Karlsruher Institut für Technologie

Addressing Low-Quality Domain Knowledge in Knowledge-Guided Machine Learning

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

Pawel Bielski

aus Gdansk (Polen)

Tag der mündlichen Prüfung:	11. Februar 2025
Referenten:	Prof. Dr. Veit Hagenmeyer
	Prof. Dr. Jochen Garcke
	apl. Prof. Dr.-Ing. Ralf Mikut
Betreuer:	Prof. Dr.-Ing. Klemens Böhm

Acknowledgments

Completing this PhD has been a challenging journey, and it would not have been possible without the support of other people.

First, I would like to thank my supervisor, Prof. Klemens Böhm, for providing me with scientific freedom and for securing the funding of my PhD. I would also like to thank Prof. Veit Hagenmeyer, Prof. Jochen Garcke, and Prof. Ralf Mikut for serving as the dissertation reviewers, with special thanks to Prof. Ralf Mikut for his invaluable guidance during the final stages of my PhD.

Next, I would like to thank my co-author and research partner, Jakob, for his strategic advice that has been crucial to the success of my PhD. I also want to acknowledge my student co-authors with whom I spent several months exploring early ideas that ultimately evolved into this thesis: Nico, Florian, Lena, Sönke, and Aleksandr.

Additional thanks go to Elaheh, Snigdha, and Angelika for our many conversations throughout the entire PhD journey. To Miki, for organizing the board game evenings. To Haoyu, for his help with the experimental data. To Mrs. Bohlinger, for her support during the final stages of my PhD. To Jan, for his indoor cycling classes accompanied by the upbeat music that helped me stay balanced throughout my PhD.

Finally, I want to thank my family and friends for their support and belief in me. The warmest thanks go to my dear Aleksandra, who has been by my side through it all. Thank you for your patience, understanding, and faith in me, and for all our small and big adventures that make my life so much more memorable.

Abstract

Many industries collect vast amounts of data, applying machine learning techniques to derive actionable insights and improve decision-making. However, deploying machine learning models in scientific and engineering contexts presents unique challenges. This is because the available data often may not sufficiently represent the true nature of the underlying phenomena. In consequence, models may not generalize well beyond the training data and are may be prone to learning spurious relationships. It is crucial to ensure patterns discovered from data are consistent with relevant domain expertise. *Knowledge-guided Machine Learning* (KGML, also known as Theory-Guided Data Science, Physics-Guided Machine Learning or Informed Machine Learning) is a subfield of Machine Learning that focuses on systematically integrating domain knowledge into machine learning. Recent studies have shown that even little information integrated in this way can substantially enhance the accuracy, consistency, and generalizability of machine learning models.

However, domain knowledge can sometimes degrade model performance, a phenomenon we refer to as low-quality domain knowledge. Low-quality domain knowledge can arise for several reasons: (1) Difficult and ineffective process of integrating domain knowledge, due to the interdisciplinary nature of the task and the need for extensive collaboration with domain experts. (2) Imperfections of domain knowledge, due to difficulties in its collection, definition and representation. (3) A mismatch between domain knowledge and task-specific requirements, where knowledge originally developed for a different purpose proves to be suboptimal.

This thesis investigates these sources of low-quality domain knowledge and proposes solutions to address them:

(1) To facilitate the process of integrating domain knowledge, we introduce novel evaluation criteria for KGML methods that are crucial for practical collaboration with domain experts but are often overlooked: the amount of domain knowledge required and intellectual effort necessary for implementation. We apply these criteria to compare methods for integrating domain knowledge into machine learning models, using time-series voltage prediction for lithium-ion batteries as a case study. Our findings reveal that while effective KGML methods typically require substantial domain knowledge, they do not always demand significant implementation effort.

(2) To handle imperfect domain knowledge, we demonstrate how to design and evaluate a new KGML approach that addresses two types of imperfections: approximate domain knowledge, when exact measurements are not possible, and incorrect domain knowledge,

arising from measurement errors. We formalize a novel form of domain knowledge, commonly encountered in temporal dynamics modeling for engineering systems, and incorporate it through a knowledge-guided loss function. Our results indicate that the proposed method reduces prediction error by up to 40%, while significantly reducing the data requirements for training, and being robust to an incorrect domain knowledge of certain degree and being able to leverage approximate domain knowledge.

(3) To address mismatches between domain knowledge and task-specific knowledge, we systematically study mismatches within Ontology-guided Machine Learning, a subfield of KGML that uses ontological knowledge. We investigate how these mismatches manifest and influence predictions, and propose a framework to quantify them. Applying this framework to image classification and patient health prediction, we find that mismatches can affect substantial portions of the knowledge base, leading to a significant decline in model performance.

These contributions provide a comprehensive framework for addressing common challenges stemming from the low-quality domain knowledge in KGML, enhancing its applicability of KGML in various scientific and engineering domains.

Zusammenfassung

Viele Industrien sammeln riesige Mengen an Daten und wenden maschinelle Lerntechniken an, um umsetzbare Erkenntnisse zu gewinnen und die Entscheidungsfindung zu verbessern. Die Implementierung von maschinellen Lernmodellen in wissenschaftlichen und ingenieurtechnischen Kontexten stellt jedoch einzigartige Herausforderungen dar. Dies liegt daran, dass die verfügbaren Daten oft nicht in ausreichendem Maße die wahre Natur der zugrunde liegenden Phänomene widerspiegeln. Infolgedessen können Modelle Schwierigkeiten haben, über die Trainingsdaten hinaus zu generalisieren und sind möglicherweise anfällig dafür, falsche Zusammenhänge zu lernen. Es ist entscheidend, sicherzustellen, dass die aus den Daten entdeckten Muster mit relevanter Expertise aus dem jeweiligen Fachgebiet übereinstimmen.

Knowledge-guided Machine Learning (KGML, auch bekannt als Theory-Guided Data Science, Physics-Guided Machine Learning oder Informed Machine Learning) ist ein Teilbereich des maschinellen Lernens, der sich darauf konzentriert, Fachwissen systematisch in maschinelle Lernverfahren zu integrieren. Neueste Studien haben gezeigt, dass bereits kleine Mengen an integriertem Fachwissen die Genauigkeit, Konsistenz und Generalisierbarkeit von maschinellen Lernmodellen erheblich verbessern können.

Allerdings kann das Einbringen von Fachwissen manchmal auch die Modellleistung verschlechtern, ein Phänomen, das wir als „Low-Quality Domain Knowledge“ (niedrigwertiges Fachwissen) bezeichnen.

Niedrigwertiges Fachwissen kann aus verschiedenen Gründen entstehen: (1) Der schwierige und ineffektive Integrationsprozess des Fachwissens, bedingt durch die interdisziplinäre Natur der Aufgabe und den Bedarf an umfangreicher Zusammenarbeit mit Fachexperten. (2) Unvollkommenheiten des Fachwissens, bedingt durch Schwierigkeiten bei der Sammlung, Definition und Darstellung. (3) Ein Missverhältnis zwischen dem Fachwissen und den spezifischen Anforderungen der Aufgabe, bei dem Wissen, das ursprünglich für einen anderen Zweck entwickelt wurde, sich als suboptimal herausstellt.

Diese Dissertation untersucht diese Quellen von niedrigwertigem Fachwissen im Kontext des KGML und schlägt Lösungen vor, um damit umzugehen:

(1) Um den Integrationsprozess von Fachwissen zu erleichtern, führen wir neuartige Bewertungskriterien für KGML-Methoden ein, die für die praktische Zusammenarbeit mit Fachexperten entscheidend sind, jedoch oft übersehen werden: die Menge des benötigten Fachwissens und der intellektuelle Aufwand, der für die Umsetzung erforderlich ist. Wir wenden diese Kriterien an, um verschiedene Methoden zur Integration von Fachwissen in

maschinelle Lernmodelle zu vergleichen, wobei wir die Vorhersage von Spannungszeitreihen für Lithium-Ionen-Batterien als Fallbeispiel verwenden. Unsere Ergebnisse zeigen, dass effektive KGML-Methoden zwar typischerweise viel Fachwissen erfordern, jedoch nicht immer mit einem erheblichen Implementierungsaufwand verbunden sind.

(2) Um mit unvollkommenem Fachwissen umzugehen, zeigen wir, wie man eine neue KGML-Methode entwirft und bewertet, die zwei Arten von Unvollkommenheiten adressiert: ungefähres Fachwissen, wenn genaue Messungen nicht möglich sind, und falsches Fachwissen, das durch Messfehler entsteht. Wir formalisieren eine neuartige Form von Fachwissen, die häufig in der Modellierung dynamischer Systeme im Ingenieurwesen vorkommt, und integrieren es durch eine wissensgesteuerte Verlustfunktion. Unsere Ergebnisse zeigen, dass die vorgeschlagene Methode den Vorhersagefehler um bis zu 40% reduziert, die Datenanforderungen für das Training signifikant senkt und robust gegenüber falschem Fachwissen in gewissem Maße ist, während sie gleichzeitig auch ungefähres Fachwissen nutzen kann.

(3) Um Missverhältnisse zwischen Fachwissen und aufgabenspezifischem Wissen zu adressieren, untersuchen wir systematisch solche Missverhältnisse innerhalb des Ontology-guided Machine Learning, einem Teilbereich des KGML, der ontologisches Wissen verwendet. Wir untersuchen, wie sich diese Missverhältnisse manifestieren und die Vorhersagen beeinflussen, und empfehlen eine Vorgehensweise, um sie zu quantifizieren. Die Anwendung dieser Vorgehensweise auf Bildklassifikationen und Vorhersagen zur Gesundheit von Patienten zeigt, dass Missverhältnisse große Teile der Wissensbasis beeinträchtigen und die Modellleistung erheblich verschlechtern können.

Diese Beiträge bieten einen umfassenden Rahmen, um gängige Herausforderungen im Zusammenhang mit niedrigwertigem Fachwissen im KGML zu adressieren, und erhöhen die Anwendbarkeit von KGML in verschiedenen wissenschaftlichen und ingenieurtechnischen Bereichen.

Contents

List of Figures

List of Tables

Part I.

Introduction

1. Motivation

1.1. Background

This section provides an overview of the foundational concepts of Knowledge-Guided Machine Learning (KGML). We begin by discussing the motivation for incorporating domain knowledge into machine learning, emphasizing its role in improving accuracy, consistency, and generalizability while also reducing data requirements. We then discuss the different forms of domain knowledge used in KGML, the strategies used to integrate this knowledge into machine learning models, and the diverse objectives that drive its application. Finally, we introduce the concept of low-quality domain knowledge, a key focus of this dissertation. Understanding this concept is crucial for tackling the challenges addressed in the subsequent chapters. Chapters 1 and 2 offer a high-level overview of these challenges, while Chapters 3 to 5 provide details on how to address them.

1.1.1. Knowledge-guided Machine Learning (KGML)

Purely Data-driven Machine Learning Industry collects huge volumes of data and uses machine learning methods to generate business value. However, using machine learning models in science and engineering comes with specific challenges [Kar+17]. A key problem is that in these fields, the available data is often limited, and the phenomena being modeled are complex. As a result, the data may not fully capture the underlying processes. For example, while routine system behaviors are well-represented, critical or rare events often lack sufficient data coverage. As a result, machine learning models that rely only on data may struggle to generalize beyond the training data. They are also prone to learning spurious patterns, particularly when the data is scarce [Beu+21; Dje+22; Kar+17; Rob+22]. While these models may perform well on the available test data, they may easily fail in real-world scenarios where conditions differ. For example, they might fail to extrapolate beyond the value ranges seen in the training data due to not capturing all the relevant factors affecting the phenomenon, or because they fail to identify the correct relationships between multiple, often correlated, features. To address these issues and improve the reliability of machine learning models, it is essential to ensure that the patterns they learn align with existing domain knowledge [Kar+17].

Domain Knowledge in Machine Learning In addition to data, there is often additional information about the machine learning task available that can help overcome the limitations of purely data-driven machine learning [Sim+91]. This additional information is commonly referred to in the literature as *domain knowledge* [Kar+17; Wil+23], *prior knowledge* [Sim+91; Rue+23], or *side information* [Abu92]. It can be obtained from other data discovery processes or directly provided by domain experts [ABH95]. In this dissertation, we occasionally refer to *domain knowledge* as simply *knowledge*.

Knowledge-guided Machine Learning Knowledge-guided Machine Learning (KGML) [KKK22] is a subfield of Machine Learning that focuses on systematically integrating domain knowledge into machine learning. It is referred in the literature by many names including: *Physics-guided Deep Learning* [WY22], *Theory-guided Data Science* [Kar+17], *Physics-informed Machine Learning* [Wil+23], or *Informed Machine Learning* [Rue+23]. Recent studies have shown that even little information integrated in this way can substantially enhance the accuracy, consistency, and generalizability of machine learning models. KGML has proven its worth in many fields, including materials research [Mur+20], energy research [Li+21], climate science [Jia+19], fluid mechanics [Cai+21], as well as in medical data processing [Cho+17] and computer vision [Che+18]. As methods and techniques for applying KGML to specific use cases continue to grow, researchers have categorized them into taxonomies in literature surveys such as [Kar+17] [Wil+23] or [Rue+23]. Based on these taxonomies, KGML methods can be categorized based on the forms of domain knowledge they use, different strategies to integrate domain knowledge, and different goals of integrating domain knowledge.

1.1.2. Forms of Knowledge in KGML

According to the survey by [Rue+23], in the context of KGML, domain knowledge from various sources can be represented in different ways.

Scientific Knowledge is a type of domain knowledge commonly found in scientific and engineering fields. It is typically represented by simplified mathematical models, based on *algebraic* or *differential equations*, or by *simulation results* derived from these models. These equation-based models describe relationships between variables using parameterizable functions, which are validated through experiments, or by constraints on the variables. The parameters of these models are empirically determined for specific applications. Scientific knowledge is the most widely used type of domain knowledge in KGML [Rue+23].

World Knowledge is a type of general knowledge that refers to general facts and information about the world that are widely known and understood. It includes intuitive understanding, and basic reasoning about the objects, concepts, and relationships between them. It is often represented symbolically, through (crisp or fuzzy) *logic rules*, *knowledge graphs*, *ontologies* or *spatial invariances*. Being qualitative in nature, world knowledge

is often more challenging to integrate into quantitative machine learning models than scientific knowledge.

Expert Knowledge is a type of less formal and intuitive knowledge known to a group of domain experts. It is typically acquired over years of experience working in a specific field. It is often represented by the direct *human feedback* using human-machine interfaces, assumption-based *probabilistic relations* as well as simple *algebraic equations*.

1.1.3. Strategies of Integrating Knowledge in KGML

According to [Kar+17] and [Wil+23] the there are five main KGML paradigms of integrating domain knowledge into machine learning. A KGML paradigm could be seen as a design pattern of integrating domain knowledge, however it is often a loose guideline rather than a precise recipe. A KGML method is then an instantiation of a KGML paradigm on a concrete use case. Both surveys ([Kar+17] and [Wil+23]) abound with concrete, useful examples for each paradigm. Conversely, this variety implies that not all KGML methods are suitable for every situation.

Knowledge-guided Loss Function This paradigm introduces a regularization term $\text{Loss}_{Physical}$ representing the conformity with the knowledge in the loss function, in addition to the training loss term Loss_{Train} (see Equation 1.1). The final loss function optimizes multiple objectives and can be seen as multi-task learning [Wil+20]. Models trained in this way tend to generate physically consistent results, and perform better, for small training data in particular [Jia+19; Daw+20].

$$\text{Loss}(\hat{y},\, y) = \text{Loss}_{Train}(\hat{y},\, y) + \lambda\, \text{Loss}_{Physical}(\hat{y},\, y). \tag{1.1}$$

Knowledge-Guided Initialization This approach involves pretraining the model on a related task or dataset, followed by fine-tuning on the target task using the target dataset. For example, in [Jia+19], the authors pretrained a model using data generated by a theoretical model and then fine-tuned it with a smaller set of real data to improve the accuracy of lake temperature dynamics modeling. The key idea is that initializing the model with physically consistent parameters, rather than random ones, helps guide the optimization process. This increases the likelihood of finding a better local optimum in a multi-modal optimization landscape, ultimately improving performance and reducing training time [Jia+21].

Knowledge-guided Architecture The idea is design or adjust a machine learning a model architecture to match the structure of the problem according to domain knowledge [WY22]. For instance, in [Daw+20], researchers adapted the recurrent neural network with physics-informed connections to encode the established monotonic relationships between density

and depth when predicting lake temperatures. Similarly, in [Wan+20], authors proposed a custom variant of a neural network with three identical encoders to model turbulent flow on different scales. In [Cho+17] the authors proposed a custom attention-based embedding layer to model the relationships between medical concepts for the patient health prediction task.

Knowledge-guided Hybrid Model This approach combines a theoretical equation-based model and a machine learning model to operate simultaneously [Wil+20]. One common method of integrating these models is to use the output of the theory-based model as an input to the machine learning model, alongside other features. This approach has been successfully applied in studies such as [Daw+17] and [Wil+98] to provide the machine learning model with missing physical terms, enabling it to achieve significantly higher predictive accuracy. Additionally, parallel models can be employed, potentially utilizing an ensemble model technique to enable gradual switching between different domain-knowledge-based and data-driven models [Mei+23; ZW19].

Knowledge-guided Residual Learning This paradigm refines the predictions of a theory-based model by training a machine learning model to predict the error (known as residual) of the theory-based model. Final predictions are obtained by summing the output of the theory-based with the residuals predicted by the machine learning model. This approach has proven effective in fields such as chemical modeling [TK94] and mechanical process modeling [Zhe+21].

1.1.4. Objectives of Integrating Knowledge in KGML

The survey by [Wil+23] introduces the concept of *application-centric objectives*, which can be viewed as the goals for integrating domain knowledge. In this thesis, we focus on the objective of improving the machine learning models.

Improving the machine learning model According to [Rue+23], the most common goal of integrating domain knowledge in KGML is improving the machine learning model. This improvement can be achieved in terms of the improved prediction accuracy, reduced data requirement, better conformity with the knowledge, or better interpretability compared to the knowledge-uninformed model. [Wil+23] emphasizes that the improvement in prediction accuracy should ideally be evident not only when compared to a knowledge-uninformed machine learning model, but also when compared to a theory-based model.

Other Integration Objectives In addition to improving the prediction model, [Wil+23] provides a taxonomy of other objectives, including discovering the governing equations, generating synthetic data, estimating process uncertainty, and approximating the existing

(simulation) model with either coarser, more computationally efficient models or finer, higher-resolution models.

1.1.5. Low-quality Domain Knowledge in KGML

While KGML methods generally outperform the knowledge-uninformed machine learning [Kar+17; Rue+23; Wil+23; WY22; Cho+17; Che+18], integrating domain knowledge sometimes can also negatively impact machine learning predictions. We refer to such problematic knowledge as *low-quality domain knowledge*.

The issue of low-quality domain knowledge in the context of KGML was first recognized by [Mit97] and [Yu07], who acknowledged that domain knowledge can be imperfect due to difficulties in its collection, definition, and representation. They stressed the importance of considering the negative impact of imperfect domain knowledge when applying KGML in practice. [Yu07] further highlighted that domain knowledge is often highly context-dependent, meaning its usefulness can vary across different tasks. [Mit97] argued, however, that even imperfect domain knowledge could still be useful if the machine learning algorithm can tolerate some level of error. Subsequent studies, such as those by [Cho+17] and [BD19c], confirmed that KGML models may, in some cases, underperform compared to knowledge-uninformed models, particularly when the domain knowledge used was either randomly distorted or created for a different purpose.

Despite these findings, apart from few recent studies that design KGML methods to work with low-quality domain knowledge (e.g., [BBD21] or [Den+14]), most KGML research does not explicitly acknowledge or address the issues stemming from low-quality domain when designing or evaluating KGML methods. One potential reason for this gap may be the use-case-specific nature of KGML studies, which makes it difficult to make generalizable observations about KGML, as noted by [RGB23]. The KGML studies typically focus on achieving prediction improvements rather than examining auxiliary issues such as enhancing the domain knowledge integration process, designing models to be robust against low-quality knowledge, or developing methods for detecting such low-quality knowledge. Given these gaps, understanding the factors that contribute to low-quality domain knowledge and identifying ways to mitigate its impact is essential for the broader adoption of KGML. This thesis investigates different sources of low-quality domain knowledge and proposes solutions to address them.

1.2. Challenges

Low-Quality Domain Knowledge In this thesis, we define low-quality domain knowledge in the context of KGML as knowledge that has a negative impact on machine learning predictions compared to knowledge-uninformed baseline models. There are several challenges that contribute to the phenomenon of low-quality domain knowledge, including issues in its integration, representation and structure:

1.2.1. Difficult Process of Integrating Knowledge

Firstly, low-quality domain knowledge can result from the ineffective and difficult process of integrating domain knowledge in practice. These difficulties typically arise from the interdisciplinary nature of the task, which often necessitates extensive collaboration with domain experts to identify domain knowledge. While research studies demonstrating how to integrate various types of domain knowledge and KGML surveys organizing the field into taxonomies and paradigms ([Kar+17; Wil+23; Rue+23]) make some aspects of integrating domain knowledge easier, two major challenges remain that need to be addressed to improve domain knowledge integration in KGML projects:

1.2.1.1. Difficult Identification of Domain Knowledge

Although domain knowledge relevant to a task is often available in addition to data [Sim+91], in practice, it is often unclear what specific domain knowledge to look for, and there are no guarantees that relevant and compatible domain knowledge exists for a specific machine learning task, and even if it does, whether it can be formalized. Identifying new forms of domain knowledge and developing KGML methods that integrate them into various use cases is a critical ongoing effort to facilitate adaptation of KGML in practice. However, many types of potentially valuable domain knowledge still remain unexplored in the literature [WY22].

1.2.1.2. Difficult Selection Process of KGML Methods

Despite the large range of KGML methods available, selecting an effective method suited to a specific task and compatible with the available domain knowledge is challenging. This difficulty arises because KGML research is highly application-driven, with many specialized approaches and limited guidance on selecting the best method for a new task [RGB23]. Comparative studies that directly evaluate KGML methods on real tasks are scarce, which further complicates method selection for specific domains. For instance, while [MAM22] compared various KGML methods, their study neither involved domain experts nor developed methods based on specific paradigms. Additionally, current evaluation metrics [YR21; RGB23] for KGML methods focus on performance and knowledge conformity but overlook practical aspects of collaborating with domain experts, which are often missing from use-case-specific studies.

1.2.2. Imperfect Representation of Knowledge

Secondly, as noted by [Mit97] and [Yu07], domain knowledge itself can be imperfect, due to difficulties in its collection, definition and representation. However, because KGML research is highly application-driven, with the focus on identifying new forms of domain knowledge and developing KGML methods for specific use cases, there has been limited

investigation into what such imperfect domain knowledge looks like, when it typically arises, and how to address it effectively. In this thesis, we identify two types of imperfect domain knowledge that are relevant in the KGML context, but have been mostly overlooked in the existing KGML literature. Both of these types of imperfect domain knowledge should ideally be accounted for during the development and evaluation of KGML methods:

1.2.2.1. Approximate Domain Knowledge

In practical situations like within engineering and natural science where measurements have uncertainties or when the problem being modeled is stochastic in nature, certain types of domain knowledge might not be possible to be measured exactly. In such situations such represented knowledge can still be useful for the task, but it requires such knowledge to be represented approximately and to be compatible with the respective KGML method. Currently, to the best of our knowledge, no studies have demonstrated effective approaches for representing and integrating such approximate domain knowledge.

1.2.2.2. Incorrect Domain Knowledge

Incorrect domain knowledge occurs when the information provided does not correctly represent the reality. As noted by [Mit97], incorrect knowledge can still be useful if the KGML method can tolerate some level of error. However, current KGML studies do not design methods explicitly to be robust against inaccuracies, nor do they evaluate these methods' resilience to incorrect domain knowledge.

1.2.3. Domain-application Knowledge Mismatch

Thirdly, while KGML methods generally outperform domain-uninformed machine learning on average [Dha+20; Kar+21; SF11], the structure of the underlying domain knowledge may not always align optimally with the specific machine learning task, which can negatively impact prediction performance, even when the knowledge itself is correct. This issue often arises when domain knowledge was originally created for different purpose than the specific KGML task. In this thesis, we refer to this phenomenon as a mismatch between domain knowledge and application knowledge. Addressing this mismatch is essential for successful KGML adaptation. However, the following two issues complicate its resolution:

1.2.3.1. Limited Awareness and Understanding of Domain-Application Mismatches

Due to limited research on the negative impacts of the domain knowledge in KGML, there is only limited awareness in the KGML research community that even structurally and semantically correct domain knowledge can sometimes reduce prediction accuracy. While the negative effects of incorrect (i.e. randomized) domain knowledge have been

observed for example in [Cho+17], there is far less recognition of the negative impact that structurally and semantically correct knowledge may have. For example, [BD19c] observed that semantically correct knowledge could, on average, negatively affect the predictions if it was originally generated for a different purpose. Similarly, [MSG17] observed that although their KGML approach outperformed the baseline on average, some subsets of data actually performed worse than the baseline. Despite these observations, the issue of domain-application mismatch remains largely unexplored. There is a lack of systematic research to understand how these mismatches manifest generally in KGML and how they impact predictions.

1.2.3.2. Lack of Methods to Quantify and Address Mismatches

Currently, there are no established methods to identify and quantify domain-application knowledge mismatches, which limits the ability to study and address them effectively. Developing a method to quantify such mismatches is an essential first step, as it would enable systematic identification, understanding, and mitigation of mismatches within the KGML context.

1.3. Contributions

To investigate and address these challenges, we conducted several studies focused on integrating domain knowledge to enhance machine learning prediction models (see Section 1.1.4). These studies explore various forms of domain knowledge and different KGML paradigms for knowledge integration.

1.3.1. Addressing the Difficult Process of Integrating Knowledge

To address difficult selection process of KGML methods, we introduced novel evaluation criteria relevant for practical collaboration with domain experts, which are often overlooked in application-specific KGML publications. These criteria include the amount of domain knowledge required, and intellectual effort necessary for implementation. To demonstrate the usefulness of these criteria we conducted a comparative study of KGML methods specifically using these criteria for the example task of time-series voltage prediction in the context of lithium-ion batteries. In that context, we identified and formalized various types of **Scientific Knowledge** and proposed concrete implementations of KGML methods based on the five established knowledge-guided machine learning paradigms (see Section 1.1.3) from the existing literature to address difficult identification of domain knowledge. We categorized each KGML paradigm based on these criteria in addition to the performance on the machine learning prediction task. Our study revealed that while effective KGML methods often require substantial domain knowledge, their implementation does not necessarily entail significant effort which is a novel observation in KGML.

1.3.2. Addressing the Imperfect Representation of Knowledge

To address the challenge of working with imperfect representations of domain knowledge, we demonstrate how to design a KGML method that accommodates approximate domain knowledge, and how to evaluate its robustness against incorrect domain knowledge. In the context of deep learning-based temporal dynamics modeling, we introduced and formalized a novel type of **Expert Knowledge** that is often available for dynamical systems: permissible system states (thus addressing difficult identification of domain knowledge). This domain knowledge describes dynamic states that the system is allowed to take during its operation. To encode this novel domain knowledge, we propose a knowledge-guided multi-state constraint enforced leveraging the knowledge-guided loss function paradigm. This constraint is applicable to deep learning models for temporal dynamics modeling in response to the input control signal. Validating our method's effectiveness in modeling the temporal behavior of a micro gas turbine, we observed a reduction in prediction error by up to 40%, a decreased dependency on extensive training data and reduced training randomness, compared to a knowledge-uninformed deep learning baseline. At the same time our method demonstrated tolerance for approximations within 25% of the permissible system states' values and maintained robustness against incorrect domain knowledge values that were up to 16 times larger or 8 times smaller.

1.3.3. Addressing the Domain-Application Knowledge Mismatch

To increase the awareness and understanding of domain-application mismatches, we conducted the study focused on ontology-guided machine learning (OGML), a subfield of KGML that uses ontological **World Knowledge**. We introduced a framework to detect these mismatches by comparing model performance with and without domain knowledge, identifying specific parts of the ontology that negatively impact predictions. To demonstrate the framework's effectiveness, we apply it to two common OGML application areas: image classification and patient health prediction. Our findings indicate that domain-application mismatches are widespread across various OGML approaches, machine-learning model architectures, datasets, and prediction tasks, and can impact up to 40% of unique domain concepts in the datasets. We also explore the potential root causes of these mismatches and discuss strategies for addressing them. This approach shows promise as a generalizable methodology for assessing ontology quality.

1.4. List of Publications

This dissertation is based on the following publications:

1. Pawel Bielski, Nico Denner, Simon Bischof, Benedikt Rzepka, and Klemens Böhm. "Theory-Guided Data-Science for Battery Modeling: Insights from a Comparative Study". In: *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*. 2024, pp. 1–10
2. Pawel Bielski, Aleksandr Eismont, Jakob Bach, Florian Leiser, Dustin Kottonau, and Klemens Böhm. "Knowledge-Guided Learning of Temporal Dynamics and its Application to Gas Turbines". In: *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*. e-Energy '24. Singapore, Singapore, 2024, pp. 279–290
3. Pawel Bielski, Lena Witterauf, Sönke Jendral, Ralf Mikut, and Jakob Bach. "Quantifying Domain-Application Knowledge Mismatch in Ontology-Guided Machine Learning". In: *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. Ed.: D. Aveiro. Vol. 2*. 2024, p. 216

We have reused content from the publications mentioned above, restructuring and rephrasing it to create a coherent monograph that incorporates their major contributions.

1.5. Dissertation Outline

The remainder of this dissertations is structured as follows: Firstly, in Chapter 2 we discuss related work relevant for our contributions. Then, in the main part of this dissertation, in Chapters 3,4 and 5 we present our contributions. Chapter 3 addresses the difficult process of integrating domain knowledge. Chapter 4 addresses the imperfect representation of domain knowledge. Chapter 5 addresses the domain-application mismatch of domain knowledge. Finally in the last part of this dissertation we wrap up our work. Chapter 6 summarizes the conclusions and Chapter 7 discusses future work.

2. Related Work

2.1. Modeling Dynamical Systems with KGML

In this section, we review related work in two key areas. First, in Section 2.1.1, we explore the role of machine learning in modeling dynamical systems within Science and Engineering, which is relevant to Chapters 3 and 4. Next, in Section 2.1.2, we focus on how domain knowledge is integrated into loss functions, a topic relevant to Chapter 4.

2.1.1. Role of Machine Learning in Modeling Dynamical Systems in Science and Engineering

A dynamical system is a system whose state changes over time. Modeling dynamical systems is a fundamental task in science and engineering [Kar+17; Wil+23; Beu+21]. Conventionally, these systems are represented using application-specific theory-based models that describe their behavior through mathematical equations. However, when the exact equations describing the system's dynamics cannot be determined, or measuring the necessary system components needed for parameterizing these equations proves impractical or excessively expensive, employing theory-based models becomes infeasible [WY22; Yin+21].

Deep learning methods are a promising data-driven alternative to theory-based models by learning the dynamical system's behavior directly from data [WY22; KM19; Wil+23]. Their flexibility and ability to learn complex relationships make them highly suitable for modeling various real-world systems [Wil+23]. However, despite recent successes, these methods have shortcomings. They can produce physically inconsistent predictions, especially when data availability is limited [Beu+21; Dje+22], Additionally, they often struggle to generalize to scenarios not adequately represented in the training data [Kar+17; Rob+22].

2.1.2. Integrating Domain Knowledge in the Loss Function for Modeling Dynamical Systems

There is a rapid growth of papers on incorporating domain knowledge into deep learning for dynamical systems, through knowledge-guided constraints in the loss function and using them in various scenarios [Wil+23; Kar+17; Rue+23; WY22]. Early work in this area

focused on defining generic types of domain knowledge independently from the application domain. For instance, [Sim+91] introduced a knowledge-guided loss function to inform a neural network about the derivatives at specific points of a generic target function. Soon after, [Abu92] proposed constraints on the loss function to encode invariances, monotonic relationships, and permissible value ranges. These generic, domain-independent types of domain knowledge enabled modeling application-specific domain knowledge in various data-driven scenarios of modeling particular dynamical systems. For example, [Mur+18] employed a combination of monotonicity and approximation constraints for predicting oxygen solubility in water, while [Jia+19] applied a monotonicity constraint on the density-depth relationship when predicting temperature in lakes.

Much less research has been done on identifying and formalizing domain knowledge that is applicable across various dynamical systems. For example, [Daw+17] proposed a constraint to encode the energy conservation law. In general, proposing and evaluating new types of domain knowledge and formalizing them with the knowledge-guided constraints in the loss function remains an active field of research [WY22]. Such research is particularly impactful if the proposed domain knowledge is applicable to a broader class of problems, allowing for the reuse of the knowledge-guided methods. To the best of our knowledge, our work is the first to identify and propose the use of domain knowledge in the form of permissible system states. None of the previously proposed knowledge-guided constraints are suitable for representing this specific type of domain knowledge. While we evaluate our method with modeling a gas turbine, the concept of permissible systems states is applicable for data-driven temporal dynamics modeling of various dynamical systems.

There also is related work that focuses on integrating multiple sources of domain knowledge into a single neural network. For example, [Abu92] balanced multiple knowledge-guided constraints by alternating between them during training, while [Mur+18] combined multiple constraints within a single model using a weighted sum. Additionally, [Elh+22] introduced a method for optimizing weights among competing knowledge-guided constraints. However, these approaches do not consider incorporating information about alternative characteristics that a target function can possess, like the permissible system states featured in our approach presented in Chapter 4.

Finally, there is related work on integrating domain knowledge into optimization problems, particularly in the context of dynamic systems, as seen in [Grö15]. Although this work does not focus on machine learning, it highlights that the integration of domain knowledge through the loss function predates machine learning, tracing back to parameter search methods in system theory. These approaches can also be extended to machine learning models that are not based on neural networks, such as support vector machine regression, as demonstrated in studies like [Don+15].

2.2. Evaluating the Quality of Ontologies in KGML

In this section, we review related work regarding approaches for ontology quality evaluation in Section 2.2.1, OGML in general in Section 2.2.2, and the issue of low-quality domain knowledge in the context of OGML in Section 2.2.3. These three areas are relevant to Chapter 5.

2.2.1. Ontology Quality Evaluation

Creating and maintaining ontologies is a highly subjective, labor-intensive process. This process is prone to errors, as there is no standard method for creating ontologies [Cap99; Bre02; Duq+11]. Additionally, ontologies are only approximations of domain knowledge, and multiple valid ontologies can exist to represent the same knowledge [HS14; MS20]. Thus, evaluating the quality of ontologies is essential for the broader adoption of ontology-informed applications [MAD17]. This process ensures that developed ontologies are useful for specific tasks or domains and helps select the most suitable ontology for the given application [Duq+11]. Evaluation methods can significantly reduce the human effort needed to create and maintain ontologies. In particular, they can guide the construction process and enable the reuse of existing ontologies instead of building them from scratch [Cap99; Bey+11; MS20]. Despite many proposed approaches to ontology quality evaluation, no universal solution exists as they address different quality aspects [MS20].

Existing methods can be grouped into two broad categories: deductive (metrics-based) and inductive (empirical) [Bur+05; HS14].

Deductive evaluation methods to evaluate ontology quality are theory-based metrics that quantify whether an ontology is correct according to structural properties and description-logic axioms [HS14; Wil+22]. Often inspired by software-engineering research on software quality, these methods use heuristic quality criteria to identify syntactic, semantic, and structural problems that are independent of the application [MS20]. However, because these deductive methods rely on various subjective interpretations of ontology quality, none of them has become standard [Bre+04]. Additionally, verifying whether an ontology meets specific formal criteria does not guarantee optimal performance for a particular purpose [Góm99; MS20].

Inductive evaluation methods assess ontology quality by empirically testing its fitness (i.e., usefulness for a specific application) rather than its syntax, semantics, or structure [Bur+05; Wil+22]. Fitness can be quantified in terms of application fitness, which evaluates performance on a specific task, or domain fitness, which assesses performance across multiple tasks within a domain. Ontology fitness is typically quantified for the entire ontology [PM04; Cla+13], but it can also be quantified for specific parts of the ontology, which can help identify improvement potentials. This process requires linking specific parts of the ontology to application performance, which is not trivial and thus often skipped in practice [PB15].

[PM04], [BGM05], [Bur+05] and [OKM11] argue that inductive evaluation, particularly task-based evaluation, offers an objective measure of ontology quality by directly evaluating the ontology's ability to solve practical problems. Despite this, research in this area is limited. Apart from the original paper introducing task-based ontology quality evaluation [PM04] and a few adaptations [Cla+13; PB15], there is little research on assessing ontology quality based on its utility for specific applications. Both [OKM11] and [Wil+22] have highlighted the need for more research in this area. Specifically, evaluating ontology quality for OGML, which we address in this work, has not been previously explored.

Recent research in confident learning and data-centric AI [WWX18; NJC21; REF23] shows that analyzing predictions from traditional machine-learning methods can uncover and address ontological issues in image label hierarchies, enhancing data quality and prediction performance. Our work follows a similar direction but focuses specifically on ontology-guided machine learning.

2.2.2. Ontology-Guided Machine Learning

Ontology-guided machine learning (OGML) is a subfield of knowledge-guided machine learning (KGML) that leverages structured ontological domain knowledge to enhance machine-learning models. This is usually accomplished with custom loss functions [Zen+17; Ju+24], ontology-aware embeddings [Ven+16; NK17; Che+18; Dha+20; Ber+20], or adapted model architectures [BD19b]. OGML methods have demonstrated significant success in domains rich in ontological background knowledge, such as medical data processing or computer vision.

In healthcare, abundant medical domain knowledge has accumulated through years of medical research, hospital administration, billing, and documentation of medical procedures. This knowledge is often organized into ontologies that group medical codes into semantically meaningful categories using parent-child relationships, e.g., the ICD-9 hierarchy of symptoms and diseases (see Section 5.3.2). OGML approaches leverage these ontologies for various automated medical data processing tasks, such as patient health prediction [Cho+17; Yin+19; Ma+19] or medical text classification [Arb+19]. These methods have been shown to improve prediction performance, especially for rare diseases that are often insufficiently represented in data.

In computer vision, domain knowledge is often structured as taxonomies of labels, reflecting the hierarchical nature of many real-world datasets, such as those in biology [SF11; Rez+22]. Even non-hierarchical datasets can be enriched with knowledge from literature or general domain-independent ontologies [Che+18; BD19b]. OGML approaches in computer vision have been applied to tasks such as image classification [Den+14; Goo+16; MSG17; Che+18; BD19b; Ber+20; Ju+24], image retrieval [Ven+16; BD19a], and semantic segmentation [Li+22].

OGML approaches typically use readily available generic or domain ontologies [Bur+05] rather than task-specific application ontologies. While research often reports that ontological domain knowledge improves average prediction performance compared to models

without it [Dha+20; Kar+21; SF11], there is limited recognition that not all data subsets may benefit equally in the context of a specific task.

2.2.3. Low-Quality Domain Knowledge in OGML

In the specific context of OGML, no studies similar to ours on the problem of domain-application knowledge mismatch have been conducted. However, several related observations have been made regarding the low quality of domain knowledge. For example, [BD19c] investigated the discrepancy between visual and semantic similarity in OGML for image classification. They observed that the overall prediction performance may decrease in some situations compared to knowledge-uninformed baselines. [Cho+17] showed that fully randomized ontological domain knowledge can decrease the overall prediction performance in healthcare OGML applications. The above studies considered the overall negative effect on average prediction performance and did not analyze the prediction performance on subsets of the data. They also did not consider identifying specific parts of ontological domain knowledge that might have caused the decrease in the prediction performance. [Den+14] and [BBD21] investigated the related problem of maximizing the utility of imprecise ontologies in OGML but did not focus on identifying potential quality issues within the ontologies themselves.

The most similar research to our work from Chapter 5 is [MSG17], where the authors analyzed the prediction performance of their OGML approach for image classification across different data subsets. They found that their OGML approach performed worse than the baselines on certain subsets of the data, attributing this to missing relationships in the ontology. While their study provided valuable insights, our work builds upon this by offering a more comprehensive framework that not only broadens the perspective on the underlying issues but also systematically quantifies and addresses them.

Part II.

Addressing Low Quality Domain Knowledge in KGML

3. Addressing the Difficult Process of Integrating Domain Knowledge

The content of this chapter is based on the following publication:

- Pawel Bielski, Nico Denner, Simon Bischof, Benedikt Rzepka, and Klemens Böhm. "Theory-Guided Data-Science for Battery Modeling: Insights from a Comparative Study". In: *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*. 2024, pp. 1–10

Keywords: Theory-guided Data Science, Domain Knowledge, Dynamical Systems Modeling, Battery Modeling

3.1. Chapter Overview

As described in Section 1.2.1, low-quality domain knowledge can result from the difficult process of integrating domain knowledge in practice. This difficulty arises because identifying relevant domain knowledge and selecting the most suitable KGML method for a specific task and available domain knowledge are both difficult. To facilitate identifying relevant domain knowledge in practice, it is essential to conduct studies that identify new types of domain knowledge and demonstrate effective methods for integration. Despite numerous KGML studies contributing to this effort, there are still many forms of domain knowledge that remain unexplored, making this an ongoing challenge. Additionally, comparing different ways of integrating domain knowledge across specific tasks and domains, while considering criteria relevant for practical collaboration with domain experts, can help practitioners select the most appropriate method for their use cases.

To address both these challenges, we propose two novel criteria crucial for domain expert collaboration: *amount of domain knowledge required* and *intellectual implementation effort necessary*. Then we integrate domain knowledge following the five different paradigms of integrating domain knowledge in KGML through a collaborative process with domain experts in an example practical machine learning task: time-series voltage prediction for lithium-ion batteries. We then compare the performance of the resulting five KGML methods and assess them based on these new criteria. All KGML methods in our study are based on neural networks, as they are very suitable to various theory-guided modifications [Wil+20]. Our study findings reveal that while effective KGML methods require substantial domain knowledge, their implementation does not necessarily require significant effort.

The example practical machine learning task we choose for the comparative study is time-series voltage prediction for lithium-ion batteries serves as a demonstration how to compare KGML methods to facilitate the integration process of domain knowledge in KGML. We chose time-series prediction, because of its importance in data mining, and dynamical systems modeling across engineering [TK94; SH19] and natural sciences [Jia+19; ZLS20]. The domain of lithium-ion battery modeling offers abundant **Scientific Knowledge** growing interest in KGML [Tu+23; Sin+23; Nas+23; Li+21], making it ideal for our comparative study. While our findings are based on the observations from this particular task and domain, our study serves as a stepping stone for addressing the challenge of ineffective process of integrating domain knowledge in practice. More studies of similar structure on scope on other tasks and domains will yield deeper insights and solidify the importance of taking into consideration practical aspects of collaborating with domain experts when applying KGML in practice.

Our **Contributions** are as follows:

1. We establish the new evaluation criteria "amount of domain knowledge required" and "intellectual implementation effort necessary" for KGML methods.
2. We describe the use case of time-series voltage prediction in the context of lithium-ion batteries, and describe the abundant available domain knowledge, which makes it a good use case for a comparative study.
3. We propose deployments of existing KGML paradigms to our use case. We instantiate five KGML methods and two baselines for reference. Note that publications introducing a new KGML method or deploying an existing one on a specific use case do not feature these criteria.
4. We propose a two-dimensional taxonomy that goes beyond existing ones. The dimensions are the criteria "amount of domain knowledge required" and "intellectual implementation effort necessary". We classify the KGML methods from our study into the taxonomy.
5. We carry out the study and report on its outcomes. An important finding is that the best-performing KGML methods often require a lot of domain knowledge, but not necessarily much implementation effort.
6. Finally, we provide recommendations on how to apply KGML paradigms in settings with a structure similar to ours. We point to open challenges in KGML and suggest directions for future work.
7. We make the code with the implementation of our experiments available online [1].

[1] `https://github.com/Energy-Theory-Guided-Data-Science/Battery-System`

3.2. Methods

In this section, we begin by outlining the preliminaries for the comparative study, which include the use case description and the available domain knowledge (see Section 3.2.1.1). Next, we introduce two novel criteria essential for domain expert collaboration (see Section 3.2.2). Finally, we propose instantiations of existing KGML paradigms for our use case (see Section 3.2.3).

3.2.1. Preliminaries

In this section, we first describe the use case used in our comparative study (see Section 3.2.1.1). Next, we discuss the available domain knowledge and explain how it can be represented using the feature engineering (see Section 3.2.1.2).

3.2.1.1. Use Case

Lithium-ion batteries are rechargeable energy storage devices widely used in mobility, consumer electronics, or electrical grids [Sin+23]. Due to their high energy density, lithium-ion batteries are sensitive to overcharging and require monitoring of their internal state to ensure they remain within a safe operating range [RBB21; Sin+23]. The internal state of a battery cannot be measured directly and needs to be approximated with physical models, equivalent circuit models, or data science models. Physical models describe the behavior of the battery on the electrochemical level with differential equations. They are the most accurate model, but difficult to parameterize as they require very detailed information about the composition of the battery. Equivalent circuit models describe the behavior of the battery in terms of its input current and output voltage, as an electrical circuit with components as voltage sources, resistors and capacitors. Their model's parameters are derived directly from the battery output voltage response, making them much easier to use in practice. Data science models describe the behavior of the battery in terms of its input current and output voltage, but without prior domain knowledge about the battery. They typically require much training data, which is not necessarily available.

Modeling dynamical systems in terms of its input and output is common in engineering [TK94; SH19] and natural sciences [Jia+19; ZLS20]. In this work, we model the behavior of the battery in terms of its input current and output voltage as a time series prediction. Figure 3.1 shows a typical nonlinear relationship between the input current and output voltage during battery charging. As the current starts flowing into the battery, its voltage gradually increases. It drops after the current stops flowing and converges to the so-called steady-state voltage. The behavior of lithium-ion batteries is heavily researched [Ple18], and there is a lot of domain knowledge on the relationship between input current and output voltage.

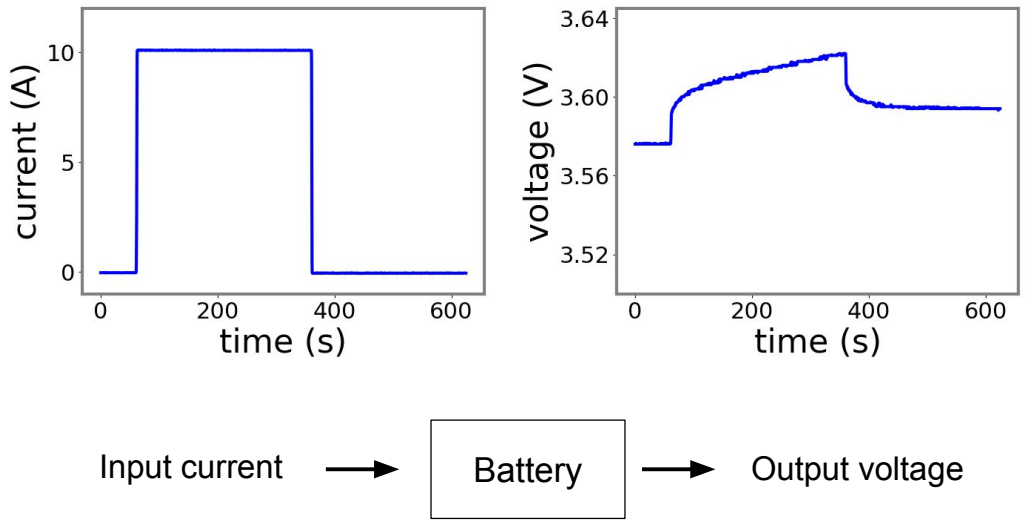


Figure 3.1.: We model the behavior of a lithium-ion battery in terms of its input and output as time series prediction.

3.2.1.2. Available Domain Knowledge

In our study, we have organized the available domain knowledge affecting the shape of the voltage curve into four factors. The first factor is the current $i(t)$ flowing through the battery resistance. It is responsible for the voltage increase during charging and the voltage drop once charging stops. However, it does not explain the increased steady-state voltage at the end of charging. The second factor is the charge $q(t)$, which is proportional to the integral of the current $i(t)$ flowing into the battery:

$$q(t) = \int_{t_0}^{t} i(\tau)\, d\tau. \tag{3.1}$$

This accumulated charge is responsible for the higher output voltage of the battery at the end of charging, compared to the voltage at the beginning of the charging process. The third factor is the initial battery charge q_0 and its corresponding initial voltage v_0. It provides information on the battery state at the beginning of the charging. The fourth factor is the nonlinear relationship between the battery charge $q(t)$ and its output voltage $v(t)$, known as open-circuit voltage (OCV) (see Figure 3.3). It explains the voltage saturation during charging and provides additional information on the steady-state voltage at the end of charging.

It is worth noting that, in addition to the four factors described above, other factors can influence the shape of the voltage curve, such as battery temperature and aging (see [Ple18]). However, including these factors would require detailed data on temperature variations and the aging process, which is not available in the dataset used for this study (see Section 3.3.1.1).

Feature Engineering is a simple and widely used technique for creating new input features, often used to incorporate domain knowledge into data science models [Hea16]. It can be applied alongside TGDS paradigms (see Section 3.3.1.2). In our study, we represent the four factors of domain knowledge as four distinct input features (see Figure 3.2). Then, in Section 3.2.3 we formalize and encode these factors using the TGDS paradigms. We systematically evaluate the performance of implemented TGDS methods across various parameter configurations, including different combinations of these input features in Section 3.3.

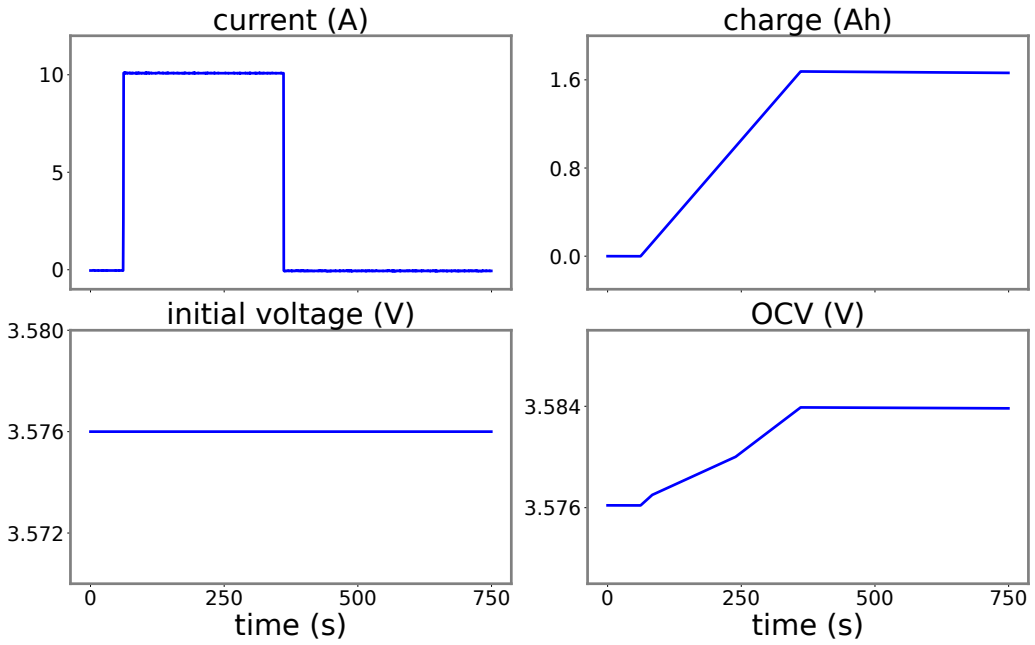


Figure 3.2.: The four input features representing the available domain knowledge in our study.

3.2.2. Establishing Evaluation Criteria

When comparing supervised machine learning methods, it's common to assess their prediction performance on a specific evaluation task. As long as these methods align with the task type (e.g., classification, prediction) and data structure (e.g., time series of numerical values), a data scientist can implement these methods for a new use case with relative ease. However, implementing KGML methods additionally requires use case specific domain knowledge, and the required implementation effort can vary significantly, making some methods easier to use than others. To account for these KGML-specific variations, we introduce two new evaluation criteria.

3.2.2.1. Amount of Domain Knowledge Required

Domain knowledge in KGML can take various forms and is often specific to the use case [Kar+17]. In our study, we observed that some KGML paradigms only need a basic understanding of the data science task for implementation, while others demand a deeper understanding of the use case specifics. To account for these differences, we propose three levels of required domain knowledge to quantify the amount of domain knowledge required for the implementation of KGML methods, from the perspective of a domain-uninformed data scientist.

- **Basic** A data scientist understands the data science task and which parts of the data serve as input and output.
- **Intermediate** A data scientist understands the factors influencing the relationship between the input and the output of the model. However, they do not combine these factors explicitly, leaving this to the prediction model.
- **Expert** In addition to *Intermediate*, a data scientist can explicitly combine several factors influencing the relationship between the input and the output to solve the prediction task.

This categorization is qualitative and is based on the observations from our study. While other quantitative categorizations may seem more objective at first glance — such as defining the required domain knowledge based on the number of physical equations available — they are not applicable to all of the KGML methods, making them hard to use in general.

3.2.2.2. Intellectual Implementation Effort Necessary

Implementing new KGML methods is not trivial because the KGML literature is very use-case specific [RGB23]. In our study, we observed that some KGML paradigms define concrete adjustments to the data science model or the training procedure, while others only serve as loose guidelines for the data scientist. In order to reflect these variations, we propose three degrees of intellectual implementation effort to quantify the time and effort needed to realize them:

- **Low** Only the inputs or the outputs of the data science model need to be adjusted or added. No adjustments to the data science model or the training procedure are required.
- **Medium** Significant adjustments to the architecture of the data science model or the training procedure are necessary, but it is clear what to adjust and how.
- **High** Significant adjustments to the architecture of the data science model or the training procedure are necessary, but it is unclear what to adjust and how.

This categorization is qualitative and is based on the observations from our study. While other quantitative categorizations might seem more objective at first glance — such as lines of code required or the time needed to implement the method — they are subjective and challenging to apply universally. This subjectivity arises from factors like the existing code base, programming language, frameworks used, and the data scientist's programming experience.

3.2.3. Instantiation of KGML Paradigms

In this section we introduce a theory baseline and a data baseline which serve as reference points in our study. Then, for each of the five existing KGML paradigms described in Section (see Section 1.1.3), we describe its principles and show how to apply it to battery-voltage prediction, on top of the Data Baseline architecture described in Section 3.2.3.2. Our selection of KGML paradigms follows the taxonomy from [Wil+20], which identifies five paradigms for integrating domain knowledge: *Loss Function*, *Initialization*, *Architecture*, and *Hybrid Model* (with a distinct subcategory, *Residual Modeling*). It is important to note that there is generally no single optimal way to instantiate a paradigm for a given task. We discuss this issue further in Chapter 7.

3.2.3.1. Theory Baseline

Principle A theory baseline is a model based on domain knowledge that has shown its worth empirically or has been deduced from physical principles [Kar+17]. Such theory-based models often rely on simplifications and may not perform well for complex processes that are not entirely understood. Theory baselines are commonly used, next to data baselines, as reference points to quantify the performance of KGML methods [Jia+19; Daw+20].

Implementation To model the behavior of the lithium-ion battery, we utilize an equivalent circuit model featuring one resistor-capacitor (RC) pair (refer to Figure 3.4). This model is described by Equation 3.2, where OCV (open-circuit voltage) represents the battery voltage at rest when no current flows, and SoC (state of charge) indicates the ratio of electrical charge $q(t)$ to the battery's maximum charge capacity Q. The function $OCV(SoC(t))$ captures the static nonlinear relationship between the State of Charge (SoC) and the open-circuit voltage (OCV), as depicted in Figure 3.3. This relationship is specific to the tested battery and is measured in a laboratory setting. Equation 3.2 mathematically captures the lithium-ion battery's behavior, including electrical charge, battery resistance, and diffusion processes. We determine the parameter values for this model based on the methodology outlined in [Ple18].

$$v(t) = \mathrm{OCV}(SoC(t)) - R_1 \cdot i_{R_1}(t) - R_0 \cdot i(t) \tag{3.2}$$

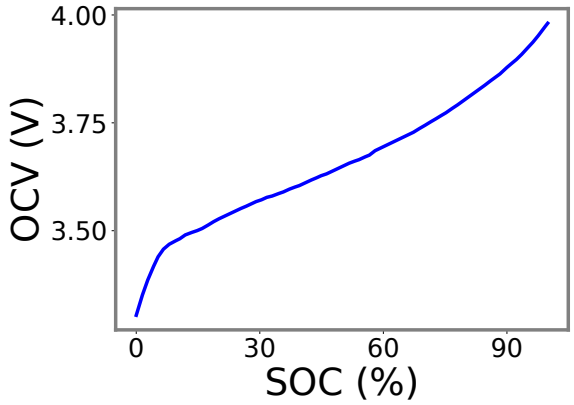


Figure 3.3.: The static nonlinear relationship between State of Charge (SoC) and the open-circuit voltage (OCV).

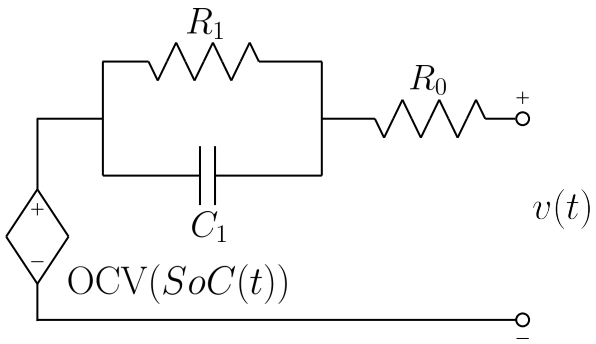


Figure 3.4.: The equivalent circuit model of the lithium-ion battery.

3.2.3.2. Data Baseline

Principle Data baselines are standard machine learning models that can be used to solve a concrete prediction task, e.g., voltage prediction. They thrive in commercial applications but are prone to learning physically inconsistent relationships in the scientific domains, especially if the available data is small [Kar+17]. Data baselines are commonly used as reference points to quantify the usefulness of domain knowledge of KGML methods [Jia+19; Daw+20].

Implementation Predicting voltage time series given the current time series is a time series prediction task. We leverage a LSTM network which excel in learning long-term dependencies through their memory cell mechanism and are commonly used in time series prediction tasks related to lithium-ion batteries [Zha+17; Wei+20; Par+20]. We use two versions of an LSTM network: one has a single layer, and the other has two layers. Both versions include 20 LSTM units per layer and a fully connected output layer. For more details about the experimental setup, see Section 3.3.1.2.

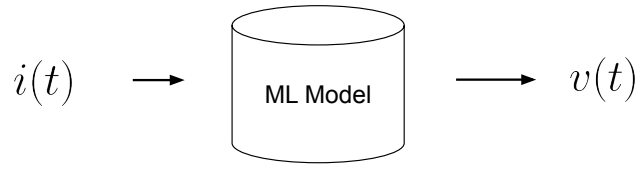
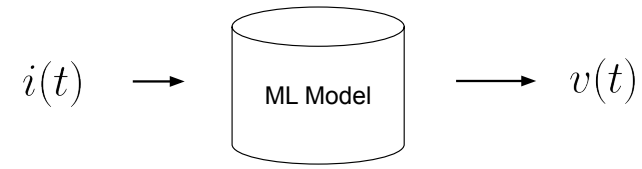


Figure 3.5.: Data Baseline is a standard LSTM neural network.

3.2.3.3. Knowledge-guided Loss Function

Implementation Building on the Knowledge-guided Loss Function paradigm, we introduce a regularization term $\text{Loss}_{Physical}$ to account for the static nonlinear relationship between the State of Charge (SoC) and voltage (see Figure 3.3) in a similar way as in [Jia+19; Daw+20]. This term penalizes the difference between the predicted and actual SoC based on voltage, as depicted in Figure 3.6. The SoC is calculated as the ratio of the electrical charge $q(t)$ to the maximum charge Q that the battery can hold.



$$Loss_{Physical}(\hat{v}, v) = MSE(SoC(\hat{v}), SoC(v))$$

Figure 3.6.: Loss penalizes deviations between predicted and actual SoC.

3.2.3.4. Knowledge-guided Initialization

Implementation To apply the Knowledge-guided Initialization paradigm, we employ a two-step transfer learning approach (refer to Figure 3.7) as described in [Jia+19]. In the first step, we pre-train the model using the actual input current $i(t)$, charge $q(t)$, and the output voltage $v_{Th}(t)$ simulated with the theory baseline for the first half of the training. Subsequently, we fine-tune the model with the actual current $i(t)$, charge $q(t)$, and voltage $v(t)$ data for the second half of the training.

3.2.3.5. Knowledge-guided Architecture

Implementation To apply the Knowledge-guided Architecture paradigm, we opt for conditional LSTM cells [Rem20] rather than the standard LSTM cells to incorporate information about the initial voltage v_0, as depicted in Figure 3.8. Unlike the original LSTM, which processes each sequence step solely based on input data and internal states, the conditional LSTM integrates extra time-invariant details to guide predictions at each sequence step. The overall network architecture remains unchanged from the Data Baseline.

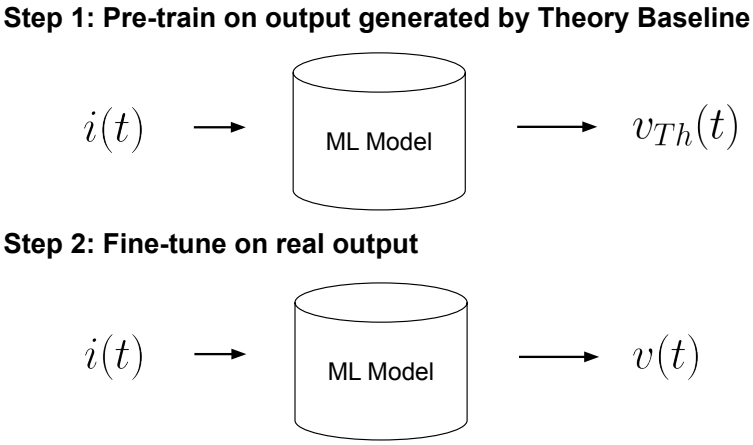


Figure 3.7.: The model is pre-trained on simulated output voltage for the first half of training, then fine-tuned on measured output voltage for the second half.

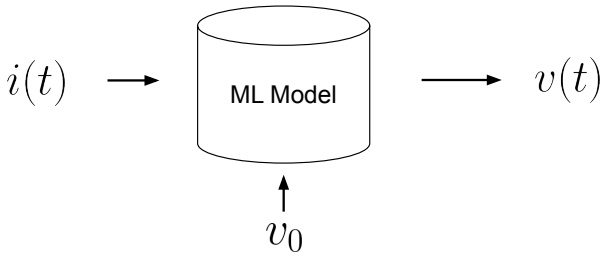


Figure 3.8.: Initial voltage is passed to the model using conditional LSTM cells.

3.2.3.6. Knowledge-guided Hybrid Model

Implementation To apply the Knowledge-guided Hybrid paradigm, we use the output of the theory model as the input to the machine learning model, in addition to other features, in a similar way to [Daw+17] and [Wil+98]. To be specific, we feed the simulated voltage output of the theory model $v_{Th}(t)$ into the machine learning model as a separate input feature (refer to Figure 3.9).

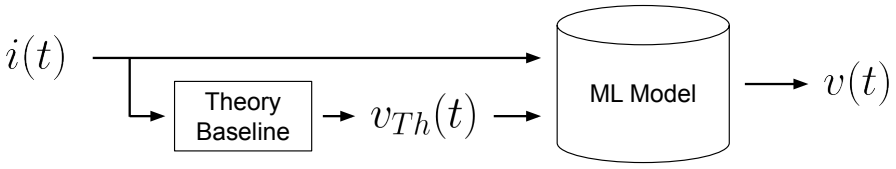


Figure 3.9.: Simulated voltage output is passed as a separate input feature.

3.2.3.7. Knowledge-guided Residual Learning

Implementation To apply Knowledge-guided Residual Learning paradigm, we adjust the machine learning model to predict the gap between the voltage output of the theory model $v_{Th}(t)$ and the actual measured voltage $v(t)$ (see Figure 3.10), in a similar way as in [Zhe+21].

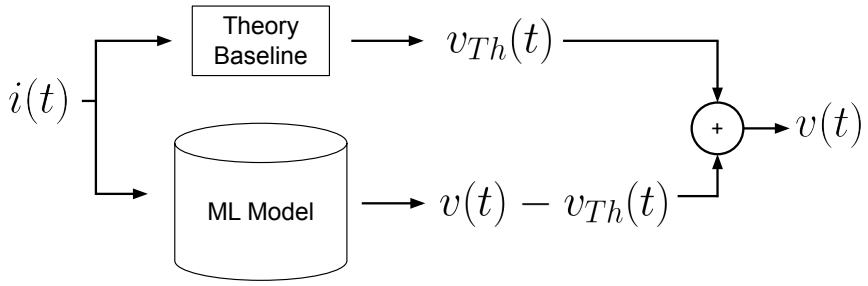


Figure 3.10.: Model predicts the gap between the theory and actual output voltage.

3.3. Experiments

In this section, we first describe the data and experimental setup for the comparative study (see Section 3.3.1). Then we report on the performance of various KGML methods on the three test profiles (Figure 3.13 and Table 3.1), and then classify the KGML methods according to our new evaluation criteria (Table 3.4), and explain our thought process.

3.3.1. Experimental Design

3.3.1.1. Data

In our experiments, we use the FOBSS dataset [Ste+19] that consists of charging and discharging profiles of varying complexity collected from a battery system with four battery packs, each containing eleven cells. Since each battery cell can have different physical characteristics, in the experiments we focus on learning the behavior of a one representative battery cell situated in the middle of the first battery pack (on position 5). Each profile consists of the input current curve and the corresponding output voltage curve. We use six profiles as training data for each of the KGML methods and the Data Baseline, with each representing a cell's response to a 10A current pulse lasting approximately 300 seconds. Depending on the battery's initial voltage, the voltage output curves differ slightly (see Figure 3.11). Note that the *Theory Baseline* is not trained with these six profiles; instead, it is parameterized using the Reproduction profile, following the methodology described in Section 3.2.3.1.

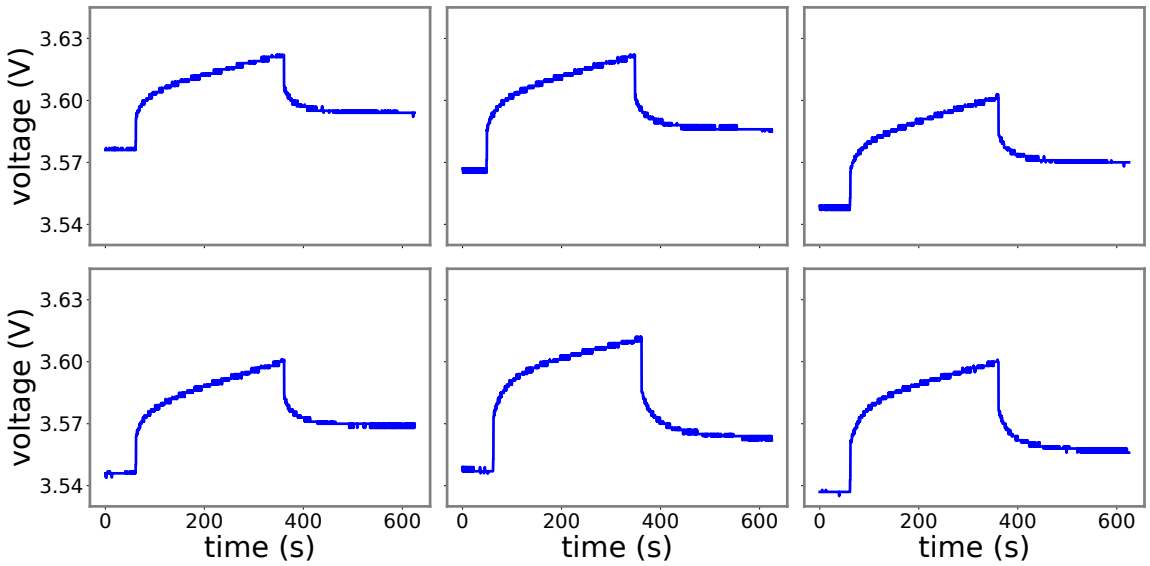


Figure 3.11.: The six voltage profiles used for training.

To evaluate the methods, we use three test profiles (Figure 3.12). Each profile captures the battery cell's behavior in different realistic scenarios.

- The **Reproduction** profile tests the performance of the models using one specific voltage profile selected from the training data. Similar to the other training data, this profile represents a cell's response to a 10A current pulse.
- The **Abstraction** profile tests whether a model can abstract into an unseen profile with similar characteristics to the training data. This profile represents a cell's response to three consecutive 10A current pulses.
- The **Generalization** profile tests whether a model can generalize to an unseen profile with different characteristics. This profile represents a cell's response to a negative 10A current pulse.

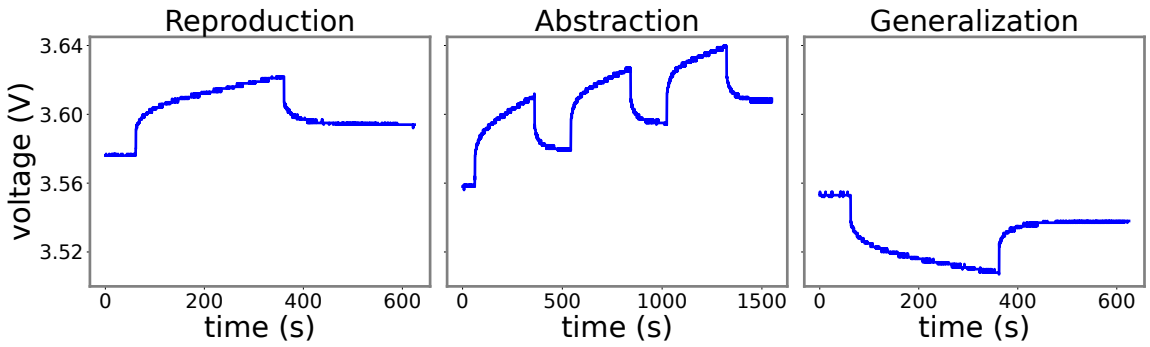


Figure 3.12.: The three test profiles: reproduction, abstraction, and generalization.

3.3.1.2. Experimental Setup

In the study, we aim to systematically evaluate the performance of different KGML methods on the battery voltage-prediction task. For the data baseline and all KGML methods, we employ the same LSTM-based network to predict the output y_t using the N=100 most recent samples $x_t, x_{t-1}, \ldots, x_{t-N+1}$ of the input, corresponding to the 50 seconds of the input current. We use Adam optimizer, a learning rate of 0.001, and the mean squared error loss function. Each network is trained with four variants of input features: 1) only current, 2) variant 1 plus charge, 3) variant 2 plus initial voltage, and 4) variant 3 plus OCV (see Figure 3.2). We test the network in two size variants (1 and 2 layers) of 20 LSTM units per layer and a fully connected output layer. Additionally, we train each network for various number of epochs (10, 20 and 50) to account for the fact that networks with different sizes and different number of input features may require different training lengths to reach the optimum. Overall, we examine 24 parameter combinations including 2 layer variants, 3 epochs variants and 4 feature variants. We repeat the experiment for each parameter combination five times to account for the network initialization randomness and use the root mean squared error (RMSE) as performance metric. For the *Theory-guided Loss Function*, we set $\lambda_{Physical} = 1$ for all experiments, as it performed best on average among values ranging from 0.001 to 1000 (refer to Equation 1.1). We use the same model for all relevant KGML methods. The *Theory Baseline* uses a single parameter configuration based on the information from all four input features and is parameterized using the Reproduction profile, following the approach outlined in Section 3.2.3.1. For further details, please refer to our published code implementation.

3.3.2. Performance of KGML Methods

Table 3.1.: Mean RMSE ± standard deviation of the best-performing parameter configurations across all 24 configurations for Data Baseline and KGML methods. The Theory Baseline uses a single parameter configuration based on the information from all four input features. Best outcomes are highlighted in bold.

Method	Reproduction	Abstraction	Generalization	Average
Theory Baseline	4.4 ± 0.0	13.4 ± 0.0	15.7 ± 0.0	11.2 ± 0.0
Data Baseline	9.0 ± 3.7	6.4 ± 1.3	6.2 ± 2.9	7.9 ± 2.8
Loss Function	9.3 ± 3.6	6.7 ± 2.6	**4.6 ± 4.5**	8.5 ± 2.9
Initialization	9.0 ± 3.9	7.0 ± 1.0	4.7 ± 1.2	8.5 ± 1.5
Architecture	9.5 ± 4.2	6.5 ± 1.0	5.6 ± 3.8	7.9 ± 1.6
Hybrid	**3.3 ± 1.5**	**3.6 ± 2.0**	5.3 ± 2.9	**4.1 ± 1.7**
Residual	3.5 ± 2.5	4.4 ± 1.1	6.2 ± 4.5	5.3 ± 3.1

Figure 3.13 shows that while the *Theory Baseline* excels at reproducing the training profile, the *Data Baseline* and other KGML methods outperform it on the unseen Abstraction and

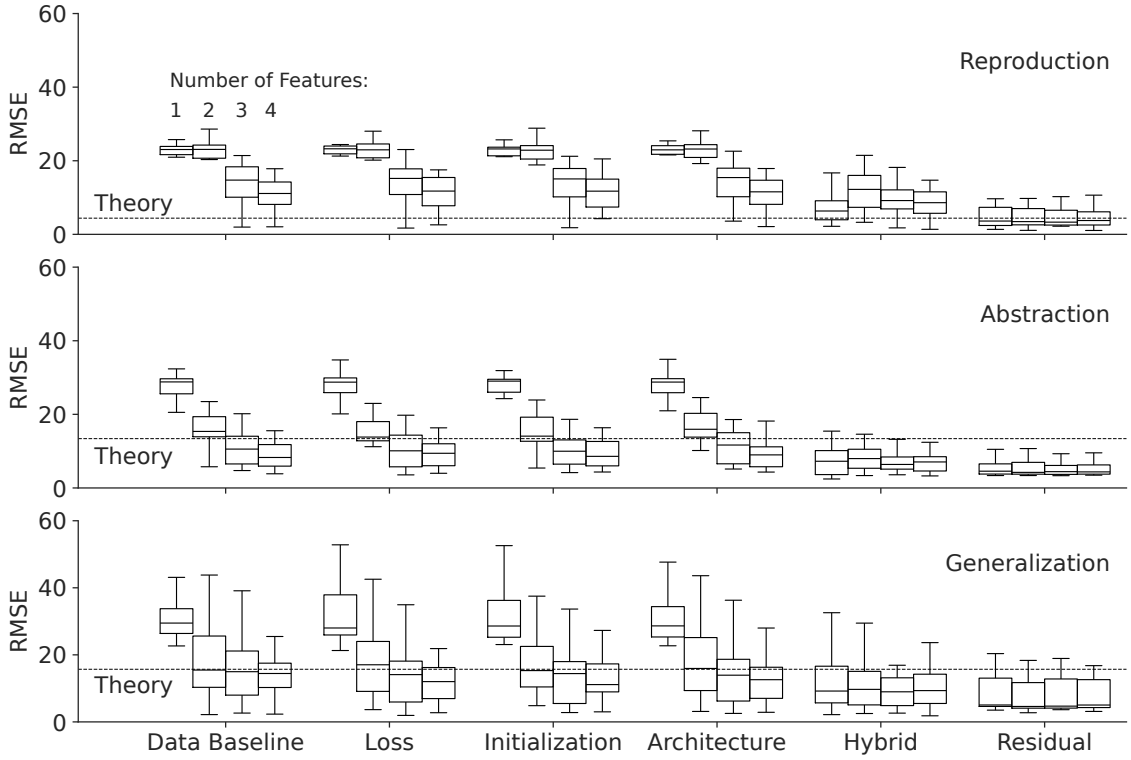


Figure 3.13.: Experimental results for all test cases. Each box plot illustrates results over two network sizes and three epoch variants, evaluated for all four variants of input features. The Theory Baseline is parameterized using the information from all four input features.

Table 3.2.: Frequency of each method achieving the best/worst performance across all 24 configurations of parameters. Theory Baseline uses a single parameter configuration based on the information from all four input features. Best outcomes are highlighted in bold.

Method	Reproduction	Abstraction	Generalization	Average
Theory Baseline	**14** / 0	0 / 1	0 / 1	0 / 1
Data Baseline	0 / 8	0 / 6	1 / 4	0 / 5
Loss Function	0 / 7	0 / 5	2 / 2	0 / 4
Initialization	0 / 4	0 / 4	0 / 10	0 / 8
Architecture	0 / 5	0 / 8	0 / 6	0 / 6
Hybrid	2 / 0	8 / 0	3 / 0	4 / 0
Residual	8 / 0	**16** / 0	**18** / 1	**20** / 0

Generalization test profiles with sufficient *Feature Engineering*. Notably, the *Hybrid* and *Residual* methods consistently outperform all other methods, regardless of the number of features used. Except for these two KGML methods, the others do not consistently surpass the *Data Baseline*. For example, incorporating the initial voltage information into

Table 3.3.: Mean RMSE ± standard deviation of the best-performing parameter configurations across the 2 network sizes and 3 epoch variants using all four input features for Data Baseline and KGML methods. The Theory Baseline uses a single parameter configuration based on the information from all four input features. Best outcomes are highlighted in bold.

Method	Reproduction	Abstraction	Generalization	Average
Theory Baseline	4.4 ± 0.0	13.4 ± 0.0	15.7 ± 0.0	11.2 ± 0.0
Data Baseline	9.0 ± 3.7	6.4 ± 1.3	7.3 ± 5.7	7.9 ± 2.8
Loss Function	9.3 ± 3.6	6.7 ± 2.6	6.5 ± 3.4	8.5 ± 2.3
Initialization	9.0 ± 3.9	7.0 ± 1.0	9.0 ± 3.6	8.6 ± 1.5
Architecture	9.5 ± 4.2	6.5 ± 1.0	6.9 ± 3.5	7.9 ± 1.6
Hybrid	6.7 ± 3.7	5.5 ± 1.4	7.2 ± 5.4	6.5 ± 1.9
Residual	**3.5 ± 2.5**	**4.7 ± 1.4**	**6.3 ± 4.3**	**5.4 ± 3.1**

the *Data Baseline* through *Feature Engineering* (using 3 features) is just as effective for Generalization, and even more effective for Reproduction and Abstraction profiles, than using *Architecture* paradigm (using 2 features).

While the best parameter configuration of *Hybrid Model* achieves the best prediction performance on all test profiles on average (see Table 3.1), *Residual Learning* performs best for 20 out of 24 parameter configurations tested (see Table 3.2). Both of these two KGML methods performed best when trained for 50 epochs, while other KGML methods performed best when trained for 20 epochs. Except for *Residual Learning*, all KGML methods achieve best performance when trained on a smaller network. For individual test profiles, the *Theory Baseline* outperforms all other methods on 14 out of 24 parameter configurations for Reproduction but performs significantly worse on average compared to the other methods for Abstraction and Generalization profiles (see Table 3.1 and Table 3.2). When considering only the parameter configurations that include all four features (for a fairer comparison with the Theory Baseline, which also uses information from all four input features), we observe in Table 3.3 that *Residual Learning* achieved the best performance across all test profiles.

3.3.3. Qualitative Evaluation of Predictions

The Figure 3.14 illustrates how including extra features affects the predictions of the Data Baseline. We notice that when the model is trained solely on current time-series data, it struggles to accurately represent the increased steady-state voltage at the end of charging. However, incorporating charge enables the model to capture this increase better. Nonetheless, the model's performance on the Reproduction test profile remains low, indicating it fails to learn the dependence on the initial voltage, resulting in a noticeable deviation from the ground truth. Introducing the initial voltage as an additional feature helps the model account for this dependence. Finally, including the OCV feature allows

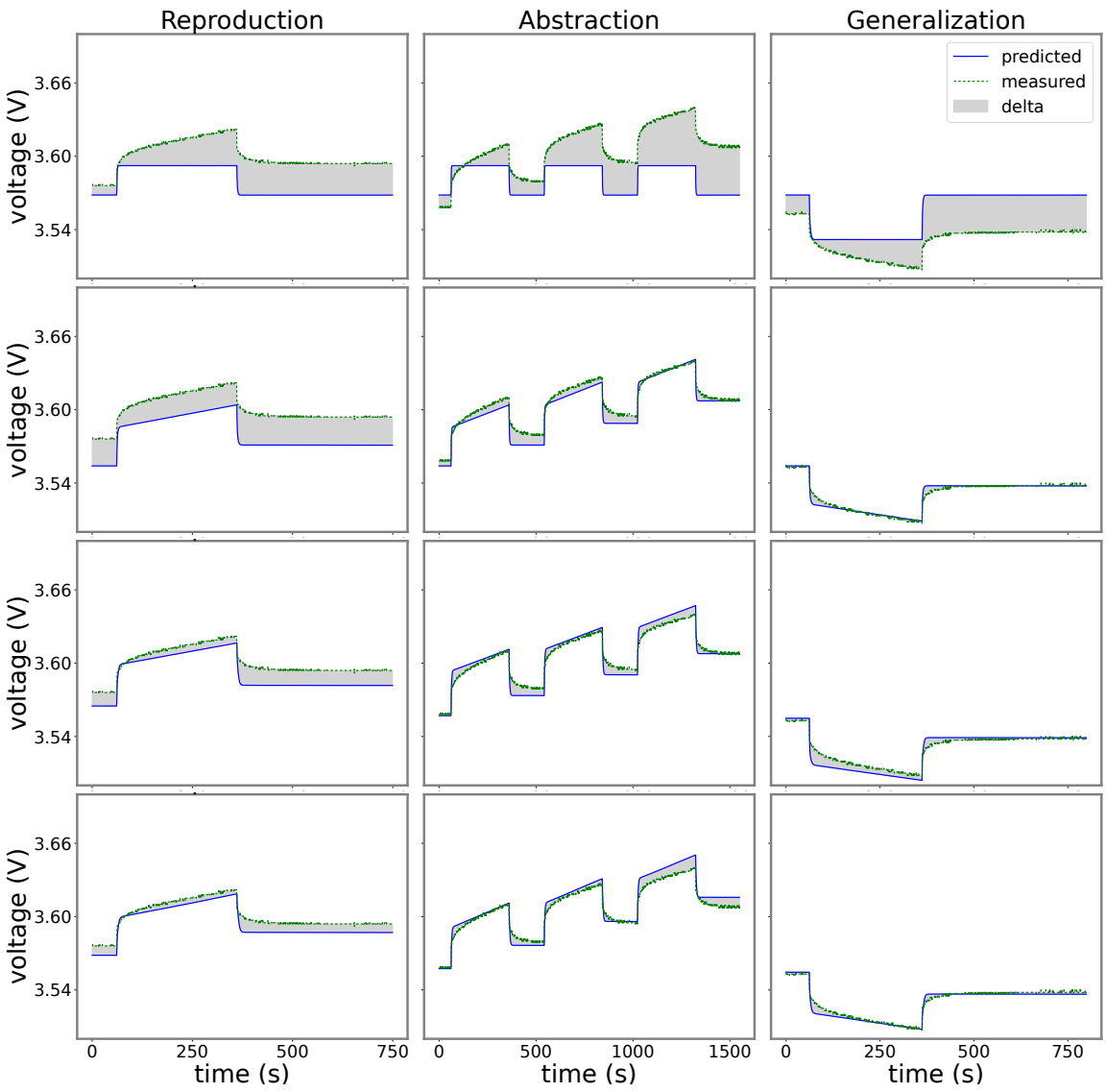


Figure 3.14.: Predictions from the best of five Data Baseline models trained with 1, 2, 3, and 4 input features (from top to bottom), using the parameter configuration that performed best on average of three test profiles (1 layer, 20 epochs).

the model to reduce the discrepancies especially in the Reproduction and Generalization profiles.

In Figure 3.15, we observe the predictions of both the Theory Baseline (top) and the best-performing Hybrid model (bottom). The Theory Baseline notably captures the Reproduction profile very accurately, which is expected since it was parameterized with that profile. However, it does not accurately represent the peak values of each curve in the Abstraction profile, and it struggles to model the decreased steady-state voltage at the end of charging in the Generalization profile. Conversely, while the Hybrid model performs slightly worse on the Reproduction profile, it effectively reduces errors in both the Ab-

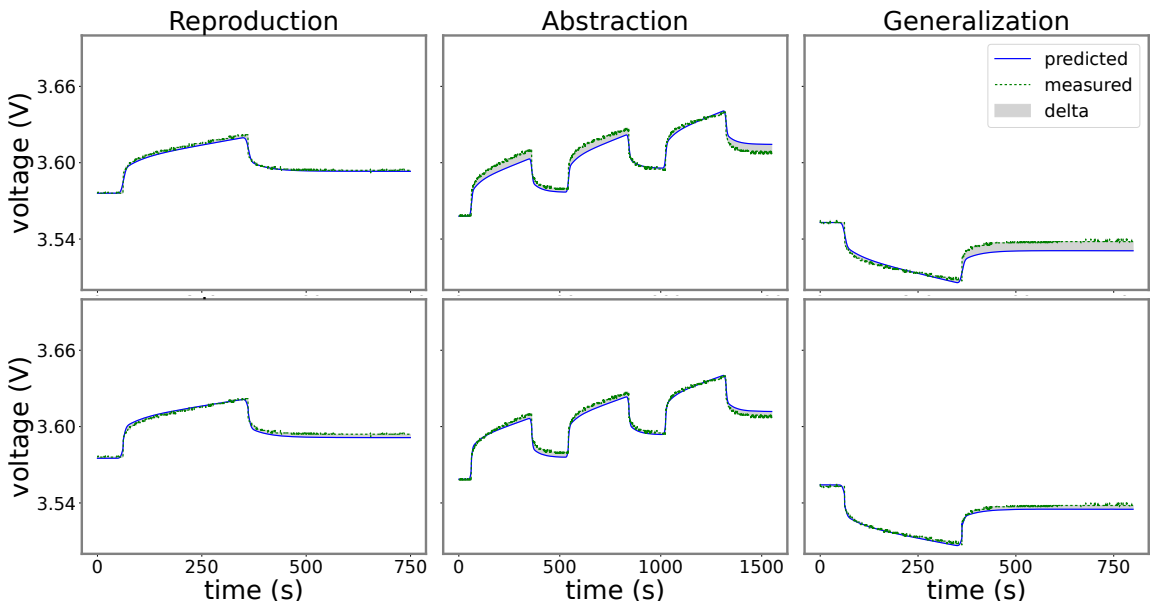


Figure 3.15.: Predictions from the Theory Baseline model (top) and the best of five Hybrid models (bottom) trained the parameter configuration that performed best on average (1 feature, 1 layer, 50 epochs).

straction and Generalization profiles, ultimately achieving the best overall performance on the three test profiles.

3.3.4. Classification of KGML Methods

Table 3.4 presents the classification of the implemented KGML paradigms into our proposed taxonomy along with the theory baseline, data baseline and feature engineering, as perceived from the perspective of a domain-uninformed data scientist. In the text, we list the concrete KGML methods in the order of their implementation in our study. We started with implementing the methods that require the least amount of domain knowledge. We did this because it seemed intuitive to us to acquire more domain knowledge on the fly while implementing various KGML paradigms. Note that this order turned out to be suboptimal in the aftermath of the study. We will discuss this further in Section 3.4.

Data Baseline required basic degree domain knowledge to understand the time-series prediction task with voltage $v(t)$ as the output. The implementation effort was low, involving the use of a standard machine learning model for time series prediction without specific modifications.

Feature Engineering required intermediate degree of domain knowledge to understand and model several factors influencing the shape of the voltage curve. The implementation effort was low, involving only adjustments to the model inputs.

Knowledge-guided Loss Function required an intermediate degree of domain knowledge to understand and encode the relationship between the battery charge $q(t)$ and voltage

$v(t)$. The intellectual implementation effort was medium, as there exist literature on how to adjust the loss function to make the output voltage consistent with the expected State of Charge.

Knowledge-guided Architecture required an intermediate degree of domain knowledge to understand and encode the initial voltage v_0. The intellectual implementation effort necessary to instantiate the method was high, as it was not straightforward to determine the necessary architectural changes.

Knowledge Baseline required expert degree of domain knowledge because it required understanding and combining several factors influencing the behavior of the battery. The intellectual implementation effort was low, because there exist detailed explanations in the literature how to implement and parameterize the theory baseline from scratch [Ple18].

Knowledge-guided Hybrid Model required an expert level of domain knowledge because it relied on a correctly parameterized theory model. Since such a model was available, the implementation effort was low, as the data scientist only needed to include the output of the theory baseline as an input feature for the model.

Knowledge-guided Residual Learning required an expert level of domain knowledge because it relied on a correctly parameterized theory model. Since such a model was available, the implementation effort was low, as the data scientist only needed to change the model's output to predict the gap between the theory baseline and the ground truth voltage.

Knowledge-guided Initialization required an expert level of domain knowledge because it relied on a correctly parameterized theory model. Since such model was available, the implementation effort was medium, as the data scientist needed to adjust the training procedure, but it was clear how to do so.

3.4. Discussion

In this section we discuss observations from the study. We first comment on the KGML methods we have implemented. Then we discuss our evaluation criteria - *amount of domain knowledge required* and *intellectual implementation effort necessary* and how they fit into the existing KGML taxonomies. We then propose a guideline for effectively applying KGML paradigms in projects with a similar problem structure to ours.

3.4.1. KGML Methods

In previous KGML projects, such as those discussed in [Jia+19] and [Daw+20], KGML methods consistently outperformed both the theory baseline and the data baseline by a significant margin. Our study confirms this finding. Interestingly, KGML methods that required medium to high implementation effort, such as *Loss Function* or *Architecture*, did not yield better results than other KGML methods. On the contrary, KGML methods

Table 3.4.: The classification of the implemented KGML paradigms according to our two evaluation criteria.

Domain Knowledge	Low	Medium	High
Expert	Theory Baseline Residual Learning Hybrid Model	Initialization	
Intermediate	Feature Engineering	Loss Function	Architecture
Basic	Data Baseline		
	Low	**Medium**	**High**
	Implementation Effort		

with low implementation effort, such as *Hybrid Model* or *Residual Learning*, directly incorporated expert knowledge from the theory baseline and outperformed other KGML methods. Overall, we observed that applying KGML methods to our specific use case has become easier over time as our understanding of the problem deepened. This underscores the importance of grasping the problem thoroughly before employing KGML methods. Throughout our study, we gained more and more domain knowledge during the project as we applied KGML paradigms. In hindsight, this iterative process led us to begin with methods that required basic knowledge, which proved to be time-inefficient.

3.4.2. Proposed Taxonomy

We've observed that the effectiveness of KGML methods tends to rise as the required domain knowledge increases. While this aligns with expectations and likely holds true for various settings aiming to model system input-output behavior over time, we did not find a clear correlation between required domain knowledge and the intellectual effort needed for implementation. Surprisingly, the top-performing KGML methods in our study, the Hybrid Model and Residual Learning, both leverage expert domain knowledge but require minimal implementation effort. Considering KGML methods in terms of required domain knowledge and implementation effort enhances existing taxonomies, as proposed in [Kar+17], [Wil+20] and [Rue+23], facilitating method selection. Given the wide variety of KGML use cases in the literature, comparative studies like ours are valuable for identifying KGML methods suited to specific data science tasks. However,

further research is needed to confirm and generalize the findings from our study beyond the time series prediction context explored in our work.

3.4.3. Recommendations for KGML Projects

From the perspective of a data scientist without domain expertise, it is essential to thoroughly understand the use case before implementing any KGML methods. Implementing and validating a *Theory Baseline* model, if available, with domain experts helps in understanding the problem and facilitates the application of various KGML paradigms with minimal effort. Along with the *Data Baseline* it provides a performance benchmark, as the KGML methods are expected to perform at least as well as these baseline methods. We recommend to start with *Hybrid Model* or *Residual Learning* as these paradigms directly leverage expert knowledge from the *Theory Baseline*, are easy to implement, and can effectively leverage both data and theory without necessarily requiring additional features to perform effectively. Good understanding of the use case also facilitates *Feature Engineering*, another simple but effective way to incorporate domain knowledge into any data science model. Starting with KGML methods that require a medium or high implementation effort, such as *Loss Function* or *Architecture* may not be time-efficient. We believe this strategy is applicable to problem settings where the goal is to model the input-output behavior of a system over time, and possibly for other problem settings as well. Additionally, we recommend training the model multiple times and selecting the best one to mitigate the effects of randomness in network initialization.

3.5. Summary

In this chapter we addressed the challenge of low-quality domain knowledge stemming from its difficult integration process in practice. We first proposed two novel criteria crucial for domain expert collaboration: *amount of domain knowledge required* and *intellectual implementation effort necessary*. Then we integrated domain knowledge following the five different paradigms of integrating domain knowledge in KGML through a collaborative process with domain experts in an example practical machine learning task: time-series voltage prediction for lithium-ion batteries. We then compared the performance of the resulting five KGML methods and assess them based on these new criteria. Our study findings revealed that while effective KGML methods require substantial domain knowledge, their implementation does not necessarily require significant effort.

3.6. Author's Contribution

The idea to apply various KGML methods to model the lithium-ion battery was mine. The initial implementations of the KGML methods, including identifying the available

domain knowledge, were carried out by Nico Denner during his Bachelor's thesis [Den21] under my supervision. The domain knowledge experts for this project were Simon Bischof and Benedikt Rzepka. Recognizing that the process of applying domain knowledge in practice is difficult was my observation. The idea to explore the practical aspects of collaboration with domain experts came from Prof. Klemens Böhm, and the criteria were defined collaboratively by him and me. The final experimental design, including the systematic evaluation of the KGML methods for various parameters, was defined, implemented, evaluated, and visualized by me. All sections of the publication were written by me, with support and feedback from Prof. Klemens Böhm.

4. Addressing the Imperfect Representation of Domain Knowledge

The content of this chapter is based on the following publication:

- Pawel Bielski, Aleksandr Eismont, Jakob Bach, Florian Leiser, Dustin Kottonau, and Klemens Böhm. "Knowledge-Guided Learning of Temporal Dynamics and its Application to Gas Turbines". In: *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*. e-Energy '24. Singapore, Singapore, 2024, pp. 279–290

Keywords: Dynamical Systems, Dynamics Modeling, Micro Gas Turbine, Domain Knowledge, Physics-Guided Deep Learning

4.1. Chapter Overview

As described in Section 1.2.2, domain knowledge can be imperfect, due to difficulties in its collection, definition and representation. In this thesis, we identify two types of imperfect domain knowledge: approximate domain knowledge and incorrect domain knowledge. Both of these types of imperfect domain knowledge should ideally be accounted for during the development and evaluation of KGML methods. However, despite numerous KGML studies introducing new forms of domain knowledge and developing methods to incorporate it across various use cases, there is limited research focused on designing and evaluating KGML approaches that can effectively represent approximate domain knowledge and remain robust against incorrect domain knowledge.

To address this challenge, we demonstrate how to design and evaluate a new KGML approach that incorporates a novel type of **Expert Knowledge**, to handle approximate and incorrect domain knowledge effectively. We define this novel type of domain knowledge as *permissible system states*, which is especially relevant for modeling the temporal dynamics of systems encountered in engineering using neural networks. While system states play a big role in control theory and can be estimated by (state) observers [RG06; BAA22], they are typically not incorporated into deep learning methodologies. The proposed *permissible system states* characterize the dynamic states a system is allowed to take during its operation. Deviations from these states are penalized while training the machine learning model, preventing it from adopting behavior inconsistent with the domain. For example, *permissible system states* can be used to inform the model about the expected

change rates of an output signal. Such knowledge is particularly useful for modeling the system's behavior in response to an input control signal during transition phases, where the output signal shows a delayed response to a rapid change of an input control signal. This is a common scenario when operating energy systems that are vital for the carbon-free modern energy grid, such as gas turbines, energy storage systems, or power-to-X systems.

Because existing knowledge-guided methods do not allow to model *permissible system states*, we propose a novel *multi-state constraint* to represent and incorporate this type of domain knowledge into a neural network, using a knowledge-guided loss function. Importantly, this proposed method can be applied on top of any deep learning model and can be used with both exact and approximate domain knowledge. Approximate domain knowledge is particularly relevant in practical applications where precisely defining *permissible system states* can be challenging due to measurement uncertainties or the stochastic nature of the dynamical system under consideration. To the best of our knowledge, our work is the first attempt to leverage this type of domain knowledge in the context of KGML.

To study the effectiveness of our method, we apply it to model the temporal dynamics of a gas turbine in a scenario where measurements from the internal system components are unavailable, which prevents applying conventional theory-based input-output models. In this context, the domain knowledge is specified as the expected change rate of the gas turbine's output during transition and stationary phases. Our findings indicate that our method allows to create an accurate data-driven model with minimal effort and data requirements. It reduces the prediction error by up to 40%, while simultaneously reducing the need for extensive data and mitigating training randomness, compared to the knowledge-uninformed deep learning baseline. Additionally, the method aids the model to behave consistent with the domain and enhances its ability to generalize to data with varying characteristics, especially in scenarios with limited available training data.

We have made the code[1] and the experimental data[2] available online. Additionally, we have contributed the data to the UCI Machine Learning Repository under the name *Micro Gas Turbine Electrical Energy Prediction*. To sum up, our contributions are as follows:

1. We identify and formalize *permissible system states*, a novel type of domain knowledge that is relevant for data-driven temporal dynamics modeling of engineering systems with unknown or partially known system structure.
2. We introduce a novel *multi-state constraint* designed to incorporate this domain knowledge into any neural network, using a knowledge-guided loss function.
3. We thoroughly evaluate our approach on the task of data-driven temporal dynamics modeling of a gas turbine. This includes performance tests with varied dataset sizes, a sensitivity analysis of hyperparameters, assessing robustness against approximate and incorrect domain knowledge, and a qualitative prediction analysis.

[1] `https://github.com/Energy-Theory-Guided-Data-Science/Gas-Turbine`
[2] `https://doi.org/10.35097/sLJiahifxvfDKMEc`

4.2. Methods

In this section, we begin by outlining the preliminaries for our proposed KGML approach to temporal dynamics modeling. This includes an introduction to temporal dynamics modeling, a description of the domain knowledge identified in our scenario, and an overview of integrating knowledge with loss functions (see Section 4.2.1). Next, we present our new KGML approach in two variants: one for exact domain knowledge representation and another for approximate representation (see Section 4.2.2).

4.2.1. Preliminaries

In this section, we present fundamentals for our work. First, we introduce the scenarios of temporal dynamics modeling in general (Section 4.2.1.1) and gas turbine modeling in particular (Section 4.2.1.2). Next, we discuss the limitations of classical deep learning in such scenarios (Section 4.2.1.3). Further, we describe the domain knowledge identified in our scenario (Section 4.2.1.4). Then, we explain how loss functions are used in knowledge-guided machine learning (Section 4.2.1.5). Finally, we review the limitations of previous work in gas turbine temporal modeling (Section 4.2.1.7).

4.2.1.1. Problem: Temporal Dynamics Modeling

In this study, we focus on discrete-time nonlinear dynamical systems that evolve over time in response to an input control signal. Such a system is characterized by an unknown function f:

$$y_t = f(x_t, x_{t-1}, \ldots, x_{t-N+1}) \tag{4.1}$$

Here, $x_t \in X$ is the input control signal at time t, and $y_t \in Y$ is the system output signal at time t, which depends on the N last samples of the input control signal. We assume to have access to a dataset comprising tuples of input-output value pairs $((x_t, y_t) \mid t \in \{1, \ldots, T\})$. T is the number of discrete time samples in the dataset. The objective of temporal dynamics modeling with deep learning is to estimate the unknown function f using a parameterized function f_θ, where $\theta \in \mathbb{R}^p$ are parameters of a neural network, to predict the output y_t given the control input $x_t, x_{t-1}, \ldots, x_{t-N+1}$.

4.2.1.2. Our Scenario: Gas Turbine Modeling

Gas turbines play an important role in modern power and heat generation systems, serving various functions such as meeting peak demand, providing backup for renewable energy sources, and enabling decentralized power generation with minimal transportation and conversion losses [BA22; GT16]. Additionally, gas turbines will play a crucial role in future cross-sector power-to-gas energy systems, which enable the conversion of electricity into gas, its storage, transportation, and subsequent conversion back to electricity [Göt+16]. Their flexibility to operate on a range of fuels, including biofuel and hydrogen, enables

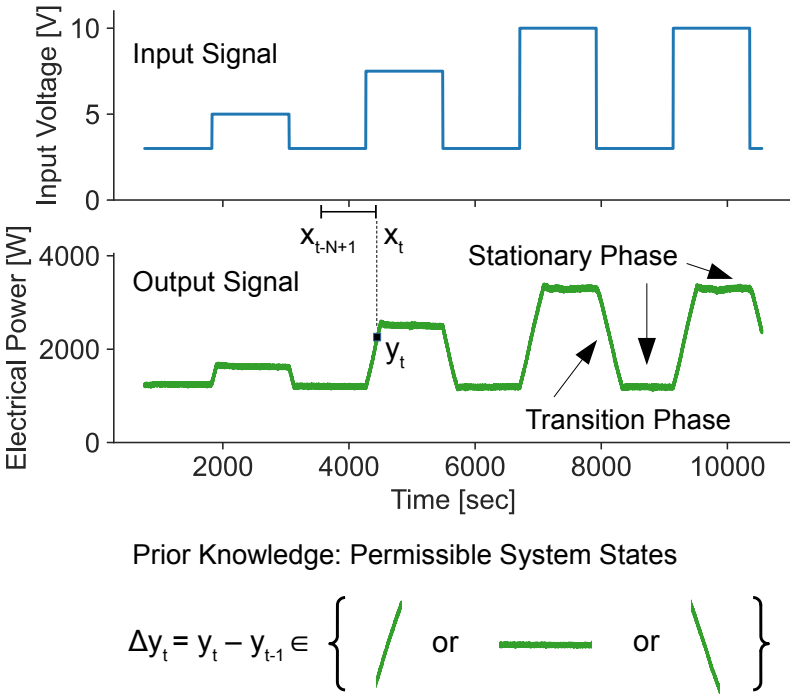


Figure 4.1.: Temporal dynamics modeling of an output signal in response to an input control signal, and the identified domain knowledge in the form of permissible system states.

them to fulfill these tasks while achieving net-zero carbon emissions [De 23]. Accurately modeling the temporal behavior of gas turbines enables precise planning of their production capacity, which is a crucial requirement for their effective use in a modern energy grid [Row83; Pas+24; GT16].

In this study, we focus on modeling the behavior of a commercial micro gas turbine designed for residential households, generating approximately 3 kW of electrical power. In our setting, we model the output power in response to an input control signal within the timeframe spanning seconds to hours (see Figure 4.1). Inputs and outputs are easily obtainable operational data, which eliminates the need for installing measurement sensors within the internal gas turbine components or conducting measurements in a laboratory. Each level of the input control signal corresponds to a stationary, i.e., constant, level of the output power. Notably, a noticeable delay occurs in the output signal during transitions in response to changes in the input control signal.

Conventional theory-based models are less suitable for our scenario, as they rely on access to measurements of internal gas turbine components for parameterization (see Section 4.2.1.7). On the other hand, rule-based and transfer-function methods are susceptible to measurement inaccuracies, or encounter difficulties in accurately representing the system's behavior during transition phases, as discussed in Section 4.2.1.7. Meanwhile, classical data-driven deep learning methods also encounter difficulties, as discussed next.

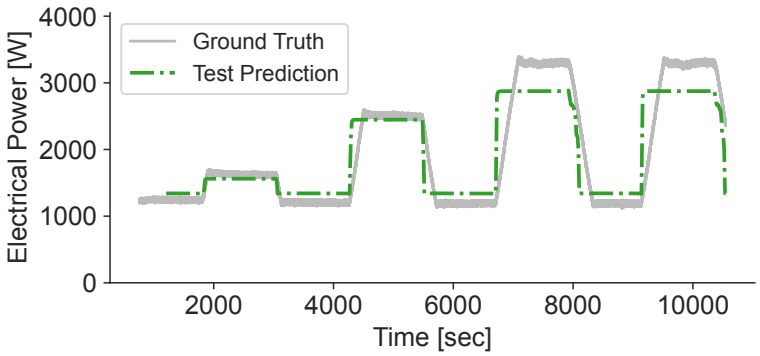


Figure 4.2.: Data-driven methods can struggle to model the transitions when training data is limited.

4.2.1.3. Issues with Classical Deep Learning

In real-world situations with limited training data, accurately modeling the temporal behavior of a gas turbine using deep learning methods proves challenging. As depicted in Figure 4.2, a recurrent neural network with LSTM cells, when trained on a small dataset (here: two time series), struggles to accurately replicate the system's behavior during both stationary and transition phases. During stationary phases, the model's predictions deviate from the actual data. The model's predictions during transition phases are inconsistent with domain knowledge, failing to represent transitions as smooth, gradual changes with a constant gradient. This inconsistency with the expected behavior of the gas turbine showcases the shortcomings of knowledge-uninformed deep learning models. In this chapter, we illustrate how to improve both the prediction accuracy and consistency on the transitions for deep learning trained on a small dataset, by applying principles from knowledge-guided machine learning.

4.2.1.4. Domain Knowledge: Permissible System States

The dynamical state of a system is a characteristic that changes over time. Some dynamical systems have a finite number of dynamical states that the system is allowed to take. We refer to this type of domain knowledge as a finite set of *permissible system states*. Domain expertise about the system can provide insights into whether such *permissible system states* exist, and how to represent and measure them. Depending on factors like measurement uncertainties or the stochastic nature of the system, domain knowledge in that form can be defined either exactly or approximately. The *multi-state constraint* that we introduce in Section 4.2.2 informs the machine learning model about these *permissible system states* by penalizing deviations from them, preventing the model from adopting a behavior inconsistent with the domain.

In temporal dynamics modeling, certain systems can be characterized by a finite set of permissible values for their output change rates. For example, in our gas turbine study,

the finite set of *permissible system states* consists of three distinct output change rates, as depicted in the lower part of Figure 4.1. We expect both the rising and falling transition phases to have a constant non-zero change rate, and stationary phases to have a change rate of zero. The values of the change rate for the transition phases can be obtained from ground-truth data by estimating the derivative of the output signal over time. This type of domain knowledge is not unique to gas turbines; similar knowledge is often available for other dynamical systems in mechanical, thermal, or electrical engineering, such as thermal engines, energy storage systems, or power-to-X systems.

4.2.1.5. Using Loss Functions in Knowledge-Guided Machine Learning

One of the most widely used paradigms to incorporate domain knowledge into neural networks is via regularization, where the loss function is augmented with additional terms based on domain knowledge [Kas+21; WY22] (refer also to Equation 1.1):

$$Loss = Loss_{ML} + \lambda_K \cdot Loss_K \tag{4.2}$$

$Loss_{ML}$ represents a conventional loss function, such as the mean squared error (MSE):

$$Loss_{ML} = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2 \tag{4.3}$$

Such a loss function is commonly employed in regression tasks, i.e., prediction tasks with continuous target variables. It does not consider domain knowledge.

$Loss_K$ has to be designed to guide the neural network towards solutions aligning with domain knowledge or general properties of the target function. The hyperparameter $\lambda_K \in \mathbb{R}$ determines its relative importance compared to $Loss_{ML}$. Typically, λ_K is determined through hyperparameter tuning. The loss term $Loss_K$ is often considered a 'soft' constraint, meaning that it does not have to be strictly met but instead imposes a penalty that increases as the constraint is violated more. We now present two examples of knowledge-guided loss constraints from the literature that encode general properties of the target function.

Example: Approximation constraint The approximation constraint, as introduced by [Abu92], specifies an expected prediction range between values $a \in \mathbb{R}$ and $b \in \mathbb{R}$. The loss term penalizes predictions outside this range based on their deviation from the range's limits:

$$Loss_{approx} = \frac{1}{T}\sum_{t=1}^{T}\begin{cases} 0 & , \ \hat{y}_t \in [a,b] \\ (\hat{y}_t - a)^2 & , \ \hat{y}_t < a \\ (\hat{y}_t - b)^2 & , \ \hat{y}_t > b \end{cases} \tag{4.4}$$

Example: Monotonicity constraint The monotonicity constraint, as introduced by [Abu92], defines a monotonic relationship between input and output variables. The loss term penalizes predictions that violate monotonicity, i.e., outputs $\hat{y}$ and $\hat{y}'$ with $\hat{y} > \hat{y}'$ for model inputs x and x' with $x < x'$. In our scenario, we are interested in the output signal over time, i.e., we would set $x := t - 1$ and $x' := t$ in the context of the monotonicity constraint. Thus, the loss term would be defined as:

$$Loss_{mono} = \frac{1}{T\text{-}1} \sum_{t=2}^{T} \begin{cases} 0 & , \hat{y}_{t-1} \leq \hat{y}_t \\ (\hat{y}_t - \hat{y}_{t-1})^2 & , \hat{y}_{t-1} > \hat{y}_t \end{cases} \tag{4.5}$$

4.2.1.6. Inapplicability of the existing constraint types to our scenario

Both discussed constraint types have limitations that make them inapplicable to our scenario in their current form. The approximation constraint does not allow to encode multiple states, as in our case, but only a single range of values. However, its capability to penalize values outside a defined range becomes useful when considering approximate state definitions of the *permissible system states*. On the other hand, the monotonicity constraint is not suitable for non-monotonous functions, as in our case. However, it can penalize inconsistencies between different points of the function, which becomes useful when we define the multi-state constraint on consecutive values.

4.2.1.7. Previous Work on Gas Turbine Modeling

Theory-based models are the conventional way to model the behavior of gas turbines, e.g., Rowen's Model [Row83], the IEEE Model [DA94], or the CIGRE Model [C4003]. These application-specific, equation-based models accurately capture the temporal behavior of gas turbines by modeling the interaction among various system components and operational conditions. This process depends on obtaining measurements from internal system components using physical sensors and involves significant parametrization efforts [SMB11; TVG09].

As an alternative of theory-based models, some researchers have turned to widely applicable, data-driven deep learning models to directly predict the output power of gas turbines from operational data. While this approach has shown promise in reducing parametrization and calibration efforts compared to theory-based models, previous studies evaluating these methods focused on scenarios where measurements of the internal system components were accessible [Sun+21; Pas+24]. However, this access may not always be guaranteed due to the associated high effort and costs, for example in the concept phases of planning decentralized energy systems [GT16].

Much less research has been done for modeling scenarios without access to internal system component measurements. [Kot23] explored the use of empirical rule-based and transfer-function models for gas turbine modeling using only easily obtainable input-control and output power data. However, the proposed rule-based ramp model is highly susceptible

to measurement uncertainties, and the transfer-function model struggles to optimize for both small and large output changes simultaneously, resulting in imprecise predictions on transitions. For this scenario, in Section 4.2.2, we propose an alternative data-driven knowledge-guided deep learning approach that, unlike the transfer-function model, models transitions accurately, and offers broader applicability and robustness against measurement uncertainties than rule-based methods. Further, our method integrates domain knowledge to address the limitations of knowledge-uninformed deep learning. This approach allows us to leverage the minimal manual effort associated with data-driven modeling while creating an accurate model with minimal data requirements.

4.2.2. Proposed Method

In this section, we present our approach. First, we propose general formulations of exact and approximate multi-state constraints (Section 4.2.2.1). Second, we show how to adapt these constraints to our scenario, i.e., gas turbine modeling (Section 4.2.2.2).

4.2.2.1. General Formulation

We define a model state $\tilde{s}$ as a value or vector describing any characteristic of a machine learning model. In the context of neural networks, the model state may be the value of a particular neuron, the network's output at a specific layer, or more generally, any combination of these values. We assume we can compute a real-valued distance $d(\cdot)$ between these model states, using a metric like the Euclidean distance. We denote the finite set of permissible model states as $S = \{s^1, s^2, \ldots, s^n\}$. These states represent the domain knowledge we would like the model to adhere to. To inform the neural network about S, we introduce a multi-state constraint as a term in the loss function. We propose two variants: an exact (Section 4.2.2.1) and an approximate (Section 4.2.2.1) version.

Exact multi-state constraint In the exact variant of the constraint (see Equation 4.6), each permissible state corresponds to a single value. The loss term penalizes model states $\tilde{s}_t$ that do not belong to the set of permissible model states S, imposing a penalty proportional to the deviation from the closest permissible state. If $\tilde{s}_t$ matches a permissible state exactly, then the penalty is zero. The ratio between the hyperparameters β^i of different permissible states allows to influence which of the permissible states is used for penalizing model states in the loss function. We provide more explanations of these hyperparameters in Section 4.3.6.

$$Loss_{MS} = \frac{1}{T} \sum_{t=1}^{T} \min_{i \in \{1,\ldots,n\}} \beta^i \cdot d(\tilde{s}_t, s^i) \tag{4.6}$$

Approximate multi-state constraint For scenarios where the model states S can only be defined approximately, we introduce an approximate multi-state constraint (see Equation 4.7). Drawing inspiration from the approximation constraint in literature (see Equation 4.4), we define an approximate state as a range of values, i.e., $s^i = [s^i_{min}, s^i_{max}]$. If a model state $\tilde{s}_t$ lies within the range of any permissible model state, then the penalty is zero. Otherwise, we apply a penalty proportional to the deviation from the nearest state range:

$$\begin{aligned} Loss_{app-MS} &= \frac{1}{T}\sum_{t=1}^{T} \min_{i\in\{1,\dots,n\}} \beta^i \cdot d_{range}(\tilde{s}_t, s^i) \\ \text{where } d_{range}(\tilde{s}_t, s^i) &= \begin{cases} 0 & , \ s^i_{min} \leq \tilde{s}_t \leq s^i_{max} \\ d(\tilde{s}_t, s^i_{min}) & , \ \tilde{s}_t < s^i_{min} \\ d(\tilde{s}_t, s^i_{max}) & , \ \tilde{s}_t > s^i_{max} \end{cases} \end{aligned} \tag{4.7}$$

Exact states are a special case of approximate states where the range is zero, i.e., the lower bound also is the upper bound.

4.2.2.2. Application to Gas Turbines

In general, it makes sense to define the model state $\tilde{s}$ in a form compatible with the form of the available domain knowledge. In our scenario, domain knowledge takes the form of a finite set of permissible values for the output change rates, as described in Section 4.2.1.4 and depicted in the lower part of Figure 4.1. To make the model state compatible with the notion of change rates, we define the model state $\tilde{s}$ as the difference between two consecutive predictions:

$$\tilde{s}_t = \Delta\hat{y}_t = \hat{y}_t - \hat{y}_{t-1} \tag{4.8}$$

Consequently, the set of *permissible model states* defines a set of permissible differences $S = \{\Delta^1, \Delta^2, \dots, \Delta^n\}$ that represent the set of change rates for two consecutive predictions of the model.

For our dataset, we define the set of permissible differences by estimating the derivative of the output signal over time $\Delta y/\Delta t$ for the transitions and assuming a difference of zero for the stationary phases. As a result, we have a set with three permissible differences: $\Delta^+ = 6.388\,\mathrm{W\,s^{-1}}$ for rising transitions, $\Delta^- = -6.388\,\mathrm{W\,s^{-1}}$ for falling transitions, and $\Delta^0 = 0\,\mathrm{W\,s^{-1}}$ for stationary phases (see the lower part of Figure 4.1). We encode these three permissible system states into the loss function with the *multi-state constraint* (see Equation 4.6), employing a squared distance to penalize deviations of the model's output from these states:

$$Loss_{MS} = \frac{1}{T}\sum_{t=2}^{T} \min\left(\beta^+(\Delta\hat{y}_t - \Delta^+)^2, \beta^-(\Delta\hat{y}_t - \Delta^-)^2, \beta^0(\Delta\hat{y}_t - \Delta^0)^2\right) \tag{4.9}$$

This equation can be simplified for our data with $\Delta^+ = -(\Delta^-) = 6.388$, $\Delta^0 = 0$, $\beta^+ = \beta^- = \beta^{trans}$, and $\beta^0 = \beta^{stat}$:

$$Loss_{MS} = \frac{1}{T}\sum_{t=2}^{T} \min\left(\beta^{trans}(|\Delta\hat{y}_t| - 6.388)^2, \beta^{stat}(\Delta\hat{y}_t)^2\right) \tag{4.10}$$

The previous two equations assume exact state definitions; in our experiments, we also evaluate the approximate variant of the constraint type, defining intervals around the exact values. In our scenario, these intervals are symmetric and expressed as a percentage of the distance between the values representing the *permissible system states*. For example, a 25% approximation range means the interval extends 25% of the distance in both directions from the exact value representing the permissible state. A 50% range extends halfway between states (in both directions), causing the lower range of one state to overlap with the upper range of the next state.

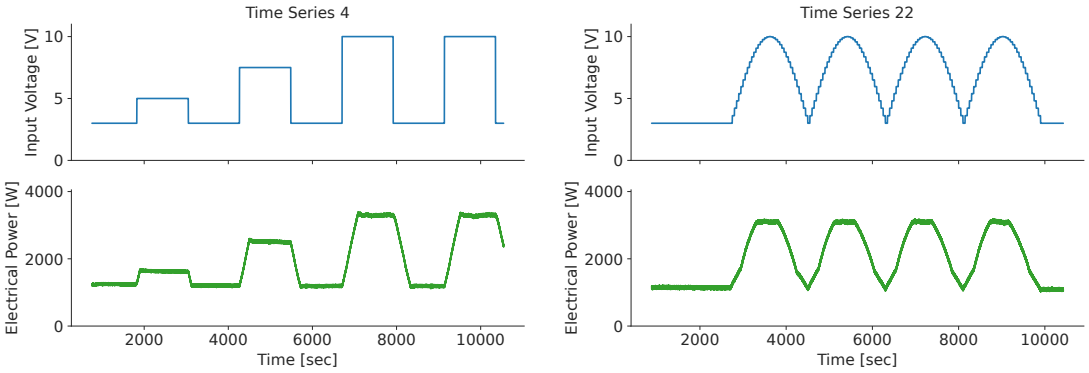
4.3. Experiments

In this section, we assess the effectiveness of our method in modeling the temporal dynamics of a gas turbine. First, we describe our general experimental design used for all the experiments (Section 4.3.1). Subsequently, we demonstrate that incorporating the multi-state constraint (both its exact and approximate formulations) consistently diminishes the prediction error across varying training set sizes (Section 4.3.2). Then, we evaluate the relationship between approximate domain knowledge and prediction error (Section 4.3.3). Next, we evaluate the method's resilience to incorrect domain knowledge (Section 4.3.4). We also showcase that our method effectively models transitions in line with domain knowledge, aiding the model in learning patterns that generalize well to test data with different characteristics (Section 4.3.5). Finally, we conduct a sensitivity analysis of the hyperparameters of our method (Section 4.3.6).

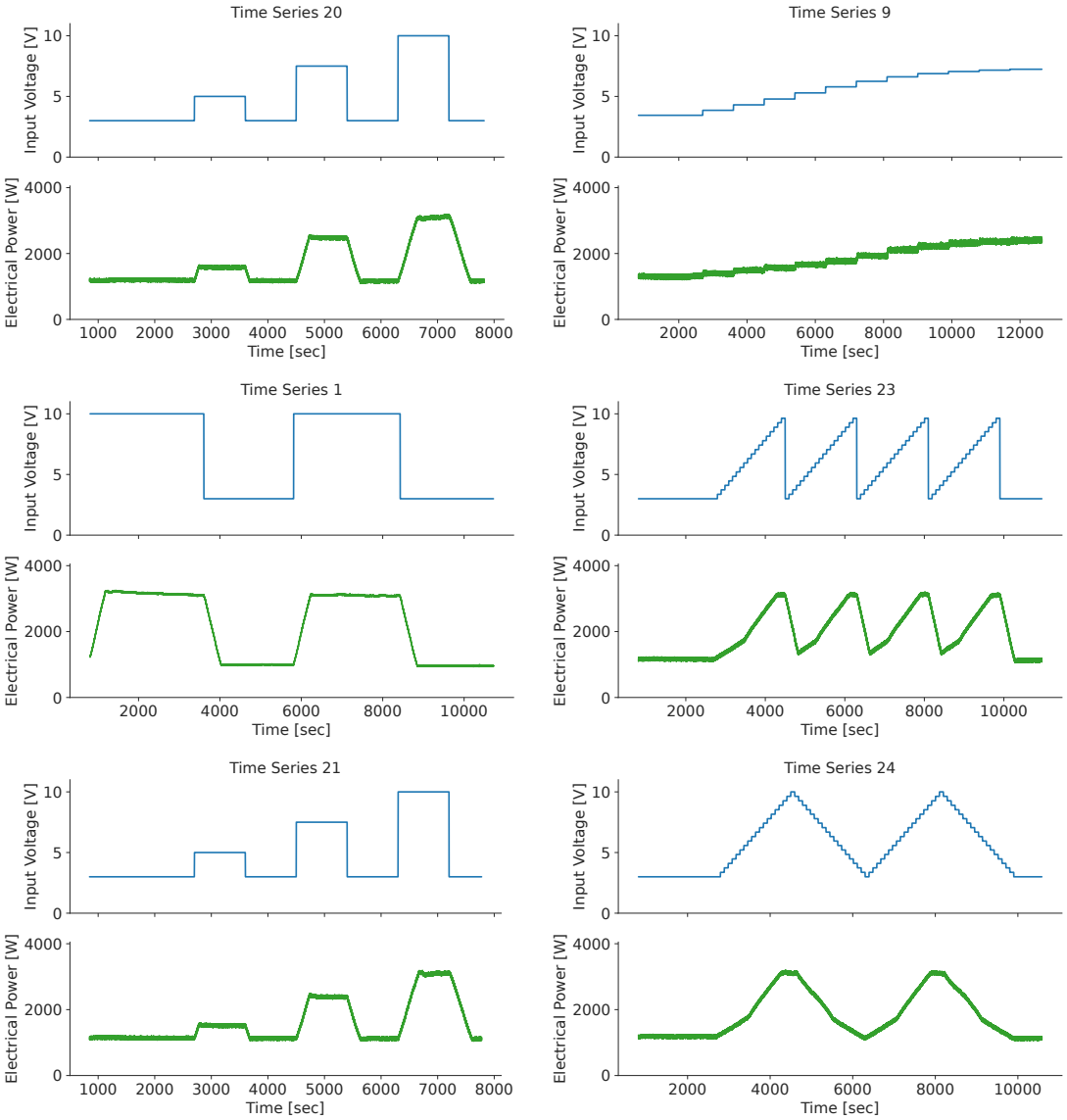
4.3.1. Experimental Design

4.3.1.1. Data

We collected our dataset from a small commercial gas turbine, described in detail in [Kot23]. The dataset comprises time-series data for both the input control signal and the corresponding output (electrical power). Collecting data from gas turbines is a challenging and time-consuming process, requiring the installation of measurement devices and direct supervision by a domain expert for extended periods during experiments. Moreover, real-life operational restrictions on the gas turbine limit the execution of certain test profiles. Our dataset comprises eight time series that depict the gas turbine's behavior under diverse conditions (see Figure 4.3). The time series vary in duration from 6,495 to 11,820 data points with a resolution of approximately 1 second, corresponding to approximately 1.8 h to 3.3 h. Four *rectangular* time series represent scenarios in which the input control signal changes instantaneously but the output power follows with a visible delay. Accurate modeling



(a) Time-series data used for testing.



(b) Time-series data used for training.

Figure 4.3.: All available time series in our dataset.

of both transitions and stationary phases is crucial for precise gas turbine modeling in these scenarios. The four remaining *continuous* time series represent scenarios in which the input control signal changes gradually and there are no visible delays in the output power. For these time series, learning transitions is less important for modeling the overall behavior.

4.3.1.2. Prediction model

As our baseline method without domain knowledge, we employ a recurrent neural network (RNN) with three hidden layers, each containing 32 LSTM units. The input size is $N = 451$, corresponding to the longest transition length in the dataset. In preliminary experiments, we determined that this architecture has sufficient capacity to learn transitions when trained with six time series, which represents the maximum training set size in our experiments. Architectures with fewer layers were incapable of modeling transitions, while architectures with more layers did not improve performance but required more training time.

Based on our sensitivity analysis (see Section 4.3.6), we set the hyperparameter λ_K to 1000 and the weighting of states between transitions and stationary phases to a ratio of $\beta^{stat}/\beta^{trans} = 0.6$ in the other experiments. We train our models using the loss function defined in Equation 4.2. For $Loss_{ML}$, we employ the mean squared error (MSE) loss (see Equation 4.3). For $Loss_K$, we utilize our proposed $Loss_{MS}$ (Equation 4.6) and $Loss_{app-MS}$ (Equation 4.7). We train the models with a learning rate of 0.001 for 300 epochs, using early stopping with a patience of 90. To evaluate prediction error in our experiments, we use the root mean squared error (RMSE).

We implemented the proposed method using *TensorFlow* [Aba+16]. On our server with an NVIDIA GeForce GTX 1080 Ti (11 GB Memory), the training took between 11 min (training size of 1) and 90 min (training size of 6), and the inference on one time series took 4 s.

4.3.2. Impact of Training Set Size

4.3.2.1. Experimental design

In this section, we the assess prediction error in terms of RMSE across varying training set sizes. We select two time series (#4, which is rectangular, and #22, which is continuous; see Figure 4.3) as the test set. From the remaining six time series, we randomly choose one to six time series for training and repeat each training five times to account for randomness in model initialization. We compare the baseline RNN with three knowledge-guided variants: (1) RNN^{MS} uses an exact multi-state constraint, representing a scenario with accurate domain knowledge. (2) $RNN^{app-MS,1\%}$ uses an approximate multi-state constraint and an error tolerance of 1%, representing a situation with sufficiently accurate but not perfect domain knowledge. (3) $RNN^{app-MS,10\%}$ uses an approximate multi-state

constraint and an error tolerance of 10%, representing a scenario with higher uncertainty, which also is realistic in practical applications. In our scenario, the value ranges for the approximate constraint are symmetric around the expected value and are defined relative to the difference between the states, which is equal to $\Delta^+ - \Delta^0 = \Delta^0 - \Delta^- = 6.388\,\mathrm{W\,s^{-1}}$ For instance, in transition phases of $RNN^{app-MS,10\%}$, with an expected value of $6.388\,\mathrm{W\,s^{-1}}$ and a 10% tolerance, the approximate state is $6.388 \pm 0.6388 = [5.7492, 7.0268]$. For stationary phases, with an expected value of 0 and a 10% tolerance, the approximate state is $0 \pm 0.6388 = [-0.6388, 0.6388]$. The tolerance for $RNN^{app-MS,1\%}$ is calculated similarly. In our experiments, we assume the same tolerance applies to all states. However, in practice, each state could be assigned a different tolerance based on specific requirements.

4.3.2.2. Results

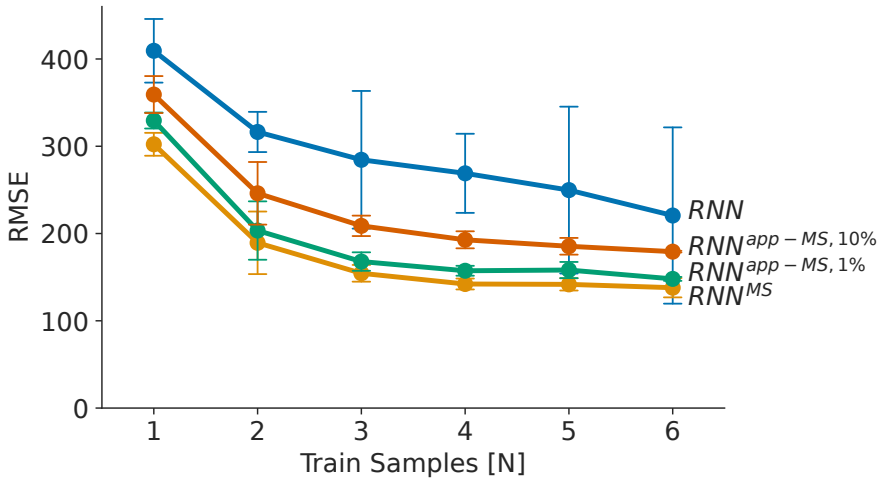


Figure 4.4.: Test prediction error (mean RMSE ± standard deviation) over training set size.

As depicted in Figure 4.4, the test prediction error consistently improves with increasing training set sizes for all three variants of multi-state constraints. In particular, RNN^{MS} and $RNN^{app-MS,1\%}$ reduce the test prediction error by 30-40%, while $RNN^{app-MS,10\%}$ reduces it by 15-20%, compared with RNN, the baseline without domain knowledge. Further, the multi-state constraint methods cause the model to reach a plateau in error with only three to four training samples, whereas the baseline does not plateau even until six samples. Additionally, the multi-state constraint methods RNN^{MS} and $RNN^{app-MS,1\%}$ achieve better test prediction error when trained on two samples than the baseline trained on six samples. This result suggests that integrating domain knowledge effectively reduces the model's dependence on extensive training data and enables reaching the optimum error with less data. Additionally, all multi-state constraint methods show lower variance between different evaluation runs than the baseline, indicating that domain knowledge may help mitigate training randomness. As the amount of training data increases, the performance gap between the baseline and the knowledge-guided approach diminishes, emphasizing the particular benefits of knowledge-guided approaches when data is limited. Furthermore,

while both approximate multi-state constraints perform worse than the exact constraint, they still outperform the baseline. These findings confirm that even an approximate definition of domain knowledge can be useful.

4.3.3. Impact of the Approximation Range

4.3.3.1. Experimental design

In this section, we assess the impact of the tolerance range of the approximate multi-state constraint on the prediction error. In particular, we vary the width of the range. As in Section 4.3.2, we use ranges that are symmetric around the expected value and define the tolerance relative to the distance between the states, i.e., 6.388 W s^{-1}. We use the same two time series as in Section 4.3.2 as test set and perform five training repetitions, each time using a random selection of four out of the remaining six time series for each value of the domain knowledge. We evaluate tolerance values from 0% to 50% with 5% increments to explore the method's limits. The tolerance of 0% represents the exact state definition.

4.3.3.2. Results

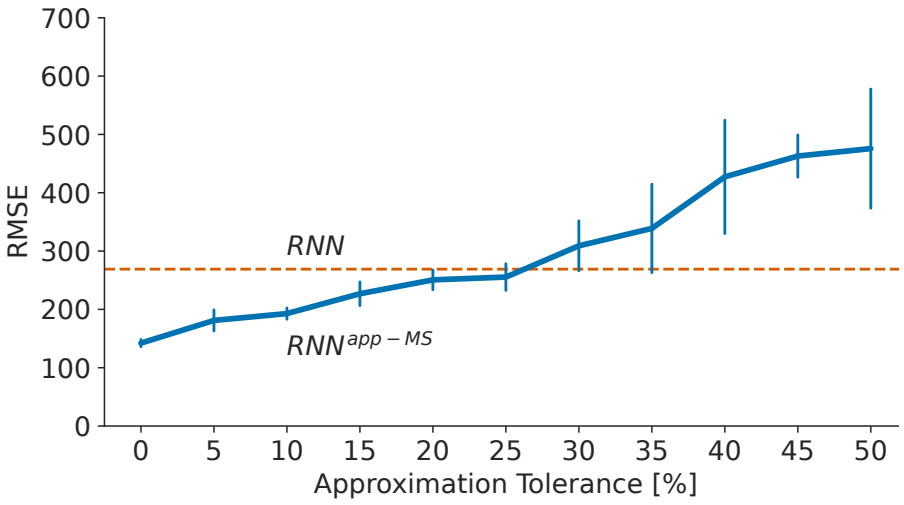


Figure 4.5.: Test prediction error (mean RMSE ± standard deviation) over approximation tolerance of domain knowledge, trained using four training samples.

As illustrated in Figure 4.5, the approximate multi-state constraint outperforms the baseline without domain knowledge (depicted as a striped line) in terms of test prediction error even with an approximation range as large as 25%. However, for a tolerance of 50%, where the lower range of one state overlaps with the upper range state of another state, the method performs significantly worse than the baseline. This indicates that the approximation tolerance between states cannot be arbitrarily large, as it can negatively impact prediction

quality, even when the tolerance ranges do not overlap. Additionally, we observe that models trained with domain knowledge with smaller tolerance have lower variance in prediction error between different evaluation runs than when using domain knowledge with larger tolerance.

4.3.4. Impact of Incorrect Domain Knowledge

4.3.4.1. Experimental design

In this section, we assess the robustness of the exact multi-state constraint when faced with incorrect domain knowledge about the permissible states. This scenario simulates a situation where the provided domain knowledge is exact but does not match the reality. We use the exact multi-state constraint and vary the values for the domain knowledge, i.e., the change rates of the gas turbine's output signal at the transitions. Again, we use the same two time series as in Section 4.3.2 as test set and perform five training repetitions, each time using a random selection of four out of the remaining six time series for each value of the domain knowledge. We test the correct value of $\Delta^+ = 6.388\,\mathrm{W\,s^{-1}}$, as well as exponentially scaled values up to 128 times smaller or larger.

4.3.4.2. Results

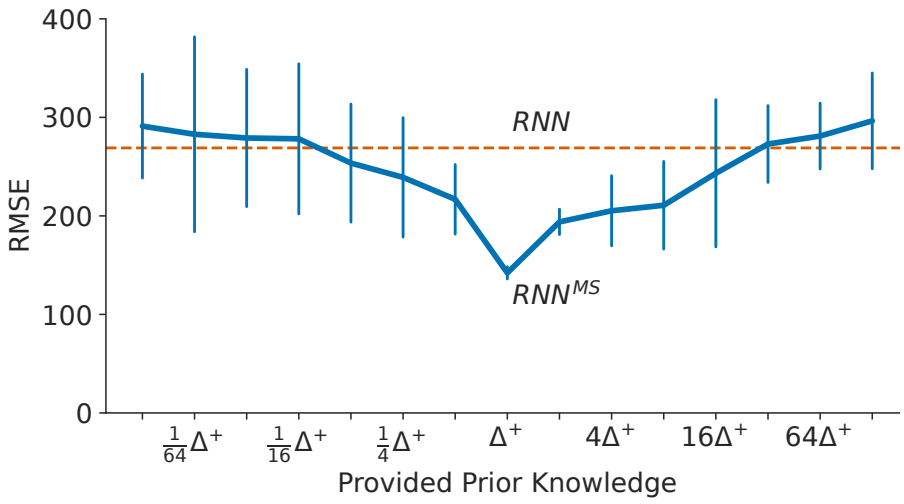


Figure 4.6.: Test prediction error (mean RMSE ± standard deviation) over incorrect domain knowledge (correct domain knowledge in center), trained using four training samples.

As depicted in Figure 4.6, the exact multi-state constraint outperforms the baseline without domain knowledge (depicted as a striped line) regarding test prediction error even if the assumed change rate is eight times smaller or larger than in reality. However, when the deviation from the correct value becomes larger, the method performs worse than the

baseline. These results suggest that even incorrect domain knowledge can be beneficial within a certain range. Additionally, we observe that the model trained with the correct domain knowledge has lower variance between different evaluation runs than with incorrect domain knowledge. Finally, the U-shaped relationship suggests that it could be technically feasible to identify the optimal value of domain knowledge by sampling different values and observing the prediction error. This could, in turn, reduce the method's reliance on providing the exact value of domain knowledge beforehand.

4.3.5. Qualitative Evaluation of Predictions

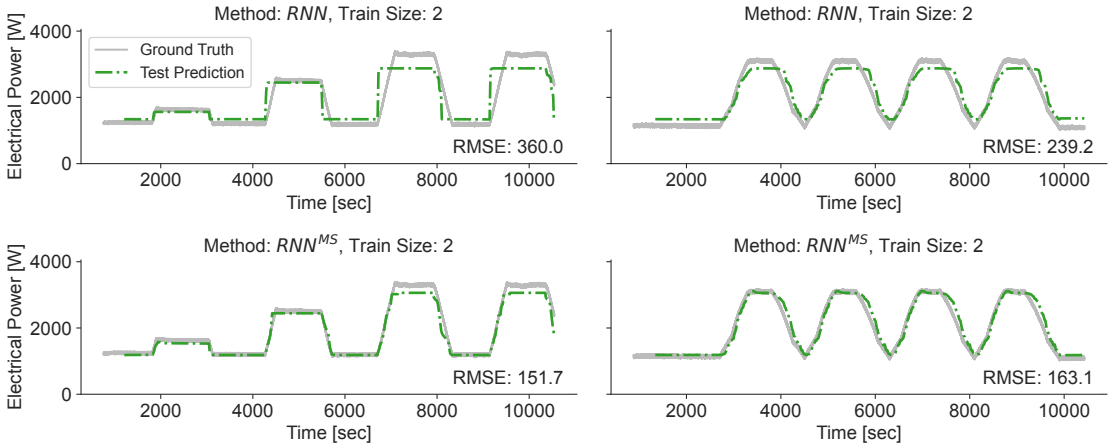
4.3.5.1. Experimental design

In this section, we showcase the effectiveness of our proposed method qualitatively. We demonstrate that our approach not only reduces the prediction error but also addresses the challenges of accurately representing stationary phases and providing consistent predictions for transition phases, as discussed in Section 4.2.1.3 and depicted in Figure 4.2. In particular, we visualize the test predictions of the exact multi-state RNN (RNN^{MS}) and the baseline (RNN) with the lowest prediction error out of the five training repetitions from the experiments in Section 4.3.2. To show how the impact of our method depends on the training set size, we visualize predictions for training sizes of two and six. Figure 4.3 plots the corresponding input signals.

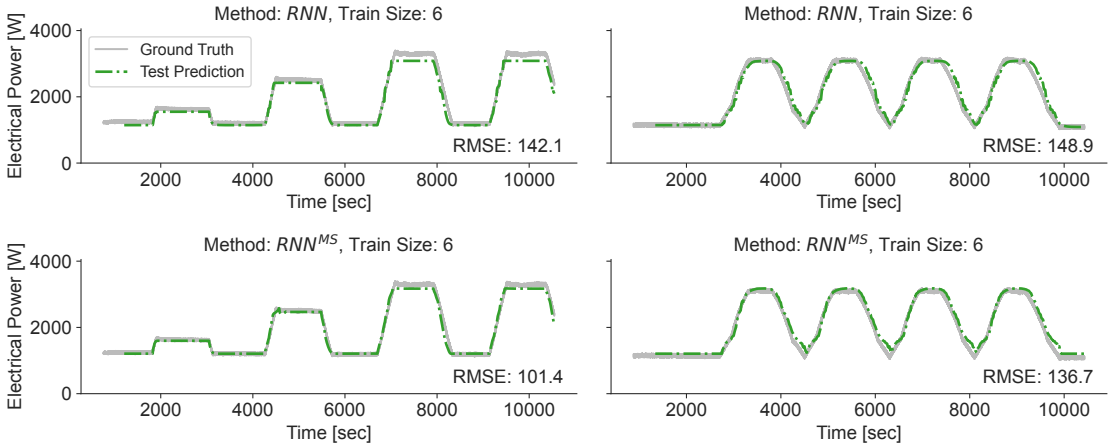
4.3.5.2. Results

The first row of Figure 4.7a shows that the baseline model RNN, trained on data from two time series, fails to accurately model the rising transition, representing it as a discrete step rather than a gradual change. We also observe that RNN does not capture the stationary levels accurately. The second row of Figure 4.7a shows results from training with a multi-state constraint, resulting in a significant RMSE reduction for both rectangular and continuous time series. Further, the representation of transitions is smoother, and the overall behavior of the gas turbine is captured better. In the first row of Figure 4.7b, we see that while the baseline trained on data from six time series is able to capture transitions of the rectangular test-set time series as a gradual change, it does not capture the stationary levels accurately, which results in a worse RMSE than RNN^{MS} has. For the continuous test-set time series, RNN^{MS} also is slightly better at following the ground truth data.

To sum up, our method proves advantageous since it improves the modeling of transitions as well as stationary phases, especially when the training set size is small. In practical applications, we recommend training the model multiple times and choosing the best one to mitigate the effects of randomness in network initialization.



(a) Trained on two time series from our dataset.



(b) Trained on six time series from our dataset.

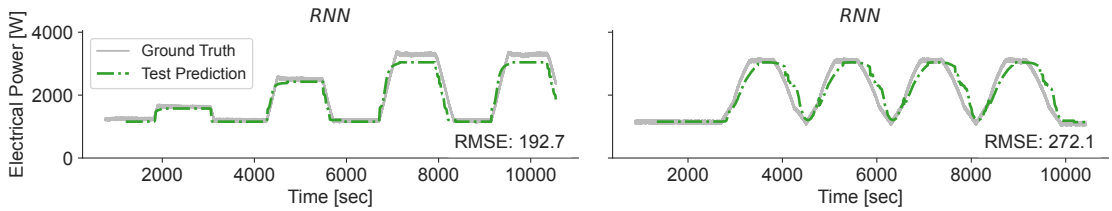
Figure 4.7.: The predictions of the best of five models on the test data.

4.3.6. Sensitivity Analysis

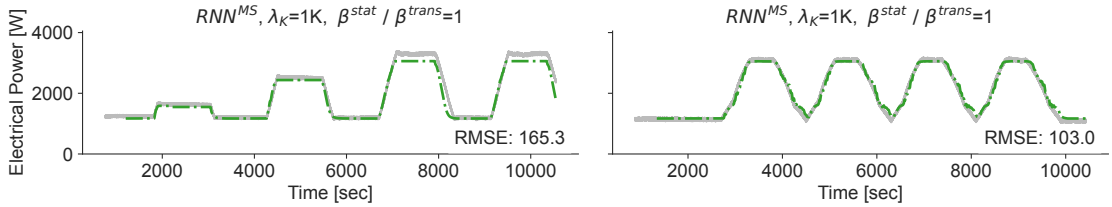
4.3.6.1. Experimental design

In this section, we investigate the two hyperparameters of our method: λ_K and the ratio $\beta^{stat}/\beta^{trans}$. For both hyperparameters, we evaluate an exponentially scaled range of values, i.e., $\{0.001, 0.01, \ldots, 10000\}$. As discussed in Section 4.2.1.5, λ_K determines the relative importance of the knowledge-guided loss $Loss_K$ in relation to $Loss_{ML}$. The ratio $\beta^{stat}/\beta^{trans}$ between the hyperparameter β^i of different permissible states allows to determine which of the permissible states is used to compute the penalty in the loss function, by influencing the decision border between different states (see Section 4.2.2.1). A ratio of 1 positions the decision boundary precisely halfway between the values for transitions ($\Delta^+ = -\Delta^- = 6.388\,\mathrm{W\,s^{-1}}$) and stationary phases ($\Delta^0 = 0\,\mathrm{W\,s^{-1}}$). Ratios less than one shift the border toward the transition value and ratios greater than one shift it toward the stationary value. The relationship between this ratio, the decision border, and the method's performance

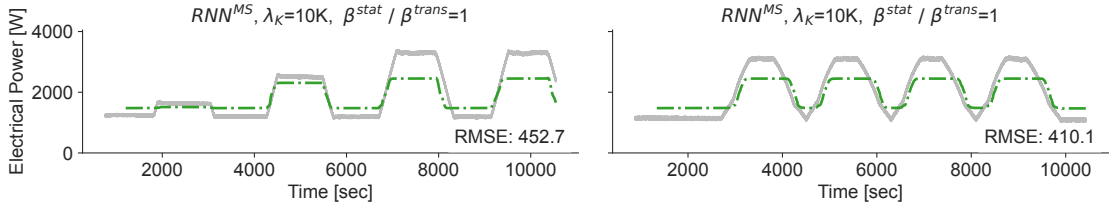
can be complex, underscoring the importance of hyperparameter tuning to find optimal values. In our analysis, we use the same two time series as in Section 4.3.2 as test set and perform five training repetitions, each time using a random selection of four out of the remaining six time series for each hyperparameter value under examination.



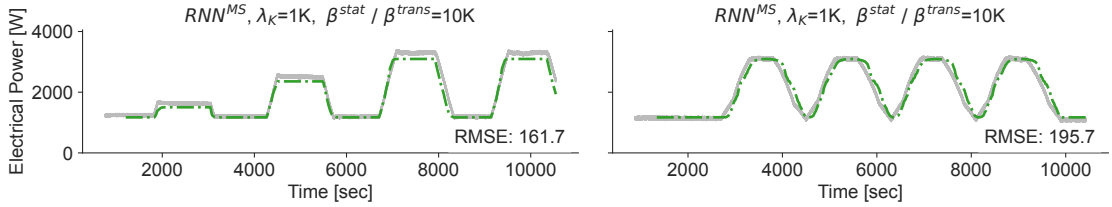
(a) Baseline model.



(b) Exact multi-state constraint with optimal hyperparameters.



(c) Exact multi-state constraint with λ_K too big.



(d) Exact multi-state constraint with β^{stat} / β^{trans} too big.

Figure 4.8.: The predictions of the best of five models trained on four time series.

4.3.6.2. Results: λ_K

As shown in Figure 4.9, the model's prediction error with the multi-state constraint approaches the baseline's error for small values of the parameter λ_K. This behavior is expected, as for small λ_K values, the knowledge-guided loss $Loss_K$ has a small impact on model training. As λ_K approaches 1000, the prediction error decreases, reaching a minimum. However, for $\lambda_K = 10000$, the error increases significantly. This behavior aligns with expectations, as excessively high values of λ_K cause the knowledge-guided loss $Loss_K$

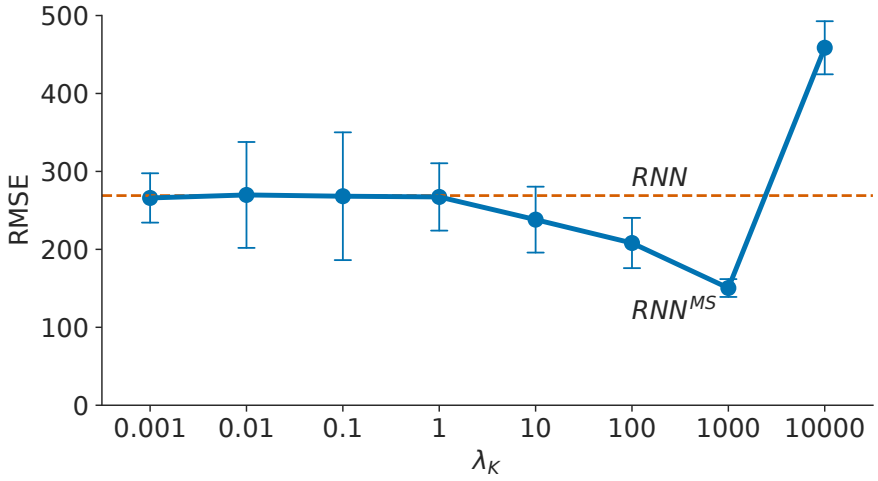


Figure 4.9.: Test prediction error over hyperparameter λ_K for $\beta^{stat}/\beta^{trans} = 1$.

to dominate over the machine learning loss $Loss_{ML}$, leading the model to disregard learning the correct stationary phases (see Figure 4.8c).

4.3.6.3. Results: $\beta^{stat}/\beta^{trans}$

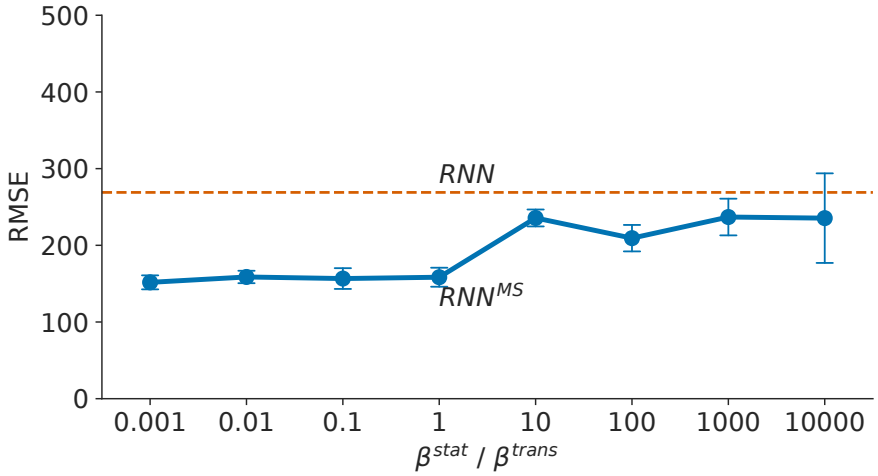


Figure 4.10.: Test prediction error over hyperparameter $\beta^{stat}/\beta^{trans}$ for $\lambda_K = 1000$.

In Figure 4.10, we observe that when the ratio $\beta^{stat}/\beta^{trans}$ is below or equal to 1, the prediction error remains consistently low, with the exact value of $\beta^{stat}/\beta^{trans}$ having minimal impact. However, for values greater than 1, the prediction error increases significantly. The results suggest that in our scenario, optimal results are achieved when the decision border is positioned closer to the value for transitions ($\Delta^+ = -\Delta^- = 6.388\,\mathrm{W\,s^{-1}}$) than the stationary phase ($\Delta^0 = 0\,\mathrm{W\,s^{-1}}$). Placing the decision border too close to the state for the stationary phase may force predictions with output change rates that are slightly above zero to converge to the output change rate for transitions, which negatively affects the

predictions for the continuous time series in particular (see Figure 4.8d). In our scenario, the typical inconsistency in model prediction arises from predicted transitions being too steep (see Figure 4.2). As a result, placing the decision border too close to the state for the transition phase does not have such a negative impact because predictions converge to the closest permissible state, which is the transition state, regardless of the positioning of the decision border.

4.4. Summary

In this chapter, we addressed the challenge of low-quality domain knowledge resulting from its imperfect representation. We tackled this issue by demonstrating how to design and evaluate a new KGML approach for effectively handling approximate and incorrect domain knowledge. First, we introduced a novel type of domain knowledge relevant to modeling dynamical systems: *permissible system states*. We then proposed a new multi-state constraint to incorporate that domain knowledge into neural networks via a knowledge-guided loss function. We proposed two variants of this constraint: one for exact domain knowledge and one for approximate domain knowledge. We evaluated both types of imperfect domain knowledge and found that even approximately defined domain knowledge can improve machine learning prediction performance. Additionally, we demonstrated how to assess the robustness of the KGML method against incorrect domain knowledge by performing a sensitivity analysis on the knowledge itself.

4.5. Author's Contribution

The idea of applying KGML to model an energy system originated from me. Prof. Giovanni De Carne suggested using the gas turbine as a case study and recommended Dustin Kottonau as the domain knowledge expert. The initial implementation of the KGML method, including identifying and formalizing the available domain knowledge, was carried out by Florian Leiser during his Master's thesis [Lei21], under my supervision. The concept of exploring the aspects of imperfect domain knowledge was mine. The final implementation of both types of the knowledge-guided loss function (exact and approximate) was completed by Aleksandr Eismont under my supervision. The experimental design was conceptualized by me and discussed with Aleksandr Eismont, who then implemented the final experiments. I conducted the evaluation of the experiments including the visualization. All sections of the publication were written by me, with support and feedback from Jakob Bach and Prof. Klemens Böhm.

5. Addressing the Domain-Application Knowledge Mismatch

The content of this chapter is based on the following publication:

1. Pawel Bielski, Lena Witterauf, Sönke Jendral, Ralf Mikut, and Jakob Bach. "Quantifying Domain-Application Knowledge Mismatch in Ontology-Guided Machine Learning". In: *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. Ed.: D. Aveiro. Vol. 2.* 2024, p. 216

Keywords: Ontology Quality Evaluation, Application Ontology, Domain Ontology, Knowledge-Guided Machine Learning

5.1. Chapter Overview

As described in Section 1.2.3, low-quality domain knowledge can arise when knowledge, although correct, has a structure that is suboptimal for the specific KGML task, especially if it was originally developed for a different purpose. This creates a mismatch between domain knowledge and application-specific knowledge, which may negatively affect certain subsets of data. The two issues that complicate mismatch resolution are: limited awareness and understanding of such mismatches in the KGML research community and lack of methods to quantify and address these mismatches. Currently, there is a lack of systematic research to understand how these mismatches manifest generally in KGML and how they impact predictions, and how to address them.

To address these challenges, we systematically study the important but underexplored issue of knowledge mismatch between domain-specific and application-specific knowledge in Ontology-guided Machine Learning (OGML), a subfield of KGML, that leverages ontologies as a source of **World Knowledge** and incorporates them into machine learning models. OGML aims to improve prediction performance, especially for rarely represented data objects, to reduce training data requirements, and to generate more interpretable results. OGML methods have shown significant success in fields like computer vision (image classification, segmentation, and retrieval) and medical data processing (including text classification and patient health prediction [Cho+17; Ma+18; Yin+19]), where rich ontological background knowledge is abundant [MW12].

Because the existing theory-based and empirical methods to evaluate ontology quality [HS14; Wil+22] are inadequate for detecting this mismatch, we propose a new OGML-aware evaluation framework based on the task-based framework [PM04]. Because the original framework was not designed for OGML, we adapt it to account for OGML-specific aspects, such as separation of the ontology and data, different task and ground truth definitions, and the stochastic nature of machine-learning algorithms. We argue that domain-application knowledge mismatch manifests as *harmful* domain knowledge, negatively impacting the prediction performance of OGML methods. Our framework identifies such harmful parts of the ontology for a specific task by comparing the performance of the OGML method with an ontology-uninformed method.

To demonstrate the effectiveness of our framework, we apply it to two common OGML application areas: image classification and patient health prediction. For image classification, we quantify the mismatch across three biological image datasets, using the Hierarchical Semantic Embedding OGML approach by [Che+18]. For patient health prediction, we quantify the mismatch across three prediction tasks within one medical dataset, using the GRAM [Cho+17] OGML approach. Our findings reveal that such mismatches are widespread across various OGML approaches, machine-learning model architectures, datasets, and prediction tasks. We also explore the potential root causes of these mismatches based on the harmful parts of the ontology identified by our framework. Furthermore, we discuss strategies to address these issues, demonstrating that our methodology shows promise as a generalizable approach for ontology quality assessment, enabling the identification of various ontological issues.

Contributions To summarize, our contributions are as follows:

- We study the important but relatively overlooked problem of domain-application knowledge mismatch in ontology-guided machine learning (OGML).
- We introduce a quality evaluation framework to quantify this mismatch and identify ontology parts that negatively impact the task performance.
- We apply our framework in two common OGML application areas to demonstrate how to detect, interpret, and address such mismatches.
- We provide the code and experimental results[1].

[1] `https://doi.org/10.35097/zv8zqgqd6ezm02vk`

5.2. Methods

Section 5.2.1 outlines our adaptation of the original task-based evaluation framework to OGML. Next, Section 5.2.2 introduces the concept of domain-application mismatch and explains how to quantify it.

5.2.1. Proposed Method

The original task-based evaluation framework for ontologies, proposed by [PM04] (see Section 2.2.1), assessed quality within ontology-informed applications by comparing task results against human-generated gold standards. While effective in its context, this framework requires significant adaptation to OGML.

Separation of Ontology and Data: In the original framework, the task is performed directly on the ontology since data and ontology are the same. In contrast, OGML distinguishes between ontology and data. The ontology is used to improve prediction performance of a machine-learning task on the data.

Task Definition and Ground Truth Data: In the original framework, tasks were specifically designed to identify ontology issues, with the ground truth defined by humans, leading to potential subjectivity errors. In OGML, however, the machine-learning process defines the task, and the ground truth is derived directly from the data. This ensures that the evaluation is more objective and less prone to errors. However, it also necessitates linking the performance of the OGML task to specific parts of the ontology, which can be achieved by using refinement metrics, as described in Section 5.2.2.

Stochastic Nature of ML Algorithms: OGML introduces stochastic elements inherent in machine learning, including retraining machine-learning models multiple times with different seed values, varying train-test splits, or varying model sizes. These factors must be considered to ensure the objectivity of results.

5.2.2. Quantifying Domain-Application Mismatch

In OGML, a *domain ontology* represents a broad field of knowledge and an *application ontology* is tailored to a specific task. A mismatch between domain knowledge and application knowledge occurs when the provided ontological knowledge for a domain is not optimally structured for the specific machine-learning task at hand. This mismatch can exist even if the domain knowledge is free of mistakes and thus has high quality from the domain perspective.

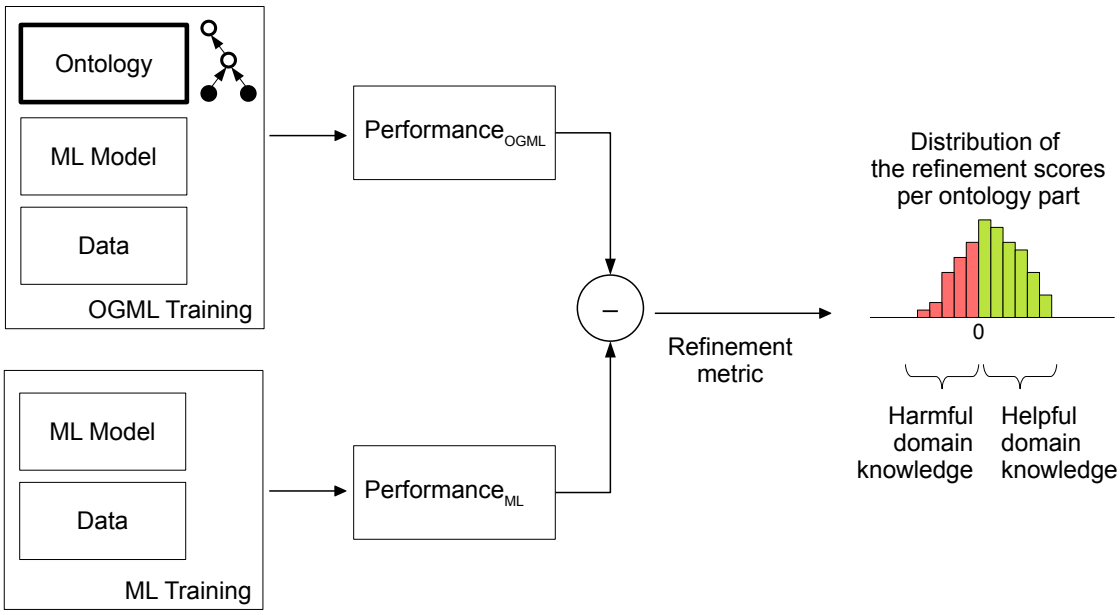


Figure 5.1.: Proposed framework to quantify the domain-application mismatch. Both OGML Training and ML Training use the same data and underlying machine learning architecture.

Because most OGML approaches leverage domain ontologies instead of application ontologies, it is often impossible to quantify the mismatch between domain and application knowledge directly by comparing OGML models with both types of ontologies. We argue that such a mismatch manifests itself through the existence of harmful parts of the ontological domain knowledge, which may exist independent from the fitness of the entire ontology. That is why we propose to approximate domain-application mismatch by measuring *harmful* domain knowledge.

Definition 1. *A particular part of domain knowledge is **harmful** (**helpful**) for a particular supervised machine-learning task if it negatively (positively) affects the prediction performance compared to a knowledge-uninformed baseline. The machine-learning task comprises the datasets for training and testing, prediction target, prediction model, and evaluation metric.*

Measuring harmful and helpful domain knowledge requires measuring the fitness of specific ontology parts, which can be achieved with our proposed framework (Figure 5.1). The core idea is to compare the performance of an ontology-guided machine learning model with that of a knowledge-uninformed model, both using the same data and underlying machine learning architecture. An ontology part is the subset of nodes and edges of the ontology that is semantically connected with a specific domain concept (e.g., unique class label) from the dataset. A *refinement metric*, which is a task-specific heuristic, links these ontology parts to application performance. We demonstrate examples of such refinement metrics in Section 5.3. In general, a refinement metric assigns a score to each part of the

domain knowledge: 0 indicates no impact on the prediction performance, a score greater than 0 indicates a positive impact, and a score less than 0 indicates a negative impact. The distribution of the refinement scores can be plotted, as shown in Figure 5.1. The parts of domain knowledge with scores below zero are harmful.

As the representation of domain knowledge may be of considerable size, e.g., for an ontology with many nodes and edges, only some parts may be harmful or helpful. Furthermore, the above definition is closely tied to *one* particular machine-learning task. In particular, some knowledge may be harmful for one task but not another. We propose to leverage our framework to quantify mismatch as follows:

Definition 2. *The **mismatch** is the ratio between the amount of harmful domain knowledge (Definition 1) and the entire amount of domain knowledge.*

For example, if the OGML approach outperforms the ontology-uninformed baseline, but 25% of relevant domain concepts perform worse than the baseline, we consider there to be a 25% mismatch. Note that different domain concepts may occur with different frequencies in the data. For example, if 25% of the domain concepts are harmful, 10% of the data for the machine-learning task may be affected if the affected concepts are relatively infrequent or 50% of the data if they are relatively frequent.

5.3. Experiments

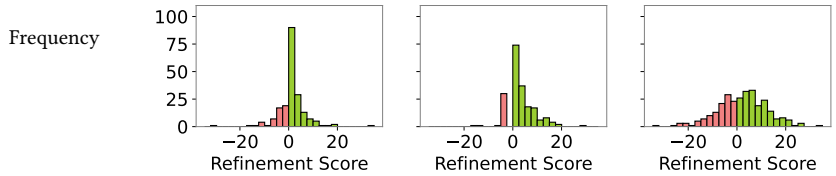
In this section, we demonstrate how to quantify a domain-application mismatch in two common OGML application areas: image classification (Section 5.3.1) and patient health prediction (Section 5.3.2). For each area, we explain how to apply our proposed framework to quantify the mismatch and present the results. Additionally, for patient health prediction, we use our framework to identify and qualitatively describe ontological issues arising from mismatches.

5.3.1. Ontology-Guided Image Classification

Scenario In the first use case, we demonstrate how to quantify domain-application mismatch in computer vision. We apply our framework to the OGML approach for image classification proposed by [Che+18]. This approach incorporates structured information about parent-child relationships between image categories and subcategories (i.e., a label taxonomy) into a deep learning model. The OGML model employs a Hierarchical Semantic Embedding framework to maintain consistency in classification across different taxonomy levels. During training, the model's predictions for higher-level categories help shape and improve predictions for lower-level categories, ensuring that the overall classification process is consistent across different levels of the taxonomy.

Table 5.1.: Comparison of the three datasets for image classification (negative refinement scores in red).

Dataset	Butterflies	Birds	VegFru
Acc. Baseline [%]	84.78	85.23	86.31
Acc. OGML [%]	85.82	88.09	88.77
Improvement [pp]	1.04	2.86	2.46
Mismatch [%]	25.50	16.50	40.21
– Data affected [%]	22.00	17.00	40.67



Experimental Setup We employ the same setup as the original paper by [Che+18]. Specifically, we use the three pretrained machine-learning models made publicly available by the authors and apply them to the corresponding hierarchical image datasets: *Butterflies*, *Birds*, and *VegFru* (Vegetables and Fruits). These datasets contain 200 unique classes for Butterflies and Birds, and 292 classes for VegFru, each organized into a taxonomy with four levels for Butterflies and Birds, and two levels for VegFru.

Refinement Metric To quantify the domain-application mismatch of ontological domain knowledge, we define the refinement metric as the per-class performance improvements, while the original paper assesses overall performance improvements. Our approach allows for a more detailed analysis of how the ontology impacts the model's performance, offering insights into which specific classes benefit from the ontological knowledge and which do not. We quantify prediction performance with top-1 accuracy, as in the original paper, on the test set. We calculate the mismatch as the percentage of classes that show a decrease in prediction performance compared to the baseline. Additionally, since classes may vary in the number of examples, we also report the proportion of the test data affected by these classes.

Results As Table 5.1 shows, the OGML method demonstrates overall improvements compared to the knowledge-uninformed baseline across all three hierarchical datasets. However, a substantial number of classes does not benefit from the ontological domain knowledge (highlighted in red on the distribution plots of refinement scores). Since the classes are relatively balanced in all datasets, we observe a similar percentage of data affected by the mismatch.

Table 5.2.: Comparison of model sizes for two variants of risk prediction (negative refinement scores in red).

	Heart Disease Pred.		Diabetes Pred.	
Architecture	Small	Large	Small	Large
Acc. Baseline [%]	71.36	79.66	70.32	85.72
Acc. OGML [%]	78.98	81.32	89.03	90.33
Improvement [pp]	7.62	1.66	18.71	4.61
Mismatch [%]	15.07	20.07	4.73	11.47
– Data affected [%]	40.79	78.76	11.46	48.90

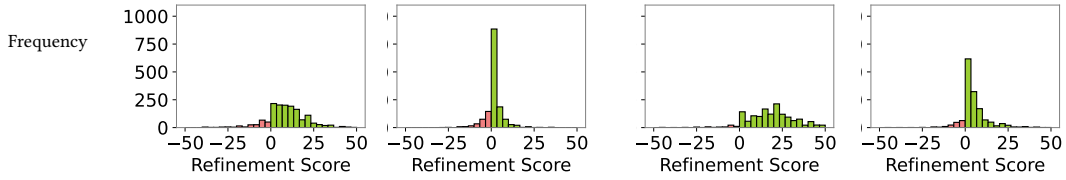
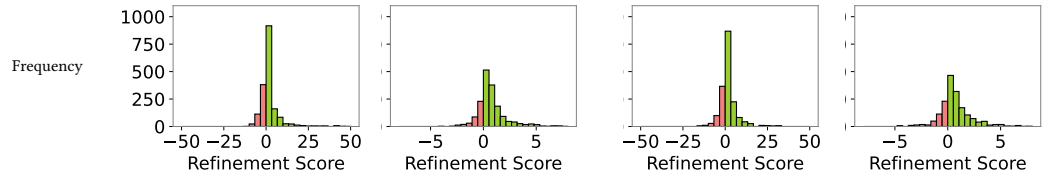


Table 5.3.: Comparison of model sizes and refinement metrics for next-visit prediction (negative refinement scores in red).

	Next-Visit Prediction			
Architecture	Small		Large	
Acc. Baseline [%]	55.10		66.19	
Acc. OGML [%]	73.87		71.32	
Improvement [pp]	18.77		5.13	
Refinement Metric	Acc.-based	Rank.-based	Acc.-based	Rank.-based
Mismatch [%]	29.60	21.47	29.71	25.56
– Data affected [%]	83.50	59.61	82.67	60.78



5.3.2. Ontology-Guided Sequential Health Prediction

Scenario In the second use case, we demonstrate how to quantify domain-application mismatch in medical data processing. We apply our proposed framework to an OGML approach for sequential patient health prediction, proposed by [Cho+17]. This approach incorporates structured information about the hierarchical relationships of varying depth between medical codes of symptoms and diseases defined by the ICD-9 classification system[2]. It processes sequences of medical codes with a graph-based attention mechanism (GRAM) to generate semantic embeddings, considering not only individual medical codes but also their hierarchical ancestors.

[2] `http://www.icd9data.com/2015/Volume1/default.htm`

Experimental Setup We employ a similar setup as the original paper by [Cho+17], utilizing the publicly available MIMIC-III healthcare dataset [Joh+16]. Different from the computer vision use case, we train the OGML model ourselves, varying the experiments across three prediction tasks: two *risk prediction* tasks – for heart diseases and diabetes – and one *next-visit prediction* task. In the dataset, each patient visit is represented by medical codes corresponding to the diagnoses and symptoms identified during that visit. For risk prediction, the goal is to predict whether the patient's next visit will include a diagnosis of heart disease or diabetes, based on their previous visits. For next-visit prediction, the goal is to predict all the diagnoses and symptoms recorded during the patient's next visit, based on their previous visits.

To address the stochastic nature of machine learning methods, we conduct five experiments for each combination of task and model size. In each experiment, we used random train-test splits – 80-20 for risk prediction and 90-10 for next-visit prediction. We then report the average performance on the test sets. Each model comprises an embedding layer, an RNN layer, and a final dense layer. The dense layer uses a sigmoid activation function for risk prediction and a softmax activation function for next-visit prediction. The larger model has an attention dimension of 100, an RNN dimension of 200, and an embedding dimension of 300, and it is trained with a batch size of 128 for 100 epochs with early stopping. The smaller model has an attention dimension of 16, an RNN dimension of 32, and an embedding dimension of 16, and it is trained with a batch size of 32 for 50 epochs with early stopping. For further details on the experimental setup, please refer to the experimental code provided along with this thesis.

Refinement Metric To quantify domain-application mismatch in both tasks, we define the refinement metric as the per-code performance improvements based on all input-output pairs where the input sequences (patient visits) include that particular medical code. For risk prediction, the improvement is measured using binary accuracy. For next-visit prediction, we define two variants of the refinement metric. The first one is accuracy-based, similar to the risk prediction task, but using top-20 accuracy to measure the improvement, in line with the evaluation metric from the original paper. The second one measures the average rank improvement between the baseline and OGML approach by comparing the rank differences for medical codes found in the ground-truth data for the respective patient visits. In both tasks, input sequences may be counted multiple times for different medical codes. As in the computer vision use case, we calculate the mismatch as the percentage of classes (codes) that show a performance decrease, and we also report the proportion of data affected by these classes (codes).

Given the smaller dataset sizes, varying train-test splits, and a larger number of unique classes compared to the computer vision use case, we report the mismatch on the entire dataset (including the train set) instead of just the test set. To handle the high number of unique medical codes (1,823 for next-visit prediction and 2,426 for risk prediction), we filter out codes that appear fewer than three times in the dataset. This results in 38.2% of unique codes being filtered out for risk prediction and 2.1% for next-visit prediction.

Results The results, summarized in Tables 5.2 and 5.3, reveal distinct patterns across the various models and tasks. First, domain-application knowledge mismatch is evident across all model sizes, prediction tasks, and refinement metrics within the dataset. However, the degree of mismatch varies, with the mismatch for diabetes risk prediction being two to three times smaller than that for heart disease risk prediction using the same model sizes. This variation is also reflected in the differences in the distribution of refinement scores shown in the bottom row of the table. Additionally, models used for diabetes risk prediction benefit more from domain knowledge than those used for heart disease risk prediction. Further, ontological domain knowledge tends to improve the performance of smaller models more than of larger models. For risk prediction, smaller OGML models perform nearly as well as their larger counterparts, while in next-visit prediction, smaller OGML models even outperform the larger ones. Additionally, smaller models generally exhibit less domain-application mismatch, i.e., they benefit more from domain knowledge than larger models. Lastly, we observe a much higher percentage of data affected by mismatches compared to the computer vision use case, likely due to the presence of multiple medical codes in each input sequence.

5.3.3. Identifying Ontological Issues

When examining the top ten medical categories with the lowest accuracy-based refinement scores for next-visit prediction, we found several potential ontological issues (with the ICD-9 hierarchy) that could decrease the prediction performance of OGML.

5.3.3.1. Similar Concepts in Different Ontological Paths

This issue arises when related ontological categories are placed under different paths and lack a common semantic ancestor. As a result, the OGML approach treats these categories as semantically independent, which can confuse the machine-learning model and negatively impact the prediction performance for these categories. For example, five of the ten medical codes with the lowest refinement scores are related to drug-related symptoms or diseases. These five categories fall into three distinct ontological paths without a shared common ancestor:

- *970.8: Poisoning by other specified central nervous system stimulants*, falls under the ontological category *Injury and Poisoning 800-999*
- *E950.0: Suicide and self-inflicted poisoning by analgesics, antipyretics, and antirheumatics* and *E950.4: Suicide and self-inflicted poisoning by other specified drugs and medicinal substances* both fall under the ontological category *Supplementary Classification of External Causes of Injury and Poisoning E000-E999*.
- *304.23 Cocaine dependence, in remission*, and *304.21 Cocaine dependence, continuous* both fall under the ontological category *Mental Disorders 290-319*.

5.3.3.2. Irrelevant Categorization

This issue arises when the categorization focuses on aspects that may be less relevant to the machine-learning task. For instance, consider the following codes:

- *E956 Suicide and self-inflicted injury by cutting and piercing instrument*
- *E950.0 Suicide and self-inflicted poisoning by analgesics, antipyretics, and antirheumatics*
- *E950.4 Suicide and self-inflicted poisoning by other specified drugs and medicinal substances*

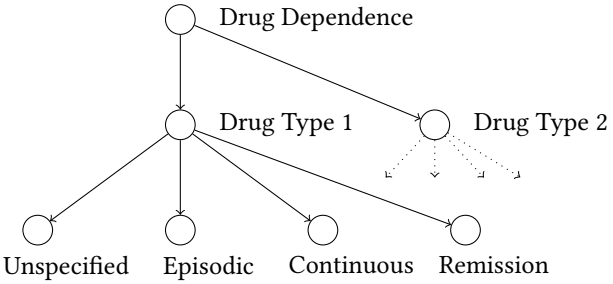
While these codes categorize different types of injuries (cutting, poisoning by drugs, etc.), they are all grouped under the broader category of *Suicide and Self-Inflicted Injury (E950-E959).* This grouping does not account for other causes of such injuries. For next-visit prediction, the focus on whether an injury is self-inflicted might be less relevant than the specific type of injury. A more effective approach could be categorizing these codes based on the type of injury (e.g., cutting, poisoning) rather than its origin, as this may be more relevant to the prediction task.

5.3.3.3. Inaccurate or Overly Broad Categories

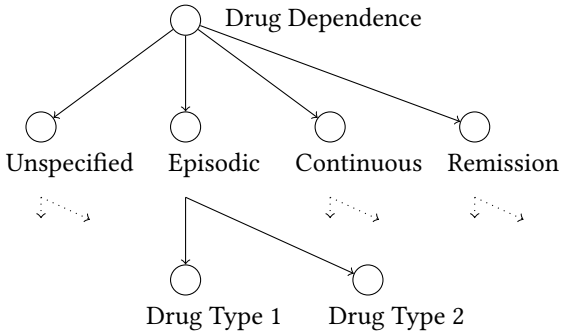
This issue arises when a category is not specific enough or is overly broad. Categories that include terms like "unspecified" or "other", or have such terms in their parent categories, are especially susceptible to this problem. For example, the code *957.1 Injury to other specified nerve(s)* is classified under the broader category *957 Injury to other and unspecified nerves.* This broad classification can include various, potentially unrelated medical codes within the same category, which may confuse the machine-learning model.

5.3.3.4. Suboptimal Ordering of Parent Categories

This issue arises when parent categories are organized to prioritize one aspect over another, which may not be optimal for the specific task. For example, the ICD-9 ontology initially classifies drug dependence by drug type and then by dependence type (Figure 5.2, top). This structure leads the OGML approach to treat continuous use of different drugs as unrelated and continuous versus episodic use of the same drug as more similar. Reordering to classify by dependence type first (Figure 5.2, bottom) could better capture the nuances of drug use. Similarly, Figure 5.3 shows that organizing wound information by location and type at the same level may not be ideal. Depending on the task, it might be more effective to classify injuries first by location and then by type, or vice versa, or even to provide both ordering paths.



(a) Example of the suboptimal parent order.



(b) Example of the improved parent order.

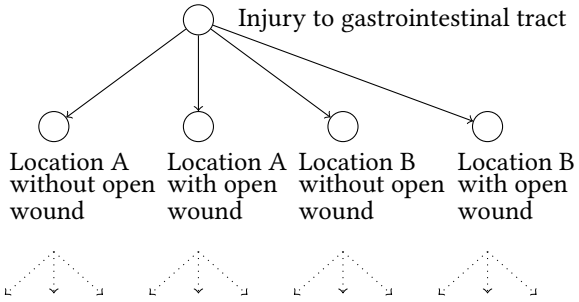
Figure 5.2.: Suboptimal parent order (top) and potential improved one (bottom) for categories related to drug dependence.

5.3.3.5. Examples of Useful Categorization

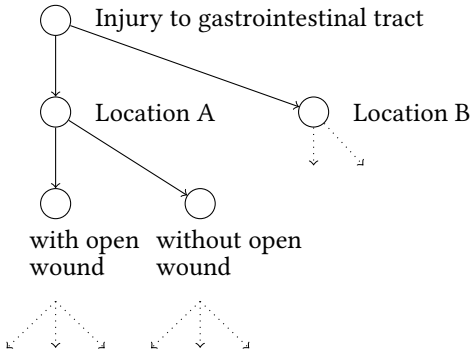
To address ontological challenges, it can be helpful to identify which categories gained the most from incorporating domain-specific ontological knowledge in the given task. In our experiments, we found that the top 10 categories with the highest refinement scores frequently involved medical codes related to newborns or children, such as:

- *Certain Conditions Originating in the Perinatal Period (760-779)*, with 6 out of 10 codes:
  - 765.28, 767.1, 767.19, 770.12, 770.89, 774.6
- *A skin abnormality present at birth or detected in the neonatal period* (757.32)
- *Liveborn Infants According to Type of Birth (V30.00)*
- *Health supervision of infant or child (V20.2)*
- *Need for prophylactic vaccination and inoculation against viral hepatitis (V05.3)*

This indicates that the ontological structure of these categories is well-suited to the prediction task. Future research should focus on identifying the structural patterns behind this success, possibly through collaboration with medical experts and a detailed analysis of the data subsets that benefited from this knowledge. We encourage readers to review



(a) Example of the suboptimal parent order.



(b) Example of the improved parent order.

Figure 5.3.: Suboptimal parent order (top) and potential improved one (bottom) for categories related to gastrointestinal tract.

the experimental results to identify additional patterns in the refinement scores across different refinement metrics and prediction tasks.

5.4. Summary

In this chapter, we addressed the critical and often overlooked issue of domain-application knowledge mismatch in ontology-guided machine learning (OGML). We developed an OGML-aware framework to quantify these mismatches and identify harmful ontology parts, which negatively affect prediction performance. Our framework offers a practical and generalizable methodology for assessing ontology quality in OGML contexts. Thus, it improves the integration of ontological knowledge into machine-learning models, leading to more effective and reliable use of ontologies. Our case studies in image classification and patient health prediction revealed that mismatches are widespread across datasets, OGML approaches, and machine-learning architectures. This highlights the importance of aligning domain ontologies with specific application requirements in OGML contexts.

5.5. Author's Contribution

The observations regarding the negative impact of ontological domain knowledge on predictions in Ontology-guided Machine Learning were made by me in collaboration with Lena Witterauf [Wit21] and Sönke Jendral [Jen22], both of whom conducted their theses under my supervision. The initial reimplementation of the GRAM framework, used for the experiments on patient health prediction, was carried out by Lena Witterauf under my supervision. The ideas concerning various aspects of low-quality ontological domain knowledge, including harmful and helpful parts of domain knowledge, were developed by me together with Lena Witterauf and Sönke Jendral. The observation that the topic of low-quality domain knowledge in OGML is underexplored was mine, as was the naming of the phenomenon "domain-application mismatch". The concept of the framework to quantify the mismatch was my idea. I was responsible for implementing the experiments to measure the mismatch, as well as for evaluating and visualizing the results. All sections of the publication were written by me, with support and feedback from Jakob Bach and Prof. Ralf Mikut.

Part III.

Conclusions

6. Conclusions

In this dissertation, we investigated three sources of low-quality domain knowledge in Knowledge-Guided Machine Learning (KGML) that negatively affect machine learning predictions and proposed solutions to address them. In particular, we made the following contributions:

In Chapter 3, we introduced novel evaluation criteria for KGML methods relevant for practical collaboration with domain experts and facilitate the complex process of integrating domain knowledge into machine learning models. We applied these criteria in a comparative study of KGML methods for lithium-ion battery voltage prediction. Our findings indicate that while effective KGML methods require substantial domain knowledge, they do not necessarily demand significant implementation effort.

In Chapter 4, we demonstrated how to account for imperfect representation of domain knowledge when designing and evaluating new KGML methods. Specifically, we proposed a knowledge-guided constraint to represent approximate domain knowledge and integrated it through a knowledge-guided loss function. Additionally, we demonstrated how to use sensitivity analysis to assess the robustness of KGML methods against inaccurate domain knowledge. Our results demonstrated that both approximate and incorrect domain knowledge can still provide value in practical applications to a certain degree which can be assessed with our proposed approach.

In Chapter 5, we presented a framework to quantify mismatches between domain knowledge and task-specific knowledge in Ontology-Guided Machine Learning. We then examined how these mismatches manifest and influence model predictions. Our findings showed that such mismatches are common and can affect substantial portions of the ontological domain knowledge, leading to a significant impact on prediction performance, particularly within specific subsets of the data. By applying our framework, these mismatches can be identified and addressed, enabling the redesign of ontological domain knowledge to improve model performance.

These contributions enhance our understanding of the important yet underexplored issue of low-quality domain knowledge in KGML and provide practical strategies for mitigating its effects. In the next chapter, we will outline potential directions for future research.

7. Future Work

In this section, we highlight potential directions for future research that could further advance the field of Knowledge-Guided Machine Learning (KGML).

Expanding KGML Across Diverse Applications As discussed in Section 1.2.1, the field of KGML remains highly application-specific, with many methods tailored to particular scenarios. In Chapter 3, we demonstrated how formalizing and integrating domain knowledge into machine learning models can serve as a reference for future implementation within the same application domain. A natural future work is to continue this effort, by applying KGML principles to more and more use cases.

Identifying and Integrating New Forms of Domain Knowledge As discussed in Section 1.2.1, the identification and integration of new forms of domain knowledge into KGML methods remains an ongoing effort in the field. As described in Chapter 4, interaction with domain experts and deep understanding of the domain is often needed to reveal novel types of domain knowledge that can lead to innovative KGML methods. These novel forms of domain knowledge may be potentially applicable to other domains as well. Future research should continue to explore and document these novel forms of domain knowledge, with a focus on methodologies that facilitate their integration across a wide range of practical use cases.

Expanding KGML Comparative Studies to Other Dynamical Systems The comparative study described in Chapter 3, that focuses on time-series voltage prediction in lithium-ion battery modeling, serves as a foundational example. Similar studies on other dynamical systems, such as electrical motors, hydraulic systems, heat exchangers, or chemical reactors, could leverage rich domain knowledge to enhance data-driven time-series prediction models. It would reinforce the proposed taxonomy of the proposed evaluation criteria, highlighting its practical value for effective collaboration with domain experts.

Expanding KGML Comparative Studies to Other Machine Learning Tasks Beyond time-series prediction, extending KGML comparative studies, such as the one described in Chapter 3, to non-temporal regression and classification tasks in engineering and natural sciences can provide valuable insights. Examples include material property prediction, fault classification, and disease diagnosis, where domain knowledge can enhance model performance. Such studies would achieve two key goals: identifying new forms of domain

knowledge and guiding the selection of appropriate KGML paradigms, thereby encouraging their adoption across various machine learning applications.

Developing Reusable Design Patterns in KGML KGML paradigms are highly adaptable, often allowing multiple instantiations for a given form of domain knowledge. Conducting comprehensive comparative studies across paradigms and their various instantiations can be resource-intensive. A more efficient approach would focus on identifying common design requirements across domains and developing reusable design patterns to meet these needs. Future studies could adopt a narrower focus, concentrating on use-case independent forms of domain knowledge and design requirements. For instance, as discussed in Section 3.2.1.2, initial battery charge and voltage illustrate the design requirement of system response based on initial conditions. Exploring KGML paradigm instantiations as reusable design patterns for similar problems could streamline the application of KGML to a broader set of tasks. Priority should also be given to evaluating widely applicable design requirements and developing specialized test benches for validating specific design needs.

Exploring the Applicability of Permissible System States Across Various Dynamical Systems In Chapter 4, we introduced and formalized a novel type of expert knowledge: permissible system states, which we applied to temporal modeling of gas turbines. Future research could investigate the broader applicability of this approach across various dynamical systems with similar characteristics, such as thermal engines, energy storage systems or power-to-X systems.

Expanding the Proposed Framework for Ontology-Guided Machine Learning to Other Methods In Chapter 5, we introduced a general framework for quantifying mismatches between domain knowledge and task-specific knowledge in OGML. We demonstrated its use in two common OGML applications: image classification and patient health prediction. Future research could extend its applicability to other domains and OGML methods.

Refining the Proposed Framework for OGML by Incorporating Statistical Significance Tests Incorporating statistical significance tests into the framework proposed in Chapter 5 would enable a more nuanced analysis of the ontology's helpful and harmful parts, addressing the sensitivity of the refinement metric to small changes in the ontology structure.

Expanding Analysis for Identifying Ontological Issues In Section 5.3.3, we demonstrated that analyzing data categories with low refinement scores can uncover ontological issues, such as irrelevant categorization or suboptimal ordering. Future work could expand this approach to validate these identified issues and potentially uncover additional ontological problems in OGML applications like patient health prediction and computer vision. It is important to note that the current OGML approach for image classification, described in

Section 5.3.1, uses a single pretrained model for each image dataset due to limited access to the training code. While a single pretrained model is sufficient to show the existence of knowledge mismatch, it may lack the statistical precision required to reliably identify ontological issues. For more precise analysis, OGML models should ideally be retrained multiple times to account for the stochastic nature of machine learning (as discussed in Section 5.2.1), similar to the approach described in Section 5.3.2. In general, identifying and understanding ontological issues requires domain expertise. Future analyses should involve collaboration with domain experts to ensure that findings are both relevant to the domain and useful for practitioners.

Evaluating Domain Fitness of Ontologies with OGML In Chapter 5, we demonstrated how to measure an ontology's application fitness, i.e. usefulness for specific task, using our proposed framework (see Section 2.2.1). Future research could expand on this by applying the framework to multiple tasks within a single domain, offering a more comprehensive evaluation of the ontology's overall domain fitness. This approach shows promise as a generalizable method for ontology quality assessment, enabling the identification of various ontological issues as illustrated in Chapter 5, section 5.3.3.

Refining the Ontological Domain Knowledge for Ontology-Guided Machine Learning As demonstrated in Chapter 5, our proposed framework identifies specific parts of the ontology that negatively impact model predictions. Future research could build on this by refining these problematic sections and reassessing their effectiveness. This refinement could involve either removing the harmful components or redesigning them to better align with the task requirements, ultimately improving the overall performance of the ontology-guided machine learning model. We have begun exploring this approach in our workshop paper [Bie+23], where preliminary results were promising. Additionally, modern large language models (LLMs) [Bro+20], could be used to suggest alternative ontology parts for those identified as harmful.

Bibliography

[Aba+16] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. arXiv:1603.04467 [cs]. Mar. 2016 (page 54).

[ABH95] Sarabjot S. Anand, David A. Bell, and John G. Hughes. "The role of domain knowledge in data mining". In: *Proceedings of the Fourth International Conference on Information and Knowledge Management*. 1995, pp. 37–43 (page 4).

[Abu92] Yaser Abu-Mostafa. "A Method for Learning From Hints". In: *Advances in Neural Information Processing Systems*. Vol. 5. Morgan-Kaufmann, 1992 (pages 4, 14, 48, 49).

[Arb+19] Aryan Arbabi, David R Adams, Sanja Fidler, and Michael Brudno. "Identifying Clinical Terms in Medical Text Using Ontology-Guided Machine Learning". en. In: *JMIR Medical Informatics* 7.2 (May 2019), e12596. ISSN: 2291-9694 (page 16).

[BA22] Reyhaneh Banihabib and Mohsen Assadi. "The role of micro gas turbines in energy transition". In: *Energies* 15.21 (2022), p. 8084 (page 45).

[BAA22] Pauline Bernard, Vincent Andrieu, and Daniele Astolfi. "Observer design for continuous-time dynamical systems". In: *Annual Reviews in Control* 53 (2022), pp. 224–248 (page 43).

[BBD21] Clemens-Alexander Brust, Björn Barz, and Joachim Denzler. "Making Every Label Count: Handling Semantic Imprecision by Integrating Domain Knowledge". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 6866–6873 (pages 7, 17).

[BD19a] Björn Barz and Joachim Denzler. "Hierarchy-Based Image Embeddings for Semantic Image Retrieval". In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. ISSN: 1550-5790. Jan. 2019, pp. 638–647 (page 16).

[BD19b] Clemens-Alexander Brust and Joachim Denzler. "Integrating Domain Knowledge: Using Hierarchies to Improve Deep Classifiers". In: *Asian Conference on Pattern Recognition*. Springer. 2019, pp. 3–16 (page 16).

[BD19c] Clemens-Alexander Brust and Joachim Denzler. “Not Just a Matter of Semantics: The Relationship Between Visual and Semantic Similarity”. In: *Pattern Recognition: 41st DAGM German Conference, DAGM GCPR 2019, Dortmund, Germany, September 10–13, 2019, Proceedings 41*. Springer. 2019, pp. 414–427 (pages 7, 10, 17).

[Ber+20] Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A. Lord. “Making Better Mistakes: Leveraging Class Hierarchies With Deep Networks”. en. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 12503–12512 (page 16).

[Beu+21] Tom Beucler, Michael Pritchard, Stephan Rasp, Jordan Ott, Pierre Baldi, and Pierre Gentine. “Enforcing Analytic Constraints in Neural Networks Emulating Physical Systems”. en. In: *Physical Review Letters* 126.9 (Mar. 2021), p. 098302. ISSN: 0031-9007, 1079-7114 (pages 3, 13).

[Bey+11] Ghassan Beydoun, Antonio A. Lopez-Lorca, Francisco García-Sánchez, and Rodrigo Martínez-Béjar. “How Do We Measure and Improve the Quality of a Hierarchical Ontology?” en. In: *Journal of Systems and Software* 84.12 (Dec. 2011), pp. 2363–2373. ISSN: 01641212 (page 15).

[BGM05] Janez Brank, Marko Grobelnik, and Dunja Mladenic. “A Survey of Ontology Evaluation Techniques”. In: *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)*. Citeseer, 2005, pp. 166–170 (page 16).

[Bie+23] Pawel Bielski, Lena Witterauf, Sönke Jendral, and Jakob Bach. “Towards Automatically Refining Low-Quality Domain Knowledge: A Case Study in Healthcare”. In: *Proceedings of the 1st European Knowledge-Guided Machine Learning Workshop at ECML-PKDD*. Part of ECML-PKDD 2023, (to be published). Turin, Italy, Sept. 2023 (page 83).

[Bie+24a] Pawel Bielski, Nico Denner, Simon Bischof, Benedikt Rzepka, and Klemens Böhm. “Theory-Guided Data-Science for Battery Modeling: Insights from a Comparative Study”. In: *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*. 2024, pp. 1–10 (pages 12, 21).

[Bie+24b] Pawel Bielski, Aleksandr Eismont, Jakob Bach, Florian Leiser, Dustin Kottonau, and Klemens Böhm. “Knowledge-Guided Learning of Temporal Dynamics and its Application to Gas Turbines”. In: *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*. e-Energy ’24. Singapore, Singapore, 2024, pp. 279–290 (pages 12, 43).

[Bie+24c] Pawel Bielski, Lena Witterauf, Sönke Jendral, Ralf Mikut, and Jakob Bach. “Quantifying Domain-Application Knowledge Mismatch in Ontology-Guided Machine Learning”. In: *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. Ed.: D. Aveiro. Vol. 2*. 2024, p. 216 (pages 12, 63).

[Bre+04] Christopher Brewster, Harith Alani, Srinandan Dasmahapatra, and Yorick Wilks. "Data driven ontology evaluation". In: *Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon, Portugal*. 2004 (page 15).

[Bre02] Christopher Brewster. "Techniques for Automated Taxonomy Building: Towards Ontologies for Knowledge Management". In: *Proceedings of the 5th Annual CLUK Research Colloquium, Leeds, United Kingdom*. 2002 (page 15).

[Bro+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901 (page 83).

[Bur+05] Andrew Burton-Jones, Veda C. Storey, Vijayan Sugumaran, and Punit Ahluwalia. "A Semiotic Metrics Suite for Assessing the Quality of Ontologies". In: *Data & Knowledge Engineering* 55.1 (2005). Publisher: Elsevier, pp. 84–102 (pages 15, 16).

[C4003] CIGRE Task Force C4.02.25. "Modeling of Gas Turbines and Steam Turbines in Combined Cycle Power Plants". In: *CIGRE Technical Brochure* 238 (2003) (page 49).

[Cai+21] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. "Physics-Informed Neural Networks (PINNs) for Fluid Mechanics: A Review". In: *Acta Mechanica Sinica* (2021) (page 4).

[Cap99] Maria Angeles Capellades. "Assessment of Reusability of Ontologies: A Practical Example". In: *Proceedings of AAAI1999 Workshop on Ontology Management, AAAI Press*. 1999, pp. 74–79 (page 15).

[Che+18] Tianshui Chen, Wenxi Wu, Yuefang Gao, Le Dong, Xiaonan Luo, and Liang Lin. "Fine-Grained Representation Learning and Recognition by Exploiting Hierarchical Semantic Embedding". en. In: *Proceedings of the 26th ACM International Conference on Multimedia*. Seoul Republic of Korea: ACM, Oct. 2018, pp. 2023–2031 (pages 4, 7, 16, 64, 67, 68).

[Cho+17] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F. Stewart, and Jimeng Sun. "GRAM: Graph-based Attention Model for Healthcare Representation Learning". en. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Halifax NS Canada: ACM, Aug. 2017, pp. 787–795 (pages 4, 6, 7, 10, 16, 17, 63, 64, 69, 70).

[Cla+13] Erik L Clarke, Salvatore Loguercio, Benjamin M Good, and Andrew I Su. "A Task-Based Approach for Gene Ontology Evaluation". en. In: *Journal of Biomedical Semantics* 4.S1 (Apr. 2013), S4. ISSN: 2041-1480 (pages 15, 16).

[DA94] F. P. De Mello and D. J. Ahner. "Dynamic Models for Combined Cycle Plants in Power System Studies". In: *IEEE Transactions on Power Systems (Institute of Electrical and Electronics Engineers)* 9.3 (1994) (page 49).

[Daw+17] Arka Daw, Anuj Karpatne, William D Watkins, Jordan S Read, and Vipin Kumar. "Physics-guided neural networks (PGNN): An application in lake temperature modeling". In: *Knowledge-guided machine learning*. 2017 (pages 6, 14, 30).

[Daw+20] Arka Daw, R Quinn Thomas, Cayelan C Carey, Jordan S Read, Alison P Appling, and Anuj Karpatne. "Physics-guided architecture (PGA) of neural networks for quantifying uncertainty in lake temperature modeling". In: *International Conference on Data Mining*. SIAM. 2020 (pages 5, 27–29, 38).

[De 23] Roberta De Robbio. "Micro Gas Turbine Role in Distributed Generation with Renewable Energy Sources". en. In: *Energies* 16.2 (Jan. 2023), p. 704. ISSN: 1996-1073 (page 46).

[Den+14] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. "Large-Scale Object Classification Using Label Relation Graphs". en. In: *Computer Vision – ECCV 2014*. Vol. 8689. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 48–64 (pages 7, 16, 17).

[Den21] Nico Denner. "Theory-Guided Data Science for Battery Voltage Prediction: a Systematic Study". Bachelor's Thesis. Karlsruhe Institute of Technology (KIT), 2021 (page 41).

[Dha+20] Ankit Dhall, Anastasia Makarova, Octavian Ganea, Dario Pavllo, Michael Greeff, and Andreas Krause. "Hierarchical Image Classification using Entailment Cone Embeddings". en. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Seattle, WA, USA: IEEE, June 2020, pp. 3649–3658 (pages 9, 16, 17).

[Dje+22] Franck Djeumou, Cyrus Neary, Eric Goubault, Sylvie Putot, and Ufuk Topcu. "Neural networks with physics-informed architectures and constraints for dynamical systems modeling". In: *Proceedings of the Learning for Dynamics and Control Conference*. PMLR. 2022, pp. 263–277 (pages 3, 13).

[Don+15] Wolfgang Doneit, Ralf Mikut, Lutz Gröll, and Markus Reischl. "Vorwissen in Funktionsapproximationen durch Support-Vektor-Regression bei schlechter Datenqualität". In: *Proceedings of the 25th Workshop Computational Intelligence, Dortmund, Germany*. Vol. 25. 2015, pp. 163–181 (page 14).

[Duq+11] Astrid Duque-Ramos, Jesualdo Tomás Fernández-Breis, Robert Stevens, and Nathalie Aussenac-Gilles. "OQuaRE: A SQuaRE-based approach for evaluating the quality of ontologies". In: *Journal of Research and Practice in Information Technology* 43.2 (2011), pp. 159–176 (page 15).

[Elh+22] Mohannad Elhamod, Jie Bu, Christopher Singh, Matthew Redell, Abantika Ghosh, Viktor Podolskiy, Wei-Cheng Lee, and Anuj Karpatne. "*CoPhy* -PGNN: Learning Physics-guided Neural Networks with Competing Loss Functions for Solving Eigenvalue Problems". en. In: *ACM Transactions on Intelligent Systems and Technology* 13.6 (Dec. 2022), pp. 1–23. ISSN: 2157-6904, 2157-6912 (page 14).

[Góm99] Asunción Gómez-Pérez. "Evaluation of taxonomic knowledge in ontologies and knowledge bases". In: *Proceedings of the 12th Banff Workshop on Knowledge Acquisition, Modeling and Management (KAW'99)*. Publisher: University of Calgary, Alberta, Canada. 1999 (page 15).

[Goo+16] Wonjoon Goo, Juyong Kim, Gunhee Kim, and Sung Ju Hwang. "Taxonomy-Regularized Semantic Deep Convolutional Neural Networks". en. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Vol. 9906. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 86–101 (page 16).

[Göt+16] Manuel Götz, Jonathan Lefebvre, Friedemann Mörs, Amy McDaniel Koch, Frank Graf, Siegfried Bajohr, Rainer Reimert, and Thomas Kolb. "Renewable Power-to-Gas: A technological and economic review". In: *Renewable Energy* 85 (Jan. 2016), pp. 1371–1390. ISSN: 0960-1481 (page 45).

[Grö15] Lutz Gröll. *Methodik zur Integration von Vorwissen in die Modellbildung*. Vol. 52. KIT Scientific Publishing, 2015 (page 14).

[GT16] Satya Gopisetty and Peter Treffinger. "Generic Combined Heat and Power (CHP) Model for the Concept Phase of Energy Planning Process". en. In: *Energies* 10.1 (Dec. 2016), p. 11. ISSN: 1996-1073 (pages 45, 46, 49).

[Hea16] Jeff Heaton. "An empirical analysis of feature engineering for predictive modeling". In: *SoutheastCon 2016*. IEEE. 2016 (page 25).

[HS14] Hlomani Hlomani and Deborah Stacey. "Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey". In: *Semantic Web Journal* 1.5 (2014). Publisher: IOS Press, pp. 1–11 (pages 15, 64).

[Jen22] Sönke Jendral. "Refining Domain Knowledge for Domain Knowledge Guided Machine Learning". Bachelor's Thesis. Karlsruhe Institute of Technology (KIT), 2022 (page 75).

[Jia+19] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan Read, Jacob Zwart, Michael Steinbach, and Vipin Kumar. "Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles". In: *International Conference on Data Mining*. SIAM. 2019 (pages 4, 5, 14, 22, 23, 27–29, 38).

[Jia+21] Xiaowei Jia, Jacob Zwart, Jeffrey Sadler, Alison Appling, Samantha Oliver, Steven Markstrom, Jared Willard, Shaoming Xu, Michael Steinbach, Jordan Read, et al. "Physics-guided recurrent graph model for predicting flow and temperature in river networks". In: *International Conference on Data Mining*. SIAM. 2021 (page 5).

[Joh+16] Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. "MIMIC-III, a freely accessible critical care database". en. In: *Scientific Data* 3.1 (May 2016). Number: 1 Publisher: Nature Publishing Group, p. 160035. ISSN: 2052-4463 (page 70).

[Ju+24] Lie Ju, Zhen Yu, Lin Wang, Xin Zhao, Xin Wang, Paul Bonnington, and Zongyuan Ge. "Hierarchical Knowledge Guided Learning for Real-World Retinal Disease Recognition". en. In: *IEEE Transactions on Medical Imaging* 43.1 (Jan. 2024), pp. 335–350. ISSN: 0278-0062, 1558-254X (page 16).

[Kar+17] Anuj Karpatne, Gowtham Atluri, James H. Faghmous, Michael Steinbach, Arindam Banerjee, Auroop Ganguly, Shashi Shekhar, Nagiza Samatova, and Vipin Kumar. "Theory-guided data science: A new paradigm for scientific discovery from data". In: *IEEE Transactions on Knowledge and Data Engineering* 29.10 (2017). Publisher: IEEE, pp. 2318–2331 (pages 3–5, 7, 8, 13, 26–28, 39).

[Kar+21] Shyamgopal Karthik, Ameya Prabhu, Puneet K. Dokania, and Vineet Gandhi. *No Cost Likelihood Manipulation at Test Time for Making Better Mistakes in Deep Networks*. arXiv:2104.00795 [cs]. Apr. 2021 (pages 9, 17).

[Kas+21] Karthik Kashinath, M. Mustafa, Adrian Albert, J. L. Wu, C. Jiang, Soheil Esmaeilzadeh, Kamyar Azizzadenesheli, R. Wang, A. Chattopadhyay, and A. Singh. "Physics-informed machine learning: case studies for weather and climate modelling". In: *Philosophical Transactions of the Royal Society A* 379.2194 (2021), p. 20200093 (page 48).

[KKK22] Anuj Karpatne, Ramakrishnan Kannan, and Vipin Kumar. *Knowledge guided machine learning: Accelerating discovery using scientific knowledge and data*. CRC Press, 2022 (page 4).

[KM19] J. Zico Kolter and Gaurav Manek. "Learning Stable Deep Dynamics Models". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019 (page 13).

[Kot23] Dustin Kottonau. "Echtzeitsimulation und Netzintegration einer Mikrogasturbine". German. PhD Thesis. Karlsruher Institut für Technologie (KIT), 2023 (pages 49, 52).

[Lei21] Florian Leiser. "Modeling Dynamical Systems using Transition Constraints". Masters's Thesis. Karlsruhe Institute of Technology (KIT), 2021 (page 62).

[Li+21] Weihan Li, Jiawei Zhang, Florian Ringbeck, Dominik Jöst, Lei Zhang, Zhongbao Wei, and Dirk Uwe Sauer. "Physics-informed neural networks for electrode-level state estimation in lithium-ion batteries". In: *Journal of Power Sources* (2021) (pages 4, 22).

[Li+22] Liulei Li, Tianfei Zhou, Wenguan Wang, Jianwu Li, and Yi Yang. "Deep Hierarchical Semantic Segmentation". en. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 1236–1247 (page 16).

[Ma+18] Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. "KAME: Knowledge-based Attention Model for Diagnosis Prediction in Healthcare". en. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. Torino Italy: ACM, Oct. 2018, pp. 743–752 (page 63).

[Ma+19] Fenglong Ma, Yaqing Wang, Houping Xiao, Ye Yuan, Radha Chitta, Jing Zhou, and Jing Gao. "Incorporating medical code descriptions for diagnosis prediction in healthcare". In: *BMC Medical Informatics and Decision Making* 19.6 (Dec. 2019), p. 267. ISSN: 1472-6947 (page 16).

[MAD17] Silvio Mc Gurk, Charlie Abela, and Jeremy Debattista. "Towards ontology quality assessment". In: *Joint Proceedings of MEPDaW and LDQ 2017*. Publisher: CEUR. 2017 (page 15).

[MAM22] Simone Monaco, Daniele Apiletti, and Giovanni Malnati. "Theory-Guided Deep Learning Algorithms: An Experimental Evaluation". In: *Electronics* (2022). ISSN: 2079-9292 (page 8).

[Mei+23] Stefan Meisenbacher, Benedikt Heidrich, Tim Martin, Ralf Mikut, and Veit Hagenmeyer. "AutoPV: Automated photovoltaic forecasts with limited information using an ensemble of pre-trained models". In: *Proceedings of the 14th ACM International Conference on Future Energy Systems*. 2023, pp. 386–414 (page 6).

[Mit97] Tom. M. Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997 (pages 7–9).

[MS20] Melinda McDaniel and Veda C. Storey. "Evaluating Domain Ontologies: Clarification, Classification, and Challenges". en. In: *ACM Computing Surveys* 52.4 (July 2020), pp. 1–44. ISSN: 0360-0300, 1557-7341 (page 15).

[MSG17] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. *The More You Know: Using Knowledge Graphs for Image Classification*. arXiv:1612.04844 [cs]. Apr. 2017 (pages 10, 16, 17).

[Mur+18] Nikhil Muralidhar, Mohammad Raihanul Islam, Manish Marwah, Anuj Karpatne, and Naren Ramakrishnan. "Incorporating Prior Domain Knowledge into Deep Neural Networks". In: *2018 IEEE International Conference on Big Data (Big Data)*. Dec. 2018, pp. 36–45 (page 14).

[Mur+20] Nikhil Muralidhar, Jie Bu, Ze Cao, Long He, Naren Ramakrishnan, Danesh Tafti, and Anuj Karpatne. "Physics-guided deep learning for drag force prediction in dense fluid-particulate systems". In: *Big Data* (2020) (page 4).

[MW12] Hua Min and Janusz Wojtusiak. "Clinical data analysis using ontology-guided rule learning". en. In: *Proceedings of the 2nd International Workshop on Managing Interoperability and CompleXity in Health Systems*. Maui Hawaii USA: ACM, Oct. 2012, pp. 17–22 (page 63).

[Nas+23] Renato G. Nascimento, Felipe AC Viana, Matteo Corbetta, and Chetan S. Kulkarni. "A framework for Li-ion battery prognosis based on hybrid Bayesian physics-informed neural networks". In: *Scientific Reports* (2023) (page 22).

[NJC21] Curtis Northcutt, Lu Jiang, and Isaac Chuang. "Confident learning: Estimating uncertainty in dataset labels". In: *Journal of Artificial Intelligence Research* 70 (2021), pp. 1373–1411 (page 16).

[NK17] Maximillian Nickel and Douwe Kiela. “Poincaré Embeddings for Learning Hierarchical Representations”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017 (page 16).

[OKM11] Mamoru Ohta, Kouji Kozaki, and Riichiro Mizoguchi. “A Quality Assurance Framework for Ontology Construction and Refinement”. In: *Advances in Intelligent Web Mastering – 3*. Vol. 86. Springer, 2011, pp. 207–216 (page 16).

[Par+20] Kyungnam Park, Yohwan Choi, Won Jae Choi, Hee-Yeon Ryu, and Hongseok Kim. “LSTM-based battery remaining useful life prediction with multi-channel charging profiles”. In: *IEEE Access* (2020) (page 28).

[Pas+24] Mostafa Pasandideh, Matthew Taylor, Shafiqur Rahman Tito, Martin Atkins, and Mark Apperley. “Predicting Steam Turbine Power Generation: A Comparison of Long Short-Term Memory and Willans Line Model”. In: *Energies* 17.2 (2024), p. 352 (pages 46, 49).

[PB15] Perrine Pittet and Jérôme Barthélémy. “Exploiting Users’ Feedbacks - Towards a Task-based Evaluation of Application Ontologies Throughout Their Lifecycle:” en. In: *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Lisbon, Portugal, 2015, pp. 263–268 (pages 15, 16).

[Ple18] Gregory Plett. *ECE4710/5710 Modeling, Simulation, and Identification of Battery Dynamics*. University Lecture accessed on 30th Jan. 2022. 2018 (pages 23, 24, 27, 38).

[PM04] Robert Porzel and Rainer Malaka. “A Task-Based Approach for Ontology Evaluation”. In: *ECAI Workshop on Ontology Learning and Population, Valencia, Spain*. Citeseer Valencia, Spain, 2004, pp. 1–6 (pages 15, 16, 64, 65).

[RBB21] Benedikt Rzepka, Simon Bischof, and Thomas Blank. “Implementing an extended Kalman filter for SOC estimation of a Li-ion battery with hysteresis: A step-by-step guide”. In: *Energies* (2021) (page 23).

[REF23] Davide Rigoni, Desmond Elliott, and Stella Frank. “Cleaner Categories Improve Object Detection and Visual-Textual Grounding”. en. In: *Image Analysis*. Ed. by Rikke Gade, Michael Felsberg, and Joni-Kristian Kämäräinen. Vol. 13885. Series Title: Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, 2023, pp. 412–442 (page 16).

[Rem20] Philippe Remy. *Conditional RNN for Keras*. accessed on 25 Jun. 2022. website: https://github.com/philipperemy/cond$_r$nn. 2020 (page 29).

[Rez+22] Pâmela M Rezende, Joicymara S Xavier, David B Ascher, Gabriel R Fernandes, and Douglas E V Pires. “Evaluating hierarchical machine learning approaches to classify biological databases”. en. In: *Briefings in Bioinformatics* 23.4 (July 2022), bbac216. ISSN: 1467-5463, 1477-4054 (page 16).

[RG06] Aaron Radke and Zhiqiang Gao. “A survey of state and disturbance observers for practitioners”. In: *2006 American Control Conference*. IEEE. 2006, 6–pp (page 43).

[RGB23] Laura von Rueden, Jochen Garcke, and Christian Bauckhage. “How Does Knowledge Injection Help in Informed Machine Learning?” In: *2023 International Joint Conference on Neural Networks (IJCNN)*. 2023 (pages 7, 8, 26).

[Rob+22] Haakon Robinson, Suraj Pawar, Adil Rasheed, and Omer San. “Physics guided neural networks for modelling of non-linear dynamics”. In: *Neural Networks* 154 (2022), pp. 333–345 (pages 3, 13).

[Row83] W. I. Rowen. “Simplified Mathematical Representations of Heavy-Duty Gas Turbines”. en. In: *Journal of Engineering for Power* 105.4 (Oct. 1983), pp. 865–869. ISSN: 0022-0825 (pages 46, 49).

[Rue+23] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Michal Walczak, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. “Informed Machine Learning – A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems”. In: *IEEE Transactions on Knowledge and Data Engineering* 35.1 (Jan. 2023), pp. 614–633. ISSN: 1558-2191 (pages 4, 6–8, 13, 39).

[SF11] Carlos N. Silla and Alex A. Freitas. “A survey of hierarchical classification across different application domains”. en. In: *Data Mining and Knowledge Discovery* 22.1-2 (Jan. 2011), pp. 31–72. ISSN: 1384-5810, 1573-756X (pages 9, 16, 17).

[SH19] Mohammadkazem Sadoughi and Chao Hu. “Physics-based convolutional neural network for fault diagnosis of rolling element bearings”. In: *IEEE Sensors Journal* (2019) (pages 22, 23).

[Sim+91] Patrice Simard, Bernard Victorri, Yann LeCun, and John Denker. “Tangent Prop - A formalism for specifying selected invariances in an adaptive network”. In: *Advances in Neural Information Processing Systems*. Vol. 4. Morgan-Kaufmann, 1991 (pages 4, 8, 14).

[Sin+23] Soumya Singh, Yvonne Eboumbou Ebongue, Shahed Rezaei, and Kai Peter Birke. “Hybrid modeling of lithium-ion battery: Physics-informed neural network for battery state estimation”. In: *Batteries* (2023) (pages 22, 23).

[SMB11] Emam Shalan, Mohamed Moustafa Hassan, and ABG Bahgat. “Parameter Estimation and Dynamic Simulation Of Gas Turbine Model In Combined Cycle Power Plants Based On Actual Operational Data”. In: *Journal of American Science* 7 (Jan. 2011) (page 49).

[Ste+19] Georg Steinbuss, Benedikt Rzepka, Simon Bischof, Thomas Blank, and Klemens Böhm. “FOBSS: monitoring data from a modular battery system”. In: *ACM International Conference on Future Energy Systems*. 2019 (page 31).

[Sun+21] Lei Sun, Tianyuan Liu, Yonghui Xie, Di Zhang, and Xinlei Xia. “Real-time power prediction approach for turbine using deep learning techniques”. en. In: *Energy* 233 (Oct. 2021), p. 121130. ISSN: 03605442 (page 49).

[TK94] Michael L Thompson and Mark A Kramer. “Modeling chemical processes using prior knowledge and neural networks”. In: *AIChE Journal* (1994) (pages 6, 22, 23).

[Tu+23] Hao Tu, Scott Moura, Yebin Wang, and Huazhen Fang. “Integrating physics-based modeling with machine learning for lithium-ion batteries”. In: *Applied Energy* (2023) (page 22).

[TVG09] M.R.B. Tavakoli, B. Vahidi, and W. Gawlik. “An Educational Guide to Extract the Parameters of Heavy Duty Gas Turbines Model in Dynamic Studies Based on Operational Data”. en. In: *IEEE Transactions on Power Systems* 24.3 (Aug. 2009), pp. 1366–1374. ISSN: 0885-8950, 1558-0679 (page 49).

[Ven+16] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. *Order-Embeddings of Images and Language*. en. arXiv:1511.06361 [cs]. Mar. 2016 (page 16).

[Wan+20] Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. “Towards Physics-informed Deep Learning for Turbulent Flow Prediction”. en. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Virtual Event CA USA: ACM, Aug. 2020, pp. 1457–1466 (page 6).

[Wei+20] Meng Wei, Min Ye, Jia Bo Li, Qiao Wang, and Xinxin Xu. “State of charge estimation of lithium-ion batteries using LSTM and NARX neural networks”. In: *IEEE Access* (2020) (page 28).

[Wil+20] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. “Integrating physics-based modeling with machine learning: A survey”. In: *arXiv preprint arXiv:2003.04919* 1.1 (2020), pp. 1–34 (pages 5, 6, 21, 27, 39).

[Wil+22] R. S. I. Wilson, J. S. Goonetillake, W. A. Indika, and Athula Ginige. “A conceptual model for ontology quality assessment”. In: *Semantic Web* Preprint (2022). Publisher: IOS Press, pp. 1–47 (pages 15, 16, 64).

[Wil+23] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. “Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems”. en. In: *ACM Computing Surveys* 55.4 (Apr. 2023), pp. 1–37. ISSN: 0360-0300, 1557-7341 (pages 4–8, 13).

[Wil+98] Robert Wilby, Tom Wigley, Declan Conway, P. Jones, Bruce Hewitson, J Main, and Daniel Wilks. “Statistical Downscaling of General Circulation Model Output: A Comparison of Methods”. In: *Water Resources Research* (1998) (pages 6, 30).

[Wit21] Lena Witterauf. “DomainML - A Modular Framework for Domain Knowledge Guided Machine Learning”. Master’s Thesis. Karlsruhe Institute of Technology (KIT), 2021 (page 75).

[WWX18] Hanzhang Wang, Hanli Wang, and Kaisheng Xu. “Categorizing concepts with basic level for vision-to-language”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4962–4970 (page 16).

[WY22] Rui Wang and Rose Yu. *Physics-Guided Deep Learning for Dynamical Systems: A Survey*. arXiv:2107.01272 [cs]. Mar. 2022 (pages 4, 5, 7, 8, 13, 14, 48).

[Yin+19] Changchang Yin, Rongjian Zhao, Buyue Qian, Xin Lv, and Ping Zhang. "Domain Knowledge Guided Deep Learning with Electronic Health Records". In: *2019 IEEE International Conference on Data Mining (ICDM)*. ISSN: 2374-8486. Nov. 2019, pp. 738–747 (pages 16, 63).

[Yin+21] Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. "Augmenting physical models with deep networks for complex dynamics forecasting". en. In: *Journal of Statistical Mechanics: Theory and Experiment* 2021.12 (Dec. 2021), p. 124012. ISSN: 1742-5468 (page 13).

[YR21] Jianyi Yang and Shaolei Ren. *A Quantitative Perspective on Values of Domain Knowledge for Machine Learning*. arXiv:2011.08450 [cs]. 2021 (page 8).

[Yu07] Ting Yu. "Incorporating prior domain knowledge into inductive machine learning: its implementation in contemporary capital markets". PhD Thesis. 2007 (pages 7, 8).

[Zen+17] Chunqiu Zeng, Wubai Zhou, Tao Li, Larisa Shwartz, and Genady Ya Grabarnik. "Knowledge Guided Hierarchical Multi-Label Classification Over Ticket Data". In: *IEEE Transactions on Network and Service Management* 14.2 (June 2017), pp. 246–260. ISSN: 1932-4537 (page 16).

[Zha+17] Yongzhi Zhang, Rui Xiong, Hongwen He, and Zhiru Liu. "A LSTM-RNN method for the lithium-ion battery remaining useful life prediction". In: *2017 Prognostics and System Health Management Conference*. IEEE. 2017 (page 28).

[Zhe+21] Guanjie Zheng, Chang Liu, Hua Wei, Porter Jenkins, Chacha Chen, Tao Wen, and Zhenhui Li. "Knowledge-based Residual Learning." In: *IJCAI*. 2021 (pages 6, 31).

[ZLS20] Ruiyang Zhang, Yang Liu, and Hao Sun. "Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling". In: *Engineering Structures* (2020) (pages 22, 23).

[ZW19] Yun Zhou and Peichao Wang. "An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence". In: *Computers & Security* 82 (2019), pp. 261–269 (page 6).