield networks and neurons, or rather vertices, get activated or deactivated based on their receiving signal strength. The activation function  $\Psi(\mathbf{x})$  in the case of modified Hopfield neural networks is the identity function.

It is also quite interesting to compare the modified Hopfield neural network model with the human learning experience. The model resembles the perceived stress by a human when seeing or interacting with an impression, e.g., an image or a task. This interaction becomes more stressful when interacting with new impressions previously not seen or interacted with. The energy function resembles the stress in interacting with these impressions. A high level of stress, attention, or tension occurs when interacting with new impressions and learning new things. On the other hand, low stress occurs when only known and memorized impressions are perceived. Regarding the learning process itself, by interacting with new impressions, the teaching experience is high. In contrast, when interacting with

already known impression, the interaction becomes less stressful and thus, there is also very little to learn. The applied learning process in equation (4.16) with more familiar graphs resulting in less adjustments of the weights of the network is also comparable to the human learning experience, as discussed within the context of Autoencoders by Sirois (2004).

In summary, modified Hopfield neural networks are easily interpretable, do not require a high effort in (hyper-)parametrization, and yield a competitive sensitivity in anomaly detection tasks without a necessary prior collection of an extensive amount of training data. Most notably, the model is data-driven and does not require domain and expert knowledge. It is concluded that modified Hopfield neural networks enable a valid anomaly detection method for graph structures by memorizing correct subgraphs and by providing estimations on how close new subgraphs, which are representing, e.g., process combinations, are to a memorized training data set.

## 5.3 Summary

This section showcases three use cases which highlight the application of modified Hopfield neural networks across different domains and for the application fields of anomaly detection, classification, and data generation.

The first use case shows an application in process planning. The model is trained and evaluated using a data set of 8674 correct process combinations with 586 different unique process types applied within the Body-in-White assembly. The objective is to identify anomalous process combinations and to provide a recommendation for correction. A comprehensive (hyper-)parametrization is conducted and shows that the model is stable regarding its input factors and for different partitions of training and test data by using a Monte Carlo cross-validation. The results of the anomaly detection model indicate a high predictive performance outmatching the state-of-the-art models of Autoencoders, one-class SVM, and Isolation Forest during a benchmarking. In summary, 96.7% of incorrect process

combinations can be detected with only 1.5% of correct combinations falsely classified as incorrect. In addition, the recommendation and explainability of the model in this use case is demonstrated and an embedding into the business processes of the Digital Factory is proposed.

The second use case shows a filtering and validation methodology for synthetic data generators of machinery sequence in production planning. Hereby, two approaches using transition matrices, a Bayesian network, and a transformer network are compared for their degree of realism. The modified Hopfield neural networks improve the presented methodology by adding a filtering via the anomaly detection capability and also by providing the validation mechanism via the classification model. The results are rather robust and indicate that the Bayesian network and transformer network are both suitable data generator models, which coincides with former results. This use case highlights, complementary to the first use case, the processing of sequence data by a modified Hopfield neural network as well as the similarity metric derived from its classification model. The application of modified Hopfield neural networks can therefore enhance data generation methodologies for simulations in production planning.

The third use case proves the applicability of the model outside of the manufacturing domain in a medical use case from the field of oncology. While the model is underperforming in a multi-class classification, it outperforms the state-of-the-art in a two-class classification and anomaly detection application within this domain in addition to a higher explainability of the results.

Subsequently, the use cases are discussed in detail and the limitations of the modified Hopfield neural networks are derived. A main limitation is the requirement of an ontological data set of correct instances for the training and the rather suboptimal performance in multi-class classification tasks. However, the explainability and high performance in two-class and anomaly detection cases are strongpoints of the introduced model. In addition, the high data requirements are a limitation of other AI models within the state-of-the-art and a trade-off for not requiring manual addition of expert knowledge.

Lastly, the relation to the common layout of neural networks is discussed. The main differences are the single fully-connected layer layout, as also common in Hopfield networks, and the activation of edges and not vertices. In addition, the evaluation of the energy function as output differentiates the layout from conventional neural networks.





## 6 Conclusion and Future Work

Lastly, this concluding section provides a summary of the key aspects and concluding remarks about this thesis, the research questions, and the proposed modified Hopfield neural network. Subsequently, an outlook and recommendations on future research based on the findings in this thesis is given.

### 6.1 Conclusion

First, the key aspects of the different section of this doctoral thesis are highlighted and discussed within the context of the whole thesis.

In Section 1, the core idea of a support system in production planning is defined. By using the knowledge of former production systems, it is possible to derive optimizations for newly planned systems. Faults in the planning data can be analyzed and countermeasures derived. Core of the support system is an AI engine, which utilizes production databases, digital models, and simulations in order to draw inferences for the planned production system. This AI engine can feedback recommendations to a planning expert. Based on these considerations, five main business-related objectives are defined which a support system should target: requirement definition, system design optimization, documentation, risk analysis, and analysis of fault events. Using these objectives, this section provides a comprehensive problem statement for the subsequent sections.

Section 2 aims at providing the reader with the required background of production planning, the Digital Factory as the paradigm which an AI support system is embedded in, knowledge-based systems, and fault detection. The context in

which an AI support system is deployed, the scope of production planning, and the use cases are therefore refined in this section. In more detail, the use case of fault detection during the rough planning phase of production planning is of particular interest since it enables a cost-efficient, feasible, and economic starting point to evaluate the planning scenarios using a support system which provides recommendations and evaluates the risks of fault events to occur. In order to provide context to planning scenarios and utilize human-understandable as well as machine-readable knowledge, the use of ontologies is recommended. Therefore, the use case of the AI support system is refined as a reasoning engine within a knowledge-based system in the Digital Factory in order to detect faults within the rough planning phases.

These use cases are then fleshed-out into two applied research questions in Section 3 which this thesis is based on. In order to find the gaps in the body of literature, a structured literature research is conducted. Using this, different state-of-the-art models can be differentiated, namely: rule-based models, optimization models, and AI models. In addition, the precise use cases in which these models are enrolled can be derived. Subsequently, the body of literature is examined in this section and a gap in the research is found. Process and resource planning use cases currently require a significant amount of manual domain knowledge in order to create rule-based models or optimization functions. Thus, an AI-based solution might circumvent the high-effort rule and optimization function definition and would enable a more efficient process planning. However, the maturity of the current state-of-the-art solutions using AI models in production planning is relatively low because the required high number of labeled data and the limited explainability are hurdles in their application. Based on these findings, two research questions are derived. The essence of these two research questions is the development of an explainable and accurate AI support system for process and resource planning without the requirement for manually added domain knowledge.

These questions are at first operationalized as principles and objectives of an AI support system in Section 4. Using these principles and objectives, the proposed AI model is introduced and its mathematical foundations explained in greater detail. The core idea is the modeling of ontological information of the planning

data as a graph network. Instances are represented by vertices and the relations between instances are the edges of the graph. This graph layout is also the base for the introduced modified Hopfield neural network. In contrast to commonly applied neural networks, the introduced modified Hopfield neural network is based on the activation of the edges not vertices. By defining an energy measurement based on Hopfield networks and the Lenz-Ising model, it is possible to evaluate the similarity of a new graph network to the set of given correct, also true negatives, graph networks applied during the training. This is possible by training the weights of the edges; high weights indicate that these connections between two vertices, ergo the combination of two underlying instances within the planning scenario of the domain application, are more commonly active than inactive. Thus, the energy criterion enables an anomaly detection and classification model. In addition, the weights of these edges provide a human-understandable and highly interpretable decision logic of the AI model. Lastly, the model should be implemented in the IT landscape considering the relevant interfaces of the support system to databases, data governance, and UX design.

Subsequently, Section 5 provides three use cases in order to prove the feasibility of the modified Hopfield neural network. The first use case from process planning is discussed in greater detail and highlights the high predictive performance of the model in finding anomalous process planning set-ups. In this use case, the introduced AI model outmatches the state-of-the-art AI models for process planning tasks. In addition, the recommendation procedure for correcting process combinations is shown and proven to be effective. The second use case shows an application in the field of simulation and synthetic data generation for machinery sequences in which the proposed AI model is successfully embedded. This use case proves the model's applicability for sequential information during process and resource planning. The third use case also shows the applicability of the AI models in use cases outside of manufacturing and production planning and thus, proves its viability as an explainable AI model on its own. Lastly, this section provides a discussion of the application scenarios, limitations, and similarities to commonly applied neural networks. In detail, the proposed AI model is limited by the graph structure of the underlying use case and requires a cleansed data

set of a sufficient number of true negative instances. However, the model is way less restrictive and complex in its necessary (hyper-)parametrization or database requirements than most state-of-the-art AI models.

The introduced AI model, the modified Hopfield neural network, is indeed capable of fulfilling all defined requirements, objectives, and research questions. Firstly, it enables a requirement and system design optimization during the rough production planning by its recommendation function, which is successfully demonstrated in the process planning use case of Section 5. Secondly, its high explanatory value also enables a clear and human-understandable method in order to analyze faults and risks, which are evaluated and detected by the anomaly detection model, within a planned layout or process combination. The structured embedding of this model is then capable of providing a clear documentation procedure for production planning by providing a case database. This case database is utilized by the classification capabilities of the introduced AI model.

The comprehensively discussed use cases demonstrate that the introduced AI support system can be successfully utilized in the under-researched fields of process and resource planning using AI models. The modified Hopfield neural network requires as a base just a structured data model, e.g., given by an ontology, but does not require any manual domain knowledge during its implementation. Therefore, the AI model is fully data-driven which separates itself from rule-based or optimization function-based approaches where a high initial manual effort in modeling is required. The required data amount is also comparably low and in addition, the model does not require fully labeled data sets but only true negative, meaning correct, instances of production systems for the training procedure. Thus, modified Hopfield neural networks overcome two common limitations of AI models: a high necessary amount of data and its labeling. It is therefore concluded that the introduced modified Hopfield neural network completely fulfills the first research question of this thesis.

Regarding the second research question, the modified Hopfield neural network fully adheres to XAI principles and enables a highly interpretable and human-understandable method for evaluating classification as well as anomaly detection

results. In addition, the model does not mask its complexity but can provide interpretations based on the user's requirements from high-level interpretations up to detailed derivations of the mathematical decision process of the model. In all use cases, the Hopfield neural network also demonstrated a high accuracy and predictive performance, even outmatching state-of-the-art models such as Autoencoders, one-class SVM, and Isolation Forest in anomaly detection tasks. Thus, it is concluded that the introduced modified Hopfield neural networks do fully encompass the requirements set by the second research question of this thesis.

To summarize, the proposed AI model fulfills the two research questions, which are derived from a structured literature review, and also enables the initially defined business case regarding support systems along the use cases of process and resource planning. Therefore, the objectives and tasks of this doctoral thesis are successfully fulfilled. This thesis provides the research and practitioner community with a novel and versatile AI model with high accuracy and XAI capabilities. This AI model in particular improves the current state-of-the-art in production planning by providing a solution for two under-investigated field within production planning.

## **6.2 Future Work**

Five relevant future research objectives and topics in the field of support systems in production planning emerge from this doctoral thesis: the development of fully autonomous planning tools, the generalization of the modified Hopfield neural network model for more use cases, the standardization of the model, the practical embedding of modified Hopfield neural networks into the business operations of a manufacturing company, and increased efforts in the elaboration of the Digital Factory definition as well as in the development and application of AI methods in production planning.

First, as discussed in the conclusion, the objective of developing an autonomous production planning is still not fully satisfied by the proposed Hopfield neural network. While it provides a recommender system which enables an accelerated and semi-automated production planning, main tasks are still manually conducted. This is in particular true for the many interfaces of the rough production planning with further planning activities in the detailed planning. An autonomous planning system would require a holistic system whereby the AI support system proposed here is a subdomain which needs to be completed by, among others, collaborative virtual environments, generative layout and process planning software, and a Cyber-Physical Systems system landscape as an enabler. This thesis adds another important piece into the objective of autonomous production planning and thus, enables further research progress regarding this topic.

Second, the modified Hopfield neural network layout proved itself highly useful in the excursus in the domain of gene networks. However, the capabilities of this novel NN layout should be carefully tested and evaluated in further use cases outside of the manufacturing domain. Furthermore, a set of more diverse parametrization should be tested in order to show the effects of, e.g., learning rate, decay, and reduced learning rate in multiple use cases and further prove for more use cases and domains that the network layout is stable with an easy and fast parametrization.

Third, the model can be further generalized for all kind of use cases in the anomaly detection and classification of graph networks. The author indeed plans to further develop the modified Hopfield neural network model in order to provide a library in the programming language Python for the research and practitioner community shared via the programming platform Git Hub. This library should enable a fast and simple embedding of modified Hopfield neural network into data science projects. Main objective of this library is a flexible set-up of the layout based on an imported ontology and a sufficient parallelization of the training procedure in order to speed up computational time.

Fourth, the demonstrated use cases are only proof-of-concepts and no comprehensive implementation is conducted of the proposed modified Hopfield neural

network model into the production planning landscape until now. However, the concept is currently under discussion by the operative management of Mercedes-Benz in order to test an embedding of the AI model into current planning tools in future projects. A full embedding and implementation into the IT landscape will most certainly bring up new practical challenges and practical questions. The author will closely accompany any implementation efforts and will provide the research community with relevant findings and lessons learned if an implementation project is effectively conducted.

Fifth, the discussion regarding Digital Twins, Cyber-Physical Systems, and the Digital Factory shows that further research is required in order to create a meaningful definition and partition of some of the most prominent research initiatives in manufacturing. On a similar note, the literature review of the state-of-the-art indicates that more novel ideas and concepts for the application of AI support systems in production planning, in particular layout and process planning, are needed. The maturity of research in process planning is still deficient and does require more attention by the research community.

This doctoral thesis builds on the knowledge of the scientific community and is intended to give subsequent ambitious doctoral students opportunities to develop their research topics. This work is a piece of the puzzle, which should make it possible to add further pieces in order to develop humanity one small step further at a time. The author is excited to see what will follow based on his five recommendations for future research topics and how the modified Hopfield neural network model will be further developed and utilized in the future.





# A Appendix

The connection matrix  $\mathbf{M}$ , which requires an edge between all pairs of active process groups and a loop if a process group is active twice, can be computed by algorithm 1 as proposed by Spoor and Weber (2024).

---

**Algorithm 1** Create a connection matrix

---

**Input:**

Process combination set  $\mathcal{C}$

**Output:**

Connection matrix  $\mathbf{M}$  of size  $V \times V$

```
1: Set  $i = 0, j = 0, a = 0, b = 0$ 
2: for  $i \leq V$  do
3:   for  $j \leq V$  do
4:     Set  $m_{ij} = -1$ 
5:     Set  $j = j + 1$ 
6:   end for
7:   Set  $i = i + 1$ 
8: end for
9: for  $a \leq \|\mathcal{C}\|$  do
10:  Set  $i$  as the  $a$ -th element of  $\mathcal{C}$  and  $b = a + 1$ 
11:  for  $b \leq \|\mathcal{P}\|$  do
12:    Set  $j$  as the  $b$ -th element of  $\mathcal{C}$  and  $b = b + 1$ 
13:    Set  $m_{ij} = 1$  and  $m_{ji} = 1$ 
14:  end for
15:  Set  $a = a + 1$ 
16: end for
```

---

The algorithms for the training, testing, and recommendation procedure of the modified Hopfield neural network are provided by Spoor and Weber (2024) using pseudo code. Algorithm 2 describes the training procedure, algorithm 3 describes the testing procedure, and algorithm 4 describes the recommendation procedure.

---

**Algorithm 2** Training Phase
 

---

**Input:**

Data set  $\mathbf{X}$  of size  $T \times V \times V$

**Parameter:**

Training rate  $\alpha$

Reduced training rate  $\beta$

Decay  $\gamma$

**Output:**

Trained weights matrix  $\mathbf{W}$  of size  $V \times V$  and correction value  $\theta$

```

1: Set  $t = 1$ 
2: for  $t \leq T$  do
3:   Set  $i = 1$ 
4:   for  $i \leq V$  do
5:     Set  $j = i$ 
6:     for  $j \leq V$  do
7:       if  $\sum_{j=0}^V m_{ij} > -V$  or  $\sum_{i=0}^V m_{ij} > -V$  then
8:         Set  $w_{ij} = w_{ij} + \alpha \left(1 - r \frac{t}{T}\right) (m_{ij} - w_{ij}) - d * w_{ij}$ 
9:         Set  $w_{ji} = w_{ij}$ 
10:      end if
11:      Set  $j = j + 1$ 
12:    end for
13:    Set  $i = i + 1$ 
14:  end for
15:  Set  $t = t + 1$ 
16: end for
17: Set  $\theta = \frac{2}{V(V+1)} \sum_{i=0}^V \sum_{j=i}^V w_{ij}$ 

```

---

---

**Algorithm 3** Testing Phase
 

---

**Input:**

Process connection matrix  $\mathbf{M}$  of size  $V \times V$

**Parameter:**

Trained weights matrix  $\mathbf{W}$  of size  $V \times V$  and correction value  $\theta$

Critical energy  $H_{crit}$

**Output:**

Measured energy  $H$

Statement if process classifies as outlier

```

1: Set  $a = 0$ ,  $i = 1$ , and  $H = 0$ 
2: for  $i \leq V$  do
3:   Set  $j = i$ 
4:   for  $j \leq V$  do
5:     Set  $H = H - w_{ij} * m_{ij}$ 
6:     if  $m_{ij} = 1$  then
7:       Set  $a = a + 1$ 
8:     end if
9:     Set  $j = j + 1$ 
10:  end for
11:  Set  $i = i + 1$ 
12: end for
13: Set  $H = H + a * \theta$ 
14: if  $H > H_{crit}$  then
15:   Outlier = TRUE
16: end if

```

---

**Algorithm 4** Recommendation Phase

---

**Input:**Process combination set  $\mathcal{C}$ **Parameter:**Trained weights matrix  $\mathbf{W}$  of size  $V \times V$  and correction value  $\theta$ Critical energy  $H_{crit}$ **Output:**List of recommendations  $\mathcal{C}^*$  and energy improvements  $H^*$  of size  $n + 1$ 

```
1: Set  $n = 0$ ,  $removed = \text{FALSE}$ 
2: Compute  $H$  and  $\mathbf{M}$  from set  $\mathcal{C}$  using algorithm 1 and 3
3: Set  $\mathcal{C}_0^* = \mathcal{C}$  and  $H_0^* = H$ 
4: while  $H_n^* > H_{crit}$  do
5:   Set  $I_{max} = 0$  and  $i = 1$ 
6:   for  $i \leq V$  do
7:     if  $\|\mathcal{C}_n^* \cap \{i\}\| > 1$  then
8:       Set  $\mathcal{C}^* = \mathcal{C}_n^* - \{i\}$ 
9:     end if
10:    if  $\|\mathcal{P}_n^* \cap \{i\}\| = 1$  then
11:      if  $removed = \text{FALSE}$  then
12:        Set  $\mathcal{C}^* = \mathcal{C}_n^* - \{i\}$  and  $removed = \text{TRUE}$ 
13:        Set  $i = i - 1$ 
14:      else
15:        Set  $\mathcal{C}^* = \mathcal{C}_n^* + \{i\}$  and  $removed = \text{FALSE}$ 
16:      end if
17:    end if
18:    if  $\|\mathcal{C}_n^* \cap \{i\}\| = 0$  then
19:      Set  $\mathcal{C}^* = \mathcal{C}_n^* + \{i\}$ 
20:    end if
21:    Compute  $H'$  and  $\mathbf{M}'$  from set  $\mathcal{C}^*$  using algorithm 1 and 3
22:    Compute  $I_i = H - H'$ 
23:    if  $I_i > I_{max}$  then
24:      Set  $I_{max} = I_i$ ,  $\mathcal{C}_{opt} = \mathcal{C}^*$ , and  $H_{opt} = H'$ 
25:    end if
26:    Set  $i = i + 1$ 
27:  end for
28:  Set  $n = n + 1$ 
29:  Set  $\mathcal{C}_n^* = \mathcal{C}_{opt}$  and  $H_n^* = H_{opt}$ 
30: end while
```

---

# List of Figures

1.1	Schematic procedure and objectives of an AI support system in early production planning phases in manufacturing. . . . .	3
1.2	Schematic set-up of an AI support system utilizing a resource, data, digital, and AI layer for the validation of planning scenarios. . . . .	4
1.3	Overview of important objectives and challenges along general requirements and the different layers of an AI support system. . . . .	7
1.4	Overview of the structure of the thesis and the subsequent guiding questions and objectives. . . . .	8
2.1	Production as an input/output transformation process as described by Dangelmaier (2009). . . . .	12
2.2	A transformation system model as described by Hubka and Eder (1988). . . . .	13
2.3	Fundamental and dispositive factors of production. . . . .	14
2.4	Manufacturing, production, part production, and assembly system based on the differentiation by Bellgran and Säfsen (2010). . . . .	14
2.5	System lifecycle of manufacturing systems based on Schenk and Wirth (2004). . . . .	16
2.6	Phases of production planning based on Hagemann (2022) and Kiefer et al. (2017). . . . .	17
2.7	The scope of production planning of the production system, assembly process, and production factors within manufacturing. . . . .	19
2.8	Schematic visualization of a car body construction assembly line with two sequential workstations. The robots are equipped with handling and joining tools. . . . .	21

2.9	Digital Factory in the lifecycle phases of a product across all business activities as defined by VDI 4499 - Part 1 (2008). The fields of the production are separated by color. . . . .	25
2.10	Digital Factory operations in the lifecycle phases of a production system as defined by VDI 4499 - Part 2 (2011). . . . .	26
2.11	Method categories within the Digital Factory concept as defined by Bracht et al. (2018). The categorization of the method used in the introduced support system is highlighted. . . . .	27
2.12	Scope, technologies, enablers, and degree of maturity of the Digital Factory, Digital Twin, Cyber-Physical Systems, and Smart Manufacturing. . . . .	31
2.13	An exemplary (and incomplete) ontology of the automotive industry and an exemplary implementation for specific instances. . .	34
2.14	Information models utilizing the PPR ontology in the rough production planning phase based on Hagemann (2022). . . . .	35
2.15	Architecture of a knowledge graph as defined by Ehrlinger and Wöß (2016). . . . .	36
2.16	Sketch of the ability to detect faults and to correct the faults during planning stages. . . . .	40
2.17	Sketch of common outlier types as described by Lindemann et al. (2021). . . . .	42
2.18	Sketch of clustering and classification tasks. . . . .	45
3.1	Schematic of the applied methodology in the literature review. . . . .	48
3.2	Syntax and keywords applied as search terms in the online search engine. . . . .	49
3.3	An exemplary conditional statement using two operands. . . . .	52
3.4	Visualization of an exemplary rule-based decision process as a tree structure. . . . .	53
3.5	Approach of a rule evaluation by a fuzzy logic following the Mamdani system. . . . .	55
3.6	Exemplary fuzzification of a continuous input variable into a fuzzy set with a membership assignment "low" with a value of 0.7 and "medium" with a value of 0.3. . . . .	56
3.7	Visualization of an optimization problem trapped within a local optimum. . . . .	60

3.8	Simplified morphological analysis of AI models in manufacturing applications. . . . .	63
3.9	An exemplary model of a Perceptron with three input features using a weighted summation and a signum activation function to create an output. . . . .	66
3.10	Schematic diagram of the presented activation functions. . . . .	69
3.11	An exemplary model of a Multi-layer network with four input features using two hidden layers in order to create a two-dimensional output. . . . .	71
3.12	An exemplary model of an Autoencoder with four input features and a latent space with only two dimensions. The decoding and encoding are conducted in one layer each. . . . .	74
3.13	Schematic concept of a SVM using two separable classes without overlap. . . . .	82
3.14	Schematic concept of a SVM using two separable classes with three overlapping objects. . . . .	83
3.15	Schematic concept of the kernel trick transforming a data set using a higher dimensional representation in which the classes become linearly separable. . . . .	86
3.16	Illustrative KNN classifier using the $k = 7$ nearest neighbor within Euclidean distance. . . . .	89
3.17	Exemplary miss-classification of linearly separable objects by a KNN classifier using the $k = 7^{th}$ neighbor within Euclidean distance. . . . .	90
3.18	Illustrative outlier detection with LOF using $k = 3$ . The neighbors (green) of the outlier (orange) have lower reachability distances and thus, higher local reachability densities. . . . .	92
3.19	Evaluation of prominence, maturity, and primarily applied models for production planning use cases in the body of literature. . . . .	95
3.20	Overview of the characteristics, advantages, limitations, and use cases of the different methodical approaches for support systems in production planning. . . . .	99
4.1	Five applied design principles of an AI support system based on Spoor et al. (2022a). . . . .	106
4.2	Objectives of an AI support system separating the associated tasks related to (1) anomaly detection and (2) classification. . . . .	109

4.3	Schematic overview of the sequential inference procedure and of the elements of the introduced support system. . . . .	110
4.4	Data preprocessing of a configuration within a specific scenario by embedding the configuration into an ontology to convert the configuration into the applied input data format. . . . .	113
4.5	Schematic display of combinations as a network using $V = 4$ objects and a maximum of $E_{max} = 10$ edges. Unused edges and vertices are grayed out. . . . .	116
4.6	Sketch of the proposed procedure in five phases for the anomaly detection and evaluation of combinations using modified Hopfield nets. . . . .	118
4.7	Sketch of the proposed procedure in five phases for the classification of combinations using modified Hopfield nets. . . . .	119
4.8	Sketch of proposed intensity thresholds for combinations with objects using intensities. . . . .	121
4.9	Training procedure of two exemplary cases using (a) only one identical combination and (b) four equally often but randomly arranged combinations. . . . .	129
4.10	Schematic display of sequences as a network using $V = 4$ objects and a maximum of $E_{max} = 16$ directed edges. Unused edges and vertices are grayed out. . . . .	133
4.11	Schematic display of ontologies with multiple concepts as a network. The resulting sub-connection matrices are differently colored, unused edges or vertices are grayed out. . . . .	136
4.12	Sketch of an annotated edge using a certain logical relation $r \in \mathcal{R}$ between two vertices representing the objects. . . . .	138
4.13	Exemplary weight matrix visualized by a heatmap plot. Highly common combinations are colored red, anomalous combinations are colored blue. . . . .	155
4.14	Exemplary delta weight matrix between two classes visualized by a heatmap plot. Shared common combinations are colored blue, distinguishing combinations are colored red. . . . .	157
4.15	Accuracy and interpretability assessment for different ML models by Angelov et al. (2021) including modified Hopfield nets for the evaluation of combinations. . . . .	159



4.16	Schematic design of the IT system architecture for the management of the required data transfer from different subsystems into the product planning tool. . . . .	161
4.17	Schematic mock-up of a software using the graph-based visualization of the weight matrix of a manufacturing cell for an explainability of a detected anomaly. . . . .	166
4.18	Schematic mock-up of a software for the recommendation procedure in order to correct the manufacturing cell of Figure 4.17. .	167
5.1	Training procedure using the moving average of the stimulated energy over 100 iterations, its 3-standard derivation area, and the equilibrium energy. . . . .	184
5.2	Distribution of the energy level of correct, random, and incorrect process combinations displayed as violin plot. . . . .	186
5.3	Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection task. . . . .	189
5.4	Resulting weight matrix after the training procedure visualized as a heatmap plot. . . . .	191
5.5	Energy levels after multiple iterations of recommendation with single activations or deactivations of processes per iterative step. . . . .	193
5.6	Layout of a side panel assembly plant. FS71 is located in the lower right of the map. The position of the exemplary process is marked in red (Spoor and Weber 2024). . . . .	196
5.7	Layout of the first cell of the manufacturing line FS71. The resources are marked in green. Robots and conveyor belt are named with arrows (Spoor and Weber 2024). . . . .	197
5.8	Analyzed process chart with 020RB 100 marked in blue, 020RB 200 marked in green, 020RB 300 marked in violet, and 020RB 400 marked in orange (Spoor and Weber 2024). . . . .	198
5.9	Proposal for an embedding of the process planning use case into the business processes of production planning by Spoor and Weber (2024). . . . .	202
5.10	Schematic methodology to generate, validate, and compare synthetic data as described by Matthes et al. (2023). . . . .	207

5.11	Schematic methodology to generate, validate, and compare synthetic data utilizing a modified Hopfield neural network for validation and filtering by Spoor et al. (2024). . . . .	209
5.12	Exemplary simple transition matrix (ST) of 3 machines for the simulation of sequences. . . . .	210
5.13	Exemplary transition matrix with added source and sink condition (TASS) of 3 machines for the simulation of sequences. . .	211
5.14	Simplified schematic transformer architecture. . . . .	212
5.15	Training procedure using the moving average of the stimulated energy over 100 iterations, its 3-standard derivation area, and the equilibrium energy (Spoor et al. 2024). . . .	220
5.16	Distribution of the real-world and synthetic data sets' energy levels displayed as violin plots (Spoor et al. 2024). . . . .	221
5.17	2D projection of the distance matrix between the energy distributions by a test set using randomly drawn subsets of the real-world data set for training of the weight matrix. . . . .	224
5.18	2D projection of the distance matrix between the energy distributions in Figure 5.16 using the Wasserstein metric (Spoor et al. 2024). . . . .	224
5.19	2D projection of the distance matrix between the trained weight matrices of randomly drawn subsets of the real-world data set. . . . .	226
5.20	2D projection of the distance matrix between the weight matrices of the real-world data set and the synthetic data sets using the Frobenius norm. . . . .	227
5.21	Dendrogram and 2D projection of the distance matrix for the different analyzed FAB subtypes of AML. . . . .	232
5.22	Training procedure of FAB subtype m4 using the moving average of the stimulated energy over 5 iterations, its 3-standard derivation area, and the equilibrium energy. . . . .	235
5.23	Energy levels of the gene combinations of all patients with FAB subtypes m3 and m4 for a weight matrix trained on subtype m4 using a 60% training data size. . . . .	236

---

5.24	Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection separating FAB subtype m3 from m4. . . . .	237
5.25	Confusion matrix of modified Hopfield neural networks for a classification of FAB subtype m1, m2, and m3. . . . .	238
5.26	Training procedure of FAB subtype m1 and m2 using the moving average of the stimulated energy over 5 iterations, its 3-standard derivation area, and the equilibrium energy. . . . .	239
5.27	Energy levels of the gene combinations of all patients with FAB subtypes m1, m2 and m3 for a weight matrix trained on subtype m1 and m2 using a 80% training data size. . . . .	240
5.28	Benchmarking of Hopfield nets, one-class SVMs, Isolation Forests, and Autoencoders by a ROC curve for an anomaly detection separating FAB subtype m1 and m2 from m3. . . . .	241



# List of Tables

3.1	Selection of rule-based models for production planning. . . . .	57
3.2	Selection of optimization models for production planning. . . . .	61
3.3	Selection of Neural Network models for production planning. . . . .	77
3.4	Selection of Tree-based models for production planning. . . . .	81
3.5	Selection of Support Vector Machines for production planning. . . .	88
3.6	Selection of neighborhood and density-based models for production planning. . . . .	93
3.7	Selection of statistical approaches for production planning. . . . .	94
5.1	Overview of the basic information of the analyzed use cases. . . . .	172
5.2	Description of the applied data set of process combinations. . . . .	176
5.3	Effect of parametrization evaluated by the FPR using a given 90.0% TPR. . . . .	180
5.4	Effect of different training sizes evaluated by the FPR using a given 90.0% TPR. . . . .	181
5.5	Applied parameters for the training procedure. . . . .	183
5.6	Recommended process group adjustments per iterative step. . . . .	194
5.7	Description of the applied data set of machinery sequences. . . . .	219
5.8	Number of patients per AML FAB subtype. . . . .	231
5.9	Accuracy of AI models for FAB subtype m3 and m4 classification. . .	234
5.10	Accuracy of AI models for FAB subtype m3 and m4 classification. .	238
5.11	Gene combinations more or less common in different FAB subtypes.	243



# Own Publications

## Journal Articles

Jan Michael Spoor. Improving customer segmentation via classification of key accounts as outliers. *Journal of Marketing Analytics*, 11(4):747–760, 2023. doi: 10.1057/s41270-022-00185-4.

Jan Michael Spoor and Jens Weber. Evaluation of process planning in manufacturing by a neural network based on an energy definition of hopfield nets. *Journal of Intelligent Manufacturing*, 35(6):2625–2643, 2024. doi: 10.1007/s10845-023-02158-5.

Jan Michael Spoor, Jens Weber, and Jivka Ovtcharova. Anomaly detection and prediction evaluation for discrete nonlinear dynamical systems. *Transactions of the Institute of Measurement and Control*, 2023. doi: 10.1177/01423312231203030.

Dawid Stade, Jan Michael Spoor, Martin Manns, and Jivka Ovtcharova. Process time distribution simulation in robotic assembly line balancing. *International Journal of Production Research*, pages 1–18, 2024. doi: 10.1080/00207543.2024.2416570.

## Conference Papers

Jan Michael Spoor and Jens Weber. Schematic Categorization and Definition of Applied and Target-oriented Digital Twins. In Sören Bergmann, Niclas

Feldkamp, Rainer Souren, and Steffen Straßburger, editors, *20. ASIM Fachtagung Simulation in Produktion und Logistik 2023*, pages 103–112, 2023. doi: 10.22032/dbt.57476.

Jan Michael Spoor, Jens Weber, Simon Hagemann, and Frederik Simon Bäumer. Concept of an Inference Procedure for Fault Detection in Production Planning. In *The Fourteenth International Conference on Pervasive Patterns and Applications*, PATTERNS '22, pages 10–17, Barcelona, Spain, 2022a.

Jan Michael Spoor, Jens Weber, and Jivka Ovtcharova. A Definition of Anomalies, Measurements, and Predictions in Dynamical Engineering Systems for Streamlined Novelty Detection. In *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 675–680, 2022b. doi: 10.1109/CoDIT55151.2022.9804125.

Jan Michael Spoor, Christian Graewe, and Jens Weber. Requirements for the Application of Knowledge Graphs in Automotive Manufacturing. In *The Fifteenth International Conference on Pervasive Patterns and Applications*, PATTERNS '23, pages 12–17, Nice, France, 2023a.

Jan Michael Spoor, Dawid Stade, Jivka Ovtcharova, and Martin Manns. Reverse engineering of feedback control systems and classification of resistance spot welding times using random forest in automotive body-in-white manufacturing. In *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 1113–1119, 2023b. doi: 10.1109/CoDIT58514.2023.10284212.

Jan Michael Spoor, Jens Weber, and Jivka Ovtcharova. A proposal for formalization and definition of anomalies in dynamical systems. In Paula Brito, José G. Dias, Berthold Lausen, Angela Montanari, and Rebecca Nugent, editors, *Classification and Data Science in the Digital Age*, pages 373–381, Cham, 2023c. Springer International Publishing. doi: 10.1007/978-3-031-09034-9\_40.

Jan Michael Spoor, Marvin Matthes, Martin Krockert, and Jens Weber. Validation Towards Realistic Synthetic Datasets in Production Planning. In



2024 *Winter Simulation Conference (WSC)*, pages 2703–2714, 2024. doi: 10.1109/WSC63780.2024.10838928.

Dawid Stade, Jan Michael Spoor, and Martin Manns. A Multimodal Distribution Model of Stochastic Process Times in Resistance Spot Welding. In Francesco Gabriele Galizia and Marco Bortolini, editors, *Production Processes and Product Evolution in the Age of Disruption*, pages 589–597, Cham, 2023. Springer International Publishing. doi: 10.1007/978-3-031-34821-1\_64.



# References

- Hervé Abdi, Dominique Valentin, Betty Edelman, and Alice J. O'Toole. A Widrow–Hoff Learning Rule for a Generalization of the Linear Auto-associator. *Journal of Mathematical Psychology*, 40(2):175–182, 1996. doi: 10.1006/jmps.1996.0017.
- Anam Abid, Muhammad Tahir Khan, and Javaid Iqbal. A review on fault detection and diagnosis techniques: basics and beyond. *Artificial Intelligence Review*, 54(5):3639–3664, 2021. doi: 10.1007/s10462-020-09934-2.
- Emad H. Abualsauod. Machine learning based fault detection approach to enhance quality control in smart manufacturing. *Production Planning & Control*, pages 1–9, 2023. doi: 10.1080/09537287.2023.2175736.
- Amina Adadi and Mohammed Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018. doi: 10.1109/ACCESS.2018.2870052.
- Sebastian Adolphy, Hendrik Grosser, Lucas Kirsch, and Rainer Stark. Method for automated structuring of product data and its applications. *Procedia CIRP*, 38:153–158, 2015. doi: 10.1016/j.procir.2015.07.063.
- Charu C. Aggarwal. *Outlier Analysis*. Springer International Publishing, Cham, 2017. doi: 10.1007/978-3-319-47578-3.
- Charu C. Aggarwal. *Neural Networks and Deep Learning*. Springer International Publishing, Cham, 2018. doi: 10.1007/978-3-319-94463-0.
- Kwabena Agyapong-Kodua, Csaba Haraszko, and István Németh. Recipe-based Integrated Semantic Product, Process, Resource (PPR) Digital Modelling

- Methodology. *Procedia CIRP*, 17:112–117, 2014. doi: 10.1016/j.procir.2014.03.118.
- Majid Al-Ruithe, Elhadj Benkhelifa, and Khawar Hameed. A systematic literature review of data governance and cloud data governance. *Personal and Ubiquitous Computing*, 23(5):839–859, 2019. doi: 10.1007/s00779-017-1104-3.
- Marcel Albus and Marco F. Huber. Resource reconfiguration and optimization in brownfield constrained Robotic Assembly Line Balancing Problems. *Journal of Manufacturing Systems*, 67:132–142, 2023. doi: 10.1016/j.jmsy.2023.01.001.
- Saleh M. Amaitik and S. Engin Kiliç. An intelligent process planning system for prismatic parts using STEP features. *The International Journal of Advanced Manufacturing Technology*, 31(9):978–993, 2007. doi: 10.1007/s00170-005-0269-5.
- Plamen P. Angelov, Eduardo A. Soares, Richard Jiang, Nicholas I. Arnold, and Peter M. Atkinson. Explainable artificial intelligence: an analytical review. *WIREs Data Mining and Knowledge Discovery*, 11(5):e1424, 2021. doi: 10.1002/widm.1424.
- Farhad Angizeh, Hector Montero, Abhinay Vedpathak, and Masood Parvania. Optimal production scheduling for smart manufacturers with application to food production planning. *Computers & Electrical Engineering*, 84:106609, 2020. doi: 10.1016/j.compeleceng.2020.106609.
- Jorge F. Arinez, Qing Chang, Robert X. Gao, Chengying Xu, and Jianjing Zhang. Artificial Intelligence in Advanced Manufacturing: Current Status and Future Outlook. *Journal of Manufacturing Science and Engineering*, 142(11):110804, 2020. doi: 10.1115/1.4047855.
- Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bernetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020. doi: 10.1016/j.inffus.2019.12.012.

- Barbara Rita Barricelli, Elena Casiraghi, and Daniela Fogli. A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications. *IEEE Access*, 7:167653–167671, 2019. doi: 10.1109/ACCESS.2019.2953499.
- Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. *SIGKDD Explor. Newsl.*, 6(1):20–29, 2004. doi: 10.1145/1007730.1007735.
- André Bauer, Simon Trapp, Michael Stenger, Robert Leppich, Samuel Kounev, Mark Leznik, Kyle Chard, and Ian Foster. Comprehensive Exploration of Synthetic Data Generation: A Survey. 2024. doi: 10.48550/arXiv.2401.02524.
- Sara Moghadaszadeh Bazaz, Mika Lohtander, and Juha Varis. Availability of Manufacturing data resources in Digital Twin. *Procedia Manufacturing*, 51: 1125–1131, 2020. doi: 10.1016/j.promfg.2020.10.158.
- Monica Bellgran and Kristina Säfsten. *Production Development*. Springer London, London, 2010. doi: 10.1007/978-1-84882-495-9.
- Luigi Bellomarini, Georg Gottlob, Andreas Pieris, and Emanuel Sallinger. Swift Logic for Big Data and Knowledge Graphs. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2–10, 2017. doi: 10.24963/ijcai.2017/1.
- J. M. Bennett, D. Catovsky, Marie-Therese Daniel, G. Flandrin, D. A. G. Galton, H. R. Gralnick, and C. Sultan. Proposals for the Classification of the Acute Leukaemias French-American-British (FAB) Co-operative Group. *British Journal of Haematology*, 33(4):451–458, 1976. doi: 10.1111/j.1365-2141.1976.tb03563.x.
- Pouria G. Bigvand, Alexander Fay, Rainer Drath, and Pablo Rodriguez Carrion. Concept and development of a semantic based data hub between process design and automation system engineering tools. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, 2016. doi: 10.1109/ETFA.2016.7733734.

- A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano. A Review on Outlier/Anomaly Detection in Time Series Data. *ACM Comput. Surv.*, 54(3):56, 2021. doi: 10.1145/3444690.
- Leonardo Borba, Marcus Ritt, and Cristóbal Miralles. Exact and heuristic methods for solving the Robotic Assembly Line Balancing Problem. *European Journal of Operational Research*, 270(1):146–156, 2018. doi: 10.1016/j.ejor.2018.03.011.
- Nils Boysen, Malte Fliedner, and Armin Scholl. A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2): 674–693, 2007. doi: 10.1016/j.ejor.2006.10.010.
- U. Bracht and T. Masurat. The Digital Factory between vision and reality. *Computers in Industry*, 56(4):325–333, 2005. doi: 10.1016/j.compind.2005.01.008.
- Uwe Bracht, Dieter Geckler, and Sigrid Wenzel. *Digitale Fabrik: Methoden und Praxisbeispiele*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2018. doi: 10.1007/978-3-662-55783-9\_2.
- Alex Brasington, Joshua Halbritter, Roudy Wehbe, and Ramy Harik. Bayesian optimization for process planning selections in automated fiber placement. *Journal of Composite Materials*, 56(28):4275–4296, 2022. doi: 10.1177/00219983221129010.
- Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324.
- Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. *SIGMOD Rec.*, 29(2):93–104, 2000. doi: 10.1145/335191.335388.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu,

- Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- Stephen G. Brush. History of the Lenz-Ising Model. *Rev. Mod. Phys.*, 39:883–893, 1967. doi: 10.1103/RevModPhys.39.883.
- Georg Buchgeher, David Gabauer, Jorge Martinez-Gil, and Lisa Ehrlinger. Knowledge Graphs in Manufacturing and Production: A Systematic Literature Review. *IEEE Access*, 9:55537–55554, 2021. doi: 10.1109/ACCESS.2021.3070395.
- Holger Burr. *Informationsmanagement an der Schnittstelle zwischen Entwicklung und Produktionsplanung im Karosserierohbau*. PhD thesis, Universität des Saarlandes, 2008. doi: 10.22028/D291-22495.
- Raffaella Cagliano and Gianluca Spina. Advanced manufacturing technologies and strategically flexible production. *Journal of Operations Management*, 18 (2):169–190, 2000. doi: 10.1016/S0272-6963(99)00022-4.
- Mauro Capestro and Steffen Kinkel. Industry 4.0 and Knowledge Management: A Review of Empirical Studies. In Marco Bettiol, Eleonora Di Maria, and Stefano Micelli, editors, *Knowledge Management and Industry 4.0: New Paradigms for Value Creation*, pages 19–52, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-43589-9\_2.
- Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly Detection: A Survey. *ACM Comput. Surv.*, 41(3):15, 2009. doi: 10.1145/1541880.1541882.
- Niechen Chen. An evolutionary neural network approach to machining process planning: A proof of concept. *Procedia Manufacturing*, 53:690–696, 2021. doi: 10.1016/j.promfg.2021.06.083.
- Ying Cheng, Yongping Zhang, Ping Ji, Wenjun Xu, Zude Zhou, and Fei Tao. Cyber-physical integration for moving digital factories forward towards smart

- manufacturing: a survey. *The International Journal of Advanced Manufacturing Technology*, 97(1):1209–1221, 2018. doi: 10.1007/s00170-018-2001-2.
- Heejeong Choi, Donghwa Kim, Joungee Kim, Jina Kim, and Pilsung Kang. Explainable anomaly detection framework for predictive maintenance in manufacturing systems. *Applied Soft Computing*, 125:109147, 2022. doi: 10.1016/j.asoc.2022.109147.
- Parames Chutima. A comprehensive review of robotic assembly line balancing problem. *Journal of Intelligent Manufacturing*, 33(1):1–34, 2022. doi: 10.1007/s10845-020-01641-7.
- Marcello Colledani, Dávid Gyulai, László Monostori, Marcello Urgo, Johannes Unglert, and Fred Van Houten. Design and management of reconfigurable assembly lines in the automotive industry. *CIRP Annals*, 65(1):441–446, 2016. doi: 10.1016/j.cirp.2016.04.123.
- Roberto Confalonieri, Ludovik Coba, Benedikt Wagner, and Tarek R. Besold. A historical perspective of explainable Artificial Intelligence. *WIREs Data Mining and Knowledge Discovery*, 11(1):e1391, 2021. doi: 10.1002/widm.1391.
- Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. doi: 10.1007/BF00994018.
- T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. doi: 10.1109/TIT.1967.1053964.
- Jovani Dalzochio, Rafael Kunst, Edison Pignaton, Alecio Binotto, Srijnan Sanyal, Jose Favilla, and Jorge Barbosa. Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. *Computers in Industry*, 123:103298, 2020. doi: 10.1016/j.compind.2020.103298.
- Simon Thevenin Dan Luo and Alexandre Dolgui. A state-of-the-art on production planning in Industry 4.0. *International Journal of Production Research*, 61(19):6602–6632, 2023. doi: 10.1080/00207543.2022.2122622.



- Wilhelm Dangelmaier. *Theorie der Produktionsplanung und -steuerung*. Springer, Berlin, Heidelberg, 2009. doi: 10.1007/978-3-642-00633-3.
- Jianxin Deng, Bin Xie, Dongdong You, Ling Wang, Xiusong Wu, Gang Liu, and Jiawei Liang. Process parameters design of squeeze casting through an improved KNN algorithm and existing data. *Journal of Manufacturing Processes*, 84: 1320–1330, 2022. doi: 10.1016/j.jmapro.2022.10.074.
- Berend Denkena, Marc-André Dittrich, Siebo Claas Stamm, and Vannila Prasanthan. Knowledge-based process planning for economical re-scheduling in production control. *Procedia CIRP*, 81:980–985, 2019. doi: 10.1016/j.procir.2019.03.238.
- Berend Denkena, Marc-André Dittrich, Hai Nam Nguyen, and Konrad Bild. Self-optimizing process planning of multi-step polishing processes. *Production Engineering*, 15(3):563–571, 2021. doi: 10.1007/s11740-021-01042-6.
- DIN EN 13306:2018-02. Maintenance – Maintenance terminology. Standard ICS 01.040.03, 03.080.10, Deutsches Institut für Normung e.V., Berlin, Germany, 2018.
- Khaled A. Alkhaledi Doraid Dalalah, Udechukwu Ojiako and Alasdair Marshall. A support vector machine model for due date assignment in manufacturing operations. *Journal of Industrial and Production Engineering*, 40(1):68–85, 2023. doi: 10.1080/21681015.2022.2059791.
- Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 210–215, 2018. doi: 10.23919/MIPRO.2018.8400040.
- Lisa Ehrlinger and Wolfram Wöß. Towards a Definition of Knowledge Graphs. In *International Conference on Semantic Systems*, 2016.

- Grischan Engel, Thomas Greiner, and Sascha Seifert. Ontology-Assisted Engineering of Cyber-Physical Production Systems in the Field of Process Technology. *IEEE Transactions on Industrial Informatics*, 14(6):2792–2802, 2018. doi: 10.1109/TII.2018.2805320.
- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, Portland, OR, 1996.
- Simon Fahle, Christopher Prinz, and Bernd Kuhlenkötter. Systematic review on machine learning (ML) methods for manufacturing processes – Identifying artificial intelligence (AI) methods for field application. *Procedia CIRP*, 93: 413–418, 2020. doi: 10.1016/j.procir.2020.04.109.
- Ludwig Fahrmeir, Hans Wolfgang Brachinger, Alfred Hamerle, and Gerhard Tutz. *Multivariate statistische Verfahren*. Walter de Gruyter, Berlin, 1996.
- Mohammad Fakhar Manesh, Massimiliano Matteo Pellegrini, Giacomo Marzi, and Marina Dabic. Knowledge Management in the Fourth Industrial Revolution: Mapping the Literature and Scoping Future Avenues. *IEEE Transactions on Engineering Management*, 68(1):289–300, 2021. doi: 10.1109/TEM.2019.2963489.
- Christina Feilmayr and Wolfram Wöß. An analysis of ontologies and their success factors for application to business. *Data & Knowledge Engineering*, 101:1–23, 2016. doi: 10.1016/j.datak.2015.11.003.
- Ederson Carvalhar Fernandes, Lucas Iuri dos Santos, Juliane Andressa Camatti, Liam Brown, and Milton Borsato. Flexible production data generator for manufacturing companies. *Procedia Manufacturing*, 51:1478–1484, 2020. doi: 10.1016/j.promfg.2020.10.205.
- Borja Ramis Ferrer, Bilal Ahmad, Daniel Vera, Andrei Lobov, Robert Harrison, and José Luis Martínez Lastra. Product, process and resource model coupling for knowledge-driven assembly automation. *at – Automatisierungstechnik*, 64(3):231–243, 2016. doi: 10.1515/auto-2015-0073.

- Evelyn Fix and Joseph L Hodges. Discriminatory analysis, nonparametric discrimination: consistency properties. 1951.
- R. Flamary, N. Courty, A. Gramfort, M.Z. Alaya, A. Boissunon, S. Chambon, L. Chapel, A. Corenflos, K. Fatras, N. Fournier, L. Gautheron, N.T.H. Gayraud, H. Janati, A. Rakotomamonjy, I. Redko, A. Rolet, A. Schutz, V. Seguy, D.J. Sutherland, R. Tavenard, A. Tong, and T. Vayer. POT: Python Optimal Transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.
- Ralph Foorthuis. On the nature and types of anomalies: a review of deviations in data. *International Journal of Data Science and Analytics*, 12(4):297–331, 2021. doi: 10.1007/s41060-021-00265-1.
- Emmanuel Francalanza, Jonathan Borg, and Carmen Constantinescu. A knowledge-based tool for designing cyber physical production systems. *Computers in Industry*, 84:39–58, 2017. doi: 10.1016/j.compind.2016.08.001.
- Marco Franke, Konstantin Klein, Karl A. Hribernik, and Klaus-Dieter Thoben. Semantic Data Integration Approach for the Vision of a Digital Factory. In Kai Mertins, Ricardo Jardim-Gonçalves, Keith Popplewell, and João P. Mendonça, editors, *Enterprise Interoperability VII*, pages 77–86, Cham, 2016. Springer International Publishing.
- Jonas Friederich and Sanja Lazarova-Molnar. Towards data-driven reliability modeling for cyber-physical production systems. *Procedia Computer Science*, 184:589–596, 2021. doi: 10.1016/j.procs.2021.03.073.
- Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002. doi: 10.1016/S0167-9473(01)00065-2.
- Mikhail Galkin, Sören Auer, Haklae Kim, and Simon Scerri. Integration Strategies for Enterprise Knowledge Graphs. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, pages 242–245, 2016. doi: 10.1109/ICSC.2016.24.
- Mikhail Galkin, Sören Auer, Maria-Esther Vidal, and Simon Scerri. Enterprise Knowledge Graphs: A Semantic Approach for Knowledge Management in the

- Next Generation of Enterprise Information Systems. In *Proceedings of the 19th International Conference on Enterprise Information Systems – Volume 2: ICEIS*, pages 88–98, 2017. doi: 10.5220/0006325200880098.
- Alberto Garre, Mari Carmen Ruiz, and Eloy Hontoria. Application of Machine Learning to support production planning of a food industry in the context of waste generation under uncertainty. *Operations Research Perspectives*, 7: 100147, 2020. doi: 10.1016/j.orp.2020.100147.
- Ludo F. Gelders and Luk N. Van Wassenhove. Production planning: a review. *European Journal of Operational Research*, 7(2):101–110, 1981. doi: 10.1016/0377-2217(81)90271-X.
- Jan Markus Gelfgren, Hélène Arvis, Simon Hagemann, and Sigrid Wenzel. Mixed-Integer Programming and State-of-the-Art Heuristic Approaches for the Scheduling of Modular Automotive Body-in-White Production. In Francesco Gabriele Galizia and Marco Bortolini, editors, *Production Processes and Product Evolution in the Age of Disruption*, pages 225–234, Cham, 2023. Springer International Publishing.
- Emilia Gelwer, Jens Weber, and Frederik Simon Bäumer. A Concept of Enabling Data Consistency Checks Between Production and Production Planning Using AI. In *Proceedings of the 17th International Conference on Applied Computing*, pages 139–142, 2020.
- Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89, 2018.
- Franco Giustozzi, Julien Saunier, and Cecilia Zanni-Merk. Context Modeling for Industry 4.0: an Ontology-Based Proposal. *Procedia Computer Science*, 126: 675–684, 2018. doi: 10.1016/j.procs.2018.08.001.
- Matthias Glawe, Christopher Tebbe, Alexander Fay, and Karl-Heinz Niemann. Knowledge-based Engineering of Automation Systems using Ontologies and

- Engineering Data. In *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2015)*, pages 291–300, Lisbon, Portugal, 2015. SCITEPRESS – Science and Technology Publications, Lda.
- Claudia V. Goldman, Michael Baltaxe, Debejyo Chakraborty, and Jorge Arinez. Explaining Learning Models in Manufacturing Processes. *Procedia Computer Science*, 180:259–268, 2021. doi: 10.1016/j.procs.2021.01.163.
- M. Goldstein and A. Dengel. Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm. In *Poster and Demo Track of the 35th German Conference on Artificial Intelligence (KI-2012)*, volume 9, pages 59–63, 2012.
- Germán González Rodríguez, Jose M. Gonzalez-Cava, and Juan Albino Méndez Pérez. An intelligent decision support system for production planning based on machine learning. *Journal of Intelligent Manufacturing*, 31(5):1257–1273, 2020. doi: 10.1007/s10845-019-01510-y.
- S. Gottschlich, C. Ramos, and D. Lyons. Assembly and task planning: a taxonomy. *IEEE Robotics & Automation Magazine*, 1(3):4–12, 1994. doi: 10.1109/100.326723.
- Michael Grieves and John Vickers. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In Franz-Josef Kahlen, Shannon Flumerfelt, and Anabela Alves, editors, *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, pages 85–113, Cham, 2017. Springer International Publishing. doi: 10.1007/978-3-319-38756-7\_4.
- Liang Guo, Fu Yan, Tian Li, Tao Yang, and Yuqian Lu. An automatic method for constructing machining process knowledge base from knowledge graph. *Robotics and Computer-Integrated Manufacturing*, 73:102222, 2022. doi: 10.1016/j.rcim.2021.102222.
- Erich Gutenberg. *Grundlagen der Betriebswirtschaftslehre, Bd. 1: Die Produktion*. Springer, Berlin, Heidelberg, New York, 24th edition, 1984.

- Simon Hagemann. *Algorithmische Konzeption von hochautomatisierten Fließmontagesystemen am Beispiel des automobilen Karosseriebaus*. PhD thesis, Technische Universität Berlin, 2022. doi: 10.14279/depositonce-16131.
- Simon Hagemann and Rainer Stark. Automated Body-in-White Production System Design: Data-Based Generation of Production System Configurations. In *Proceedings of the 4th International Conference on Frontiers of Educational Technologies*, ICFET '18, pages 192–196, 2018. doi: 10.1145/3233347.3233373.
- Simon Hagemann and Rainer Stark. An optimal algorithm for the robotic assembly system design problem: An industrial case study. *CIRP Journal of Manufacturing Science and Technology*, 31:500–513, 2020. doi: 10.1016/j.cirpj.2020.08.002.
- Simon Hagemann, Atakan Sünnetcioglu, and Rainer Stark. Hybrid Artificial Intelligence System for the Design of Highly-Automated Production Systems. *Procedia Manufacturing*, 28:160–166, 2019. doi: 10.1016/j.promfg.2018.12.026.
- T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, New York, 2009. doi: 10.1007/978-0-387-84858-7.
- Constantin Hildebrandt, André Scholz, Alexander Fay, Tizian Schröder, Thomas Hadlich, Christian Diedrich, Martin Dubovy, Christian Eck, and Ralf Wiegand. Semantic modeling for collaboration and cooperation of systems in the production domain. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, 2017. doi: 10.1109/ETFA.2017.8247585.
- John J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10):3088–3092, 1984. doi: 10.1073/pnas.81.10.3088.
- Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 1985. doi: 10.1017/CBO9780511810817.

- Vladimir Hubka and W. Ernst Eder. *Theory of Technical Systems*. Springer-Verlag Berlin, Heidelberg, 1988. doi: 10.1007/978-3-642-52121-8.
- IEEE. IEEE Taxonomy. Version 1.3, The Institute of Electrical and Electronics Engineers (IEEE), 2023.
- Paul Jaccard. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. 1901. doi: 10.5169/seals-266440.
- Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, New York, Heidelberg, Dordrecht, London, 2013.
- Matthias Jarke and Yanis Vassiliou. Data warehouse quality – a review of the DWQ project. In *Proceedings of the Conference on Information Quality*, pages 299–313, 1997.
- Nasr Kasrin, Maliha Qureshi, Simon Steuer, and Daniela Nicklas. Semantic Data Management for Experimental Manufacturing Technologies. *Datenbank-Spektrum*, 18(1):27–37, 2018. doi: 10.1007/s13222-018-0274-0.
- Siddhartha Kumar Khaitan and James D. McCalley. Design Techniques and Applications of Cyberphysical Systems: A Survey. *IEEE Systems Journal*, 9(2):350–365, 2015. doi: 10.1109/JSYST.2014.2322503.
- Jens Kiefer, Sebastian Allegretti, and Theresa Breckle. Quality- and Lifecycle-oriented Production Engineering in Automotive Industry. *Procedia CIRP*, 62: 446–451, 2017. doi: 10.1016/j.procir.2016.06.086.
- Jens Kiefer, Theresa Breckle, Ralf Stetter, and Martin Manns. Digital assembly planning using graph-based design languages. *Procedia CIRP*, 72:802–807, 2018. doi: 10.1016/j.procir.2018.03.063.
- Kyoung-Yun Kim, David G. Manley, and Hyungjeong Yang. Ontology-based assembly design and information sharing for collaborative product development. *Computer-Aided Design*, 38(12):1233–1250, 2006. doi: 10.1016/j.cad.2006.08.004.

- Konstantin Klein, Marco Franke, Karl Hribernik, Eva Coscia, Silke Balzert, Jan Sutter, and Klaus-Dieter Thoben. Potentials of Future Internet technologies for Digital Factories. In *2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA)*, pages 734–741, 2014. doi: 10.1109/AICCSA.2014.7073273.
- C. Kober, V. Adomat, M. Ahanpanjeh, M. Fette, and J.P. Wulfsberg. Digital Twin Fidelity Requirements Model for Manufacturing. In D. Herberger and M. Hübner, editors, *Proceedings of the Conference on Production Systems and Logistics: CPSL 2022. Hannover*, pages 595–611, 2022. doi: 10.15488/12145.
- Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982. doi: 10.1007/BF00337288.
- Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, Cambridge, Massachusetts, 2009.
- Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018. doi: 10.1016/j.ifacol.2018.08.474.
- Martin Krockert, Marvin Matthes, Torsten Munkelt, and Sven Völker. Generierung realitätsnaher Testdaten für die Simulation von Produktionen. In Jörg Franke and Peter Schuderer, editors, *19. ASIM Fachtagung Simulation in Produktion und Logistik 2021*, pages 565–574, Göttingen, 2021. Cuvillier Verlag.
- Andreas Kuhnle, Nicole Röhrig, and Gisela Lanza. Autonomous order dispatching in the semiconductor industry using reinforcement learning. *Procedia CIRP*, 79:391–396, 2019. doi: 10.1016/j.procir.2019.02.101.
- S.P. Leo Kumar. Knowledge-based expert system in manufacturing planning: state-of-the-art review. *International Journal of Production Research*, 57(15-16):4766–4790, 2019. doi: 10.1080/00207543.2018.1424372.
- Andrew Kusiak. Smart manufacturing. *International Journal of Production Research*, 56(1-2):508–517, 2018. doi: 10.1080/00207543.2017.1351644.



- Andrew Kusiak. Smart Manufacturing. In Shimon Y. Nof, editor, *Springer Handbook of Automation*, pages 973–985, Cham, 2023. Springer International Publishing. doi: 10.1007/978-3-030-96729-1\_45.
- Izabela Kutschenreiter-Praszkiewicz. Application of artificial neural network for determination of standard time in machining. *Journal of Intelligent Manufacturing*, 19(2):233–240, 2008. doi: 10.1007/s10845-008-0076-6.
- Mirko Kück and Michael Freitag. Forecasting of customer demands for production planning by local k-nearest neighbor models. *International Journal of Production Economics*, 231:107837, 2021. doi: 10.1016/j.ijpe.2020.107837.
- Minna Lanz, Fernando Garcia, Timo Kallela, and Reijo Tuokko. Product-Process Ontology for Managing Assembly Specific Knowledge Between Product Design and Assembly System Simulation. In Svetan Ratchev and Sandra Koelemeijer, editors, *Micro-Assembly Technologies and Applications*, pages 99–108, Boston, MA, 2008. Springer US.
- Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539.
- Edward A. Lee. Cyber Physical Systems: Design Challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369, 2008. doi: 10.1109/ISORC.2008.25.
- Chunquan Li, Yaqiong Chen, and Yuling Shang. A review of industrial big data for decision making in intelligent manufacturing. *Engineering Science and Technology, an International Journal*, 29:101021, 2022. doi: 10.1016/j.jestech.2021.06.001.
- Huaxia Li, Minjie Zou, Georg Hogrefe, Daria Ryashentseva, Michael Sollfrank, Gennadiy Koltun, and Birgit Vogel-Heuser. Application of a multi-disciplinary design approach in a mechatronic engineering toolchain. *at – Automatisierungstechnik*, 67(3):246–269, 2019. doi: 10.1515/auto-2018-0097.

- Jingshan Li and Semyon M. Meerkov. *Production Systems Engineering*. Springer Science+Business, New York, 2009. doi: 10.1007/978-0-387-75579-3.
- Don Libes, David Lechevalier, and Sanjay Jain. Issues in synthetic data generation for advanced manufacturing. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1746–1754, 2017. doi: 10.1109/BigData.2017.8258117.
- Chun-Cheng Lin, Der-Jiunn Deng, Yen-Ling Chih, and Hsin-Ting Chiu. Smart Manufacturing Scheduling With Edge Computing Using Multiclass Deep Q Network. *IEEE Transactions on Industrial Informatics*, 15(7):4276–4284, 2019. doi: 10.1109/TII.2019.2908210.
- Benjamin Lindemann, Benjamin Maschler, Nada Sahlab, and Michael Weyrich. A survey on anomaly detection for technical systems using LSTM networks. *Computers in Industry*, 131:103498, 2021. doi: 10.1016/j.compind.2021.103498.
- Lukas Lingitz, Viola Gallina, Fazel Ansari, Dávid Gyulai, András Pfeiffer, Wilfried Sihm, and László Monostori. Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer. *Procedia CIRP*, 72:1051–1056, 2018. doi: 10.1016/j.procir.2018.03.148.
- Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining, ICDM’08*, pages 413–422, Pisa, Italy, 2008. doi: 10.1109/ICDM.2008.17.
- Qihao Liu, Xinyu Li, and Liang Gao. A Novel MILP Model Based on the Topology of a Network Graph for Process Planning in an Intelligent Manufacturing System. *Engineering*, 7(6):807–817, 2021. doi: doi.org/10.1016/j.eng.2021.04.011.
- Yi-Hung Liu, Han-Pang Huang, and Yu-Sheng Lin. Dynamic scheduling of flexible manufacturing system using support vector machines. In *IEEE International Conference on Automation Science and Engineering, 2005.*, pages 387–392, 2005. doi: 10.1109/COASE.2005.1506800.

- Thiago Cantos Lopes, C.G.S. Sikora, Rafael Gobbi Molina, Daniel Schibelbain, L.C.A. Rodrigues, and Leandro Magatão. Balancing a robotic spot welding manufacturing line: An industrial case study. *European Journal of Operational Research*, 263(3):1033–1048, 2017. doi: 10.1016/j.ejor.2017.06.001.
- Hehe Ma, Yi Hu, and Hongbo Shi. Fault detection and identification based on the neighborhood standardized local outlier factor method. *Industrial & Engineering Chemistry Research*, 52(6):2389–2402, 2013. doi: 10.1021/ie302042c.
- Shuaiyin Ma, Yingfeng Zhang, Jingxiang Lv, Yuntian Ge, Haidong Yang, and Lin Li. Big data driven predictive production planning for energy-intensive manufacturing industries. *Energy*, 211:118320, 2020. doi: 10.1016/j.energy.2020.118320.
- Xiaoxiao Ma, Jia Wu, Shan Xue, Jian Yang, Chuan Zhou, Quan Z. Sheng, Hui Xiong, and Leman Akoglu. A Comprehensive Survey on Graph Anomaly Detection with Deep Learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021. doi: 10.1109/TKDE.2021.3118815.
- I. Mahdavi, A. H. Fekri Moghaddam Azar, and M. Bagherpour. Applying fuzzy rule based to flexible routing problem in a flexible manufacturing system. In *2009 IEEE International Conference on Industrial Engineering and Engineering Management*, pages 2358–2364, 2009. doi: 10.1109/IEEM.2009.5373001.
- Miro Mannino and Azza Abouzied. Is this real? generating synthetic data that looks real. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, page 549–561, 2019. doi: 10.1145/3332165.3347866.
- Luis Martí, Nayat Sanchez-Pi, José Manuel Molina, and Ana Cristina Bicharra Garcia. Anomaly detection based on sensor data in petroleum industry applications. *Sensors*, 15(2):2774–2797, 2015. doi: 10.3390/s150202774.
- Fernando Mas, José Ríos, Alejandro Gómez, and Juan Carlos Hernández. Knowledge-based application to define aircraft final assembly lines at the industrialisation conceptual design phase. *International Journal of Computer*

- Integrated Manufacturing*, 29(6):677–691, 2016. doi: 10.1080/0951192X.2015.1068453.
- Marvin Matthes, Oliver Guhr, Torsten Munkelt, Martin Krockert, and Sven Völker. Maschinelles Lernen von Maschinenfolgen für den simulationsbasierten Test von Verfahren der Produktionsplanung und -steuerung. In Sören Bergmann, Niclas Feldkamp, Rainer Souren, and Steffen Straßburger, editors, *20. ASIM Fachtagung Simulation in Produktion und Logistik 2023*, pages 167–176, Ilmenau, 2023. Universitätsverlag Ilmenau. doi: 10.22032/dbt.57476.
- Anjela Mayer, Jean-Rémy Chardonnet, Polina Häfner, and Jivka Ovtcharova. Collaborative Work Enabled by Immersive Environments. In Alexandra Shajek and Ernst Andreas Hartmann, editors, *New Digital Work: Digital Sovereignty at the Workplace*, pages 87–117, Cham, 2023. Springer International Publishing. doi: 10.1007/978-3-031-26490-0\_6.
- Sebastian Mayer, Tobias Classen, and Christian Endisch. Modular production control using deep reinforcement learning: proximal policy optimization. *Journal of Intelligent Manufacturing*, 32(8):2335–2351, 2021. doi: 10.1007/s10845-021-01778-z.
- Kenneth N. McKay, Frank R. Safayeni, and John A. Buzacott. A review of hierarchical production planning and its applicability for modern manufacturing. *Production Planning & Control*, 6(5):384–394, 1995. doi: 10.1080/09537289508930295.
- Kishan G. Mehrotra, Chilukuri K. Mohan, and HuaMing Huang. *Anomaly Detection Principles and Algorithms*. Springer, Cham, 2019.
- G. Michalos, A. Fysikopoulos, S. Makris, D. Mourtzis, and G. Chryssolouris. Multi criteria assembly line design and configuration – an automotive case study. *CIRP Journal of Manufacturing Science and Technology*, 9:69–87, 2015. doi: 10.1016/j.cirpj.2015.01.002.
- Adalberto Sato Michels, Thiago Cantos Lopes, Celso Gustavo Stall Sikora, and Leandro Magatão. The Robotic Assembly Line Design (RALD) problem:

- Model and case studies with practical extensions. *Computers & Industrial Engineering*, 120:320–333, 2018. doi: 10.1016/j.cie.2018.04.010.
- X. G. Ming and K. L. Mak. Intelligent setup planning in manufacturing by neural networks based approach. *Journal of Intelligent Manufacturing*, 11(3): 311–333, 2000. doi: 10.1023/A:1008975426914.
- L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda. Cyber-physical systems in manufacturing. *CIRP Annals*, 65(2):621–641, 2016. doi: 10.1016/j.cirp.2016.06.005.
- Junhyung Moon, Gyuyoung Park, and Jongpil Jeong. POP-ON: Prediction of Process Using One-Way Language Model Based on NLP Approach. *Applied Sciences*, 11(2):864, 2021. doi: 10.3390/app11020864.
- Dimitris Mourtzis. Simulation in the design and operation of manufacturing systems: state of the art and new trends. *International Journal of Production Research*, 58(7):1927–1949, 2020. doi: 10.1080/00207543.2019.1636321.
- Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, Cambridge, Massachusetts, 2012.
- Manuel Müller, Timo Müller, Behrang Ashtari Talkhestani, Philipp Marks, Nasser Jazdi, and Michael Weyrich. Industrial autonomous systems: a survey on definitions, characteristics and abilities. *at – Automatisierungstechnik*, 69(1): 3–13, 2021. doi: 10.1515/auto-2020-0131.
- Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-Scale Knowledge Graphs: Lessons and Challenges: Five Diverse Technology Companies Show How It’s Done. *Queue*, 17(2): 48–75, 2019. doi: 10.1145/3329781.3332266.
- Yui Okubo and Taiga Mitsuyuki. Ship Production Planning Using Shipbuilding System Modeling and Discrete Time Process Simulation. *Journal of Marine Science and Engineering*, 10(2):176, 2022. doi: 10.3390/jmse10020176.

- Gözde Öngelen and Tülin İnkaya. LOF weighted KNN regression ensemble and its application to a die manufacturing company. *Sāadhanā*, 48(4):246, 2023. doi: 10.1007/s12046-023-02283-0.
- Jeff Z. Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, Russa Biswas, Gerard de Melo, Angela Bonifati, Edlira Vakaj, Mauro Dragoni, and Damien Graux. Large Language Models and Knowledge Graphs: Opportunities and Challenges. 2023. doi: 10.48550/arXiv.2308.06374.
- You-Jin Park, Shu-Kai S. Fan, and Chia-Yu Hsu. A Review on Fault Detection and Process Diagnostics in Industrial Processes. *Processes*, 8(9):1123, 2020. doi: 10.3390/pr8091123.
- Paul Parsons. Understanding Data Visualization Design Practice. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):665–675, 2022. doi: 10.1109/TVCG.2021.3114959.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.
- Bastian Pokorni, Jan Zwerina, and Moritz Hämmerle. Human-centered design approach for manufacturing assistance systems based on design sprints. *Procedia CIRP*, 91:312–318, 2020. doi: 10.1016/j.procir.2020.02.181.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- Theofanis P. Raptis, Andrea Passarella, and Marco Conti. Data Management in Industry 4.0: State of the Art and Open Challenges. *IEEE Access*, 7:97052–97093, 2019. doi: 10.1109/ACCESS.2019.2929296.

- Samuel Ching Xin Ren, Jun Kit Chaw, Yee Mei Lim, Wah Pheng Lee, Tin Tin Ting, and Cheng Weng Fong. Intelligent Manufacturing Planning System Using Dispatch Rules: A Case Study in Roofing Manufacturing Industry. *Applied Sciences*, 12(13):6499, 2022. doi: 10.3390/app12136499.
- Christina Reuter and Felix Brambring. Improving Data Consistency in Production Control. *Procedia CIRP*, 41:51–56, 2016. doi: 10.1016/j.procir.2015.12.116.
- Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
- Jože M. Rožanec, Blaž Kažič, Maja Škrjanc, Blaž Fortuna, and Dunja Mladenić. Automotive oem demand forecasting: A comparative study of forecasting algorithms and strategies. *Applied Sciences*, 11(15):6787, 2021. doi: 10.3390/app11156787.
- Jacob Rubinovitz, Joseph Bukchin, and Ehud Lenz. RALB – A Heuristic Algorithm for Design and Balancing of Robotic Assembly Lines. *CIRP Annals*, 42(1):497–500, 1993. doi: 10.1016/S0007-8506(07)62494-9.
- Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The Earth Mover’s Distance as a Metric for Image Retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000. doi: 10.1023/A:1026543900054.
- Omer Sagi and Lior Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. doi: 10.1002/widm.1249.
- M. Salehi, G. Wald, T. L. Schmitz, R. Haas, and J. Ovtcharova. Probabilistische Modellierung und Vorhersage der Standzeit und Zuverlässigkeit eines Fräswerkzeugs mittels der Bayesianischen Statistik. *Forschung im Ingenieurwesen*, 84(2):129–139, 2020. doi: 10.1007/s10010-019-00391-0.
- Giulio Salierno, Giacomo Cabri, and Letizia Leonardi. Different perspectives of a factory of the future: An overview. In Henderik A. Proper and Janis Stirna, editors, *Advanced Information Systems Engineering Workshops*, pages 107–119, Cham, 2019. Springer International Publishing.

- Omar Santander, Vidyashankar Kuppuraj, Christopher A. Harrison, and Michael Baldea. A bayesian approach to improving production planning. *Computers & Chemical Engineering*, 173:108226, 2023. doi: 10.1016/j.compchemeng.2023.108226.
- Michael Schenk and Siegfried Wirth. *Fabrikplanung und Fabrikbetrieb. Methoden für die wandlungsfähige und vernetzte Fabrik*. Springer, Berlin, Heidelberg, 2004.
- Nicole Schmidt, Arndt Lüder, Heinrich Steininger, and Stefan Biffl. Analyzing requirements on software tools according to the functional engineering phase in the technical systems engineering process. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation*, ETFA, pages 1–8, Barcelona, Spain, 2014. doi: 10.1109/ETFA.2014.7005144.
- Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471, 2001. doi: 10.1162/089976601750264965.
- E. Schubert, J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)*, 42(3):1–21, 2017. doi: 10.1145/3068335. Article 19.
- Guenther Schuh, Jan-Philipp Prote, Melanie Luckert, and Philipp Hünnekes. Knowledge Discovery Approach for Automated Process Planning. *Procedia CIRP*, 63:539–544, 2017. doi: 10.1016/j.procir.2017.03.092.
- Günther Schuh. *Produktionsplanung und -steuerung: Grundlagen, Gestaltung Und Konzepte*. VDI-Buch. Springer, Dordrecht, 2007.
- Simon F. Schäfer, Nils T. Gorke, Cihan Cevirgen, Yeong-Bae Park, and Peter Nyhuis. Elemente der „Fabrik der Zukunft“. Teil 1: Digitale Fabrik, Industrie 4.0 und BIM. *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, 117(1-2):20–24, 2022a. doi: doi:10.1515/zwf-2022-1002.



- Simon F. Schäfer, Nils T. Gorke, Cihan Cevirgen, Yeong-Bae Park, and Peter Nyhuis. Elemente der „Fabrik der Zukunft“. Teil 2: Smart Plant – der Digitale Zwilling des Fabrikgesamtsystems. *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, 117(3):151–156, 2022b. doi: doi:10.1515/zwf-2022-1029.
- Hanmin Sheng, Jian Xiao, Yuhua Cheng, Qiang Ni, and Song Wang. Short-term solar power forecasting based on weighted gaussian process regression. *IEEE Transactions on Industrial Electronics*, 65(1):300–308, 2018. doi: 10.1109/TIE.2017.2714127.
- Sylvain Sirois. Autoassociator networks: insights into infant cognition. *Developmental Science*, 7(2):133–140, 2004. doi: 10.1111/j.1467-7687.2004.00330.x.
- Laurène Surbier, Gülgün Alpan, and Eric Blanco. A comparative study on production ramp-up: state-of-the-art and new challenges. *Production Planning & Control*, 25(15):1264–1286, 2014. doi: 10.1080/09537287.2013.817624.
- R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Massachusetts, 2 edition, 2020.
- Daniel Tan, Manu Suvarna, Yee Shee Tan, Jie Li, and Xiaonan Wang. A three-step machine learning framework for energy profiling, activity state prediction and production estimation in smart process manufacturing. *Applied Energy*, 291: 116808, 2021. doi: 10.1016/j.apenergy.2021.116808.
- Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018. doi: 10.1016/j.jmsy.2018.01.006.
- The Cancer Genome Atlas Research Network. Genomic and Epigenomic Landscapes of Adult De Novo Acute Myeloid Leukemia. *New England Journal of Medicine*, 368(22):2059–2074, 2013. doi: 10.1056/NEJMoa1301689. PMID: 23634996.
- L. Joseph Thomas and John O. McClain. Chapter 7 – an overview of production planning. In *Logistics of Production and Inventory*, volume 4 of *Handbooks*

- in Operations Research and Management Science*, pages 333–370. Elsevier, 1993. doi: 10.1016/S0927-0507(05)80187-2.
- Michael C. Thrun, Elisabeth K. M. Mack, Andreas Neubauer, Torsten Haferlach, Miriam Frech, Alfred Ultsch, and Cornelia Brendel. A Bioinformatics View on Acute Myeloid Leukemia Surface Molecules by Combined Bayesian and ABC Analysis. *Bioengineering*, 9(11):642, 2022. doi: 10.3390/bioengineering9110642. PMID: 36354555.
- Alicia Troncoso Lora, José C. Riquelme, José Luís Martínez Ramos, Jesús M. Riquelme Santos, and Antonio Gómez Expósito. Influence of kNN-Based Load Forecasting Errors on Optimal Energy Production. In Fernando Moura Pires and Salvador Abreu, editors, *Progress in Artificial Intelligence*, pages 189–203, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- Emre Tuncel, Abe Zeid, and Sagar Kamarthi. Solving large scale disassembly line balancing problem with uncertainty using reinforcement learning. *Journal of Intelligent Manufacturing*, 25(4):647–659, 2014. doi: 10.1007/s10845-012-0711-0.
- Enn Tyugu. *Algorithms and architectures of artificial intelligence*, volume 159. IOS Press, Amsterdam, 2007.
- Juan Pablo Usuga Cadavid, Samir Lamouri, Bernard Grabot, Robert Pellerin, and Arnaud Fortin. Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0. *Journal of Intelligent Manufacturing*, 31(6):1531–1558, 2020. doi: 10.1007/s10845-019-01531-7.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- VDI 4499 - Part 1. Digital Factory – Fundamentals. Standard ICS 03.100.50, 35.240.50, Verein Deutscher Ingenieure e.V., Düsseldorf, Germany, 2008.

- VDI 4499 - Part 2. Digital Factory – Operations. Standard ICS 03.100.50, 35.240.50, Verein Deutscher Ingenieure e.V., Düsseldorf, Germany, 2011.
- S. Völker, T. Döring, and T. Munkelt. The generation of large test data for the empirical analysis of heuristic procedures for production planning and control. In Bernhard Fleischmann, Rainer Lasch, Ulrich Derigs, Wolfgang Domschke, and Ulrich Rieder, editors, *Operations Research Proceedings*, pages 266–270, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. doi: 10.1007/978-3-642-56656-1\_41.
- Waldemar Walla. *Standard- und Modulbasierte digitale Rohbauprozesskette – Frühzeitige Produktbeeinflussung bezüglich Produktionsanforderungen im Karosserierohbau der Automobilindustrie*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2015. doi: 10.5445/IR/1000061966.
- Baicun Wang, Fei Tao, Xudong Fang, Chao Liu, Yufei Liu, and Theodor Freiheit. Smart Manufacturing and Intelligent Manufacturing: A Comparative Review. *Engineering*, 7(6):738–757, 2021. doi: 10.1016/j.eng.2020.07.017.
- Hao-Xiang Wang and Hong-Sen Yan. An interoperable adaptive scheduling strategy for knowledgeable manufacturing based on SMGWQ-learning. *Journal of Intelligent Manufacturing*, 27(5):1085–1095, 2016. doi: 10.1007/s10845-014-0936-1.
- Hongwei Wang and Gongzhuang Peng. *Collaborative Knowledge Management Through Product Lifecycle: A Computational Perspective*. Springer Nature Singapore, Singapore, 2023. doi: 10.1007/978-981-19-9626-9.
- Junliang Wang, Jie Zhang, and Xiaoxi Wang. Bilateral LSTM: A Two-Dimensional Long Short-Term Memory Model With Multiply Memory Units for Short-Term Cycle Time Forecasting in Re-entrant Manufacturing Systems. *IEEE Transactions on Industrial Informatics*, 14(2):748–758, 2018. doi: 10.1109/TII.2017.2754641.
- Richard Y. Wang and Diane M. Strong. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, 12(4):5–33, 1996.

- Bernd Waschneck, André Reichstaller, Lenz Belzner, Thomas Altenmüller, Thomas Bauernhansl, Alexander Knapp, and Andreas Kyek. Optimization of global production scheduling with deep reinforcement learning. *Procedia CIRP*, 72:1264–1269, 2018. doi: 10.1016/j.procir.2018.03.212.
- Agus Wahyu Widodo, Gusti Eka Yulastuti, Agung Mustika Rizki, and Wayan Firdaus Mahmudy. An Efficient Adaptive Neuro Fuzzy Inference System for Product Demand Forecasting. In *Proceedings of the 5th International Conference on Sustainable Information Engineering and Technology*, pages 90–94, 2021. doi: 10.1145/3427423.3427443.
- Martin Emmerich Witte, Christian Diedrich, and Helmut Figalist. Model-based development in automation. *at – Automatisierungstechnik*, 66(5):360–371, 2018. doi: 10.1515/auto-2017-0125.
- Günther Wöhe, Ulrich Döring, and Gerrit Brösel. *Einführung in die Allgemeine Betriebswirtschaftslehre*. Vahlen, München, 28th edition, 2023.
- Max Wuennenberg, Konstantin Muehlbauer, Johannes Fottner, and Sebastian Meissner. Towards predictive analytics in internal logistics – an approach for the data-driven determination of key performance indicators. *CIRP Journal of Manufacturing Science and Technology*, 44:116–125, 2023. doi: 10.1016/j.cirpj.2023.05.005.
- Wei Xian, Kan Yu, Fengling Han, Le Fang, Dehua He, and Qing-Long Han. Advanced Manufacturing in Industry 5.0: A Survey of Key Enabling Technologies and Future Trends. *IEEE Transactions on Industrial Informatics*, pages 1–15, 2023. doi: 10.1109/TII.2023.3274224.
- Muhammad Yahya, John G. Breslin, and Muhammad Intizar Ali. Semantic Web and Knowledge Graphs for Industry 4.0. *Applied Sciences*, 11(11):5110, 2021. doi: 10.3390/app11115110.
- Xifan Yao, Nanfeng Ma, Jianming Zhang, Kesai Wang, Erfu Yang, and Maurizio Faccio. Enhancing wisdom manufacturing as industrial metaverse for industry and society 5.0. *Journal of Intelligent Manufacturing*, 2022. doi: 10.1007/s10845-022-02027-7.

- Xu Zhang, Zhixue Liao, Lichao Ma, and Jin Yao. Hierarchical multistrategy genetic algorithm for integrated process planning and scheduling. *Journal of Intelligent Manufacturing*, 33(1):223–246, 2022a. doi: 10.1007/s10845-020-01659-x.
- Yi Zhang, Haihua Zhu, Dunbing Tang, Tong Zhou, and Yong Gui. Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 78:102412, 2022b. doi: 10.1016/j.rcim.2022.102412.
- Yue Zhao, Zain Nasrullah, and Zheng Li. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research*, 20(96):1–7, 2019.
- Pai Zheng, Honghui wang, Zhiqian Sang, Ray Y. Zhong, Yongkui Liu, Chao Liu, Khamdi Mubarak, Shiqiang Yu, and Xun Xu. Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives. *Frontiers of Mechanical Engineering*, 13(2):137–150, 2018. doi: 10.1007/s11465-018-0499-5.
- Peng Zheng, Junliang Wang, Jie Zhang, Changqi Yang, and Yongqiao Jin. An adaptive CGAN/IRF-based rescheduling strategy for aircraft parts remanufacturing system under dynamic environment. *Robotics and Computer-Integrated Manufacturing*, 58:230–238, 2019. doi: 10.1016/j.rcim.2019.02.008.
- Bin Zhou, Jinsong Bao, Jie Li, Yuqian Lu, Tianyuan Liu, and Qiwan Zhang. A novel knowledge graph-based optimization approach for resource allocation in discrete manufacturing workshops. *Robotics and Computer-Integrated Manufacturing*, 71:102160, 2021. doi: 10.1016/j.rcim.2021.102160.
- Pengwei Zhou, Zuhua Xu, Xudong Peng, Jun Zhao, and Zhijiang Shao. Long-term prediction enhancement based on multi-output Gaussian process regression integrated with production plans for oxygen supply network. *Computers & Chemical Engineering*, 163:107844, 2022. doi: 10.1016/j.compchemeng.2022.107844.