
Semantic-guided Predictive Modeling and Relational Learning within Industrial Knowledge Graphs

Martin Ringsquandl



München 2019

Semantic-guided Predictive Modeling and Relational Learning within Industrial Knowledge Graphs

Martin Ringsquandl

Dissertation

an der Fakultät für Mathematik, Informatik und Statistik der
Ludwig-Maximilians-Universität München

vorgelegt von
Martin Ringsquandl
aus Rosenheim

München, den 21.02.2019

Erstgutachter:	Prof. Dr. Peer Kröger
Zweitgutachter	Prof. Dr. Pascal Hitzler
Drittgutachter:	Prof. Dr. Evgeny Kharlamov
Tag der mündlichen Prüfung:	25.09.2019

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. .5.)

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Ringsquandl, Martin

Name, Vorname

Rosenheim, 30.12.2019

Ort, Datum

Ringsquandl, Martin

Unterschrift Doktorand/in

Formular 3.2

Contents

Abstract	iv
Zusammenfassung	vi
1 Introduction	1
1.1 Industrial Knowledge Graphs	1
1.2 Operational Data and Industrial KGs	5
1.3 Research Topics in this Thesis	6
1.4 Outline	8
2 Semantic Data Models for Industrial Knowledge Graphs	10
2.1 Knowledge Graphs and the Semantic Web	11
2.1.1 Knowledge Representation	13
2.1.1.1 The Resource Description Framework	13
2.1.1.2 Ontologies and Logical Reasoning	15
2.1.2 Ontology-based Data Access	17
2.2 Existing Semantic Modeling Approaches for Industrial Knowledge Graphs .	18
2.3 Contribution	20
3 Domain knowledge-driven Feature Selection for Predictive Models within Industrial Knowledge Graphs	22
3.1 Background on knowledge-based Predictive Modeling	23
3.1.1 Classification and Regression Models	23
3.1.2 Feature Selection	25
3.1.3 Graph-structured Data	28
3.1.4 Feature Selection and Extraction for RDF Data	29
3.2 Existing Domain Knowledge-driven Feature Selection Approaches	32
3.3 Contribution	34
4 Completion of Missing Facts in Industrial Knowledge Graphs	36
4.1 Statistical Relational Learning	37
4.1.1 Statistical Relational Learning for KG Completion	39

4.2	Existing Background-enhanced KG Completion Approaches	42
4.2.1	Joint Embedding Models	42
4.2.2	Graph Convolutional Networks	43
4.3	Contribution	43
5	Conclusion	46
5.1	Summary	46
5.2	Outlook	47
	List of Figures	49
	List of Tables	50
	Bibliography	51
	A Publications	

Acknowledgements

During the years I have been working on this thesis, I had the privilege to get support from several amazing people. First of all, I want to thank my supervisor at Ludwig-Maximilians Universität, Peer Kröger, who guided me along the academic journey. Our talks have always been helpful, whether I needed feedback or just some calming words to sort things out. I'm also very grateful for the support of Steffen Lamparter and Raffaello Lepratti, who not only initialized this Ph.D. project, but also provided brilliant advice throughout every of its stages based on their academic and business experience. Further I would also like to thank Evgeny Kharlamov and Daria Stepanova for their tremendous help in sharpening my research and their contributions to our publications. It has been a pleasure getting all the work done towards the submission deadlines with your great expertise. I am also thankful for the discussions with colleagues from the SMR team at Siemens, especially Marcel Hildebrandt, Mitchell Joblin, and Thomas Hubauer. Equally important to me personally and to the shape of this thesis have been the ethical, philosophical, and musically-flavored talks with Hans Leonhardt, who has been a role model and a muse at the same time. I cannot thank you enough.

A special thank goes to my parents. This work would not have been possible without your support.

Finally, my dearest Muffin, you are the best inspiration I can imagine.

Abstract

The ubiquitous availability of data in today's manufacturing environments, mainly driven by the extended usage of software and built-in sensing capabilities in automation systems, enables companies to embrace more advanced predictive modeling and analysis in order to optimize processes and usage of equipment. While the potential insight gained from such analysis is high, it often remains untapped, since integration and analysis of data silos from different production domains requires high manual effort and is therefore not economic. Addressing these challenges, digital representations of production equipment, so-called digital twins, have emerged leading the way to semantic interoperability across systems in different domains. From a data modeling point of view, digital twins can be seen as industrial knowledge graphs, which are used as semantic backbone of manufacturing software systems and data analytics. Due to the prevalent historically grown and scattered manufacturing software system landscape that is comprising of numerous proprietary information models, data sources are highly heterogeneous. Therefore, there is an increasing need for semi-automatic support in data modeling, enabling end-user engineers to model their domain and maintain a unified semantic knowledge graph across the company. Once the data modeling and integration is done, further challenges arise, since there has been little research on how knowledge graphs can contribute to the simplification and abstraction of statistical analysis and predictive modeling, especially in manufacturing.

In this thesis, new approaches for modeling and maintaining industrial knowledge graphs with focus on the application of statistical models are presented. First, concerning data modeling, we discuss requirements from several existing standard information models and analytic use cases in the manufacturing and automation system domains and derive a fragment of the OWL 2 language that is expressive enough to cover the required semantics for a broad range of use cases. The prototypical implementation en-

ables domain end-users, i.e. engineers, to extend the basis ontology model with intuitive semantics. Furthermore it supports efficient reasoning and constraint checking via translation to rule-based representations. Based on these models, we propose an architecture for the end-user facilitated application of statistical models using ontological concepts and ontology-based data access paradigms.

In addition to that we present an approach for domain knowledge-driven preparation of predictive models in terms of feature selection and show how schema-level reasoning in the OWL 2 language can be employed for this task within knowledge graphs of industrial automation systems. A production cycle time prediction model in an example application scenario serves as a proof of concept and demonstrates that axiomatized domain knowledge about features can give competitive performance compared to purely data-driven ones. In the case of high-dimensional data with small sample size, we show that graph kernels of domain ontologies can provide additional information on the degree of variable dependence. Furthermore, a special application of feature selection in graph-structured data is presented and we develop a method that allows to incorporate domain constraints derived from meta-paths in knowledge graphs in a branch-and-bound pattern enumeration algorithm.

Lastly, we discuss maintenance of facts in large-scale industrial knowledge graphs focused on latent variable models for the automated population and completion of missing facts. State-of-the art approaches can not deal with time-series data in form of events that naturally occur in industrial applications. Therefore we present an extension of learning knowledge graph embeddings in conjunction with data in form of event logs. Finally, we design several use case scenarios of missing information and evaluate our embedding approach on data coming from a real-world factory environment.

We draw the conclusion that industrial knowledge graphs are a powerful tool that can be used by end-users in the manufacturing domain for data modeling and model validation. They are especially suitable in terms of the facilitated application of statistical models in conjunction with background domain knowledge by providing information about features upfront. Furthermore, relational learning approaches showed great potential to semi-automatically infer missing facts and provide recommendations to production operators on how to keep stored facts in synch with the real world.

Zusammenfassung

Die umfassende Verfügbarkeit von Daten in Produktionsbetrieben, hauptsächlich getrieben durch den erhöhten Einsatz von Sensorik und entsprechender Software in industriellen Automatisierungssystemen, ermöglicht es Unternehmen vermehrt auf prädiktive statistische Modellierung zu setzen, um damit ihre Prozesse und die Nutzung von Anlagen zu optimieren. Das große Potential von aus diesen Analysen zu gewinnenden Erkenntnissen bleibt jedoch weitgehend unerschlossen, da die Integration und Analyse von Datensilos aus verschiedenen Produktions-Domänen mit hohem manuellen Aufwand verbunden und damit nicht ökonomisch ist. Mit dem Ziel die semantische Interoperabilität zwischen Systemen zu erhöhen, hat sich der Trend hin zu digitalen Repräsentationen von (Teil-)Anlagen, den sogenannten digitalen Zwillingen, etabliert. Aus Sicht der Datenmodellierung können digital Zwillinge als industrielle Wissensgraphen angesehen werden, welche damit als umfassendes semantisches Fundament der Produktionsoftware und Analyse fungieren. Mit der vorherrschenden Vielzahl an Datenquellen und bestehenden Informationsmodellen in Produktionssystemen herrscht allerdings noch großer Bedarf an semi-automatisierter Unterstützung für Endanwender bei der semantischen Modellierung ihrer Produktionsdomäne und bei der unternehmensweiten Wartung von Wissensgraphen. Neben der Modellierung und Datenintegration ergeben sich neue Herausforderungen, insofern sich bisher wenige Untersuchungen damit beschäftigten, inwiefern Wissensgraphen die Anwendung von statistischen Analysen und prädiktiven Modellen vereinfachen und abstrahieren können, speziell in der Produktionsdomäne.

In dieser Arbeit stellen wir neue Ansätze zur Modellierung und Wartung von industriellen Wissensgraphen mit dem Fokus auf der Anwendung von prädiktiven Modellen vor. Als erstes wird das Problem der Datenmodellierung betrachtet, indem Anforderungen aus verschiedenen standardisierten Informationsmodellen in Produktionsumgebungen und Automatisierungssystemen gesammelt werden, um daraufhin ein Fragment der OWL

2 Ontologiesprache festzulegen, welches die Expressivität der geforderten Semantik für ein breites Spektrum an Anwendungsfällen abdeckt. Eine prototypische Systemimplementierung erlaubt es Endanwendern, die Basis-Modelle der Ontologie semantisch intuitiv zu erweitern. Desweiteren unterstützt es effizientes Reasoning und Constraint-Überprüfung durch Übersetzung in eine regelbasierte Repräsentation. Auf Basis dieser Modelle wird eine allgemeine Architektur skizziert, die eine erleichterte Anwendung von prädiktiven Modellen unter Benutzung von ontologischen Konzepten und dem Paradigma des ontologiebasierten Datenbankzugriffes erlaubt.

Anschließend wird ein Ansatz zur Domänenwissen-gestützten Vorbereitung von statistischen Modellen im Hinblick auf die Auswahl von Variablen vorgestellt. Es wird gezeigt, wie OWL 2 Reasoning auf Schemaebene für diesen Zweck in industriellen Wissensgraphen angewendet werden kann. Als Machbarkeitsnachweis dient ein Vorhersage-Modell von Produktionsdurchlaufzeiten in einem simulierten Beispielszenario, worin sich erweist, dass axiomatisiertes Domänenwissen über Variablen im Vergleich mit herkömmlichen datengetriebenen Ansätzen qualitativ ähnliche Ergebnisse liefern kann. Im Fall von hochdimensionalen Daten mit wenigen Beispielen zeigen wir, dass Graph-Kernel Methoden, angewandt auf Domänen-Ontologien, wertvolle Informationen zu Abhängigkeiten zwischen Variablen extrahieren können. Weiterhin wird eine spezielle Anwendung des Problems der Variablenauswahl in graph-strukturierten Daten gezeigt, für die eine Methode entwickelt wurde, welche die Eingliederung von Domänen-Constraints in Bezug auf einen Wissensgraphen in eine Branch-and-Bound Mustersuche integriert.

Darauf folgend wird das Problem der Wartung von Fakten in industriellen Wissensgraphen mit Hilfe von latenten Variablen Modellen zur automatischen Population und Vervollständigung von fehlenden Fakten untersucht. Der aktuelle Stand der Technik kann nicht mit Daten in Form von Event-Zeitreihen umgehen, weshalb hier eine Erweiterung von Repräsentations-Lernverfahren für Wissensgraphen in Verbindung mit Zeitreihen, Events im speziellen, vorgestellt wird. Abschließend wird dieses Lernverfahren in verschiedenen gezielten Anwendungsfällen von fehlenden Fakten an realen Daten aus einer Produktionsumgebung evaluiert.

Als Fazit kann festgestellt werden, dass industrielle Wissensgraphen ein mächtiges Werkzeug sind, das von Endanwendern zur Datenmodellierung und Überprüfung verwendet werden kann. Vor allem im Hinblick auf die erleichterte Anwendung von prädiktiven statistischen Modellen in Verbindung mit formalem Domänenwissen können Wis-

sensgraphen erfolgreich eingesetzt werden, indem sie Wissen über die Merkmale vorab zur Verfügung stellen. Weiterhin erweisen sich relationale Lernverfahren in Wissensgraphen als vielversprechend, um semi-automatisiert fehlende Fakten zu ergänzen und Vorschläge an Administratoren zu geben, wie sie die Wissensbasis mit der realen Umwelt synchronisieren können.

Chapter 1

Introduction

We can only see a short distance ahead, but we can see plenty there that needs to be done

Alan Turing

1.1 Industrial Knowledge Graphs

Due to the paradigm shift towards mass-customization, manufacturing companies are challenged by the need to meet high quality demands of customers, while at the same time higher flexibility of processes, equipment and people is required to enable fast introduction of new products [Sch14]. Enabling more flexibility comes with the price of entangling multitudinous inter-dependencies between production engineering and execution, including coherent changes of bill of processes, bill of materials and equipment configurations. Enterprise surveys have shown that this is a complex task that requires novel decision support and traditional manufacturing execution systems (MES) have not been designed to hold up to these new standards [JKSB14].

In order to bridge the gap between decision support software applications, i.e. MES, and physical processes, sensing and communication technologies are increasingly deployed on production equipment and products, such that it is possible to collect data throughout their life-cycle and reason about their current context [TCQ⁺18, LHSS12]. On a high-level, these developments should serve the “need of enterprises to enhance transparency

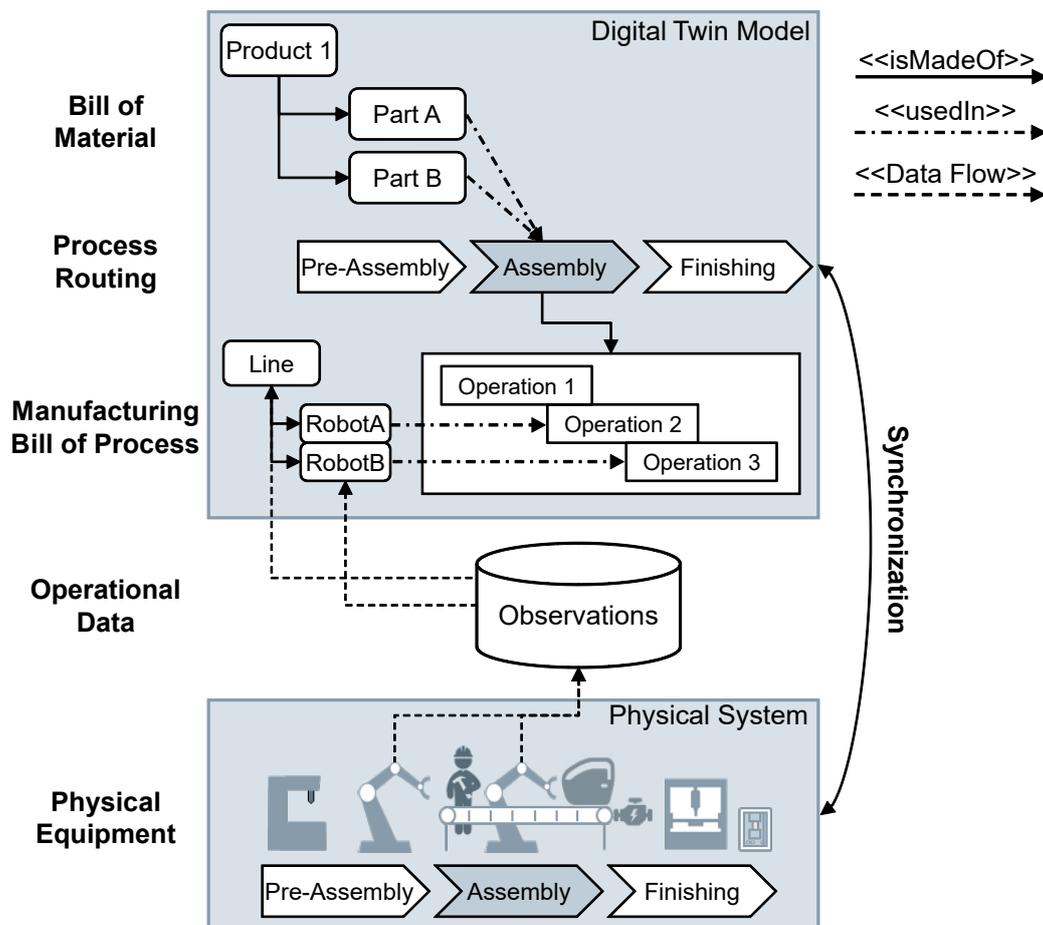


Figure 1.1: Digital Twin of an industrial automation system showing the three parts: the physical system, its digital twin model, and connected data.

and the derivation of short-term production control as well as process optimization based on near-real time data” [USL⁺17, p.114]. However, today, for example factory design engineers spend between 41% to 74% on manual data acquisition and preparation to design new variants [Paw14, p.357].

The aspiration of facilitated utilization of production data has led to an envisioned logical separation of real-world physical systems and digital representations – a concept referred to as the *digital twin* [Dat16]. Borrowing from its original definition in the product life-cycle management domain, a digital twin of an industrial automation system consists of three parts: the actual physical system, the virtual system (model), and connected data that ties the real and virtual worlds together [MRP⁺05]. By transferring these ideas to

the manufacturing and automation system domain, this can be conceptually visualized as shown in Figure 1.1. At the top of the figure, the digital twin model represents information about the product to be manufactured, exemplified here on a simple bill of material as partonomy. It further contains information about manufacturing processes routing and sequential relationships between processes, as well as the more detailed manufacturing bill of process which contains equipment models and their associated process operations. At the bottom of the figure, the actual physical automation system is depicted. The physical system generates operational data, i.e. sensor observations, which ties the physical and the model of the digital twin together.

From a technical point of view, the digital twin concept does not contribute to increased engineering flexibility or decision support. It is merely an analogy that summarizes the overall data integration and usage problems that the historically grown manufacturing and automation systems face. In this sense, the digital twin model can be seen as a conceptual data model that serves as a common ground in production data integration, i.e. the data model is a semantic representation of (fragments of) the digital twin model that is used to integrate operational data from the physical system.

An example of such a semantic data model is shown in Figure 1.2, reflecting an ontological view of the concepts in the digital twin model of Figure 1.1. As before, the vital concepts **Device**, **Process**, and **Material** are represented. An example of a specialized subclass of **Device** is given here as **SensingDevice**, which is further defined to have an existential restriction on the **observes** object property (relation) to an **InformationEntity**. We can further identify that the assembly process in the depicted automation system above should be an instance of the concept **AssemblyProcess**.

In most industry domains today, heterogeneous data models are prevalent and massive amounts of instance data is stored in disparate data sources and in a variety of formats, collected at high frequency, therefore matching every dimension of the definition of Big Data [JONK14]. The complexity of Big Data in industry has given rise to facilitated data integration and access based on semantic data models, which is an active research field in several industrial domains [KSÖ⁺14, KHJR⁺15]. However, in contrast to the above mentioned works on semantic models for facilitated industrial end-user data access and works concentrating on semantic interoperability between MES and other production software systems [SMS11], the focus of this thesis is the application of predictive models within semantically integrated production data. Thus, in accordance to the machine

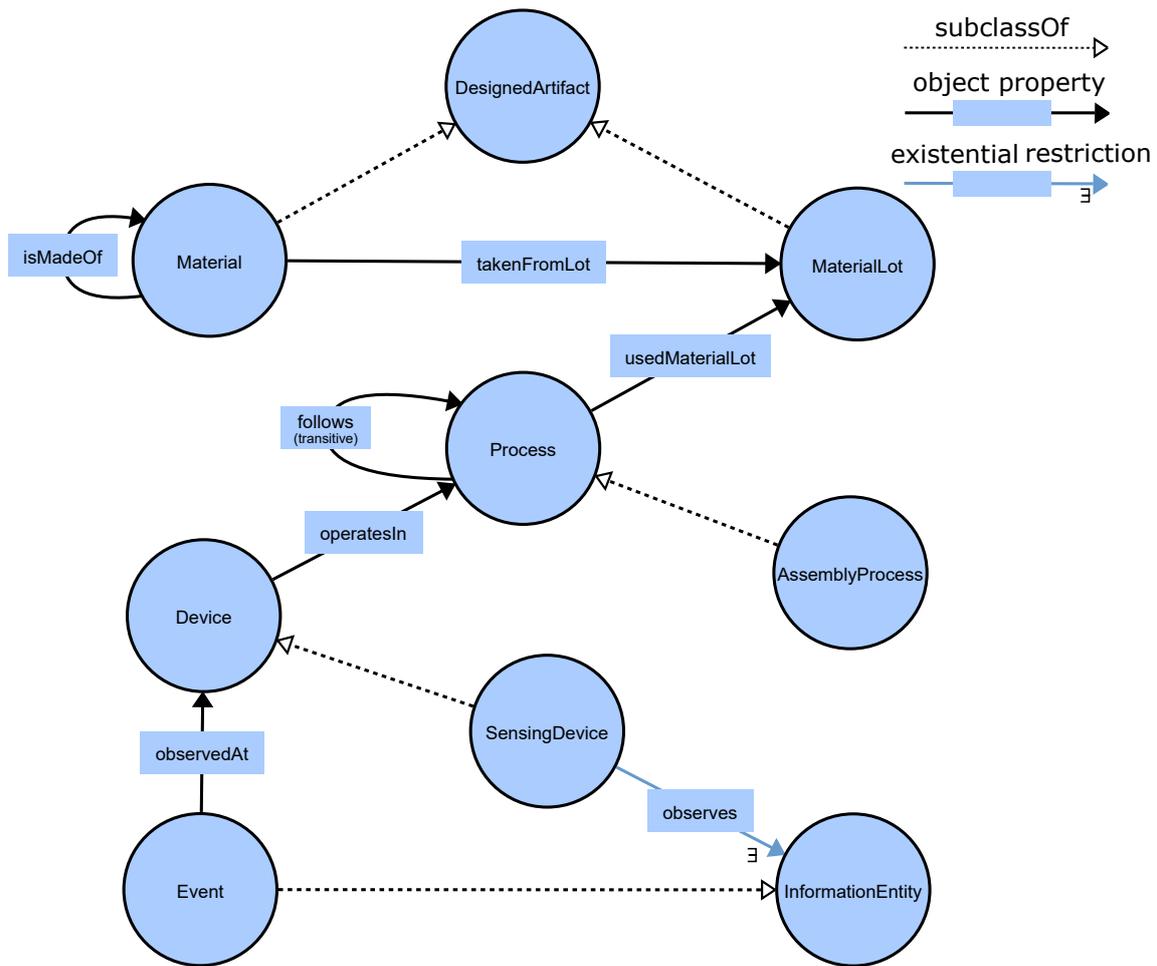


Figure 1.2: Excerpt of a semantic data model reflecting the concepts of the digital twin model in 1.1.

learning community, stressing the importance of the relational instance data, we will refer to semantically integrated production data in the digital twin model as *industrial knowledge graphs* (KGs). When a clear distinction is needed we will use the term semantic data model or *ontology* to explicitly refer to the schema only.

The Web Science¹ and Semantic Web Community have only recently picked up the direction of statistical methods on top of such industrial knowledge graphs as research field [PHGg17]. Further investigations towards the application of predictive models, e.g. machine learning, within industrial knowledge graphs is needed, especially in decision

¹<https://industrial-knowledge-graphs.github.io/2017/>

support scenarios that require to predict future behavior or give recommendations to production engineers based on time-series and operational data [Grö15].

1.2 Operational Data and Industrial KGs

A key characteristic that separates industrial knowledge graphs from general-purpose ones, such as DBPedia², is the availability of operational data that reflects the functional behavior of the underlying physical systems. This poses challenges on how operational data can be used in combination with KGs in terms of making predictions about the system's behavior as well as vice-versa inferring when the KG does not correctly reflect the physical system. These two points are the different directions of the synchronization problem, as indicated on Figure 1.1.

Towards the first point, there is a plethora of work on classical predictive models such as predictive maintenance, quality control, and production scheduling [WIT14, IOA09, LLBaK13]. However, the definition of predictive models in industry is challenging for data scientists, who typically lack the engineering knowledge and expertise of the production devices, processes, and products. To take away the need for specialized data scientists, domain-specific languages that allow domain experts to define analytic models have been studied [LNR14]. Although semantic data models are becoming widely adopted for data access and integration, there has still little research been done in considering how to further model concepts that facilitate predictive model preparation, execution and evaluation in industry. Rule-based approaches for device diagnostics have been studied that try to remove the dependence on data sources specific characteristics and therefore hide these details from the domain expert end-users [MKS⁺17]. However, this has not been done for more complex machine learning models and industrial KGs.

Concerning the second point, while text data is commonly used to populate and retract facts of general-purpose KGs, a detailed textual description of devices, processes, and products is usually not given in industry. There has been little work concerned towards the usage of operational production data to automatically extract or complete missing facts within industrial knowledge graphs. One such example is to extract physical dependence relations between devices given quality control data of production lines [BDMJ17]. These approaches are still in their infancy and not generic enough to be applied to different

²<https://wiki.dbpedia.org/>

datasets.

Furthermore, in large-scale general-purpose KGs, it has been shown that overall facts are usually highly incomplete. For example, the Freebase KG is missing the birthplace relation for about 71 % of all people entities [NMTG15]. This example shows that even simple facts that might be thought of as trivially retrievable can not be reliably managed with common KG construction techniques, whether manual curation or semi-automatic extraction from text. Exploiting statistical patterns to complete missing facts in general-purpose KGs has emerged as a big topic in machine learning research. These techniques, however, have not been sufficiently studied for industrial purposes, where the evolution of facts over time is much more significant and operational data is vital to understand the evolution of facts in the KG.

1.3 Research Topics in this Thesis

As mentioned in the above introduction, the focus of this thesis is to study issues related to predictive modeling in industrial knowledge graphs. This area is further divided into three concrete research topics.

Topic 1: Requirements analysis of conceptual information models for industrial knowledge graphs The first topic of this work concerns the study of information model requirements for industrial knowledge graphs with respect to:

Q.1 *What kind of conceptual modeling constructs need to be supported in the modeling language for typical industrial analytic use cases and how can these constructs be made available in an ontology-agnostic and domain-user friendly editor for industrial engineers?*

Q.2 *How can the conceptual model further be used to formulate statistical analysis within the ontology-based data access paradigm?*

For research question **Q.1**, the aim is to define ontological concepts for the widely adopted industrial standard information models in ANSI ISA-88/95 (IEC/ISO 62264). This is the foundation of the Business-To-Manufacturing-Markup-Language (B2MML) ³, and the de facto standard for the interface between Enterprise Resource Systems (ERP) and MES.

³<http://www.mesa.org/en/B2MML.asp>

End-user friendly management of these conceptual models is needed for practical application in the industrial domain, since concepts evolve over time. This requires to enable end-users (e.g. industrial engineers) to edit concepts and specify rules, such as data integrity, without having explicit training in formal description logics.

The next step in Q.2 is to evaluate how the ontology-based data access paradigm can not only facilitate access to heterogeneous data sources, but also enable further statistical analysis that goes beyond existing descriptive metrics, such as manufacturing key performance indicators. The semantic model makes cross-data source connections explicit and therefore opens the doors for more holistic statistical analysis across disparate production domains. We refer to this as the formulation of *advanced manufacturing analytics*, where the goal is to limit the manual effort needed for the preparation and execution of statistical analysis.

The aim is to assess these requirements for one concrete domain and use case scenario, since production environments are very diverse and a comprehensive analysis is not in the scope of this thesis.

Topic 2: Feature selection for predictive models within industrial knowledge graphs As mentioned above, the definition of predictive models in industry is challenging for data scientists, because they usually lack the domain expertise and specific insights into how industrial systems function. The second topic in this thesis is concerned with the study of how such domain knowledge can be captured and exploited, in order to ensure that predictive models use the right variables (features).

Q.4 *What kind of domain knowledge of industry experts can be captured in semantic data models to make feature selection more efficient w.r.t a large number of instance data points?*

Q.5 *Can domain knowledge-based feature selection prevent overfitting and therefore enhance accuracy of predictive models?*

We refer to the usage of ontological concept definitions for feature selection as *semantic-guided feature selection*. The concepts should allow to capture known physical engineering dependencies to make feature selection search more efficient. Additionally, it is of interest to study the effects on the statistical predictive power, i.e. enhance performance by preventing overfitting.

Furthermore, acceptance and trust of domain experts in predictive models can only be elevated with explainable selection of data. A semantic-guided feature selection should establish trust and explicability for domain experts by implicitly removing spurious correlations.

Topic 3: Completion of missing facts in industrial knowledge graphs The third topic is concerning the application of relational learning to the task of knowledge graph completion in industrial scenarios. The following research questions are posed:

- Q.6 How can relational learning be applied to the completion of missing facts in industrial knowledge graphs?*
- Q.7 What are possible extensions to existing representation learning models such that they can incorporate operational data into the completion task?*
- Q.8 Does operational data increase completion performance with regards to accuracy and can we identify different impact of operational data on different fragments of the industrial KGs?*

Since there have not been any applied studies in this area, the research questions in this topic are more fundamental. The ultimate goal is to phrase the synchronization of digital and physical twin as a knowledge graph completion problem. The hypothesis here is that traditional approaches that do not allow to incorporate operational data are not sufficient and need to be extended. To this regard, we aim to carry out machine learning experiments that study the effects on different parts of industrial knowledge graphs based on data from real-world production facilities.

1.4 Outline

The remainder of this thesis is structured as follows:

Chapter 2 presents an introduction to the information modeling research topic of this thesis. First, it provides the necessary background on semantic data models and knowledge graphs in general. Then an overview of related work in semantic industrial information modeling is presented. Finally, we outline the contribution of this thesis and refer to the respective publication.

Chapter 3 focuses on predictive modeling withing knowledge graphs. A fundamental overview of statistical predictive modeling is given with related tasks of feature selection, especially in graph-structured data. Then related work of domain knowledge-driven feature selection is presented and finally the contributions of this thesis are outlined.

Chapter 4 is concerned with the third research topic of knowledge graph completion in industrial settings. It introduces necessary background on statistical relational learning and recent representation learning approaches. Afterwards, a summary of related works in background-enhanced relational learning is presented and the chapter concludes with the contribution of this thesis and list the respective publications.

Chapter 5 presents concluding remarks in terms of a short summary that highlights again the contributions and how they fit into a bigger picture, as well as an outlook that states promising directions of future work.

Chapter 2

Semantic Data Models for Industrial Knowledge Graphs

Die Grenzen meiner Sprache
bedeuten die Grenzen meiner Welt

Ludwig Wittgenstein

Concerning the first challenge, defining shared vocabularies and concepts across several manufacturing domains, each connected to a plethora of existing standardized information models, is an elaborate task. It has been shown to require the involvement of domain experts as well as data engineers in data modeling [ALGM13]. Established industrial standards lack clearly defined semantics and are mostly not compatible to each other. For example, the OPC Foundation’s Unified Architecture (OPC UA), International Electrotechnical Commission standard (IEC 62541), for platform-independent communication and information modeling of automation systems, and AutomationML (IEC 62714) for describing production plants in engineering, both come with pre-defined basis information models that are not aligned [HS14]. Re-usage of such models is not well supported, since they often rely on proprietary formats, and further alignment of multiple information models is expensive as well as failure prone, i.e. leads to inconsistencies. While tooling support for these models is in its infancy, the Semantic Web technologies have come to a maturity that is ready to be used at industrial scale by applying state-of-the-art ontology-based data access (OBDA) modules. The work in this thesis consequently builds upon existing Semantic Web - World Wide Web Consortium (W3C) standards, such as the

Web Ontology Language, for the development of well-defined data models of industrial knowledge graphs.

2.1 Knowledge Graphs and the Semantic Web

Knowledge graphs recently have emerged as backbone of many applications, such as question answering [BGWB14], web search [DGH⁺14], and data integration [KSÖ⁺14]. In recent years the term *knowledge graph* has been adopted by the artificial intelligence research community, originally popularized by Google’s work on extending the Freebase project [Sin12], in order to emphasize the large-scale aspect and refraining from previous terms such as knowledge base or ontology.

There are several different notions of KGs and therefore no clear formal definition exists to this date. However, the most commonly accepted characteristics of a knowledge graph can be given in reference to [Pau15] as follows:

Definition 1 (Characteristics of Knowledge Graphs).

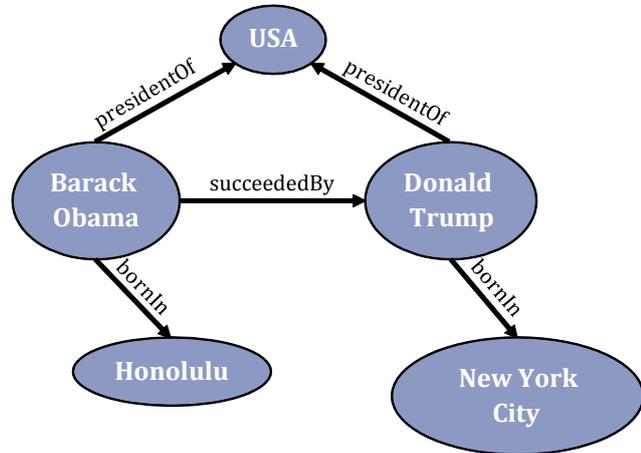
A knowledge graph

- i) mainly describes real world entities and their interrelations, organized in a graph,*
- ii) defines possible classes and relations of entities in a schema,*
- iii) allows for potentially interrelating arbitrary entities with each other and*
- iv) covers various topical domains*

From this set of characteristics it becomes evident that KGs are typically understood as databases that store facts about the world as instance data with an optional schema. Entities are the nodes in the graph and can represent real-world physical things, like persons and locations, or information objects, for example the theory of relativity. However, the most important feature of KGs are the directed edges, that is the relations between entities, which represent a certain type of relation. This means that in contrast to ordinary directed graphs, the typed relations define the multi-relational graph structure of KGs. For example person entities may be connected via the relation `friendOf`, whereas the `bornIn` relation may connect persons to locations. A more formal definition of these graphs is given in 3.1.3.



(a) Search result



(b) Graph Representation

Figure 2.1: Excerpt of a knowledge graph as part of search engine results

From an application perspective, KGs provide background knowledge that is both human and machine-readable. Most prominently, the Google search engine not only suggests websites, but also delivers additional information to the entities that are found in a given search query. For example, search for “Barack Obama”, would yield a set of facts, including birthplace and education, about the 44th president of the United States as shown in Figure 2.1a.

Both graphically and mathematically, these facts can be expressed as a multi-relational graph with entities as nodes and edges of different labels as the relations that connect them. An excerpt of the knowledge graph retrieved by the previous example query may look like the one in Figure 2.1b. Here, facts about birthplaces are denoted by connecting an edge labeled *bornIn* from the person entity node to its respective birthplace entity node.

In the subsequent sections we will introduce the data model behind KGs, see section 2.1.1.1, as well as some of the formal ontology languages for describing KG schemata in

section 2.1.1.2.

2.1.1 Knowledge Representation

Knowledge representation is a research sub-field of artificial intelligence with its roots in the development of software programs that can answer questions based on declarative facts about the world, rather than based on procedural computing instructions, therefore to some extent mimic the human reasoning process. Most prominently the so-called *expert systems* were designed to incorporate a knowledge base of facts including rules and an inference engine that is able to draw conclusions.

More advanced formalism to define facts plus allowing automated reasoning have been studied, such as semantic networks and frame languages [RN10].

The most recent developments in this area have been made towards the vision of a machine-readable web of data, the so-called *Semantic Web*. Pursuing this vision, description logics (DL) as knowledge representation language have been adopted for the development of an ontology language to describe resources on the web, which itself was heavily influenced by the earlier formalisms underlying frames and semantic net systems, realizing the trade-off between high expressive power and decidability (complexity) [BCM⁺03]. Before describing the formal background of DL and ontologies in 2.1.1.2, we first introduce the basic building block of the Semantic Web, its standard data model – the Resource Description Framework.

2.1.1.1 The Resource Description Framework

In order to enable the automatic exchange of information on the Web, i.e. unified machine-readable content, the W3C standardized a data model specification called the Resource Description Framework (RDF)¹. In contrast to other conceptual modeling approaches, such as the Unified Markup Language (UML) class diagrams, RDF is based on statements in form of triples (**subject**, **predicate**, **object**), which establish relationships between resources. These resources need to be uniquely identified by means of so-called Uniform Resource Identifiers (URIs). Every RDF entity is a resource and linking resources via triples results in a very loosely coupled structure that does not restrict the data to follow a specific schema, but rather allows the schema to evolve with the data. In simple terms,

¹<https://www.w3.org/TR/rdf11-mt/>

subject	predicate	object
dbr:Barack_Obama	rdf:type	dbo:Politician
dbr:Barack_Obama	dbo:bornIn	dbr:Honolulu

Table 2.1: RDF triples about entity `dbr:Barack_Obama`

an RDF data model is a directed labeled graph, where subjects and objects are nodes and predicates are labeled edges. There are special cases to this that do not fit this graph representation. It is allowed that predicates can also act as nodes. Also the notion of blank nodes is an exception, where an entity does not have to be identified by an actual URI and it serves mainly the purpose of reification, i.e. making statements about statements. For machine learning purposes these special characteristics are usually not considered [RP16a].

Besides the notion of triples, RDF also specifies some general pre-defined vocabulary, such as the *type* property, which is itself identified by the URI ² and is used to indicate the type of a resource. Full URIs are usually shortened by using prefixes to refer to certain namespaces. For example, one would specify the RDF vocabulary to resort in the namespace ³ and define a prefix `rdf` as shorthand notation. Hence, the RDF type property can be referred to by simply declaring `rdf:type`. Other popular vocabularies for sharing information have been developed by the Semantic Web community, such as friend-of-a-friend (FOAF) for social relationships, DBpedia for the semantic Wikipedia, or the Dublin Core for general document properties. Revisiting the presidential example, Table 2.1 lists two triple statements taken from DBpedia about the entity referred to by the URI `dbr:Barack_Obama`, specifying its type and birthplace. The prefixes `dbr` and `dbo` define namespaces for instance data and schema-level definitions respectively.

It is important to note that the URI is a symbolic identifier of the entity and does not have to be connected with the actual surface name (label) of the entity.

Another W3C standard building block is the RDF Schema (RDFS), which extends RDF to a larger vocabulary. One of the most important features of RDFS is the definition of classes, class-hierarchies (taxonomies), as well as definitions about relations, such as the domain and range, i.e. the types of the subject and object entities. In Table 2.2, RDFS predicates are used to define that `dbr:Politician` is a sub-class of `dbr:Person`, that is

²<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

³<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

subject	predicate	object
dbo:Politician	rdfs:subClassOf	dbo:Person
dbo:bornIn	rdfs:domain	dbo:Person
dbo:bornIn	rdfs:range	dbo:Place

Table 2.2: RDF triples using RDFS vocabulary for class hierarchy and domain and range definitions

every entity of type `dbr:Politician` is also of type `dbr:Person`, as well as the domain and range definition of the `dbo:bornIn` relation. The formal model behind RDFS allows basic logical inference based on a set of entailment rules, such as in the type inference in the sub-class example.

Besides logical inference RDF-based knowledge graphs also provide means for querying. The standard language for querying data in RDF-based knowledge graphs is SPARQL [HS13]. Due to the RDF graph structure, it is conceptually different from other declarative query languages, as for example SQL, which is based on relational algebra. Similar to the triple notation of RDF, SPARQL queries are built by specifying graph patterns in form of RDF triples, where variables can be specified as placeholder for subjects, predicates and objects. Since the RDF data model represents RDF and RDFS triples likewise in the same graph, a convenient property of SPARQL is the capability to intertwine queries on schema and instance-level, also referred to as *schema-aware querying*. As an example, the following query defines the variable `?P` to retrieve all instances which are of type `dbo:Politician` who were born in `dbr:Honolulu`:

```
SELECT ?P WHERE {?P rdf:type dbo:Politician . ?P dbo:bornIn dbr:Honolulu .}
```

, which would bind `dbr:Barack_Obama` as an answer to `?P`.

2.1.1.2 Ontologies and Logical Reasoning

Borrowing the term from philosophy, Gruber defines an ontology as an “explicit specification of a conceptualization” [Gru93], which means a formal, explicit way of describing a simplified model of a domain.

Although RDFS provides some support for the specification of ontologies based on RDF resources, the Web Ontology Language (OWL) family was introduced to allow more

expressive definitions building on top of the theories of description logics⁴, which are decidable fragments of first-order logic.

In terms of OWL an ontology is made of the following elements:

- concepts (classes)
- taxonomic relations between classes,
- datatype properties, i.e. attributes defined by a simple datatype,
- objects properties, i.e. named relations,
- individuals, i.e. instances of classes and properties,
- restrictions, i.e. constraints on properties.

Due to its DL foundations, in OWL terms the set of instances is referred to as *ABox* (assertions, denoted \mathcal{A}), whereas the set of concept definitions is called *TBox* (terminology, denoted \mathcal{T}). As it is more common outside of the Semantic Web community, also in this work, the term knowledge graph is by default used to refer to the instances only and the term ontology is used to explicitly refer to schema-level definitions. The same goes for the size of a knowledge graph, usually addressing the number of individuals and not the number of concepts.

Description logics are designed to allow tractable logical reasoning in ontologies under the so-called *open-world assumption* (OWA). Contrary to the closed-world semantics of classical databases, where it is assumed that data is complete, OWA inherently assumes incompleteness of data, that is the absence of a fact does not consequently mean that it is not true. This assumption is crucial in order to deal with linked open data that can be flexibly integrated, so one cannot assume that the knowledge in the KG is complete.

Given a knowledge base \mathcal{KB} as the tuple of a TBox and an ABox, $\mathcal{KB} = \langle \mathcal{T}, \mathcal{A} \rangle$, the standard reasoning tasks in DLs include:

Concept satisfiability: Given concept C : there exists an interpretation of \mathcal{KB} in which the concept C contains a non-empty set of individuals, hence $\mathcal{KB} \models C$. The unsatisfiable concept is denoted as \perp .

⁴*SROIQ* for OWL 2 <https://www.w3.org/TR/owl2-direct-semantics/>

Instance retrieval: Given concept C retrieve all of its individuals. A sub-task to this is the *instance-check*, given an individual a , answer true or false if it is assigned to C , denoted as $\mathcal{KB} \models C(a)$.

Other reasoning tasks, such as axiom entailment and classification (concept subsumption) can be reduced to concept satisfiability. For an in-depth discussion about DL reasoning, see [Rud11].

In general, DLs are very expressive and therefore can be complex, both for in terms of modeling and performing computations. Hence, the second version of the OWL family, named OWL 2, comprises different profiles with varying degree of expressiveness. For instance, the *OWL 2 QL* profile is the basis language for ontology-based data access (OBDA), see section 2.1.2, as it supports query answering that can be implemented by rewriting queries into relational query language (e.g. SQL) with efficient data complexity.

Another profile is *OWL 2 RL*, which allows axioms to be translated into rules, and therefore can be efficiently implemented in logic programming paradigms and rule systems based on Horn clauses, such as Datalog, which poses some restrictions on the composition of rule predicates in terms of recursion and negation. For example, simple type inference in OWL 2 RL according to its rule specification:

$$C(?x) \leftarrow D(?x) \wedge \text{SubClassOf}(D, C)$$

, where the left-hand side of the implication is called the *head* and the right-hand side is the *body* of the rule, in this case on variable $?x$.

This leads to rule-based reasoning that is polynomial in the size of the ontology [MLD⁺09]. Today, many reasoning systems incorporate a dedicated Datalog engine for rule-based inference in RDFS and fragments of OWL, such as Jena⁵ or the RDFox triple store [NPM⁺15].

2.1.2 Ontology-based Data Access

Due to the wide proliferation of relational databases as storage of industrial software systems, native RDF-based knowledge graphs have only been adopted sporadically in industrial applications. However, for heterogeneous relational data sources, the ontology-based data access paradigm has gained increasing attention in industry for the translation

⁵jena.apache.org/

of native data sources to a unified KG. The ontology in this case provides a convenient vocabulary for end-user queries and also allows some basic inference to give results even for incomplete data.

The main concept of OBDA is based on query rewriting of conjunctive queries over OWL 2 QL ontologies [RMKZ13]. The TBox \mathcal{T} serves as vocabulary abstraction to query the underlying data source's schema. Given a query \mathbf{q} over \mathcal{T} one obtains a first-order rewriting of \mathbf{q} and \mathcal{T} , denoted as \mathbf{q}' , that is the logically equivalent expansion of \mathbf{q} by entailment. Retrieving the set of individuals a from \mathcal{A} as the answers to \mathbf{q}' over $\langle \mathcal{T}, \mathcal{A} \rangle$, first the axioms of \mathcal{T} are applied to \mathcal{A} to retrieve the *canonical model*, then it is checked if all interpretations \mathcal{I} of $\langle \mathcal{T}, \mathcal{A} \rangle$ yield $\mathcal{I} \models \mathbf{q}'(a)$.

In practice, access to the data within the vocabulary of the ontology requires the definition of so-called *mappings*, which define a declarative specification of how concepts of the ontology translate to entities and attributes in the relational schema. Given a query in the language of the ontology (e.g. SPARQL), OBDA systems employs rewriting as explained above. In a second step, the rewritten query is translated using the defined mappings, in order to produce the data sources native query language, e.g. SQL in case of relational schemata – this is also called query *unfolding*.

A mapping is a set of rules of the form:

$$S(x) \leftarrow \mathbf{q}(x)$$

,where S is a concept or a property of the ontology and $\mathbf{q}(x)$ a SQL query. That is, the answers to the query instantiate concepts or properties in the ontology. The W3C standard language for the specification of mappings is R2RML [DSC12]. One of the major OBDA systems is the Ontop⁶ framework developed at the University of Bolzano.

2.2 Existing Semantic Modeling Approaches for Industrial Knowledge Graphs

Semantic models for automation systems share a considerable overlap with the semantic models in the Internet-of-Things (IoT) community. The focus of these works is on the device modeling end, typically concerned with automated device discovery and capability

⁶<https://ontop.inf.unibz.it/>

matchmaking [HBC17]. Hirmer et al. develop a modeling approach that enables automated device monitoring based on so-called device blueprints in smart environments. The authors build on top of the SensorML ontology⁷ and a publish-subscriber message broker architecture. Monitoring applications are automatically deployed based on the semantic device descriptions. However, they do not elaborate on the semantic expressiveness of their model and how to facilitate integration with existing industrial information modeling standards. They further propose the idea of a model editor that uses device blueprints (templates) to allow end-users to extend the ontology, but it is unclear how the users would interact with the semantic model, e.g. phrasing of axioms that could be used for data integrity checks.

Recently, Thuluva et al. [TDW⁺17a] introduced a layered model approach for automation systems, building on top of the W3C WoT Thing Description⁸. For the purpose of sensing property modeling they employ the Semantic Sensor Network⁹ and QUDT¹⁰. Their tooling support, however, is very limited. It allows end-users to create rules to discover devices. The actual semantic models are created by ontology experts using general-purpose ontology engineering tools, e.g. Protégé. Also the authors do not discuss how to align existing industry standard information models to their semantics.

Other approaches from the domain of MES have been concerned with building of ontologies that reflect the ISA-88/95 industrial information model standard for data exchange between production machines and the MES [SSF10]. This work is merely a terminological translation of the standard and lacks a detailed analysis of the expressiveness of the ontology. They describe the mapping of preconditions on data acquisition, however, using not ontological constructs, but `rdfs:comment` text attributes.

The work most closely related to the one in this thesis was done by Petersen et al. [PHGg17]. They have shown a prototypical RDF-based information model specifically for accessing equipment data in an MES-like application using SPARQL. Their approach follows a classical data integration methodology. The conceptual model was hand-crafted by analyzing SQL-dumps and queries, conducting expert interviews, and review of existing information models, without generalizing their insights. They then also follow the typical OBDA workflow in designing R2RML mappings for the Ontop framework to access the

⁷<https://www.opengeospatial.org/standards/sensorml>

⁸<https://www.w3.org/TR/wot-thing-description>

⁹<https://www.w3.org/TR/vocab-ssn>

¹⁰<http://www.qudt.org>

different data sources. Although the use case evaluation gives promising results, a deeper analysis of the needed expressiveness of the semantic models is missing, which is vital for data integrity checks.

In conclusion, all of the mentioned works above lack a further description how the domain knowledge in the semantic model can support the development of analytic (predictive) models.

2.3 Contribution

The works in this thesis present a modeling approach based on a fragment of the OWL 2 language for the schema-level modeling of industrial KGs, including standard concept hierarchies as well as the formulation of data integrity constraints that need to be addressed in common industrial information models. A prototypical system developed on top of the WebProtégé framework demonstrates how these modeling capabilities can be realized within an end-user-oriented model editor. The prototype contains an efficient constraint checking via translation of constraints to Datalog rules, which are injected into a state-of-the-art Datalog inference engine. Based on the modeling capabilities, the chapter further introduces an architecture for ontology-based data access in order to obtain semantic end-user access and preparation of predictive models on top of industrial domain-specific ontologies. These contributions are outlined in the following publications:

[KGJR⁺16] Evgeny Kharlamov, Bernardo Cuenca Grau, Ernesto Jiménez-Ruiz, Steffen Lamparter, Gulnar Mehdi, Martin Ringsquandl, Yavor Nenov, Stephan Grimm, Mikhail Roshchin, and Ian Horrocks. *Capturing industrial information models with ontologies and constraints*. In Proceedings of the 15th International Semantic Web Conference, pages 325–343, 2016

[RLL16] Martin Ringsquandl, Steffen Lamparter, and Raffaello Lepratti. *Graph-based predictions and recommendations in flexible manufacturing systems*. In Proceedings of the IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, pages 6937–6942, 2016

The modeling approach was developed by main author Evgeny Kharlamov published in *Capturing industrial information models with ontologies and constraints* [KGJR⁺16]. Martin Ringsquandl supported the requirements analysis considering expressiveness of

conceptual descriptions in the production domain, created a use case and an ontology with associated instance data and logical constraints. Ernesto Jimenez-Ruiz and Yavor Nenov were responsible for the prototype implementation. Gulnar Mehdi provided a gas turbine application case. Steffen Lamparter, Stephan Grimm, Mikhail Roshchin and Ian Horrocks gave feedback. In the second publication *Graph-based predictions and recommendations in flexible manufacturing systems* [RLL16], the architecture of ontology-based preparation for predictive models and the proof-of-concept implementation was done by Martin Ringsquandl. Steffen Lamparter and Raffaello Lepratti provided feedback on the use case.

Chapter 3

Domain knowledge-driven Feature Selection for Predictive Models within Industrial Knowledge Graphs

Le présent ne contient rien de plus
que le passé et ce qu'on trouve dans
l'effet était déjà dans la cause.

Henri Bergson

Besides sensing and communication capabilities of digital twins, the thereby introduced ubiquitous availability of data about the overall manufacturing operations enables manufacturing companies to embrace more advanced predictive modeling and analysis, in order to optimize processes and usage of equipment [LLBaK13]. As an example, in 2016 the Robert Bosch GmbH launched a public data science competition on the Internet platform Kaggle¹, challenging participants to predict internal failures of production lines based on thousands of measurements and tests collected for each component.

However analysis of such data is challenging without extensive domain background knowledge, since it not only is of great volume and variety, including time-series, graph-structured and semi-structured [JONK14], but also generated by engineered systems that exhibit known physical dependencies and constraints which are hard to uncover using

¹<https://www.kaggle.com/c/bosch-production-line-performance>

common predictive models. Revisiting the example, the aforementioned Kaggle competition was also featured in the 2016th IEEE Big Data conference and one of the biggest problems for the participating data scientists was the processing of this dataset into meaningful representations [MK16].

Although there is a clear motivation for using predictive modeling, such as machine learning, for predictive analysis, the lifecycle management of these models from first experiments to deployment is hard to manage, since each phase must be guided by domain expert knowledge to ensure proper preparation and execution [LNR14].

3.1 Background on knowledge-based Predictive Modeling

3.1.1 Classification and Regression Models

Statistical classification and regression are among the most common *supervised* learning tasks in the field of machine learning for the approximation of an unknown function over a datasets with existing label information. The goal is to find patterns in the data in order to predict the label of previously unseen data samples. Given a dataset $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^p$ is a p -dimensional feature vector of instance i and $y_i \in \mathcal{Y}$ is the true label of instance i , formally, a supervised model corresponds to a function $f : \mathbb{R}^p \rightarrow \mathcal{Y}$. The statistical inference task is to fit the model's parameters \mathbf{w} such that the approximation error is minimized. In case of a linear regression model, $\mathcal{Y} = \mathbb{R}$, and the learned function is of the form $\hat{y} = f(\mathbf{w}) = \mathbf{w}^\top \mathbf{x}$, where \mathbf{w} is the linear model's parameter vector. In ordinary classification y_i is a categorical variable, e.g. binary $y_i \in \{0, 1\}$, but further tasks extend this to non-overlapping multiple classes (multi-class classification) and multiple labels (multi-label classification). The multi-label classification setting, where the label of an instance can be any subset of a set of labels $y_i \in 2^{\mathcal{Y}}$, can be formulated as $k - 1$ distinct binary classification problems. (in case independence between labels is assumed)

In theory, function approximation is an optimization problem and the quality of the learned function approximation is typically measured by a certain loss or cost function,

e.g. mean-squared error (MSE) for regression:

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

Unfolding the mean-squared error for linear regression it is straightforward to solve this optimization problem by taking the partial derivative of the loss w.r.t. the model parameters:

$$\begin{aligned} \hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 \\ \hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmin}} (\mathbf{X}\mathbf{w} - \mathbf{y})^\top (\mathbf{X}\mathbf{w} - \mathbf{y}) \\ \frac{\partial}{\partial \mathbf{w}} &= 2(\mathbf{X}\mathbf{w} - \mathbf{y})^\top \mathbf{X} \\ &= 2\mathbf{X}^\top \mathbf{X}\mathbf{w} - 2\mathbf{X}^\top \mathbf{y} \end{aligned}$$

Setting the derivative to zero, one can solve for \mathbf{w} that minimize the loss:

$$\begin{aligned} \frac{\partial}{\partial \mathbf{w}} &= 0 \\ \mathbf{X}^\top \mathbf{X}\mathbf{w} &= \mathbf{X}^\top \mathbf{y} \\ \mathbf{w} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} \end{aligned}$$

In this simple case of ordinary least squares regression, the loss function is quadratic and there is exactly one global minimum and no further constraints need to be considered in finding the optimal parameters. When moving to a more complex family of functions, e.g. multi-layer perceptrons with non-linear activations that are the basis of neural networks, the optimization problem can no longer be solved in this straightforward way, because there may be several local minima. With the notion that a ‘good’ local minimum is still sufficient in most cases, fitting neural networks parameters is usually done with stochastic gradient descent algorithms.

The theoretical foundations of machine learning algorithms usually come with a probabilistic interpretation, such that the predicted outcome can be interpreted as a probability and therefore includes uncertainty by nature. The problem statement is formulated as

finding the parameter setting that maximizes the conditional probability, also called *likelihood function*, $P(y|\mathbf{x}; \mathbf{w})$. The optimal parameter values $\hat{\mathbf{w}}$ are then found according to the maximum likelihood estimate principle.

3.1.2 Feature Selection

Given a dataset of p random variables the task of feature selection for predictive models is to obtain a subset of these variables (features) that are given for each instance, which are then input to the predictive model. In the supervised setting, the predictive model is used to solve a classification or regression problem, that is, the chosen features represent the independent variables to predict a given dependent variable. The number of variables is also referred to as *dimensionality* of the dataset. Feature selection can be seen as search through the powerset consisting of all $2^p - 1$ subsets, which is again an optimization problem on its own.

The importance of feature selection has grown over the last decade due to the increasing availability of high-dimensional datasets. Since generalization of machine learning models over such data, i.e. the capability of a model to fit the data correctly without being to biased towards the given samples (training data), becomes exponentially harder as the number of variables increases [Dom12]. This is closely connected to the problem of *overfitting* in machine learning literature, as every learning algorithms come with a certain *bias* and *variance* towards its parameters. Highly biased algorithm need lots of evidence in the data to change their initial parameter choice, while high-variance algorithms very quickly fit to special peculiarities of datasets. Overfitting happens when the model parameters are over-specific to the characteristics of the training data and fail to generalize to unseen samples.

Another related challenge of high-dimensional datasets is the *curse of dimensionality*, which holds for a broad family of data distributions and distance measures [HKK⁺10]. Formally this effect has been shown that with increasing data dimensionality the proportional difference between the farthest-point distance d_{max} and the closest-point distance d_{min} converges to zero: on a dataset consisting of p features $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\} \in \mathbb{R}^{n \times p}$, when $p \rightarrow \infty$, $d_{min}(D) = \min_{i,j} dist(\mathbf{x}_i, \mathbf{x}_j)$, then we have $\frac{d_{max} - d_{min}}{d_{min}} = 0$. Since a stable distance or measure is vital for many predictive models, reducing high-dimensionality is an important challenge.

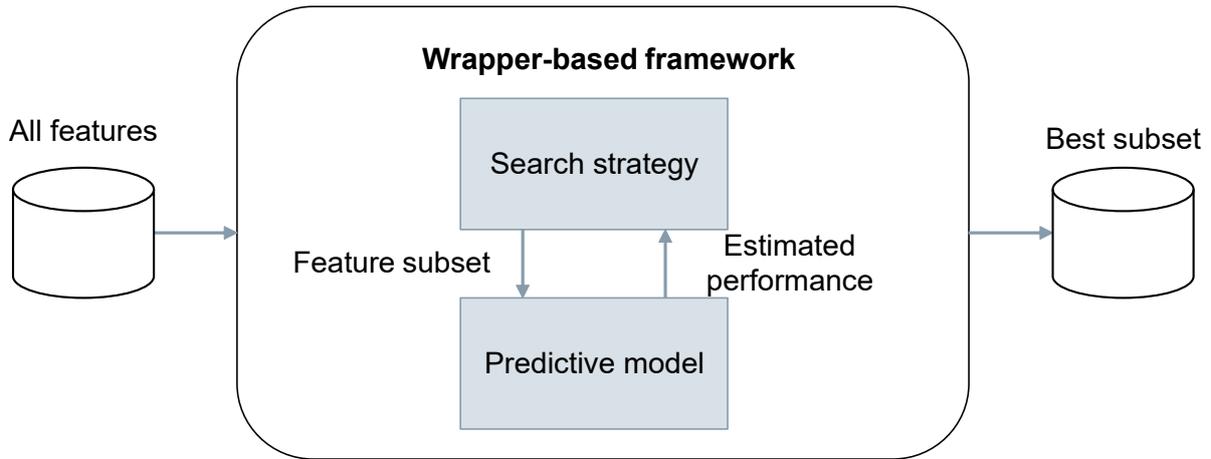


Figure 3.1: The wrapper-base feature selection framework is an iterative approach to select the best subset of features according to a search strategy

In order to avoid this problem, it is necessary to develop robust and yet cost-efficient predictive models by decreasing the data dimensionality beforehand. This reduction of input feature dimensionality is part of the well-known problem of feature selection. There are various feature selection methods that try to judge the usefulness of features towards the objective function of the predictive model. They can be grouped in to the following categories, cf [TAL14]: **filter**, **wrapper**, **embedded**.

The *wrapper*-based framework is shown in Figure 3.1. In this iterative approach, feature selection is performed by a search strategy (e.g. greedy) that selects a subset of all features. Then the predictive model is evaluated on this subset with respect to some performance metric, e.g. accuracy. Finally, the best subset according to the metric is chosen. This generic framework is agnostic of the predictive model and can therefore be applied universally. In contrast to embedded methods, the wrapper setting uses an external evaluation criterion that is not built into the predictive model itself. This is why wrapper approaches are computationally more expensive and the crucial part is to come up with an efficient search strategy that does not need exhaustive enumeration.

One form of *embedded* feature selection approaches can be seen as adding constraints on the model parameters, a technique called *regularization*. The L_1 -norm penalty on the model parameters induces a sparse feature representation. In linear regression models this is called the Lasso, or Laplacian prior. For very high variance models such as neural networks with multiple stacked layers of hidden parameters best practices to avoid over-

fitting are applied to the objective function by introducing a penalty on the magnitude of parameters. For example, given a parameter matrix \mathbf{W} the L_2 -norm $\|\mathbf{W}\|_2$ is added to the objective function, therefore favoring solutions with small magnitude of the parameters. To increase accuracy of prediction on unseen data, these regularization techniques have been applied to linear models as well, e.g. in Ridge or Lasso regression models. Such linear models are also used in cases when p is greater than the number of data points n ($p > n$). Here, further regularization schemes have been developed, especially focused on collinearity between predictor variables [ZH05]. Removing predictor variables from the model also has the benefit to increase interpretability of results.

Filter feature selection methods are based on characteristics of the features itself, such as their variance. This is closely related to *dimensionality reduction* approaches that transform the original feature space into a lower-dimensional one that still keeps most of the original variance in the dataset. This can be accomplished by combining highly correlated features, as in principal component analysis (PCA) for example, which computes the eigenvectors of the feature's covariance matrix over the whole dataset. Hence, most of the original variance can be retained by a subset of eigenvectors used as low-dimensional projection. The shortcoming of techniques like PCA are that they retain only linear correlations within the dataset. Furthermore, the low-dimensional projection can be hard to interpret in later explanation of the predictive model to domain experts.

In general, both dimensionality reduction as well as feature selection approaches have been targeted towards traditional tabular, i.e. propositionalized, data structures. However, the proliferation of complex-structured data has pushed the demand for feature selection approaches that directly deal with these representations, e.g. connected datasets [GH11], common single-relational graphs (networks) [AW06], and multi-relational graphs [GLT⁺16]. Due to the difficulty of applying predictive models to complex-structured data, the border between pure feature selection and feature extraction vanishes and the result of applying these methods to graph-structured data again results in a tabular data representation, thereby enabling efficient statistical inference, which will be introduced in Chapter 4.

3.1.3 Graph-structured Data

Many real-world datasets are naturally encoded in a graph structure, such as social networks, where a node corresponds to an individual and a edges between individuals represent some form of interaction between two individuals [HKP06]. Mathematically, the most general definition of a graph is the undirected graph. Formally the graph is denoted as a tuple $G = \langle V, E \rangle$ with vertex (or node set) V and edge set E . A graph consisting of n nodes defines $V = \{v_i\}_i^n$ and the corresponding edges as $(v_i, v_j) \in E$. Another way of representing graphs is given by the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where

$$a_{ij} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E \\ 0, & \text{otherwise} \end{cases}$$

For **undirected** graphs this matrix is *symmetric*, i.e. $(v_i, v_j) \implies (v_j, v_i)$. In case of a **weighted** undirected graph, edges have a weight attribute w attached $e_{ij} = (v_i, v_j, w_{ij})$. This means that the adjacency matrix is extended from binary values to any real value. In a statistical sense, edge weights are often used to denote the probability of an edge to exist with respect to some predictive model. This notion will be further discussed in section 4.1. **Directed** graphs are a special form of graphs which no longer posses the symmetry of edges, i.e. (v_i, v_j) expresses a directed connection from v_i to v_j which is different from (v_j, v_i) .

As mentioned before, the (subject, predicate, object) triples in RDF² data can be formulated as labeled directed multi-graph, where there are finite sets of vertex and edge labels L_V, L_E and two maps ℓ_V, ℓ_E assigning labels to vertices and edges respectively. Hence, $G = \langle V, E, L_V, L_E, \ell_V, \ell_E \rangle$. In this case E is a multiset of ordered pairs of vertices and $\ell_E : E \rightarrow \{L_E\}$, assigning a label to each ordered edge. The set of edge labels is also referred to as the set of edge or link types.

We further define the notion of a subgraph: H is a subgraph of G ($H \subseteq G$) if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.

A graph isomorphism between two graphs G and H is a bijection between the vertices of the two graphs $p : V_G \rightarrow V_H$, such that if $(v_i, v_j) \in E_G$ then $(p(v_i), p(v_j)) \in E_H$. On labeled graphs, p defines a re-labeling such that adjacency between labels is preserved.

²<https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>

3.1.4 Feature Selection and Extraction for RDF Data

For graphs consisting of only a single edge type where nodes have no labels or attributes attached, spectral graph theory and the concept of graph Laplacian provides a natural way to extract continuous representations out of the discrete graph representation. The Laplacian of a graph is defined as:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

where $\mathbf{D} = \text{diag}(\sum_{j,j \neq i} a_{ij})$ is the degree matrix. The eigendecomposition of the graph Laplacian is used in tasks like spectral clustering to extract continuous node representations as the eigenvectors of the non-zero eigenvalues (spectral embeddings). These representations allow to solve relaxed versions of the NP-hard graph partitioning problem.

Moving to more complex graph-structured data, RDF's labeled directed multi-graphs or so-called heterogeneous networks with different types of nodes, the spectral theory does no longer provide a closed framework for extracting node representations. Statistical classification models in graph-structured data for tasks such as node classification, need to consider variables attached to an individual node as well as the variables of the nodes that are connected to it. Therefore common feature selection techniques do not suffice to capture features in graph-structured data. Simplifications of feature selection like feature aggregation or propositionalisation may be performed. These techniques compute some form of aggregation function over node neighborhoods, therefore condensing graph information into tabular structure. For example in a social network analysis a feature aggregation approach could be to take the mean value over the age attribute of nodes in a neighborhood. It has been shown for RDF data that these methods suffer from substantial loss of information [RP14]. On the other hand, once the notion of first-order neighborhood is extended to also include attributes of higher order neighbors, it becomes evident that the feature selection search space grows exponentially.

To this regard, special graph kernel methods have been developed to directly encode similarity of local graph structures instead of manually engineered features. Kernel methods are a popular machine learning technique, particularly as part of support vector machine (SVM) classifiers, because kernels can be readily plugged into the SVM maximum-margin objective. Since the RDF data model represents data in form of a graph, graph

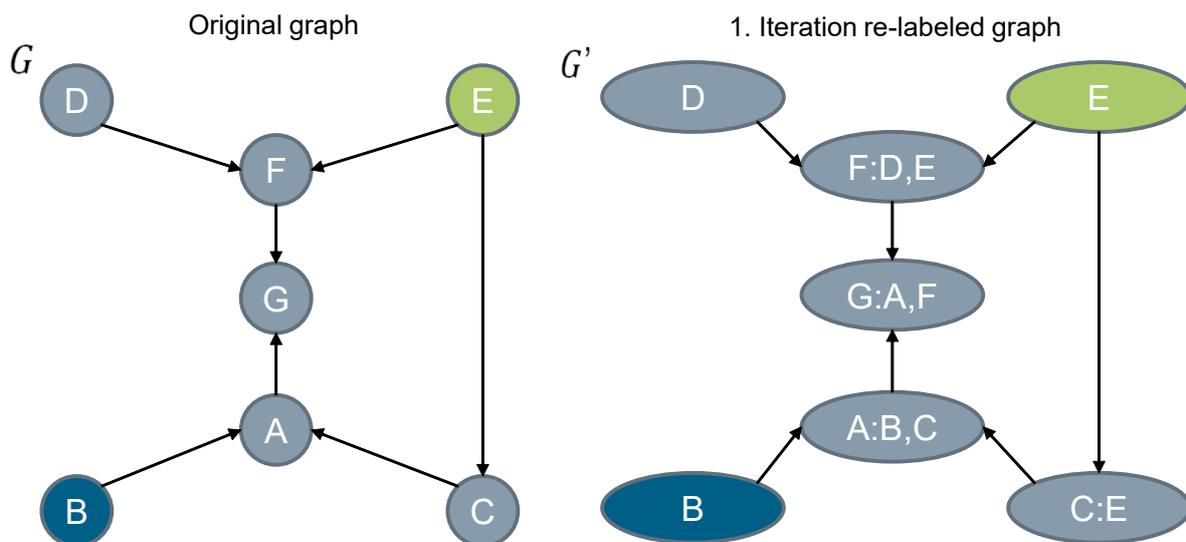


Figure 3.2: One iteration of Weisfeiler-Lehman re-labeling for graph isomorphism

kernels allow to directly apply machine learning algorithms on RDF data. A nice property of these methods is that the whole graph, including RDF schema definitions can be taken into account for similarity calculation.

Weisfeiler-Lehman Subtree graph kernel A graph kernel $\kappa : G \times G \rightarrow \mathbb{R}$ measures the similarity between two graphs. In order to apply them to encode features for a pair of nodes (entities), v_i, v_j , one first extracts the two subgraphs $G_i, G_j \subset G$ up to a certain depth of neighborhood and then applies an appropriate graph kernel on these two subgraphs. The Weisfeiler-Lehman subtree graph kernel calculates the number of matching subtrees based on the Weisfeiler-Lehman test for graph isomorphism. The isomorphism test performs node iterative node re-labeling, collecting the multi-set of labels of node neighbors, then computing a hash to compress the multi-set of labels into a new one. The new label is then used in the next iteration.

As an example of the Weisfeiler-Lehman subtree graph kernel, Figure 3.2 shows a simplified RDF graph G , without considering edge labels. In this example, the kernel between nodes v_E and v_B , node with label B and the node with label E shall be calculated. Applying one iteration of label propagation, we obtain G' , where the original label is concatenated with the neighboring ones.

It can be seen that the labels of nodes which do not have any incoming edges (B, D, E) do not change through iterations.

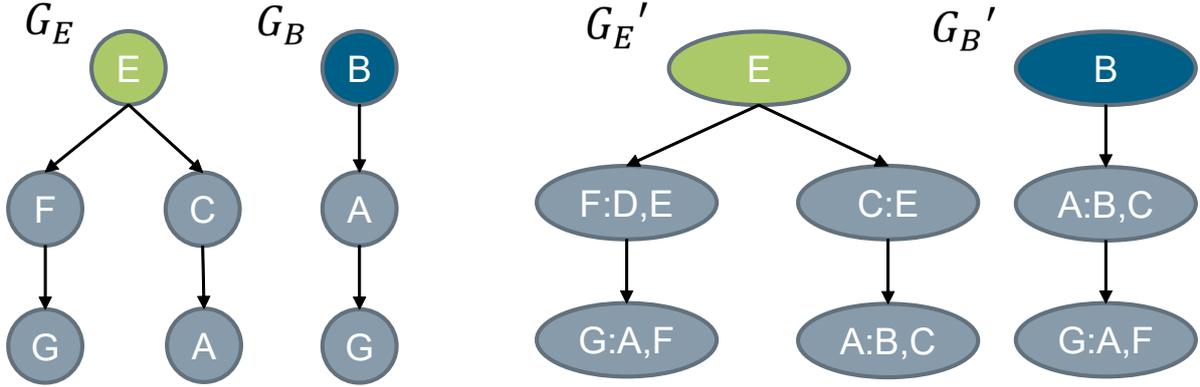


Figure 3.3: Subgraphs of nodes B and E up to depth 2 before and after one iteration of node re-labeling

As a next step, to compute the kernel for nodes E and B , Figure 3.3 presents the results for extracting the two subgraphs G_E and G_B and taking the first iteration re-labeling into account, resulting in G'_B and G'_E . The feature vector representations of G_E and G_B (x_E, x_B) are then given by counting the number of node labels in each iteration. Let $\sigma_l : G \rightarrow \mathbb{R}$ define the count of label l in graph G , the original node label feature space is: $\{\sigma_A, \sigma_B, \sigma_C, \sigma_E, \sigma_F, \sigma_G, \sigma_E\}$. After one iteration of re-labeling, the feature space is extended with: $\{\sigma_B, \sigma_{(F:DE)}, \sigma_{(C:E)}, \sigma_{(G:AF)}, \sigma_{(A:BC)}\}$. The features can then be regarded as the size of the intersection set of the iteratively extracted subtrees. Finally, the actual kernel computation is given via the dot product of the two feature vectors. The following equations show this computation, where $[\cdot]$ is the vector concatenation operation:

$$\begin{aligned}
 \mathbf{x}_B &= [(\sigma_A(G_B), \sigma_B(G_B), \sigma_C(G_B), \sigma_E(G_B), \sigma_F(G_B), \sigma_G(G_B), \sigma_E(G_B)); \\
 &\quad (\sigma_B(G'_B), \sigma_{(F:DE)}(G'_B), \sigma_{(C:E)}(G'_B), \sigma_{(G:AF)}(G'_B), \sigma_{(A:BC)}(G'_B))] \\
 \mathbf{x}_E &= [(\sigma_A(G_E), \sigma_B(G_E), \sigma_C(G_E), \sigma_E(G_E), \sigma_F(G_E), \sigma_G(G_E), \sigma_E(G_E)); \\
 &\quad (\sigma_B(G'_E), \sigma_{(F:DE)}(G'_E), \sigma_{(C:E)}(G'_E), \sigma_{(G:AF)}(G'_E), \sigma_{(A:BC)}(G'_E))] \\
 \\
 \mathbf{x}_B &= (1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1) \\
 \mathbf{x}_E &= (1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1) \\
 \kappa(G_B, G_E) &= \mathbf{x}_B^\top \mathbf{x}_E = 4
 \end{aligned}$$

It is known that the task of subgraph isomorphism is NP-complete, and therefore these kernel methods become computationally very expensive. Despite these limitations, the graph kernel approach has been extended to the more complex settings, e.g. including RDF literal nodes, ontology concept descriptions [BS07, LBR12, De 13]. The best performing variant of the RDF2Vec model also operates on walks generated at different iterations of the Weisfeiler-Lehman kernel, however, this is not applicable for large-scale RDF graphs [RP16a]. An analogy of the recently proposed graph convolutional networks, which will be discussed in 4.2.2, to graph kernels has been discovered, in the sense that the neural network is as the hash function to compress node labels of the subtree patterns.

Graph Pattern Mining For the tasks that operate on a database of graphs, $D = \{G_1, G_2, \dots, G_n\}$, such as graph classification, it has been shown that frequent graph patterns can be effectively used as feature vector representations of complete graphs. Given a set of frequent graph patterns (frequent subgraphs) mined from dataset D , g_1, g_2, \dots, g_p , a graph instance x_{G_i} can be represented as $\mathbf{x}_{G_i} = (f_{g_1}, f_{g_2}, \dots, f_{g_p})$, where $f_{g_j} = 1$ if $g_j \subseteq G_i$, 0 otherwise. While the pattern search space still grows exponentially with increasing subgraph size, pattern mining approaches can be more scalable compared to graph kernels when the graph instances are large, when efficient search space pruning is possible. For example, the gSpan algorithm exploits the anti-monotonicity criterion of the frequency of subgraphs [YH02]. Further approaches have extended this to mining of discriminative, rather than pure frequent, patterns, using branch-and-bound search strategies such as LEAP [YCHY08] and for multi-graphs gMGFL [WZZY14]. A general qualitative advantage of pattern-based approaches is that they can be more intuitively interpreted.

3.2 Existing Domain Knowledge-driven Feature Selection Approaches

While ontology-based data integration is a well-understood problem area, exploiting domain knowledge contained in ontologies for the application of predictive models has not been studied extensively thus far. To a limited degree, it has been argued that a formal understanding of the predictive modeling (or knowledge discovery) process is helpful for data and model preparation as well as execution. Few projects have been concerned with

the development of ontologies that formally describe the knowledge discovery process, hence allowing to follow best-practices and standard work-flows [KSFB14, PDS08]. Existing general-purpose ontologies such as *OntoDM* formally represent the structure of data mining investigations and datasets, which mainly serves to support the user in creating a proper data mining process.

Manually engineered features can be extracted from RDF graphs using dedicated SPARQL queries and transformation (propositionalization) strategies [RP14]. Propositionalization approaches of knowledge graphs have been built using domain-specific SPARQL queries as preprocessing, where the presence of a relation is transformed into a binary feature and numerical attributes are aggregated across the graph. This means that domain users are responsible for feature selection and generation in order to create meaningful representations ready for statistical analysis.

In the medical domain, background knowledge in form of rules has been introduced to select features from diagnostic text documents [BP01]. The authors argue that features need to be both *plausible* for a medical domain expert and *useful* for the predictive model. In their implementation they enrich the text documents with medical concept definitions and then use an association rule mining algorithm to extract frequent and high confidence rules as features.

Instead of feature selection, representation learning models, that represent knowledge graphs as latent variable models, can be used to extract structure out of RDF graphs. A more detailed discussion of these models is given in chapter 4. Usually a walk-based co-occurrence objective is employed to embed the graph-structure into a low-dimensional vector representation in an unsupervised manor. For example, inspired by the success of representation learning in natural language processing, the RDF2Vec model embeds a co-occurrence of random walks into a vector space [RP16a]. These approaches have proven to be very successful for the task of link prediction. The problem with these generic similarity-inducing approaches is that the original graph-structure is lost and results can be difficult to interpret for domain-experts, since they can no longer explain which components of the graph effectively contributed to the model's decision.

A general weakness of these approaches is that cannot directly deal with numeric or text attributes attached to nodes (entities). For example, a machine entity in the digital twin model may have thousands of sensor measurements attached. Computing correlations between numeric attributes falls in the domain of classical feature selection and

directly exploiting schema-level information for this task is promising, for example towards eliminating redundant features in the model’s feature space. Background knowledge in form of rules can be introduced into the learning objective, however, despite the fact that lots of knowledge about the data in semantic models is already captured at schema-level (ontology), logical reasoning has not been applied to feature selection thus far [RP16b].

In the graph pattern mining setting for graph classification, constraints as background knowledge have been introduced. For example simple label count constraints can be incorporated into the graph pattern mining, as they preserve the anti-monotonicity criterion. In [HKP06] further possible constraints are listed, such as value-sum constraints, where the sum of node label values must be smaller or higher than a given threshold. For mining frequent patterns in RDF data it has further been studied on how to introduce constraints given by the ontology, such as domain and range of object properties in order to reduce the pattern search space [CGWJ16].

3.3 Contribution

Concerned with the semantic guidance for predictive modeling the work in this thesis discusses how schema-level reasoning in the OWL 2 language can support feature selection for statistical classification and regression models within industrial KGs. A cycle time prediction model as an example application scenario serves as a proof of concept and demonstrates that axiomatized domain knowledge about features can give competitive performance compared to purely inductive ones. The ontological concepts allow instance retrieval reasoning to propagate prior knowledge about feature and label dependence through the feature space.

For graph-structured event data of a manufacturing system, an approach to incorporate knowledge graph path constraints in subgraph pattern mining is presented. Domain experts can specify common paths in the KG that are known to be dependent (must-link), and such that they are known to be independent (cannot-link) of the classification label. The mined patterns then serve as features for a system failure classification task. These contributions are contained in the following publications:

[RLBL15] Martin Ringsquandl, Steffen Lamparter, Sebastian-Philipp Brandt, and Raffaello Lepratti. *Semantic-guided Feature Selection for Industrial Automation Sys-*

tems. In Proceedings of the 14th International Semantic Web Conference, pages 225–240, 2015

[RLT⁺16] Martin Ringsquandl, Steffen Lamparter, Ingo Thon, Raffaello Lepratti, and Peer Kröger. *Knowledge Graph Constraints for Multi-label Graph Classification*. In Proceedings of the IEEE 16th International Conference on Data Mining Workshops (ICDMW), pages 121–127, 2016

In the contribution from *Semantic-guided Feature Selection for Industrial Automation Systems* [RLBL15] the idea and the concept for logical reasoning for feature selection as well as the implementation were developed by main author Martin Ringsquandl. Co-author Steffen Lamparter supported the formulation of the motivation, Sebastian Brandt answered questions about the complexity of the OWL reasoning. Thomas Hubauer and Raffaello Lepratti gave feedback on the industrial application in automation. In *Knowledge Graph Constraints for Multi-label Graph Classification* [RLT⁺16], the idea of applying graph mining to production event data, problem formulation, and constraint evaluation algorithm were developed by main author Martin Ringsquandl. Ingo Thon provided hints to related work. Steffen Lamparter and Raffaello Lepratti gave feedback to the application domain. Peer Kröger gave feedback to the technical concept.

Chapter 4

Completion of Missing Facts in Industrial Knowledge Graphs

There is nothing permanent except
change

Heraclitus

The final challenge addressed in this work is concerned with the constant evolution of industrial manufacturing systems over time. This happens, for example, when a plant's layout changes as new devices are deployed at the shop floor. Consequently, the information in the system's digital twin is no longer valid. However, the duality of physical and digital representation requires to continuously synchronize the digital twin model to its physical environment. As of today, human experts need to manually align device data models throughout multiple software systems (engineering tools, control systems, MES, etc.), which is not only failure prone, but also expensive. Therefore, there is a need for decision support giving model change recommendations to human operators based on historical data [KWH13]. This synchronization is vital for applications that rely on knowledge graphs as unified semantic model, e.g. equipment monitoring and quality control systems, since these expect all the facts in the KG to be true and consistent. In this work, the model synchronization problem is phrased in the same spirit as the completion of missing facts in KGs.

In contrast to general-purpose KGs, the problem of completing missing facts becomes even more challenging in highly-dynamic environments, e.g. industrial automation sys-

tems, which are subject to frequent changes of the physical reality. When facts about real-world entities are modified, the previously known facts in the KG cease to hold true and get outdated as time progresses. On the other hand, new facts are not automatically introduced in the KG and therefore missing. Additionally, the automated extraction of facts from text corpora, e.g. news documents, is usually not an option, since such documents are not available in industrial domains. The synchronization can again be shown conceptually using Figure 1.1 in the introduction, where the physical system (at the bottom) generates observations and these have to be integrated into the existing information models, the digital twin (at the top). Synchronization means that, essentially, the digital twin model needs to correctly reflect physical reality. In order to deal with the challenges of dynamic environments, where facts contained in KGs need to be synchronized with continuously changing physical processes, common approaches for statistical inference in KGs, e.g. relational learning methods, are not sufficient. Hence, there is a need to extend these approaches to also consider additional information (background) from the industrial environment. Operational data and in particular event logs of production machines reflect the environments dynamics. Therefore, it is desirable to consider these as sources for the inference of missing facts.

4.1 Statistical Relational Learning

In recent years, a machine learning paradigm for multi-relational labeled graphs, referred to as *relational learning*, has gained popularity. It aims to fit statistical models to relational data, such as knowledge graphs or relational databases. In this chapter we define a KG as a set of triples $\mathcal{K} = \{\langle h, r, t \rangle\}$, with a set of entities \mathcal{E} ($h, t \in \mathcal{E}$) and a set of relation types (edge labels) \mathcal{R} ($r \in \mathcal{R}$). In RDF-terminology relations correspond to predicates, and as mentioned in the definition of a KG in 2.1 the set of triples correspond to object property assertions in the ABox. In contrast to the Semantic Web community, in relational learning literature, it is customary to refer to triples as (head, relation, tail). The notation in this thesis follows this convention.

One of the main tasks of relational learning is *link prediction*, i.e. the inference of missing or unobserved edges. Link prediction can be defined as given \mathcal{K} , a query relation type r and two entities h and t to calculate a score $f(h, r, t)$ that represents the likelihood of h being linked to t via relation r . The score is determined by some notion of proximity

or similarity. In single edge type networks this can be simple models such as calculating the length of the shortest path between two nodes.

Rule-based inference has been employed to predict links in KGs, where rules are either given by human experts or they are obtained by rule mining methods, such as the AMIE system [GT13] which employs a support-based search for frequent rules. As an example, a rule mining system might come up with the following rule (expressed as Horn clause) for the prediction of birthplaces based on parental residency:

$$\text{bornIn}(h, t) \leftarrow \text{hasParent}(h, z) \wedge \text{livesIn}(z, t)$$

However, link prediction in large-scale KGs is challenging due to two problems: incomplete and imperfect knowledge. For general-purpose KGs it is known that they are highly incomplete and may also contain contradicting facts [DGH⁺14]. For example, person entities having two distinct birthplaces), therefore contradicting the functional aspect of the *bornIn* relation. High-recall rules are difficult to mine with imperfect facts and maintaining KGs by applying only high-precision logical inference rules is not satisfactory [LC10]. To still enable inference in KGs, statistical relational learning approaches have been proposed that extend classical rule mining approaches and can account for uncertainty and imperfect knowledge. These approaches can be categorized by which kind of features of the graph are taken into account to model for link prediction: **observed** and **latent feature** models. Observed feature models can be seen as extensions to rule learning where weight parameters are attached to rules in order to reflect uncertainty. One of the first approaches to do this was the path-ranking system [LC10, LMC11] which follows the idea that links can be predicted by learning how to weight a set observed chains of facts (paths) in the local neighborhood of an entity. Paths P which start at a certain query entity h are then turned into binary features $f_{h,P(t)}$, and $f_{h,P(t)} = \text{true}$ if query entity h reaches target t via path P . In the original version logistic regression is then applied to learn a set of so-called experts that weight the path features for link prediction.

A major benefit of observed feature models is that they are easily interpretable and can capture local interactions very well.

Other kinds of approaches express the statistical model for link prediction in KGs in terms of latent (unobserved) variables. The recently emerging representation learning ap-

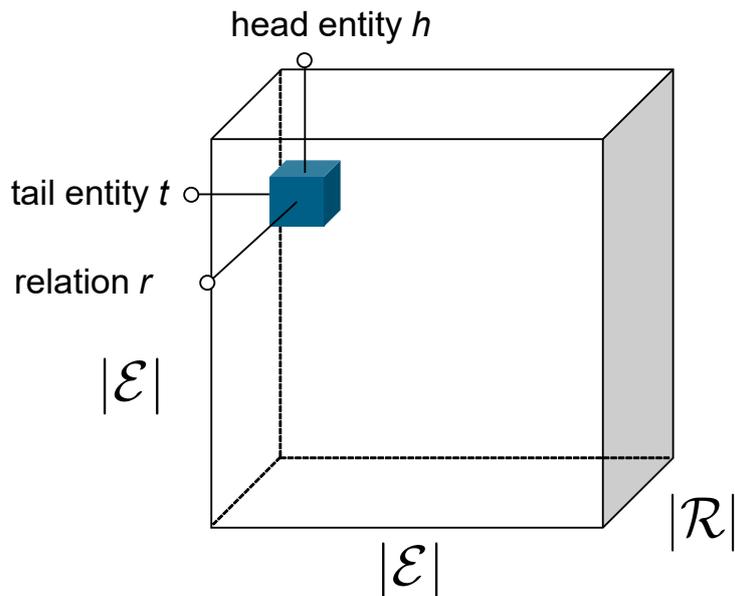


Figure 4.1: Knowledge Graph as a 3-dimensional tensor \mathcal{T}

proaches fall under this category [NMTG15]. It has been shown that these models outperform observed feature approaches by embedding patterns in the relational structure of the KG into low-dimensional vector representations of entities and relations. Different notions of these latent representations have emerged using factorization [NTK11, TDW⁺17b], neural networks [SCMN13, SW17, KW16], or translation-based approaches [BUWY13].

Since these approaches are not limited to perform link prediction for a given query relation, but employ a holistic framework that models the likelihood of the overall KG, we introduce the task of KG completion as an extension to link prediction.

4.1.1 Statistical Relational Learning for KG Completion

A knowledge graph can be defined in a probabilistic way, in a sense that facts do not have to be restricted to the binary domain of *true* or *unknown*. One can define facts to be modeled as random variables, such that they can take any value in the range from $[0, 1]$. More precisely, each fact or triple $\langle h, r, t \rangle$, with head entity h , relation r , and tail entity t , is defined to follow a Bernoulli distribution:

$$\begin{aligned} \mathcal{K}_{h,r,t} &\sim \text{Ber}(p_{h,r,t}) \\ p_{h,r,t} &\approx f(\theta_{\mathbf{h},\mathbf{r},\mathbf{t}}) \end{aligned}$$

where f is a scoring function that is proportional to the likelihood of the triple being true with respect to a set of parameters θ that either weight observed features or represent latent features of the respective triple. Given these preliminaries, one can define the task of statistical KG completion as an extension to link prediction that is concerned with inferring missing links in the overall knowledge graph.

Problem of KG Completion By representing the KG as an adjacency tensor \mathcal{T} , see Figure 4.1, formally:

$$\mathcal{T}_{h,r,t} = \begin{cases} 1, & \text{if } \langle h, r, t \rangle \in \mathcal{K} \\ 0, & \text{else} \end{cases}$$

Assuming triples are independent given their parameters θ , the KG completion can then be defined as computing the maximum likelihood estimate:

$$\hat{\mathcal{T}} = \operatorname{argmax}_{\mathcal{T}} \prod_{h \in \mathcal{E}} \prod_{r \in \mathcal{R}} \prod_{t \in \mathcal{E}} \operatorname{Ber}(\mathcal{T}_{h,r,t} | f(\theta_{\mathbf{h},\mathbf{r},\mathbf{t}}))$$

This can be seen as calculating a likelihood over all all possible worlds, as combinations of $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$. For realistic KGs this is a huge space, however, only a small fraction of triples are likely to be true. Hence, a relational learning model should exploit this sparseness and find a lower-dimensional representation.

One of the first KG completion approaches was the tensor factorization-based RESCAL model [NTK11]. It allows a concise definition of the KG completion problem using the notion of a KG as probabilistic tensor. RESCAL computes a bi-linear model as an approximation $\hat{\mathcal{T}}$ of the original tensor:

$$\mathcal{T}_{h,r,t}^{\hat{}} = f(\theta_{\mathbf{h},\mathbf{r},\mathbf{t}}) = \mathbf{h}^{\top} \mathbf{W}_r \mathbf{t}$$

where the parameters of the entities h , t and relations r occurring in \mathcal{K} are to be embedded in a low-dimensional, say d -dimensional, vector space as vectors $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ as the entity parameter matrix $\mathbf{W}_{\mathcal{E}}$. The relation parameters define the core tensor consisting of matrices $\mathbf{W}_r \in \mathbb{R}^{d \times d}$. Hence, relating to the above definitions, $\theta = \{\mathbf{W}_{\mathcal{E}}\} \cup \{\mathbf{W}_r\}_{r \in \mathcal{R}}$. These parameters are typically referred to as *latent representations* or simply *embeddings*. The approximated tensor now holds a probability-like score of all possible facts in the KG.

Another major family of models are the so-called vector translation-based models, such as TransE, which have been one of the forerunners in the representation learning domain. In the TransE model [BUWY13], given \mathcal{K} such f relies on distance or similarity between vectors of entities and relations. Intuitively, TransE follows the intuition that there is a linear relation for triples $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, hence the scoring function is defined as a dissimilarity measure (e.g. ℓ^2 -norm) $f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. This means that *translating* entity h with relation r should end up close to its tail entity t in the latent d -dimensional space. In order to prevent overfitting, the magnitudes of parameters in TransE are normalized after each mini-batch to unit-norm vectors, i.e. $\forall e \in \mathcal{E} : \|\mathbf{e}\| = 1$.

Interestingly, many of these representation learning model families have been shown to be effectively trained by using a ranking loss with the objective that true triples should be ranked before false/unknown ones according to the scoring function. This learning objective is formulated as minimizing a margin-based ranking loss:

$$\mathcal{L}_{\mathcal{K}} = \sum_{(h,r,t) \in \mathcal{K}} \sum_{(h',r,t') \in N} \max(0, \gamma + f(h, r, t) - f(h', r, t')), \quad (4.1)$$

where h, r, t are observed in \mathcal{K} and h', r, t' are sampled from N , which is a set of negative examples, i.e. presumably false triples not contained in \mathcal{K} . This loss is minimized when the true triples outscore the false ones by a constant margin γ . In practice the training is done using mini-batches of \mathcal{K} , instead of iterating over all triples, together with stochastic-gradient descent (SGD), since this introduces more variance in the embedding parameter updates and can prevent early convergence in local optima. RESCAL as well as its simpler variant DistMult can also be formulated within this negative sampling framework of weight updating, instead of closed-form Alternating-Least-Squares updates.

Note that the closed-form solution of RESCAL corresponds to taking the closed-world assumption (CWA), since every unobserved fact is considered to be false. In terms of negative sampling, different assumptions can be made, such as the local-closed world assumption (LCWA). In LCWA the, for a particular predicate subject pair h and r it is assumed that any $\langle h, r, - \rangle$ that is not observed in the KG is indeed false and can be used as negative sample [NMTG15].

4.2 Existing Background-enhanced KG Completion Approaches

Knowledge graph embedding approaches face issues when dealing with entities for which only few or no facts at all are present, since no meaningful representation can be calculated for these entities. This problem is generally known as data *sparsity*, or specifically *KG sparsity*.

4.2.1 Joint Embedding Models

We refer to background-enhanced or joint embedding models as representation learning models which not only take the KG as input, but also another data source that is in some way connected to the KG. Most commonly, the additional data source is text data that describes a subset of the entities in \mathcal{E} , such as entity Wikipedia pages. However, also models that include images or rules that are linked to entities or relations have been studied [MBXR18].

Due to its simple but effective objective, TransE has been used as a basis embedding model subject to several extensions of the original triple based embeddings. In terms of incorporating text corpora as additional source of information, joint embeddings have been proposed. The assumption here is that by also embedding co-occurrence between entities mentioned in the text, sparsity in the KG can be partially solved.

In the text-enhanced setting, the TEKE model is state-of-the-art [WL16]. It is an extension to translation-based approaches by using a joint embedding of the co-occurrence of entities in text-corpora. It includes $\mathbf{n}(h)$ as the weight-averaged neighborhood word vector embedding and then applies a linear combination $\hat{\mathbf{h}} = \mathbf{A}\mathbf{n}(h) + \mathbf{h}$ for final triple scoring. Furthermore, the relation embeddings are also modified with a merged neighborhood word embedding for pairs of entities h, t to $\hat{\mathbf{r}} = \mathbf{B}\mathbf{n}(h, t) + \mathbf{r}$. Further text-enhanced models are the semantic space projection (SSP) model [XHMZ17] which is a joint embedding approach that projects word embeddings into the TransE space and the convolutional neural network-based approach by Xie et al. [XLJ⁺16].

In the rules-enhanced setting, the KALE model [GWW⁺16] is a state-of-the-art approach. This model interprets the triple score as a truth value in fuzzy logic $I(h, r, t) = 1 - \frac{1}{3\sqrt{d}} \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. Then the model applies fuzzy t-norm operators to allow the translation

objective to incorporate manually defined rules of the form $\langle h, r_1, t \rangle \wedge \langle t, r_2, t' \rangle \rightarrow \langle h, r_3, t' \rangle$. Further works have incorporated ontological type-constraints and schema definitions to make negative sampling more efficient [KBT15, MDFE16].

The enhanced models outperform classic translation-based approaches typically in scenarios with KG sparseness. However, none of the existing background-enhanced models has been devoted to the study of time-series data such as event logs.

4.2.2 Graph Convolutional Networks

Recent developments in representation learning have brought up a new family of models, the so-called Graph Convolutional Networks (GCN) [KW16]. Intuitively, these models apply a smoothing across neighboring node representation in a message-passing fashion. The node representations are propagated by stacking convolution layers. An extension to multi-relational data has been proposed [SKB⁺18]:

$$\mathbf{h}^{(l+1)} = \sigma \left(\sum_{r \in \mathcal{R}} \sum_{\langle h, r, t \rangle \in \mathcal{K}} \frac{1}{C_{h,r}} \mathbf{w}_r^{(l)} \mathbf{t}^{(l)} + \mathbf{W}_0^{(l)} \mathbf{h}^{(l)} \right) \quad (4.2)$$

The number of stacked layers l can be seen as the maximal depth of adjacent nodes to which message-passing is done. The weight matrices \mathbf{W}_r for each relation perform the message-passing with a linear transformation of the respective neighboring nodes. The weights \mathbf{W}_0 can be seen like a skip connection to the node itself through the layers [HYL17]. One of the benefits of GCN models compared to translation-based and factorization-based ones is that they can fuse node features and graph structure into a single node representation. When node features are present, e.g. one-hot-encoded text data associated to entities in a KG, the initial node representations $\mathbf{h}^{(0)}$ contain these features. This makes them a potential candidate for the background-enhanced KG completion.

4.3 Contribution

The contribution in this thesis on representation learning models for the automated completion of industrial KGs is a realistic scenario definition (Q.6) of the digital twin model synchronization problem as KG completion. An extension of state-of-the-art learning

models in conjunction with time-series data in form of event logs is developed that takes time-dependent information into the joint embedding objective, which is in contrast to existing background-enhanced approaches that are devoted specifically to text corpora, images, or rules. Multiple novel joint embedding architectures are presented and evaluated this on a real-world industrial KG and factory data (Q.7). We show that this can significantly improve KG completion metrics and also deal with zero-shot learning, i.e. lift unseen event entities into the KG. Further a study on the impact of incorporating event data on different fragments of the KG is carried out (Q.8) demonstrating where machine event logs impact the completion of missing facts. These contributions are contained in the following publications:

[RLLK17] Martin Ringsquandl, Steffen Lamparter, Raffaello Lepratti, and Peer Kröger. *Knowledge Fusion of Manufacturing Operations Data using Representation Learning*. In Proceedings of the IFIP International Conference on Advances in Production Management Systems, pages 302–310, 2017

[RSL⁺17] Martin Ringsquandl, Daria Stepanova, Steffen Lamparter, Raffaello Lepratti, Evgeny Kharlamov, Ian Horrocks, and Peer Kröger. *On Event-Driven Knowledge Graph Completion in Digital Factories*. In Proceedings of the IEEE International Conference on Big Data, pages 1676–1681, 2017

[RSL⁺18] Martin Ringsquandl, Daria Stepanova, Steffen Lamparter, Raffaello Lepratti, Evgeny Kharlamov, Ian Horrocks, and Peer Kröger. *Event-enhanced Learning for Knowledge Graph Completion*. In Proceedings of the 15th Extended Semantic Web Conference (ESWC), pages 541–559, Heraklion, 2018

In the contribution *Knowledge Fusion of Manufacturing Operations Data using Representation Learning* [RLLK17] the idea to use representation learning to complete missing facts in semantically integrated production data, the model formulation, and prototypical implementation were done by main author Martin Ringsquandl. Steffen Lamparter, Raffaello Lepratti, and Peer Kröger gave feedback on the use case and the evaluation. In *On Event-Driven Knowledge Graph Completion in Digital Factories* [RSL⁺17] the event data analysis and the various evaluation scenarios as well as the expanded representation learning models in [RSL⁺18] were developed by Martin Ringsquandl. Daria Stepanova supported the problem formulation. Evgeny Kharlamov gave input on the motivation.

Steffen Lamparter, Raffaello Lepratti, Marcel Hildebrandt, Peer Kröger, and Ian Horrocks gave general feedback.

Chapter 5

Conclusion

5.1 Summary

Motivated by the need for flexibility and the upcoming trend of integrating sensor data to form digital representations of manufacturing systems, this thesis studied new approaches for modeling and maintaining digital twins in the form of industrial knowledge graphs with focus on the application of predictive models.

Analogous to knowledge graphs in the Semantic Web, industrial knowledge graphs need schema-level information models that provide domain concepts and relations, i.e. ontologies. The development of conceptual models that describe information flow in production plants as well as the overall manufacturing operations management is challenging, since well-established domain information models often lack defined semantics and are not compatible to each other. In chapter 2, this thesis presented a semantic modeling approach backed by a prototypical model editor that allows domain experts to intuitively extend industrial information models grounded in the OWL 2 ontology language.

The vast amounts of data generated by sensors and software systems further motivates the application of statistical models for predictive decision support, such as predicting when non-conformances are going to happen in production processes. However, several obstacles remain to effectively enable domain end-users to formulate statistical models with respect to their background knowledge about the physical dependencies of equipments and engineering processes. One of the most expertise-intensive tasks to this date is feature selection. While common feature selection methods rely on computation of correlation between dependent and independent variables, and therefore grow more

expensive alongside the number of samples in the data, reasoning about formalized background knowledge about variables, as presented in Chapter 3, is agnostic of the amount of samples and provides a static recommendation on which variables to select that only needs to be re-calculated when the semantic model changes. Similar considerations can be made for graph-structured data, where domain constraints can speed up selection of sub-graph patterns and improve precision and recall of graph classification.

Lastly, incompleteness of facts has been shown to be a challenging issue for common general-purpose knowledge graphs. This effect amplifies for large-scale industrial knowledge graph, because dynamics of manufacturing environments are high and documentation about changes is scarce. Therefore the task of maintaining all the facts in conjunction with field device observations that give indication about changes introduced to production processes can be seen as synchronizing digital twins with their physical counterpart. According to this intuition we formulated this synchronization as knowledge graph completion problem in Chapter 4 and extended state-of-the art representation learning models to the case of time-series data, discrete events in particular. We showed that jointly embedding events significantly increases KG completion performance, especially in case of missing links that are closely connected to the process flow. In the zero-shot learning setting we further demonstrated that this architecture is able to deal with event entities that have not been observed in the KG.

5.2 Outlook

The proliferation of digital twin models is introducing a new notion on how to exploit existing knowledge for the integration and alignment of production data sources as well as making prediction about future behavior. There is great potential in semi-automated extraction of semantic data models from machine data, including sensor and event logs. Machine learning-based recommendations could be a fruitful modeling support for domain experts. Related to the work in this thesis, a particular promising direction seems to be the conjunction of an automated data scientists approach within ontology-based data access and analytics systems, where ontologies are used as central analytics and data repository. For example, by deploying machine learning on top of OBDA, a coupling of the presented feature selection approach in Chapter 3 and ontology-based queries could reveal positive synergy effects, both in performance and maintainability.

Future work on the completion of digital twin knowledge graphs could be the incorporation of formal concept definitions as well as constraints, for example existential property or cardinality restrictions, similar to the rule-enhancement [WWG15]. Another interesting application of joint embeddings is to use them not for the KG completion task, but for event prediction or classification.

In general, digital twins are also perceived to incorporate detailed simulation models, e.g. finite-element simulation of machine parts. The combination of simulations and analytics based on data from the physical equipments opens new perspectives on many applications such as predictive maintenance, process optimizations, and real-time adaption of simulation models.

Lastly, there is an emerging research direction towards studying the evolution and generation of KGs over time in the machine learning community [TDWS17, YLY⁺18]. These models try to capture the generative process that can be used to predict structural changes in the future. This could be applied to an industrial setting, when taking snapshots of digital twin models as engineers modify them over time, thus trying to predict the next most likely modification to the model.

List of Figures

- 1.1 Digital Twin of an industrial automation system showing the three parts:
the physical system, its digital twin model, and connected data. 2
- 1.2 Excerpt of a semantic data model reflecting the concepts of the digital twin
model in 1.1. 4

- 2.1 Excerpt of a knowledge graph as part of search engine results 12

- 3.1 The wrapper-base feature selection framework is an iterative approach to
select the best subset of features according to a search strategy 26
- 3.2 One iteration of Weisfeiler-Lehman re-labeling for graph isomorphism . . . 30
- 3.3 Subgraphs of nodes B and E up to depth 2 before and after one iteration
of node re-labeling 31

- 4.1 Knowledge Graph as a 3-dimensional tensor \mathcal{T} 39

List of Tables

2.1	RDF triples about entity <code>dbr:Barack_Obama</code>	14
2.2	RDF triples using RDFS vocabulary for class hierarchy and domain and range definitions	15

Bibliography

- [ALGM13] Lisa Abele, Christoph Legat, Stephan Grimm, and Andreas Müller. *Ontology-based validation of plant models*. In Proceedings of the 11th IEEE International Conference on Industrial Informatics, pages 236–241. Ieee, jul 2013.
- [AW06] Ralitsa Angelova and Gerhard Weikum. *Graph-based text classification: learn from your neighbors*. In Proceedings of the 29th annual international ACM SIGIR conference on Research and Development in Information Retrieval, pages 485–492, 2006.
- [BCM⁺03] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge, second edition, 2003.
- [BDMJ17] Agniva Banerjee, Raka Dalal, Sudip Mittal, and Karuna Pande Joshi. *Generating Digital Twin models using Knowledge Graphs for Industrial Production Lines*. In Workshop on Industrial Knowledge Graphs, pages 1–5, 2017.
- [BGWB14] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. *A Semantic Matching Energy Function for Learning with Multi-relational Data*. Machine Learning, 94(2):233–259, 2014.
- [BP01] Catherine Blake and Wanda Pratt. *Better rules, fewer features: a semantic approach to selecting features from text*. In Proceedings of IEEE International Conference on Data Mining, pages 1–8, 2001.

- [BS07] Stephan Bloehdorn and York Sure. *Kernel methods for mining instance data in ontologies*. In Proceedings of the 6th International Semantic Web Conference, pages 58–71, 2007.
- [BUWY13] Antoine Bordes, Nicolas Usunier, Jason Weston, and Oksana Yakhnenko. *Translating Embeddings for Modeling Multi-Relational Data*. Advances in NIPS, 26:2787–2795, 2013.
- [CGWJ16] Yang Chen, Sean Goldberg, Daisy Zhe Wang, and Soumitra Siddharth Johri. *Ontological Pathfinding*. Proceedings of the 2016 International Conference on Management of Data, pages 835–846, 2016.
- [Dat16] Shoumen Datta. *Emergence of Digital Twins*. arXiv:1610.06467, 2016.
- [De 13] Gerben K. D. De Vries. *A fast approximation of the Weisfeiler-Lehman graph kernel for RDF data*. In Proceedings of ECML-PKDD’13, pages 606–621, 2013.
- [DGH⁺14] Xin Dong, Evgeniy Gabrilovich, Jeremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. *Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion*. In Proceedings of the 20th ACM SIGKDD’14, pages 601–610, 2014.
- [Dom12] Pedro Domingos. *A few useful things to know about machine learning*. Communications of the ACM, 55:78, 2012.
- [DSC12] Souripriya Das, Seema Sundara, and Richard Cyganiak. *RDB to RDF mapping language*. W3C Recommendation, World Wide Web Consortium, 2012. <http://www.w3.org/TR/r2rml/> (retrieved 2017-11-01).
- [GH11] Quanquan Gu and Jiawei Han. *Towards Feature Selection in Network*. Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM), pages 1175–1184, 2011.
- [GLT⁺16] Huan Gui, Jialu Liu, Fangbo Tao, Meng Jiang, Brandon Norick, and Jiawei Han. *Large-Scale Embedding Learning in Heterogeneous Event Data*. pages 907–912, 2016.

- [Grö15] Christoph Gröger. *Advanced Manufacturing Analytics: Datengetriebene Optimierung von Fertigungsprozessen*. Dissertation, Universität Stuttgart, 2015.
- [Gru93] Thomas R. Gruber. *A translation approach to portable ontology specifications*. Knowledge Acquisition, 5(2):199–220, 1993.
- [GT13] La Galárraga and Christina Teflioudi. *Amie: association rule mining under incomplete evidence in ontological knowledge bases*. Proceedings of the 22nd Conference on World Wide Web, pages 413–422, 2013.
- [GWW⁺16] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, Li Guo, National Computer, Network Emergency, and Response Technical. *Jointly Embedding Knowledge Graphs and Logical Rules*. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP-16), pages 192–202, 2016.
- [HBC17] Pascal Hirmer, Uwe Breitenb, and Ana Cristina. *Automating the Provisioning and Configuration of Devices in the Internet of Things*. Complex Systems Informatics and Modeling Quarterly, (9):28–43, 2017.
- [HKK⁺10] Michael E. Houle, Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. *Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?* In Proc of the 22nd Int. Conf. on Scientific and Statistical Database Management (SSDBM), 2010.
- [HKP06] Jiawei Han, Micheline Kamber, and Jian Pei. *Chapter 9: Graph Mining, Social Network Analysis, and Multirelational Data Mining*. Data mining: concepts and techniques, pages 535–589, 2006.
- [HS13] Steve Harris and Andy Seaborne. *SPARQL 1.1 Query Language*. W3C Recommendation, World Wide Web Consortium, 2013. <http://www.w3.org/TR/r2rml/> (retrieved 2017-11-01).
- [HS14] Robert Henßen and Miriam Schleipen. *Interoperability between OPC UA and AutomationML*. Procedia CIRP, 2014.

- [HYL17] William L. Hamilton, Rex Ying, and Jure Leskovec. *Inductive Representation Learning on Large Graphs*. Advances in Neural Information Processing Systems 30 (NIPS 2017), (Nips):1024–1034, 2017.
- [IOA09] Ruhaizan Ismail, Zalinda Othman, and Azuraliza Abu Bakar. *Data Mining in Production Planning and Scheduling: A Review*. In Conference on Data Mining and Optimization, 2009.
- [JKSB14] Jens Jäger, Andreas Kluth, Anja Schatz, and Thomas Bauernhansl. *Complexity patterns in the advanced complexity management of value networks*. Procedia CIRP, 17:645–650, 2014.
- [JONK14] Vaclav Jirkovsky, Marek Obitko, Petr Novak, and Petr Kadera. *Big Data Analysis for Sensor Time-Series in Automation*. In Proc. of Emerging Technology and Factory Automation (ETFA), pages 1–8, 2014.
- [KBT15] Denis Krompaß, Stephan Baier, and Volker Tresp. *Type-Constrained Representation Learning in Knowledge Graphs*. Proc. of the 14th Int. Sem. Web Conf. (ISWC), 2015.
- [KGJR⁺16] Evgeny Kharlamov, Bernardo Cuenca Grau, Ernesto Jiménez-Ruiz, Steffen Lamparter, Gulnar Mehdi, Martin Ringsquandl, Yavor Nenov, Stephan Grimm, Mikhail Roshchin, and Ian Horrocks. *Capturing industrial information models with ontologies and constraints*. In Proceedings of the 15th International Semantic Web Conference, pages 325–343, 2016.
- [KHJR⁺15] Evgeny Kharlamov, Dag Hovland, Ernesto Jiménez-Ruiz, Davide Lanti, Hallstein Lie, Christoph Pinkel, Martin Rezk, Martin G Skjæveland, Evgenij Thorstensen, Guohui Xiao, Dmitriy Zheleznyakov, and Ian Horrocks. *Ontology Based Access to Exploration Data at Statoil*. In The Semantic Web - ISWC 2015, pages 93–112, Cham, 2015. Springer.
- [KSFB14] Jörg Uwe Kietz, Floarea Serban, Simon Fischer, and Abraham Bernstein. *“Semantics inside!” but let’s not tell the data miners: Intelligent support for data mining*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8465 LNCS(231519):706–720, 2014.

- [KSÖ⁺14] Evgeny Kharlamov, Nina Solomakhina, Özgür Lütfü Özçep, Dmitriy Zheleznyakov, Thomas Hubauer, Steffen Lamparter, Mikhail Roshchin, Ahmet Soylu, and Stuart Watson. *How semantic technologies can enhance data access at siemens energy*. In ISWC, pages 601–619, 2014.
- [KW16] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. pages 1–14, 2016.
- [KWH13] Henning Kagermann, Wolfgang Wahlster, and Johannes Helbig. *Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0*. Technical Report April, acatech, 2013.
- [LBR12] Uta Lösch, Stephan Bloehdorn, and Achim Rettinger. *Graph Kernels for RDF Data*. In Proc. of the 9th Extended Semantic Web Conference (ESWC’12), 2012.
- [LC10] Ni Lao and William W. Cohen. *Relational retrieval using a combination of path-constrained random walks*. Machine Learning, 81(1):53–67, 2010.
- [LHSS12] Matthias Loskyll, Ines Heck, Jochen Schlick, and Michael Schwarz. *Context-Based Orchestration for Control of Resource-Efficient Manufacturing Processes*. Future Internet, 4(4):737–761, aug 2012.
- [LLBaK13] Jay Lee, Edzel Lapira, Behrad Bagheri, and Hung an Kao. *Recent advances and trends in predictive manufacturing systems in big data environment*. Manufacturing Letters, 1(1):38–41, 2013.
- [LMC11] Ni Lao, Tom Mitchell, and William W. Cohen. *Random walk inference and learning in a large scale knowledge base*. In Proceedings of the EMNLP Conference, pages 529–539, 2011.
- [LNR14] David Lechevalier, Anantha Narayanan, and Sudarsan Rachuri. *Towards a domain-specific framework for predictive analytics in manufacturing*. Proc. of the 2014 IEEE International Conference on Big Data, pages 987–995, 2014.

- [MBXR18] Aditya Mogadala, Umanga Bista, Lexing Xie, and Achim Rettinger. *Knowledge Guided Attention and Inference for Describing Images Containing Unseen Objects*. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web*, pages 415–429, Cham, 2018. Springer International Publishing.
- [MDFE16] Pasquale Minervini, Claudia D’Amato, Nicola Fanizzi, and Floriana Esposito. *Leveraging the schema in latent factor models for knowledge graph completion*. *Proceedings of the 31st Annual ACM Symposium on Applied Computing - SAC ’16*, pages 327–332, 2016.
- [MK16] Ankita Mangal and Nishant Kumar. *Using big data to enhance the bosch production line performance: A Kaggle challenge*. In *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pages 2029–2035, 2016.
- [MKS⁺17] Gulnar Mehdi, Evgeny Kharlamov, Ognjen Savković, Guohui Xiao, Elem Güzel Kalayci, Sebastian Brandt, Ian Horrocks, Mikhail Roshchin, and Thomas Runkler. *Semantic Rule-Based Equipment Diagnostics*. In *The Semantic Web – ISWC 2017*, pages 314–333, Cham, 2017. Springer.
- [MLD⁺09] Boris Motik, Carsten Lutz, Giuseppe De Giacomo, Jim Hendler, Peter F Patel-Schneider, Uli Sattler, and Michael Schneider. *OWL 2 Web Ontology Language Profiles*, 2009.
- [MRP⁺05] Ravi M. Rangan, Steve Rohde, Russell Peak, Bipin Chadha, and Plamen Bliznakov. *Streamlining Product Lifecycle Processes: A Survey of Product Lifecycle Management Implementations, Directions, and Challenges*. *Journal of Computing and Information Science in Engineering*, 5:227, 2005.
- [NMTG15] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. *A Review of Relational Machine Learning for Knowledge Graph*. *Proceedings of the IEEE*, 104(1):11–33, 2015.

- [NPM⁺15] Yavor Nenov, Robert Piro, Boris Motik, Ian Horrocks, Zhe Wu, and Jay Banerjee. *RDFox : A Highly-Scalable RDF Store*. In International Semantic Web Conference, pages 3–20, 2015.
- [NTK11] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. *A three-way model for collective learning on multi-relational data*. Proceedings of the 28th International Conference on Machine Learning (ICML), 2011.
- [Pau15] Heiko Paulheim. *Automatic Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods*. Semantic Web Journal, 1(20), 2015.
- [Paw14] Günther Pawellek. *Ganzheitliche Fabrikplanung: Grundlagen, Vorgehensweise, EDV-Unterstützung*. VDI-Buch. Springer Berlin Heidelberg, 2014.
- [PDS08] Panče Panov, Sašo Džeroski, and Larisa N. Soldatova. *OntoDM: An ontology of data mining*. Proceedings - IEEE International Conference on Data Mining Workshops, ICDM Workshops 2008, pages 752–760, 2008.
- [PHGg17] Niklas Petersen, Lavdim Halilaj, and Irlán Grangel-gonzález. *Realizing an RDF-based Information Model for a Manufacturing Company – A Case Study*. 16th International Semantic Web Conference (ISWC 2017), 15054(Iswc):1–16, 2017.
- [RLBL15] Martin Ringsquandl, Steffen Lamparter, Sebastian-Philipp Brandt, and Raffaello Lepratti. *Semantic-guided Feature Selection for Industrial Automation Systems*. In Proceedings of the 14th International Semantic Web Conference, pages 225–240, 2015.
- [RL16] Martin Ringsquandl, Steffen Lamparter, and Raffaello Lepratti. *Graph-based predictions and recommendations in flexible manufacturing systems*. In Proceedings of the IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, pages 6937–6942, 2016.
- [RLLK17] Martin Ringsquandl, Steffen Lamparter, Raffaello Lepratti, and Peer Kröger. *Knowledge Fusion of Manufacturing Operations Data using Representation*

- Learning*. In Proceedings of the IFIP International Conference on Advances in Production Management Systems, pages 302–310, 2017.
- [RLT⁺16] Martin Ringsquandl, Steffen Lamparter, Ingo Thon, Raffaello Lepratti, and Peer Kröger. *Knowledge Graph Constraints for Multi-label Graph Classification*. In Proceedings of the IEEE 16th International Conference on Data Mining Workshops (ICDMW), pages 121–127, 2016.
- [RMKZ13] Mariano Rodríguez-Muro, Roman Kontchakov, and Michael Zakharyashev. *Ontology-based data access: Ontop of databases*. In Proc. of the 12th Int. Sem. Web Conf., 2013.
- [RN10] Stuart J Russel and Peter Norvig. *Semantic Networks*. In Artificial Intelligence: A Modern Approach, chapter 12, page 454. Pearson, third edition, 2010.
- [RP14] Petar Ristoski and Heiko Paulheim. *A comparison of propositionalization strategies for creating features from linked open data*. CEUR Workshop Proceedings, 1232:6–16, 2014.
- [RP16a] Petar Ristoski and Heiko Paulheim. *RDF2Vec: RDF graph embeddings for data mining*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9981 LNCS:498–514, 2016.
- [RP16b] Petar Ristoski and Heiko Paulheim. *Semantic Web in data mining and knowledge discovery: A comprehensive survey*. Web Semantics: Science, Services and Agents on the World Wide Web, 36:1–22, 2016.
- [RSL⁺17] Martin Ringsquandl, Daria Stepanova, Steffen Lamparter, Raffaello Lepratti, Evgeny Kharlamov, Ian Horrocks, and Peer Kröger. *On Event-Driven Knowledge Graph Completion in Digital Factories*. In Proceedings of the IEEE International Conference on Big Data, pages 1676–1681, 2017.
- [RSL⁺18] Martin Ringsquandl, Daria Stepanova, Steffen Lamparter, Raffaello Lepratti, Evgeny Kharlamov, Ian Horrocks, and Peer Kröger. *Event-enhanced*

- Learning for Knowledge Graph Completion*. In Proceedings of the 15th Extended Semantic Web Conference (ESWC), pages 541–559, Heraklion, 2018.
- [Rud11] Sebastian Rudolph. *Foundations of description logics*. In Axel Polleres, Claudia D’Amato, Marcelo Arenas, Siegfried Handschuh, Paula Kroner, Sascha Ossowski, and Peter Patel-Schneider, editors, Reasoning Web. Semantic Technologies for the Web of Data, pages 76–136. Springer, 2011.
- [Sch14] Alexander Schließmann. *iProduction, die Mensch-Maschine-Kommunikation in der Smart Factory*. In Thomas Bauernhansl, Michael ten Hompel, and Birgit Vogel-Heuser, editors, Industrie 4.0 in Produktion, Automatisierung und Logistik, pages 451–480. Springer Vieweg, 2014.
- [SCMN13] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. *Reasoning With Neural Tensor Networks for Knowledge Base Completion*. Proceedings of the Advances in Neural Information Processing Systems 26 (NIPS 2013), pages 1–10, 2013.
- [Sin12] Amit Singhal. *Introducing the Knowledge Graph: things, not strings*, 2012.
- [SKB⁺18] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. *Modeling Relational Data with Graph Convolutional Networks*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10843 LNCS(1):593–607, 2018.
- [SMS11] Miriam Schleipen, Ansgar Münnemann, and Olaf Sauer. *Interoperabilität von Manufacturing Execution Systems (MES)*. at - Automatisierungstechnik, 59(7):413–424, jul 2011.
- [SSF10] Miriam Schleipen, Olaf Sauer, and Lidmilla Fuskova. *Logical interface between Manufacturing Execution Systems (MES) and machine - semantic integration by means of ontologies*. Intelligent Computation in Manufacturing Engineering, 7:2–5, 2010.
- [SW17] Baoxu Shi and Time Weninger. *ProjE : Embedding Projection for Knowledge Graph Completion*. AAAI 2017, pages 1–14, 2017.

- [TAL14] Jiliang Tang, Salem Alelyani, and Huan Liu. *Feature Selection for Classification: A Review*. In *Data Classification: Algorithms and Applications*, pages 37–60. CRC Press, 2014.
- [TCQ⁺18] Fei Tao, Jiangfeng Cheng, Qinglin Qi, Meng Zhang, He Zhang, and Fangyuan Sui. *Digital twin-driven product design, manufacturing and service with big data*. *International Journal of Advanced Manufacturing Technology*, 94(9-12):3563–3576, 2018.
- [TDW⁺17a] Aparna Saisree Thuluva, Kirill Dorofeev, Monika Wenger, Darko Anicic, and Sebastian Rudolph. *Semantic-Based Approach for Low-Effort Engineering of Automation Systems*. In *Proceedings of ODBASE*, pages 497–512, Cham, 2017. Springer International Publishing.
- [TDW⁺17b] Théo Trouillon, Christopher R. Dance, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. *Knowledge Graph Completion via Complex Tensor Factorization*. *CoRR*, abs/1702.0, 2017.
- [TDWS17] Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. *Know-Evolve: Deep Temporal Reasoning for Dynamic Knowledge Graphs*. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.
- [USL⁺17] Thomas H.J. Uhlemann, Christoph Schock, Christian Lehmann, Stefan Freiberger, and Rolf Steinhilper. *The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems*. *Procedia Manufacturing*, 9:113–120, 2017.
- [WIT14] Thorsten Wuest, Christopher Irgens, and Klaus-dieter Thoben. *An approach to monitoring quality in manufacturing using supervised machine learning on product state data*. *Journal on Intelligent Manufacturing*, 25:1167–1180, 2014.
- [WL16] Zhigang Wang and Juanzi Li. *Text-Enhanced Representation Learning for Knowledge Graph*. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1293–1299, 2016.

- [WWG15] Quan Wang, Bin Wang, and Li Guo. *Knowledge base completion using embeddings and rules*. In Proceedings of the International Joint Conference on Artificial Intelligence, pages 1859–1866, 2015.
- [WZZY14] Jia Wu, Xingquan Zhu, Changqi Zhang, and Phillip Yu. *Bag Constrained Structure Pattern Mining for Multi-Graph Classification*. IEEE Transactions on Knowledge and Data Engineering, 26(10):2382–2396, 2014.
- [XHMZ17] Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. *SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions*. AAAI, pages 1–10, 2017.
- [XLJ⁺16] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. *Representation Learning of Knowledge Graphs with Entity Descriptions*. IJCAI 2016, pages 2659–2665, 2016.
- [YCHY08] Xifeng Yan, Hong Cheng, Jiawei Han, and Philip S. Yu. *Mining significant graph patterns by leap search*. Proceedings of the 2008 ACM SIGMOD Int. conf. on Management of Data, pages 433–444, 2008.
- [YH02] Xifeng Yan and Jiawei Han. *gSpan: Graph-Based Substructure Pattern Mining*. Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM’02)., 1(d):721–724, 2002.
- [YLY⁺18] Jiaxuan You, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec. *Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation*. Number 32nd Conference on Neural Information Processing Systems (NIPS), pages 1–11, 2018.
- [ZH05] Hui Zou and Trevor Hastie. *Regularization and variable selection via the elastic net*. Journal of the Royal Statistical Society, 67:301–320, 2005.

Appendix A

Publications

Capturing Industrial Information Models with Ontologies and Constraints

Evgeny Kharlamov¹(✉), Bernardo Cuenca Grau¹, Ernesto Jiménez-Ruiz¹,
Steffen Lamparter², Gulnar Mehdi², Martin Ringsquandl², Yavor Nenov¹,
Stephan Grimm², Mikhail Roshchin², and Ian Horrocks¹

¹ University of Oxford, Oxford, UK

evgeny.kharlamov@cs.ox.ac.uk

² Siemens AG, Corporate Technology, Munich, Germany

Abstract. This paper describes the outcomes of an ongoing collaboration between Siemens and the University of Oxford, with the goal of facilitating the design of ontologies and their deployment in applications. Ontologies are often used in industry to capture the conceptual information models underpinning applications. We start by describing the role that such models play in two use cases in the manufacturing and energy production sectors. Then, we discuss the formalisation of information models using ontologies, and the relevant reasoning services. Finally, we present SOMM—a tool that supports engineers with little background on semantic technologies in the creation of ontology-based models and in populating them with data. SOMM implements a fragment of OWL 2 RL extended with a form of integrity constraints for data validation, and it comes with support for schema and data reasoning, as well as for model integration. Our preliminary evaluation demonstrates the adequacy of SOMM’s functionality and performance.

1 Introduction

Software systems in the domain of industrial manufacturing have become increasingly important in recent years. Production machines, such as assembly line robots or industrial turbines, are equipped with and controlled by complex and costly pieces of software; according to a recent survey, over 40% of the total production cost of such machines is due to software development and the trend is for this number only to continue growing [35]. Additionally, many critical tasks within business, engineering, and production departments (e.g., control of production processes, resource allocation, reporting, business decision making) have also become increasingly dependent on complex software systems.

Recent global initiatives such as Industry 4.0 [9, 18, 34] aim at the development of *smart factories* based on fully computerised, software-driven, automation of production processes and enterprise-wide integration of software components. In smart factories, software systems monitor and control physical

This work was partially funded by the Royal Society under a University Research Fellowship, the EU project Optique [24] (FP7-ICT-318338), and the EPSRC projects MaSI3, DBOnto, ED3.

processes, effectively communicate and cooperate with each other as well as with humans, and are in charge of making decentralised decisions. The success of such ambitious initiatives relies on the seamless (re)development and integration of software components and services. This poses major challenges to an industry where software systems have historically been developed independently from each other.

There has been a great deal of research in recent years investigating key aspects of software development in industrial manufacturing domains, including life-cycle costs, dependability, compatibility, integration, and performance (e.g., see [41] for a survey). This research has highlighted the need for enterprise-wide *information models*—machine-readable conceptualisations describing the functionality of and information flow between different assets in a plant, such as equipment and production processes. The development information models based on ISA and IEC standards¹ has now become a common practice in modern companies [30] and Siemens is not an exception in this trend.

In practice, however, many types of models co-exist, and applications typically access data from different kinds of machines and processes designed according to different models. These information models have been independently developed in different (often incompatible) formats using different types of proprietary software; furthermore, they may not come with a well-defined semantics, and their specification can be ambiguous. As a result, model development, maintenance, and integration, as well as data exchange and sharing pose major challenges in practice.

Adoption of semantic technologies has been a recent development in many large companies such as IBM [11], the steel manufacturer Arcelor Mittal [2], the oil and gas company Statoil [21], and Siemens [1, 4, 19, 20, 22, 25, 32]. An important application of these technologies has been the formalisation of information models using OWL 2 ontologies and the use of RDF for storing application data. OWL 2 provides a rich and flexible modelling language that seems well-suited for describing industrial information models: it not only comes with an unambiguous, standardised, semantics, but also with a wide range of tools that can be used to develop, validate, integrate, and reason with such models. In turn, RDF data can not only be seamlessly accessed and exchanged, but also stored directly in highly scalable RDF triple stores and effectively queried in conjunction with the available ontologies. Moreover, legacy and other data that must remain in its original format and cannot be transformed into RDF can be virtualised as RDF using ontologies following the Ontology-Based Data Access (OBDA) approach [21, 23, 29].

In this paper, we describe the outcomes of an ongoing collaboration between Siemens Corporate Technology in Munich and the University of Oxford, with the goal of facilitating deployment of ontology-based industrial information models. We start by describing the key role that information models play in two use cases in the manufacturing and energy production sectors. Then, we present industrial information models that are used for describing manufacturing and

¹ International Society of Automation and International Electrotechnical Commission.

energy plants, and discuss how they can be captured using ontologies. In our discussion, we stress the modelling choices made when formalising these models as ontologies and identify the key OWL constructs required in this setting. Our analysis revealed the need for integrity constraints for data validation [27, 37], which are not available in OWL 2. Hence, we discuss in detail what kinds of constraints are needed in industrial use cases and how to incorporate them. We then illustrate the use of reasoning services, such as concept satisfiability, data constraint validation, and query answering for addressing Siemens' application requirements.

Ontologies are currently being created and maintained in Siemens by qualified R&D personnel with expertise in ontology languages and ontology engineering. In order to widen the scope of application of semantic technologies in the company it is crucial to make ontology development accessible to other teams of engineers. To this end, we have developed the Siemens-Oxford Model Manager (SOMM)—a tool that has been designed to fulfil industrial requirements and which supports engineers with little background on semantic technologies in the creation and use of ontologies. SOMM provides a simple interface for ontology development and enables the introduction of instance data via automatically generated forms that are driven by the ontology and which help minimising errors in data entry. SOMM implements a fragment of the OWL 2 RL profile [26] extended with database integrity constraints for data validation; the supported language is sufficient to capture the main features of ISA and ICE based information models used by Siemens. SOMM is built on top of Web-Protégé [40], which provides built-in functionality for ontology versioning and collaborative development. It relies on HermiT [10] for ontology classification and LogMap [16] to support model alignment and merging. For query answering and constraint validation, SOMM requires a connection to a triple store or a rule inference system that supports Datalog reasoning and stratified negation-as-failure.

We showcase the practical benefits of our tool using two ontologies in the manufacturing and power generation domains. Both ontologies have been developed using SOMM by Siemens engineers to capture information models currently in use. Based on these ontologies, we conducted an empirical evaluation of SOMM's performance in supporting constraint validation and query answering over realistic manufacturing and gas turbine data. In our experiments, we coupled SOMM with the rule inference engine IRIS [3], which is available under the LGPL license.² Our evaluation demonstrates the adequacy of SOMM's functionality and performance for industrial applications.

2 Industrial Information Models

Conceptual information models can be exploited in a wide range of manufacturing and energy production applications. In this Section, we discuss two concrete use cases and describe the underpinning models and their limitations.

² <http://www.iris-reasoner.org/>.

2.1 Applications in Manufacturing and Energy Production

In manufacturing and energy production plants it is essential that all processes and equipment run smoothly and without interruptions.

In a typical manufacturing plant, data is generated and stored whenever a piece of equipment consumes material or completes a task. This data is then accessed by plant operators using *manufacturing execution systems (MES)*—software programs that steer the production in a manufacturing plant. MESs are responsible for keeping track of the material inventory and tracing their consumption, thus ensuring that equipment and materials needed for each process are available at the relevant time [30]. Similarly, turbines in energy plants are equipped with sensors that are continuously generating data. This data is consumed by *remote monitoring systems (RMS)*, which analyse turbine data to prevent faults, report anomalies and ensure that the turbines operate without interruption. In both application scenarios, the use of information models is twofold.

1. Models are used to provide machine-readable specifications for the data generated by equipment and processes, and for the data flow across assets and processes in a plant.
2. Models provide a schema for constructing and executing complex queries. In particular, monitoring tasks in MESs are realised by means of queries issued to production machines and data hubs; similarly, anomaly detection in an RMS relies on queries spanning the structure of the turbines, the readings of their sensors, and the configuration of turbines within a plant.

2.2 Information Models Based on Industrial Standards

We next describe the information models in Siemens relevant to the aforementioned applications. These models have been developed in compliance with ISA, IEC, and ISO/TS international standards.

Manufacturing Models. For many manufacturing applications it is a common practice to rely on information models that are based on the international standard ISA-88/95.

The ISA-88/95 standard provides general guidelines for specifying the functionality of and interface between manufacturing software systems. The standard consists of UML-like diagrammatic descriptions accompanied with tables and unstructured text, which are used to extend the diagrams with additional information and examples. Figure 1 presents an excerpt of the ISA-88/95 standard modelling materials, equipment, personnel, and processes in a plant. For instance, one of these diagrams establishes that pieces of equipment can be composed by other pieces of equipment and are described by a number of specified ‘equipment properties’. The table complementing this diagram indicates that each piece of equipment must have a numeric ID and may have a textual

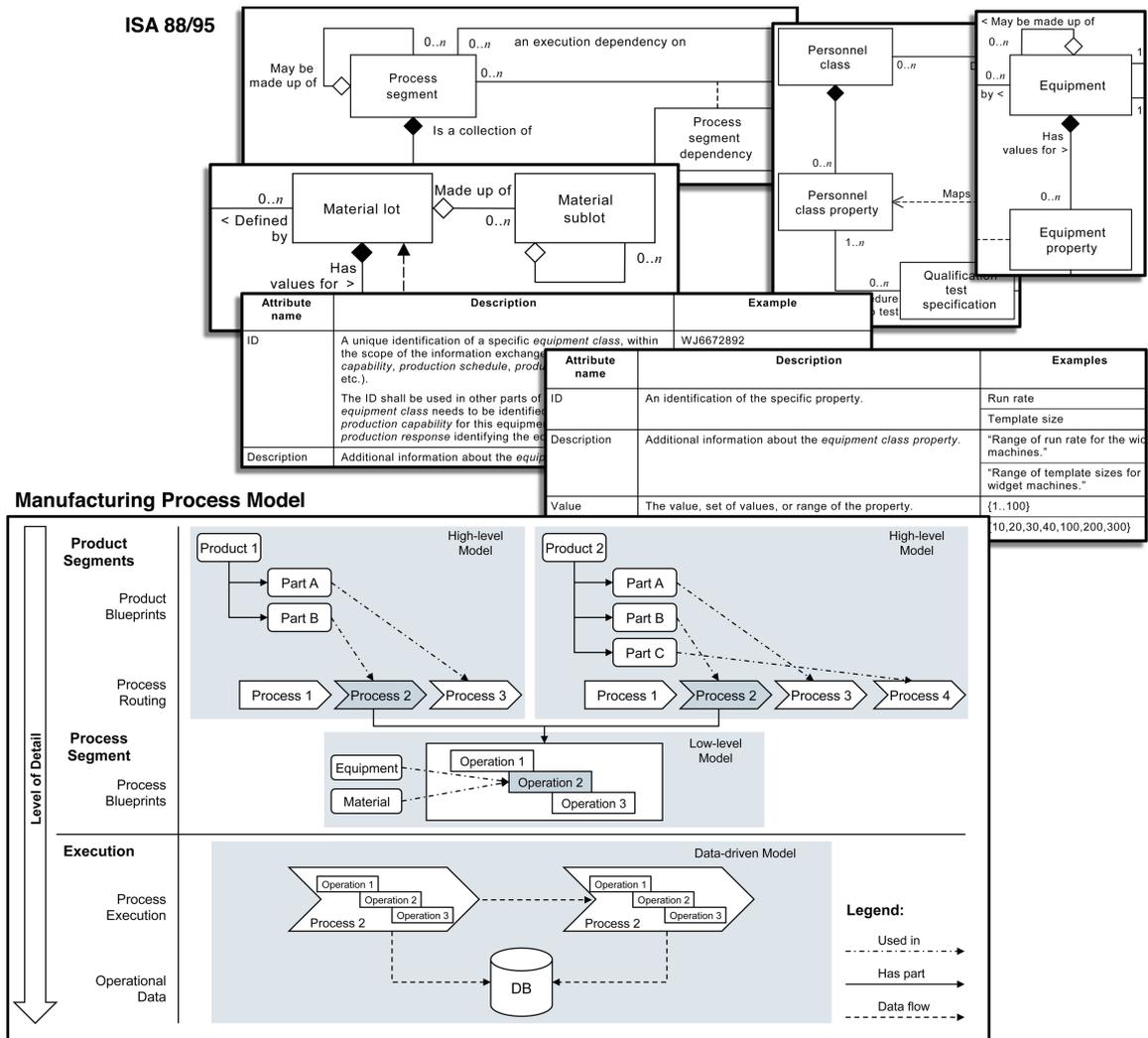


Fig. 1. Fragment of ISA 88/95 and an example model based on it.

description; additional properties of equipment can be introduced by providing an ID, a textual description of the property, and a value range.

Figure 1 provides a simplified version of an information model based on the standard ISA-88/95. The model is organised in three layers: *product*, *process*, and *execution*. On the product level, we can see the specification of two products and their relationship to production processes; for instance, *Product1* consists of *PartA* and *PartB*, which are manufactured by two consecutive processes. The process segment level provides more fine-grained specifications of the structure of each process; for instance, *Process2* consists of three operations, where the second one relies on specific kinds of materials and equipment. Finally, at the execution level, we can see how data is stored and accessed by individual processes.

Energy Plant Models. Information models for energy plants are often based on the *Reference Designation System for Power Plants (RDS-PP)* and *Kraftwerk-*

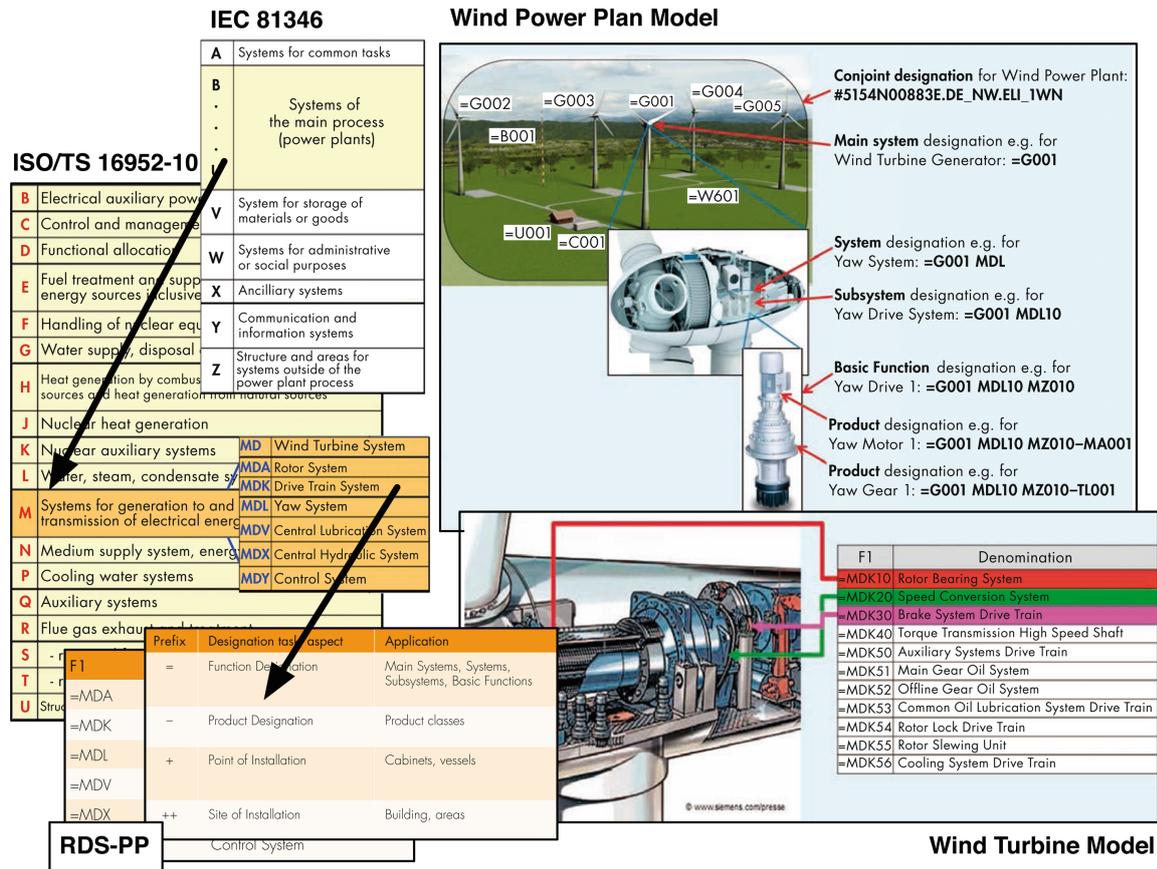


Fig. 2. Designation models IEC 81346, ISO/TS 16952-10, and RDS-PP and example energy information model for an energy plant [31].

Kennzeichensystemen (KKS) standards, which are in turn extensions for the energy sector of the IEC 81346 and ISO/TS 16952-10 international standards.

IEC 81346 and ISO/TS 16952-10 provide a generic dictionary of codes for designating and classifying industrial equipment. Figure 2 provides an excerpt of these standards and their dependencies. For instance, in IEC-81346 letters ‘B’ to ‘U’ are used for generically designating systems in power plants. ISO/TS 16952-10 makes this specification more precise by indicating, for example, that letter ‘M’ refers to systems for generating and transmitting electricity, and that we can append ‘D’ to ‘M’ to refer to a wind turbine system. RDS PP and KKS provide a more extensive vocabulary of codes for equipment, their functionality and locations, as well as a system for combining such codes.

A typical energy plant model describes the structure of a plant by providing the functionality and location of each equipment component using RDS PP and KKS codes. Having this information in a machine-readable format is important for planning and construction, as well as for the software-driven operation and maintenance of the plant. Figure 2 shows how a specific plant is represented in a model; for instance, code =G001 MDL10 denotes that the yaw drive system number 10 of type MDL is located in the wind turbine generator number 001.

2.3 Technical Challenges

The development and use of information models in practice poses major challenges.

1. Model development is costly, as it requires specialised training and proprietary tools; as a result, model development often cannot keep up with the arrival of new equipment and introduction of new processes.
2. Models are difficult to integrate and share since they are often independently developed using different types of proprietary software and they are based on incompatible data formats.
3. Monitoring queries are difficult to compose and execute on top of information models: they must comply with the requirements of the models (e.g., refer to specific codes in the energy use case), and their execution requires access to heterogeneous data from different machines and processes.

In order to overcome these challenges Siemens has recently applied semantic technologies in a number of applications [13, 15, 19, 22, 32]. In particular, OWL 2 has been used for describing information models. The choice of OWL 2 is not surprising since it provides a rich and flexible modelling language that is well suited for addressing the aforementioned challenges: it comes with an unambiguous, standardised semantics, and a wide range of tools and infrastructure. Moreover, RDF provides a unified data exchange format, which can be used to seamlessly access and exchange data, and hence facilitate monitoring tasks based on complex queries.

3 From Information Models to Ontologies and Constraints

In this section we describe the ontologies that we have developed to capture manufacturing and energy production models presented in Sect. 2. The goal of our ontologies is to eventually replace their underpinning models in applications. Thus, their design has been driven towards fulfilling the same purposes as the models they originate from; that is, to act as schema-level templates for data generation and exchange, and to enable the formulation and execution of monitoring queries.

The representation of industrial information models and standards using ontologies has been widely acknowledged as a non-trivial task [5, 12, 14, 36]. In Sect. 3.1 we discuss the modelling choices underpinning the design of our ontologies and identify a fragment of OWL 2 RL that is sufficient to capture the basic aspects of the information models. Our analysis of the models, however, also revealed the need to incorporate database integrity constraints for data validation, which are not supported in OWL 2 [27, 37]. Thus, we also discuss the kinds of constraints that are relevant to our applications.

Finally, in Sect. 3.2 we discuss how the OWL 2 RL axioms and integrity constraints can be captured by means of rules with stratified negation for the

purpose of data validation and query answering. We assume basic familiarity with Datalog—the rule language underpinning OWL 2 RL and SWRL—as well as with stratified negation-as-failure (see [6] for an excellent survey on Logic Programming).

3.1 Modelling

From an ontological point of view, most building blocks of the the typical industrial information models are rather standard in conceptual design and naturally correspond to OWL 2 classes (e.g., *Turbine*, *Process*, *Product*), object properties (e.g., *hasPart*, *hasFunction*, *locatedIn*) and data properties (e.g., *ID*, *hasRotorSpeed*).

The main challenge that we encountered was to capture the constraints of the models using ontological axioms. We next describe how this was accomplished using a combination of OWL 2 RL axioms and integrity constraints.

Standard OWL 2 RL Axioms. The specification of the models suggests the arrangement of classes and properties according to subsumption hierarchies, which represent the skeleton of the model and establish the basic relationships between their components. For instance, in the energy plant model a *Turbine* is specified as a kind of *Equipment*, whereas *hasRotorSpeed* is seen as a more specific relation than *hasSpeed*. The models also suggest that certain properties must be declared as transitive, such as *hasPart* and *locatedIn*. Similarly, certain properties are naturally seen as inverse of each other (e.g., *hasPart* and *partOf*). These requirements are easily modelled in OWL 2 using the following axioms written in functional-style syntax:

$$\text{SubClassOf}(\textit{Turbine} \textit{Equipment}) \quad (1)$$

$$\text{SubDataPropertyOf}(\textit{hasRotorSpeed} \textit{hasSpeed}) \quad (2)$$

$$\text{TransitiveObjectProperty}(\textit{hasPart}) \quad (3)$$

$$\text{InverseObjectProperties}(\textit{hasPart} \textit{partOf}) \quad (4)$$

These axioms can be readily exploited by reasoners to support query answering; e.g., when asking for all equipment with a rotor, one would expect to see all turbines that contain a rotor as a part (either directly or indirectly).

Additionally, the models describe *optional relationships* between entities. In the manufacturing model certain materials are optional to certain processes, i.e., they are compatible with the process but they are not always required. Similarly, certain processes can optionally be followed by other processes (e.g., conveying may be followed by packaging). Universal (i.e., *AllValuesFrom*) restrictions are well-suited for attaching an optional property to a class. For instance, the axiom

$$\text{SubClassOf}(\textit{Conveying} \text{ObjectAllValuesFrom}(\textit{followedBy} \textit{Packaging})) \quad (5)$$

states that only packaging processes can follow conveying processes; that is, a conveying process can be either terminal (i.e., not followed by any other process)

or it is followed by a packaging process. As a result, when introducing a new conveying process we are not forced to provide a follow-up process, but if we do so it must be an instance of *Packaging*.

All the aforementioned types of axioms are included in the OWL 2 RL profile. This has many practical advantages for reasoning since OWL 2 RL is amenable to efficient implementation using rule-based technologies.

Constraint Axioms. In addition to optional relationships, the information models from Sect. 2 also describe relationships that are inherently *mandatory*, e.g., when introducing a new turbine, the energy model requires that we also provide its rotors.

This behaviour is naturally captured by an integrity constraint: whenever a turbine is added and its rotors are not provided, the application should flag an error. Integrity constraints are not supported in OWL 2; for instance, the axiom

$$\text{SubClassOf}(\textit{Turbine} \text{ ObjectSomeValuesFrom}(\textit{hasPart} \textit{Rotor})) \quad (6)$$

states that every turbine must contain a rotor as a part; such rotor, however, can be possibly unknown or unspecified.

The information models also impose cardinality restrictions on relationships. For instance, each double rotor turbine in the energy plant model is specified as having exactly two rotors. This can be modelled in OWL 2 using the axioms

$$\text{SubClassOf}(\textit{TwoRotorTurbine} \text{ ObjectMinCardinality}(2 \textit{hasPart} \textit{Rotor})) \quad (7)$$

$$\text{SubClassOf}(\textit{TwoRotorTurbine} \text{ ObjectMaxCardinality}(2 \textit{hasPart} \textit{Rotor})) \quad (8)$$

Such cardinality restrictions are interpreted as integrity constraints in many applications: when introducing a specific double rotor turbine, the model requires that we also provide its two rotors. The semantics of axioms (7) and (8) is not well-suited for this purpose: on the one hand, (7) does not enforce a double rotor turbine to explicitly contain any rotors at all; on the other hand, if more than two rotors are provided, then (8) non-deterministically enforces at least two of them to be equal.

There have been several proposals to extend OWL 2 with integrity constraints [27, 37]. In these approaches, the ontology developer explicitly designates a subset of the OWL 2 axioms as constraints. Similarly to constraints in databases, these axioms are used as checks over the given data and do not participate in query answering once the data has been validated. The specifics of how this is accomplished semantically differ amongst each of the proposals; however, all approaches largely coincide if the standard axioms are in OWL 2 RL.

3.2 Data Validation and Query Answering

Our approach to data validation and query answering follows the standard approaches in the literature [27, 37]: given a query Q , dataset \mathcal{D} , and OWL 2 ontology \mathcal{O} consisting of a set \mathcal{S} of standard OWL 2 RL axioms and a set \mathcal{C} of axioms marked as constraints, we proceed according to Steps 1–4 given next.

Table 1. OWL 2 RL axioms as rules. All entities mentioned in the axioms are named. By abuse of notation, we use SubPropertyOf and AllValuesFrom to refer to both their Object and Data versions in functional syntax.

OWL 2 Axiom	Datalog Rules
SubClassOf($A B$)	$B(?x) \leftarrow A(?x)$
SubPropertyOf($P_1 P_2$)	$P_2(?x, ?y) \leftarrow P_1(?x, ?y)$
TransitiveObjectProperty(P)	$P(?x, ?z) \leftarrow P(?x, ?y) \wedge P(?y, ?z)$
InverseObjectProperties(P_1, P_2)	$P_2(?y, ?x) \leftarrow P_1(?x, ?y)$ and
	$P_1(?y, ?x) \leftarrow P_2(?x, ?y)$
SubClassOf($A \text{ AllValuesFrom}(P B)$)	$B(?y) \leftarrow P(?x, ?y) \wedge A(?x)$

1. Translate the standard axioms \mathcal{S} into a Datalog program $\Pi_{\mathcal{S}}$ using the well-known correspondence between OWL 2 RL and Datalog.
2. Translate the integrity constraints \mathcal{C} into a Datalog program $\Pi_{\mathcal{C}}$ with stratified negation-as-failure containing a distinguished binary predicate *Violation* for recording the individuals and axioms involved in a constraint violation.
3. Retrieve and flag all integrity constraint violations. This can be done by computing the extension of the *Violation* predicate.
4. If no constraints are violated, answer the user's query Q using the query answering facilities provided by the reasoner.

Steps 3 and 4 can be implemented on top of RDF triple stores with support for OWL 2 RL and stratified negation (e.g., [28]), as well as on top of generic rule inference systems (e.g., [3]). In the remainder of this Section we illustrate Steps 1 and 2, where standard axioms and constraints are translated into rules. **Standard Axioms.** Table 1 provides the standard OWL 2 RL axioms needed to capture the information models of Sect. 2 and their translation into negation-free rules. In particular, the axioms (1)–(5) are equivalent to the following rules:

$$\textit{Equipment}(?x) \leftarrow \textit{Turbine}(?x) \quad (9)$$

$$\textit{hasSpeed}(?x, ?y) \leftarrow \textit{hasRotorSpeed}(?x, ?y) \quad (10)$$

$$\textit{hasPart}(?x, ?z) \leftarrow \textit{hasPart}(?x, ?y) \wedge \textit{hasPart}(?y, ?z) \quad (11)$$

$$\textit{Packaging}(?y) \leftarrow \textit{Conveying}(?x) \wedge \textit{followedBy}(?x, ?y) \quad (12)$$

Constraint Axioms. Table 2 provides the constraint axioms required to capture the models of Sect. 2 together with their translation into rules with negation. Our translation assigns a unique id to each individual axiom marked as an integrity constraint in the ontology, and it introduces predicates not occurring in the ontology in the heads of all rules. Constraint violations are recorded using the fresh predicate *Violation* relating individuals to constraint axiom ids.

The constraint (6) from Sect. 3.1 is captured by the following rules:

$$\textit{hasPart_Rotor}(?x) \leftarrow \textit{hasPart}(?x, ?y) \wedge \textit{Rotor}(?y) \quad (13)$$

$$\textit{Violation}(?x, \alpha) \leftarrow \textit{Turbine}(?x) \wedge \mathbf{not} \textit{hasPart_Rotor}(?x) \quad (14)$$

Table 2. Constraints axioms as rules. All entities are named, $n \geq 1$, and α is the unique id for the given constraint. *SomeValuesFrom*, *HasValue*, *FunctionalProperty*, *MaxCardinality* and *MinCardinality* denote both their Object and Data versions.

OWL Axiom	Datalog rules
SubClassOf(A SomeValuesFrom(R B))	$R_B(?x) \leftarrow R(?x, ?y) \wedge B(?y)$ and $Violation(?x, \alpha) \leftarrow A(?x) \wedge \mathbf{not} R_B(?x)$
SubClassOf(A HasValue(R b))	$Violation(?x, \alpha) \leftarrow A(?x) \wedge \mathbf{not} R(?x, b)$
FunctionalProperty(R)	$R_2(?x) \leftarrow R(?x, ?y_1) \wedge R(?x, ?y_2) \wedge$ $\mathbf{not} owl:sameAs(?y_1, ?y_2)$ and $Violation(?x, \alpha) \leftarrow R_2(?x)$
SubClassOf(A MaxCardinality(n R B))	$R_{(n+1)}_B(?x) \leftarrow \bigwedge_{1 \leq i \leq n+1} (R(?x, ?y_i) \wedge B(?y_i))$ $\bigwedge_{1 \leq i < j \leq n+1} (\mathbf{not} owl:sameAs(?y_i, ?y_j))$ and $Violation(?x, \alpha) \leftarrow A(?x) \wedge R_{(n+1)}_B(?x)$
SubClassOf(A MinCardinality(n R B))	$R_n_B(?x) \leftarrow \bigwedge_{1 \leq i \leq n} (R(?x, ?y_i) \wedge B(?y_i))$ $\bigwedge_{1 \leq i < j \leq n} (\mathbf{not} owl:sameAs(?y_i, ?y_j))$ and $Violation(?x, \alpha) \leftarrow A(?x) \wedge \mathbf{not} R_n_B(?x)$

Rule (13) identifies all individuals with a rotor as a part, and stores them as instances of the auxiliary predicate *hasPart_Rotor*. In turn, Rule (14) identifies all turbines that are not known to be instances of *hasPart_Rotor* (i.e., those with no known rotor as a part) and links them to the constraint α they violate.

Integrity constraints based on cardinalities require the use of the OWL 2 equality predicate *owl:sameAs*. For instance, the constraint axiom (7) from Sect. 3.1, to which we assign the id β_1 , is translated into the following rules:

$$\begin{aligned}
 hasPart_2_Rotor(?x) &\leftarrow \bigwedge_{1 \leq i \leq 2} (hasPart(?x, ?y_i) \wedge Rotor(?y_i)) \wedge \\
 &\quad \wedge (\mathbf{not} owl:sameAs(?y_1, ?y_2)) \\
 Violation(?x, \beta_1) &\leftarrow TwoRotorTurbine(?x) \wedge \mathbf{not} hasPart_2_Rotor(?x)
 \end{aligned}$$

The first rule infers an instance of the auxiliary predicate *hasPart_2_Rotor* if it is connected to two instances of *Rotor* that are not known to be equal; in turn, the second rule infers that all instances of *TwoRotorTurbine* that are not known to be instances of the auxiliary predicate violate the constraint (7). Similarly, axiom (8), to which we assign the id β_2 , is translated as follows:

$$\begin{aligned}
 hasPart_3_Rotor(?x) &\leftarrow \bigwedge_{1 \leq i \leq 3} (hasPart(?x, ?y_i) \wedge Rotor(?y_i)) \wedge \\
 &\quad \wedge \bigwedge_{1 \leq i < j \leq 3} (\mathbf{not} owl:sameAs(?y_i, ?y_j)) \\
 Violation(?x, \beta_2) &\leftarrow TwoRotorTurbine(?x) \wedge hasPart_3_Rotor(?x)
 \end{aligned}$$

Analogously to the previous case, the first rule infers that an individual is an instance of *hasPart_3_Rotor* if it is connected to three instances of *Rotor* that are

not known to be equal; in turn, the second rule infers that every such individual that is also an instance of *TwoRotorTurbine* violates the constraint axiom (8).

To conclude this section, we note that our translation in Table 2 yields a stratified program for any set \mathcal{C} of constraints. We can always define a stratification where the lowest stratum consists of the predicates in \mathcal{C} and *owl:sameAs*, the intermediate stratum contains all predicates of the form *R_B*, *R_n_B*, and *R_n*, and the uppermost stratum contains the special *Violation* predicate.

4 SOMM: An Industrial Ontology Management System

We have developed the *Siemens-Oxford Ontology Management (SOMM)* tool³ to support engineers in building ontologies and inserting data based on their information models. The interface of SOMM is restricted to support only the kinds of standard OWL 2 RL axioms and constraints discussed in Sect. 3.

SOMM is built on top of the Web-Protégé platform [40] by extending its front-end with new visual components and its back-end to access a Datalog-based triple store or a generic rule inference system for query answering and constraint validation, the OWL 2 reasoner HermiT [33] for ontology classification, and LogMap [16] to support ontology alignment and merging. Our choice of WebProtégé was based on Siemens' requirements for the platform underpinning SOMM, namely that it (i) can be used as a Web application; (ii) is under active development; (iii) is open-source and modular; (iv) includes built-in functionality for ontology versioning and collaborative development; (v) provides a form-based and end-user oriented interface; and (vi) enables the automatic generation of forms to insert instance data. Although we considered other alternatives such as Protégé-desktop [39], NeON toolkit [8], OBO-Edit [7], and TopBraid Composer [38], we found that only WebProtégé satisfied all the aforementioned requirements.

In the remainder of this section, we describe the main features of SOMM.

Insertion of axioms and constraints. We have implemented a form-based interface for editing standard axioms and constraints. Figure 3 shows a screenshot of the SOMM class editor representing the following axioms about *SteamTurbine* (abbreviated below as *ST*), where all but the last axiom represent constraints.

```
SubClassOf(ST ObjectSomeValuesFrom(hasState State))
SubClassOf(ST DataSomeValuesFrom(hasId xsd:string))
SubClassOf(ST ObjectMinCardinality(1 hasConfig STConfig))
SubClassOf(ST ObjectMaxCardinality(3 hasConfig STConfig))
SubClassOf(ST ObjectAllValuesFrom(hasProductLine ProductLine))
```

The interface shows that the class *SteamTurbine* has three mandatory properties (*hasState*, *hasID* and *hasConfig*) marked as 'Required' and interpreted as constraints, and an optional property (*hasProductLine*) interpreted as a standard

³ <http://www.cs.ox.ac.uk/isg/tools/SOMM/>.

axiom. Object and data properties are indicated by blue and green rectangles, respectively. For each property we can specify their filler using a WebProtégé autocompletion field. Finally, the fields ‘Min’ and ‘Max’ are used to represent cardinality constraints on mandatory properties.

Property (*)	Required?	Min.	Max.	Range (*)	Value
hasState	<input checked="" type="checkbox"/>	1	max	State	Enter individual
hasId	<input checked="" type="checkbox"/>	1	max	xsd:string	Enter literal value
hasConfig	<input checked="" type="checkbox"/>	1	3	SteamTurbineConfig	Enter individual
hasProductLine	<input type="checkbox"/>	min	max	ProductLine	Enter individual
Enter property name	<input type="checkbox"/>	min	max	Enter datatype or class	Individual or literal

Fig. 3. SOMM editor to attach properties to classes.

Property (*)	Value
hasState (*)	Select a value
hasId (*)	turbine_987
hasConfig (*)	SteamTurbineConfiguration
hasProductLine	Select a value

Fig. 4. Data insertion in SOMM.

Automatically generated data forms. SOMM exploits the capabilities of the ‘knowledge acquisition forms’ in Web-Protégé to guide engineers during data entry. The main use of data forms that we envision is ontology validation during the time of ontology development. The forms are automatically generated for each class based on its relevant mandatory and optional properties. For this, SOMM considers (i) the explicitly provided properties; (ii) the inherited properties; and (iii) the properties explicitly attached to its descendant classes. The latter were deemed useful by Siemens engineers, e.g., although *Turbine* does not have directly attached properties, the SOMM interface would suggest adding data for the properties attached to its subclass *SteamTurbine*. Figure 4 shows an example of the property fields for an instance of the class *SteamTurbine*, where required fields (i.e., those for which a value must be provided) are marked with (*).

Extended hierarchies. In addition to subsumption hierarchies, SOMM allows also for hierarchies based on arbitrary properties. These can be seen as a generalisation of partonomy hierarchies, and assume that the dependencies between

classes or individuals based on the relevant property are ‘tree-shaped’. Figures 5a and b show the hierarchy for the *follows* property, which determines which kinds of processes can follow other processes; for instance, *Conveying* follows *Loading* and is followed by *Testing*.

Alignment. SOMM integrates the system LogMap [16] to support model alignment and merging. Users can select and merge two Web-Protégé projects, or import and merge an ontology into the active Web-Protégé project. Although LogMap supports interactive alignment [17], it is currently used in SOMM in an automatic mode; we are planning to extend SOMM’s interface to support user interaction in the alignment process.

Reasoning. SOMM relies on Hermit [10] to support standard reasoning services such as class satisfiability and ontology classification. Data validation and query answering support is currently provided on top of the IRIS reasoner [3], as described in Sect. 3.2. Figures 5c and d illustrates the supported reasoning services. The left-hand-side of the figure shows that the class *GasTurbineModes* is satisfiable and *Process* is an inferred superclass. On the right-hand-side we can see that *steam_turbine_987* violates one of the integrity constraints; indeed, as shown in Fig. 4, *steam_turbine_987* is missing data for the property *hasState*, which is mandatory for all steam turbines (see Fig. 3).

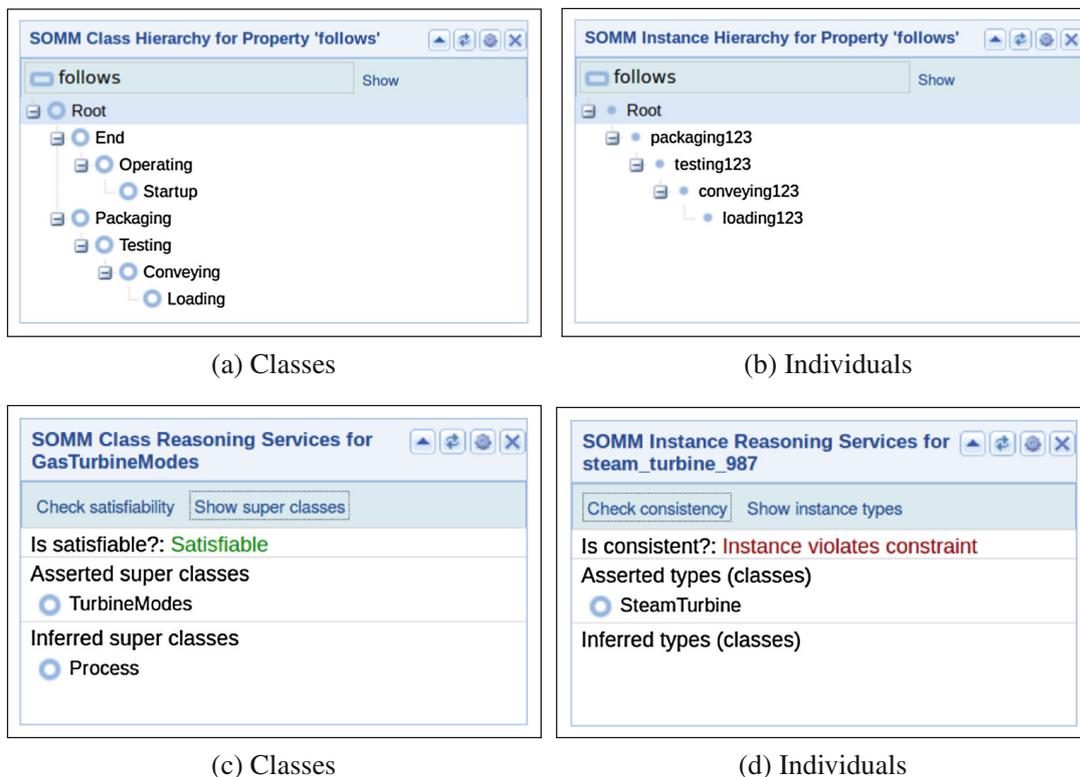


Fig. 5. Above: tree-like navigation of the ontology classes and individuals in SOMM. Below: reasoning services for ontology classes and individuals in SOMM

5 Evaluation

We have evaluated the practical feasibility of the data validation and query answering services provided by SOMM. For this, we have conducted two sets of experiments for the manufacturing and energy turbine scenarios, respectively. In the first experiment, we simulated the operation of a manufacturing plant using a synthetic generator that produces realistic product manufacturing data of varying size; in the second experiment, we used real anonymised turbine data.⁴ All our experiments were conducted on a laptop with an Intel Core i7-4600U CPU at 2.10 GHz and 16 GB of RAM running Ubuntu 14.04 (64 bits). We allocated 15 GB to Java 8 and set up IRIS with its default configuration.

Manufacturing Experiments. In our experiments for the manufacturing use case we used the ontology, data and queries given next.

- The ontology capturing the manufacturing model illustrated in Fig. 1 from Sect. 2.1. The ontology contains 79 standard axioms and 20 constraints.
- A data generator used by Siemens engineers to simulate manufacturing of products of two types based on the aforementioned model. We used two configurations of the generator: configuration (*C1*) simulates a situation where products were manufactured in violation of the model specifications (e.g., they used too much material of some kind); in (*C2*), each product is manufactured according to specifications.
- A sample of three monitoring queries commonly used in practice. The first query asks for all products that use material from a given lot; the second asks for all material lots used in a given product; finally, the third one asks for the total quantity of material in lots of a specific kind.

We generated data for 6 different sizes, ranging from 50 triples to 1 million triples. For each size, we generated one dataset for each configuration of the generator. We set up configuration *C1* so that 35 % of the manufactured products violate specification. Our experiments follow Steps 1–4 in Sect. 3.2. We checked validity of each dataset against the ontology using Steps 1–3; then, for each dataset created using *C2* we also answered all test queries (Step 4). We repeated the experiment 5 times for each dataset and configuration (i.e., 10 times for each dataset size).

Our results are summarised in Fig. 6. Times for each data size are *wall clock* time averages (in *ms*). Constraint validation time (grey bar) correspond to Step 3 in Sect. 3.2. Query answering times (blue bar) measure the time for answering the use case queries (Step 4); here, only datasets satisfying the constraints (i.e., generated using *C2*) are considered. The figure also provides the average number of constraint violations in data generated according to *C1*, and the number of triples after constraint validation.

Our results demonstrate the feasibility of our ontology-based approach to model validation and query answering in realistic manufacturing scenarios. In

⁴ We are in the process of sorting out the licenses for the ontologies and data used in our experiments; they cannot be made publicly available at this point.

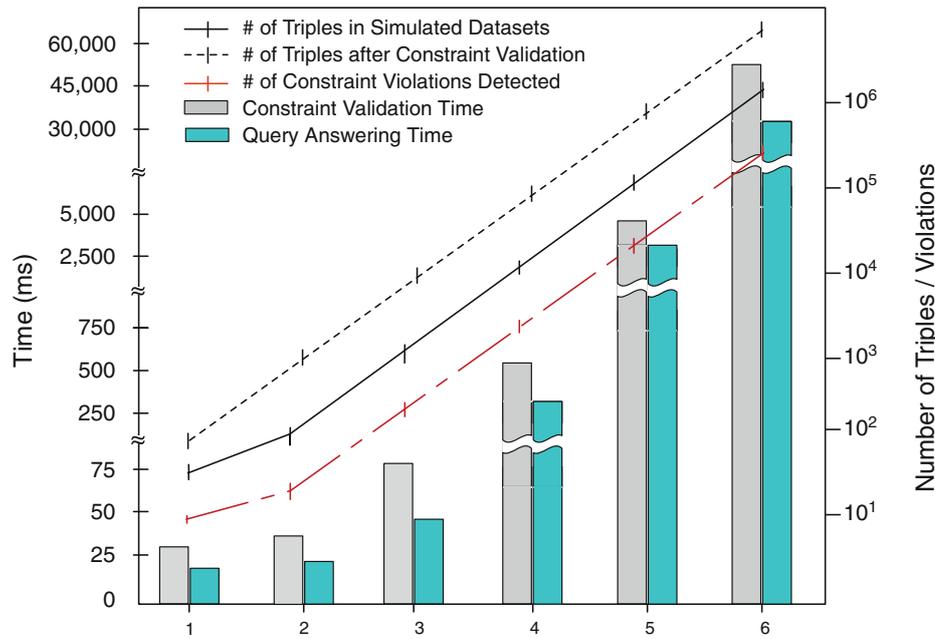


Fig. 6. Experimental results

particular, constraint validation and query answering were feasible within 87s on stock hardware over datasets containing over 1 million triples.

Gas Turbine Experiment. In this experiment we used the following data:

- The ontology capturing the energy plant model illustrated in Fig. 2 from Sect. 2. The ontology contains 121 standard axioms and 25 constraints.
- An anonymised dataset describing the structure of 800 real gas turbines, their sensor readings (temperature, pressure, rotor speed and position), and associated processes (e.g., expansion, compression, start up, shut down). The dataset was converted from a relational DB into RDF, and contains 25,090 triples involving 4,076 individuals.
- Three commonly used test queries. The first query asks for the core parts, equipment and current state of all turbines of a given type; the second asks for all components involved in a compression process; the last query asks for the temperature readings of turbines of a given type.

We followed the same steps as in the previous experiments, with very positive results. Constraint checking was completed in 2s and generated 27,007 additional triples; we found 1,582 constraint violations, which is especially interesting given that the data is real. Query answering over the valid subset took 1s on average.

6 Lessons Learned and Future Work

We have studied the use of ontologies to capture industrial information models in manufacturing and energy production applications.

Our study of the requirements of information models revealed that many key aspects of information models naturally correspond to integrity constraints and hence cannot be captured by standard OWL 2 ontologies. This demonstrates intrinsic limitations of OWL 2 for industrial modelling and gives a clear evidence of why constraints are essential for such modelling.

We also learned that even a rather simple form-based interface such as the one of SOMM is sufficient to capture most of the manufacturing and energy information models based on ISA and ICE standards. This was an important insight for us since at the beginning of this research project it was unclear whether designing such a simple tool to write ontologies of practical interest to our use cases would be feasible.

Finally, we have received a very positive feedback from Siemens engineers about the usability of SOMM at informal workshops organised as part of the project. This was encouraging since the development of a tool that is accessible to users without background in semantic technologies was one of the main motivations of our work.

In the future, we plan to conduct a formal user study where—with the help of SOMM—Siemens engineers will design elaborate information models and perform various tasks on these models, including validation and merging. We also plan to conduct more extensive scalability experiments. SOMM is a research prototype and, depending on the outcome these studies, we would like to deploy it in production departments.

References

1. Abele, L., Legat, C., Grimm, S., Muller, A.W.: Ontology-based of plant models. In: INDIN (2013)
2. Arancón, J., Polo, L., Berrueta, D., Lesaffre, F., Abajo, N., Campos, A.M.: Ontology-based knowledge management in the steel industry. In: *The Semantic Web: Real-World Applications from Industry* (2007)
3. Bishop, B., Ficsher, F.: IRIS - integrated rule inference system. In: *Workshop on Advancing Reasoning on the Web* (2008)
4. Calvanese, D., et al.: Optique: OBDA solution for big data. In: *ESWC (Satellite Events), Revised Selected Papers* (2013)
5. Classification and Product Description. <http://www.eclass.eu/>
6. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity, expressive power of logic programming. *ACM Comput. Surv.* **33**(3), 374–425 (2001)
7. Day-Richter, J., Harris, M.A., Haendel, M., Lewis, S.: OBO-Edit - an ontology editor for biologists. *Bioinformatics* **23**(16), 2198–2200 (2007)
8. Erdmann, M., Waterfeld, W.: Overview of the NeOn toolkit. In: *Ontology Engineering in a Networked World* (2012)
9. Forschungsunion. Fokus: Das Zukunftsprojekt Industrie 4.0, Handlungsempfehlungen zur Umsetzung. In: *Bericht der Promotorengruppe KOMMUNIKATION* (2012)
10. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: HermiT: An OWL 2 reasoner. *J. Autom. Reasoning* **53**(3), 245–269 (2014)
11. Gliozzo, A., Biran, O., Patwardhan, S., McKeown, K.: Semantic technologies in IBM watson. In: *Teaching NLP and CL Workshop (TNLP) at ACL* (2013)

12. Grangel-González, I., Halilaj, L., Coskun, G., Auer, S., Collarana, D., Hoffmeister, M.: Towards a semantic administrative shell for industry 4.0 components. In: ICSC (2016)
13. Grimm, S., Watzke, M., Hubauer, T., Cescolini, F., Embedded \mathcal{EL} + reasoning on programmable logic controllers. In: ISWC (2012)
14. Hepp, M., de Bruijn, J.: GenTax: a generic methodology for deriving OWL and RDF-S ontologies from hierarchical classifications, thesauri, and inconsistent taxonomies. In: ESWC (2007)
15. Hubauer, T., Lamparter, S., Pirker, M.: Automata-based abduction for tractable diagnosis. In: DL (2010)
16. Jiménez-Ruiz, E., Cuenca Grau, B.: LogMap: logic-based and scalable ontology matching. In: ISWC (2011)
17. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: algorithms and implementation. In: ECAI (2012)
18. Kagermann, H., Lukas, W.-D.: Industrie 4.0: Mit dem internet der Dinge auf dem Weg zur 4. industriellen Revolution. In: VDI Nachrichten (2011)
19. Kharlamov, E., et al.: Enabling semantic access to static, streaming distributed data with optique: demo. In: ACM DEBS (2016)
20. Kharlamov, E.: How semantic technologies can enhance data access at siemens energy. In: ISWC (2014)
21. Kharlamov, E., et al.: Ontology based access to exploration data at statoil. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 93–112. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-25010-6_6](https://doi.org/10.1007/978-3-319-25010-6_6)
22. Kharlamov, E.: Ontology-based integration of streaming and static relational data with optique. In: ACM SIGMOD (2016)
23. Kharlamov, E., et al.: Optique: ontology-based data access platform. In: ISWC (P&D) (2015)
24. Kharlamov, E., et al.: Optique: towards OBDA systems for industry. In: Cimiano, P., Fernández, M., Lopez, V., Schlobach, S., Völker, J. (eds.) ESWC 2013. LNCS, vol. 7955, pp. 125–140. Springer, Heidelberg (2013). doi:[10.1007/978-3-642-41242-4_11](https://doi.org/10.1007/978-3-642-41242-4_11)
25. Kharlamov, E., et al.: Semantic access to siemens streaming data: the optique way. In: ISWC (P&D) (2015)
26. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 web ontology language profiles (Second Edition). W3C Recommendation (2012)
27. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. *J. Web Sem.* **7**(2), 41–60 (2009)
28. Nenov, Y., Piro, R., Motik, B., Horrocks, I., Wu, Z., Banerjee, J.: RDFox: a highly-scalable RDF store. In: ISWC (2015)
29. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. *J. Data Semant.* **10**, 237–271 (2008)
30. Qiu, R.G., Zhou, M.: Mighty MESs; state-of-the-art, future manufacturing execution systems. *IEEE Robot. Automat. Mag.* **11**(1) (2004)
31. Richnow, J., Rossi, C., Wank, H.: Designation of wind power plants with the reference designation system for power plants - RDS-pp. *VGB PowerTech.* **94**, 2 (2014)
32. Ringsquandl, M., Lamparter, S., Brandt, S., Hubauer, T., Lepratti, R.: Semantic-guided feature selection for industrial automation systems. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 225–240. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-25010-6_13](https://doi.org/10.1007/978-3-319-25010-6_13)
33. Shearer, R., Motik, B., Horrocks, I.: HermiT: a highly-efficient OWL reasoner. In: OWLED (2008)

34. Siemens. Modeling new perspectives: digitalization - the key to increased productivity, efficiency and flexibility (White Paper). In: DER SPIEGEL, 6 2015
35. Stetter, R.: Software im maschinenbau-laestiges anhangsel oder chance marktfuehrerschaft? In: VDMA, ITQ (2011). <http://www.software-kompetenz.de/en/>
36. Stolz, A., Rodriguez-Castro, B., Radinger, A., Hepp, M.: PCS2OWL: a generic approach for deriving web ontologies from product classification systems. In: Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8465, pp. 644–658. Springer, Heidelberg (2014). doi:[10.1007/978-3-319-07443-6_43](https://doi.org/10.1007/978-3-319-07443-6_43)
37. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Integrity constraints in OWL. In: AAAI (2010)
38. Top Quadrant. TopBraid Composer. <http://www.topquadrant.com/>
39. Tudorache, T., Noy, N.F., Tu, S.W., Musen, M.A.: Supporting collaborative ontologydevelopment in protégé. In: ISWC (2008)
40. Tudorache, T., Nyulas, C., Noy, N.F., Musen, M.A.: WebProtégé: a collaborative ontology editor and knowledge acquisition tool for the web. In: Semantic Web 4.1 (2013)
41. Vyatkin, V., Engineering, S.: Software engineering in industrial automation: state-of-the-art review. IEEE Trans. Ind. Inf. **9**(3), 2351–2362 (2013)

Graph-based Predictions and Recommendations in Flexible Manufacturing Systems

Martin Ringsquandl

Ludwig-Maximilians Universität
Munich, Germany,

Email: martin.ringsquandl@gmail.com

Steffen Lamparter

Siemens AG, Corporate Technology
Munich, Germany

Email: steffen.lamparter@siemens.com

Raffaello Lepratti

Siemens S.p.A, Digital Factory
Genoa, Italy

Email: raffaello.lepratti@siemens.com

Abstract—Due to the emerging paradigm of mass-customization, manufacturing processes are becoming increasingly complex. Management of this complexity requires system support that goes beyond traditional MES capabilities, such as discovery of patterns throughout massive networks of interdependent processes. As of today, Manufacturing Analytics offer only limited decision support focused on descriptive metrics that cannot account for predictive and prescriptive decision support, such as detection of systematic fault patterns. The application of predictive models in manufacturing environments is non-trivial, because they need to reflect system domain constraints and preserve semantics of manufacturing operations. Recent approaches of so-called Advanced Manufacturing Analytics try to fill this gap by applying standard data mining algorithms with customized data preparation for domain-specific use cases.

In order to overcome the problem of high customization efforts, we introduce a graph-based analytics framework derived from a comprehensive requirements analysis. Additionally, we demonstrate applicability of the presented framework on two exemplary manufacturing analytics use cases.

I. INTRODUCTION

Today, manufacturing companies are challenged by the need to meet high quality demands of customers, while at the same time shorter product life cycles and mass-customization require high flexibility of operations and shop floor equipment to support fast introduction of new products [1]. Due to these emerging paradigms, manufacturing processes are becoming increasingly complex by means of various combinations and interdependencies between production operations, material definitions, and equipment configurations. It has been shown that dealing with this growing complexity is beyond capabilities of traditional Manufacturing Execution Systems (MES) and Enterprise Manufacturing Intelligence (EMI) solutions and therefore has become a key competitive factor [2].

On the other hand, the trend towards end-to-end digitalization of processes makes huge amounts of manufacturing operations and process data available for new data-centric applications that can help to reduce complexity by offering predictive decision support. For example, the detection of equipment configuration patterns in production program execution records could be used to diagnose systematic equipment configuration faults that have historically emerged due to unmanaged dependencies. These kind of data-driven applica-

tions are sometimes referred to as *Advanced Manufacturing Analytics* [3].

While integration of operations data such as inventory transactions and process data like machine alarms still remains one of the major challenges, the advancement of massively parallelized data processing has made the application of advanced analytic models to company-wide networks of interdependent processes more feasible [4]. Nevertheless, as of today, there is a lack of research devoted to filling the gap between MES (and EMI), which are still focused on limited descriptive metrics, i.e. Key Performance Indicators (KPIs), and the development of advanced analytic approaches that respect the special requirements of manufacturing environments, which are heavily influenced by structural dependencies, e.g. connections between equipment, and engineering knowledge, e.g. bill of material. We argue that semantic data models are a natural fit to bridge the gap between domain-specific requirements for analytics and their integration into existing manufacturing systems.

Considering the data integration problem, it has been shown that the strengths of semantic data models can thrive as common backbone of data integration for industrial application [5]. One step further, optimally, such a common manufacturing data model enabling data integration should also facilitate the formulation of Advanced Manufacturing Analytics, i.e. limit the manual effort needed for the preparation, execution, and respect manufacturing-specific environments. Up to now, this area of research has not been intensively studied yet.

In this paper, we derive four requirements as main enablers of *Advanced Manufacturing Analytics* by an examination of recent literature, see Section III. Based on these findings, we present a framework that is capable of formulating predictive and recommendation models with domain background knowledge in a graph-based setting. A semantic data model by design of this approach allows to respect implicit dependencies of manufacturing operations and flexible integration of different data sources, both structured and unstructured, see Section IV. Additionally, we show the applicability of the framework in several exemplary real-world use cases in Section V.

II. RELATED WORK

Recently, there have been a number of works concerned with to the development of advanced analytic frameworks

for manufacturing, e.g. for the monitoring of process states based on cluster prediction using Support Vector Machines (SVM) [6], Quality Control using Genetic Algorithms [7] and process optimization on top of a holistic manufacturing data warehouse and standard Data Mining techniques [3], [8]. Usage of semantics as model guidance has been studied in an aerospace use case [9].

Other approaches have been suggested for the automatic learning of abnormal behavior of whole production systems using Rule Induction [10], [11]. Also Artificial Neural Networks (ANN) have been proposed focused on the aspect of process interdependencies and error propagation in multistage manufacturing [12].

The shortcoming of all the mentioned approaches is that they do not respect the domain constrained nature of manufacturing operations, e.g. considering known structural dependencies. There is also a lack of interpretability for complex models (ANN, SVM) since they function more or less as a black box. Plus these approaches rely on a customized model design in terms of feature engineering, which usually requires high domain expertise. This point has also been recently stressed under the consideration of a domain-specific language for predictive manufacturing analytics [13].

In this paper, we present a framework that overcomes both of these problems by using semantic abstraction and a graph-based representation of the data.

III. REQUIREMENTS ANALYSIS

In this section we want to shed light on what challenges are still to be met in order to enable efficient execution of *Advanced Manufacturing Analytics*. As an introduction, consider the predictive analytic task:

Example 1: Given a set of operation execution instances D , where each instance d_i references a set of equipment parameters C , e.g. programs loaded onto production machines, and a set of bill of material (BOM) definition properties B , e.g. mixing ratio of ingredients. We want to predict, if an execution will yield an excessive amount of scrap, labeled as $y_i = bad$. In a highly flexible environment, assuming each parameter can take on p different values, we have $p^{|D| \times |C|}$ different combinations of equipment and BOM configurations. A predictive model should be able to find patterns of the form $\{S | S \subseteq C \cup B\} \Rightarrow y$, i.e. a systematic combination of equipment parameters and bill of material properties that influence quality of produced final material.

Based on a comparison of recent trends and existing approaches in literature, we gather four major requirements that should be considered when implementing predictive analytics in a manufacturing environment [14], [3]. The template in Table I specifies for each derived requirement the problem of existing approaches, a possible solution and its benefits.

The importance of *Adaptability* becomes evident in dynamic environments with constantly changing configurations where predictive model must still provide robust and reliable results

TABLE I
REQUIREMENTS OF ADVANCED MANUFACTURING ANALYTICS DERIVED FROM EXISTING APPROACHES

Requirement 1: Adaptability	
Problem	Customized realizations of analytics that are constrained to certain data models and use cases
Solution	Abstracting analytics from predefined sources, fixed data models and variables
Benefits	Reduce manual design effort and speed up introduction of changes
Requirement 2: Comprehensiveness	
Problem	Analytics for dedicated process steps ignore potential interdependencies
Solution	Incorporate a holistic view of all available aspects of manufacturing processes
Benefits	Broader range of optimization potential and discovery of unknown dependencies
Requirement 3: Domain-specificity	
Problem	General-purpose data analytics do not reflect domain knowledge
Solution	Allow specification of manufacturing domain constraints
Benefits	Increase chance of finding accurate, non-trivial, and unknown insights
Requirement 4: Interpretability	
Problem	Lost semantics prohibit interpretation of gained results for domain experts
Solution	Preserve context, domain vocabulary and background knowledge in results
Benefits	Directly actionable insights ensured via preserved domain vocabulary

[15]. This means, referring to the previous example, new combinations of configurations and properties should not require any changes to pattern detection model. Especially, since there is a gap between data analytic expertise and manufacturing domain knowledge, it is needed to limit the amount of manual work that is put into customization.

The need for *Comprehensiveness* is widely accepted in multi-stage manufacturing where interactions between processes need to be considered [7]. Plus, with the growing trend towards increasing availability of data, even of small manufacturing companies [13], this data should be exploited as much as possible. For instance in Example 1, including multiple processes the growing sizes of B and C simply entail an expanding space of possible explanations, which means more potential patterns should be detected.

There are some special characteristics of manufacturing data, due to the fact that MES and also EMI solutions have been developed using relational schema definitions, usually based on standards such as ISA-95 (IEC-62264). Common approaches need to prepare this relational data for analysis with some form of propositionalisation that flattens entities linked through multiple relations into one relation, cf. [8]. This is not an efficient representation of manufacturing data, where equipment structure, material, and process definitions

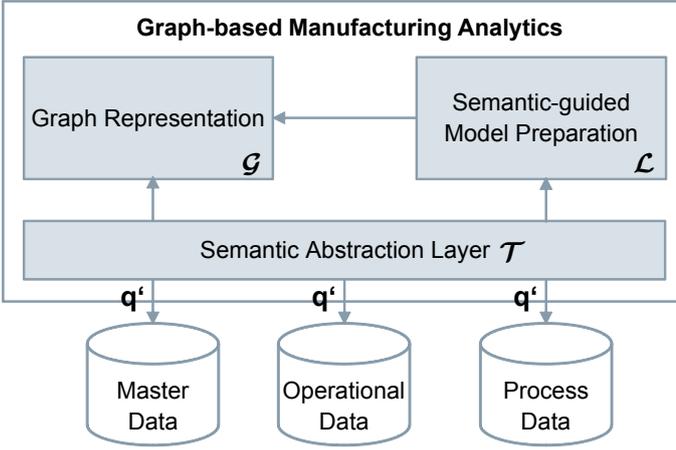


Fig. 1. Components of Graph-based Analytics Architecture

shape the way data is generated. Advanced analytics should respect these dependencies, i.e. support some form of built-in *Domain-specificity*. Consider the case in Example 1 when it is known that two consecutive operation executions d_i and d_j process the same work piece, there is an implicit connection in the data that a robust pattern detection algorithm must be aware of. In order to make analytic results as insightful as possible, *Interpretability* is one of the major challenges for manufacturing. This is why, for example, Decision Trees have been very popular as predictive models for manufacturing analytics in the past [8]. However, as stated above, propositionalisation of relational data significantly limits interpretation of relations between entities. Therefore, there is a need for other approaches that preserve relations with their natural semantics.

In contrast to common approaches, we directly express Advanced Manufacturing Analytics in a graph-based representation that keeps natural semantics of the data, does not require propositionalisation, and facilitates specification of domain constraints.

IV. ARCHITECTURE

In the following, we present a proposal for the fulfillment of all requirements described in the previous section, which is built around a semantic data model that serves as abstraction to the actual data and the analytic models.

The resulting components of this graph-based architecture are shown in Figure 1.

A. Semantic Abstraction Layer: Ontology-based Data Access

This component is responsible for mapping instance data from sources into a common schema described by the ontology \mathcal{T} . For our system, we use an ontology-based data access (OBDA) technique.

OBDA relies on the definition of *mappings*, e.g. using the W3C standard *R2RML* language. These mappings relate the signature of \mathcal{T} to the vocabulary of each data source. This allows an OBDA system to take a query q that is formulated in the abstract signature (terminology) of ontology \mathcal{T} and re-write it into a logically equivalent, but semantically-enriched

query q' . Thus, q' absorbs parts of the ontology to retrieve all entailed answers. After re-writing q' can readily be translated into an SQL query to the underlying source using the previously defined mappings.

Example 2: Consider the tables `Tags` and `OperationExecution` in two relational databases. The following mapping establishes the semantic **occursAt** relation between execution records and events:

```
occursAt ≡ SELECT t.Id, o.Id FROM Tags t
JOIN OperationExecution o ON t.TimeStamp
BETWEEN o.TimeSpan
```

Further, it is possible to instantiate concepts, e.g. for the class **RFID-Event** like follows:

```
RFID-Event :- SELECT t.Id FROM Tags t WHERE
Message IN ['Glimpsed', 'Observed']
```

Also as part of the ontology we define implicit dependencies of manufacturing environments that are typically well-known to operators and engineers.

Definition 1 (Dependency Links): The set of dependency links \mathcal{L} reflects engineering background knowledge about known causality between two entities e_i and e_j , as assertion $\text{dependsOn}(e_i, e_j)$.

Figure 2 gives an overview of these dependency structures. It can be seen that manufacturing processes are specified throughout different levels of detail. Starting at high-level representations including BOM and Process Routing from engineering up to more detailed Manufacturing BOM. During the actual execution, when data D is stored, the dependency information is typically lost. However, by defining these structures explicitly as part of the semantic abstraction layer, connections can be restored during data integration.

For example, we can express the dependency between *Process 4* and *Product 2* –since this process step is unique to this product – by asserting: $\text{dependsOn}(\text{Process4}, \text{Product2})$.

B. Model Preparation

This component takes care of instance and feature selection. In contrast to traditional statistical approaches, feature selection is done directly using dependency links without looking at actual instance data [16]. For example, if it is known that some particular equipment configuration properties are always co-occurring, there is no need to include such redundant variables as features in the model.

C. Graph-based Representation

After integration and preparation, final data D is represented in form of a set of multi-graphs – as it is common in Linked Data, e.g. via Resource Description Framework (RDF).

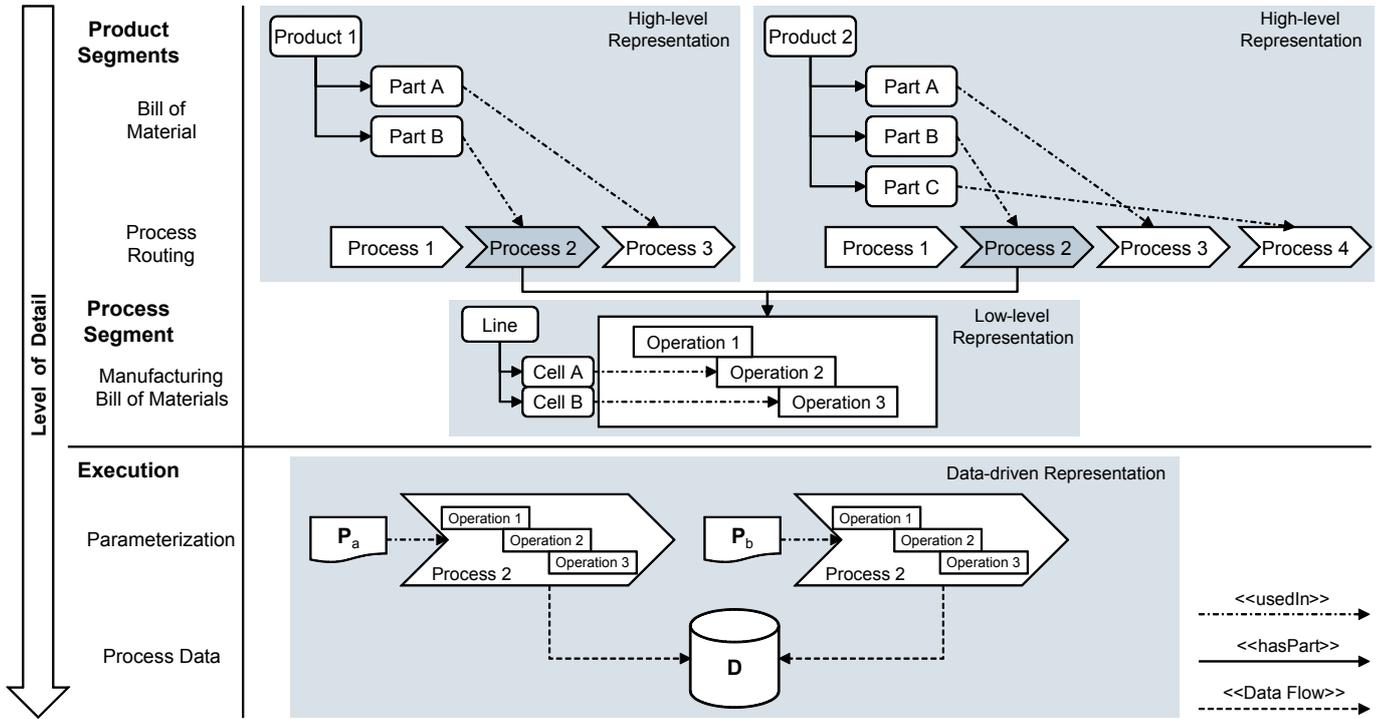


Fig. 2. Material, Process and Equipment Dependencies at different Layers of Manufacturing Data.

Definition 2 (Graph Representation): Instance data in D is represented as multi-graph $G = \langle V, E_l \rangle$, where entities are in the vertex set V and pairs of entities connected through relation l in the edge set E_l . As defined above, there is a special dependency sub-graph \mathcal{L} . The set of these multi-graphs is denoted as \mathcal{G} .

The benefits of a multi-graph representation is that the unique semantics of entities and relations are preserved.

D. Requirements Discussion

To summarize, we briefly discuss the major pinpoints of how this framework satisfies requirements of Section III.

First, flexible data integration (*Adaptability*) is possible by abstracting from the concrete physical model of each data source via *mappings*. This way, instance data can be integrated in an on-demand fashion. In addition to that, concept definitions of ontology \mathcal{T} ensure usage of domain vocabulary (*Interpretability*) that is independent of individual schemata, e.g. table definitions and column identifiers. The benefit of this is that results of analytics are not presented by using identifiers, for example, machine tags, but using domain terminology as agreed upon in the concept definitions. Naturally, the graph representation enables an holistic view over all relevant manufacturing processes (*Comprehensiveness*) by being able to simply integrate additional data sources. This means that analytics can be scaled from single to multiple processes without any changes. Furthermore, background knowledge in form of dependencies can be exploited by analytics (Domain-specificity). This is important for real-world applications of

Manufacturing Analytics where inferring statistical dependencies between entities that are already known in advance is not efficient. If background knowledge already indicates causation, this should be considered by analytic models.

In the following sections, we show how this architecture can be readily applied for detection of patterns and providing recommendations.

V. ADVANCED MANUFACTURING ANALYTICS USE CASES

As a running example, we refer to a similar scenario of production executions as discussed in the introductory example of section III having different production equipment and bill of material properties in a flexible manufacturing environment.

A. Predictive Model: Root Cause Analysis

1) *Use Case Scenario:* Root Cause Analysis (RCA) is the task of finding the source of an undesirable state or event. In this scenario, the undesirable state occurs if an excessive amount of scrap is produced by an automated welding operation executed with respect to a certain program. Given data records D consisting of operation execution entities that are labeled as good or bad, e.g. based on the amount of scrap produced, RCA is defined as detecting patterns that distinguish good from bad operation executions.

Figure 3 shows how this scenario looks like in our graph-based framework. For sake of simplicity, in this example only equipment structure is used, as defined in IV. From the bottom of Figure 3, the relational schema of operation executions is integrated through the semantic abstraction layer. Additionally, a second data source of events is aligned with the execution records.

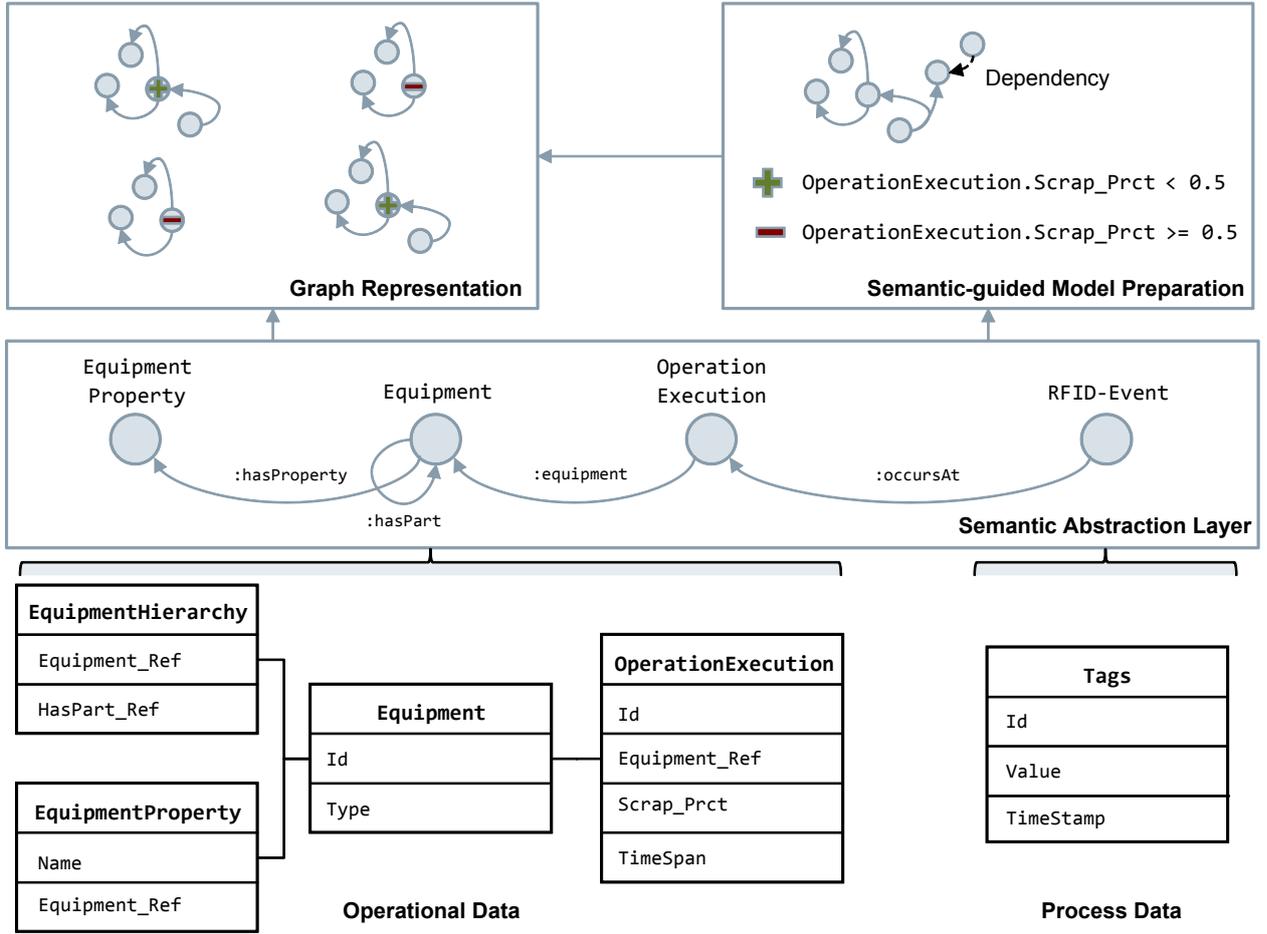


Fig. 3. Operation Executions: From Relational to Graph Representation

2) *Frequent Pattern Detection*: For pattern detection, operation executions are represented in form of positive and negative labeled graphs \mathcal{G}^+ and \mathcal{G}^- , therefore we want to find frequent sub-graph patterns S w.r.t. the following measure:

$$S = \max \frac{|\{g \in \mathcal{G}^+ | S \not\subseteq g\}| + |\{g \in \mathcal{G}^- | S \subseteq g\}|}{|\mathcal{G}^+| + |\mathcal{G}^-|} \quad (1)$$

This means interesting patterns are the ones that occur frequently in negatively labeled graphs and rarely in positive ones. As an example, Figure 4 shows two frequent sub-graph patterns that occur in *bad* executions. This pattern suggests that material check-in did not correctly identify the RFID tag, i.e. only "glimpsed" it, and therefore the false program was loaded onto the welding machine.

3) *Causation Patterns*: Since some dependency links are known to the system, it is more efficient to consider patterns as interesting that only contain causation entities, i.e. the right-hand side of dependsOn relation.

Through the dependency links, it is possible guide pattern search, see Figure 4, where pattern S and S' are separately detected, but actually only the event pattern is relevant.

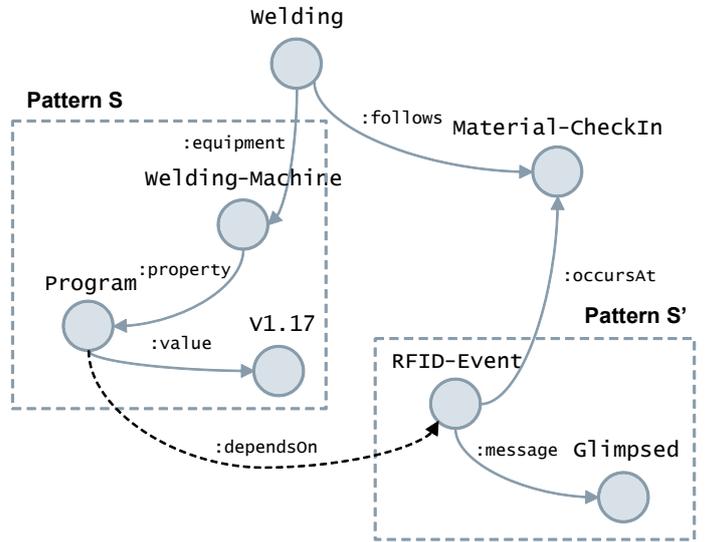


Fig. 4. Example Pattern Extracted from Operation Executions

B. Equipment Parameter Recommendations

1) *Use Case Scenario*: Going one step further from predictions to recommendations, we want to present patterns of

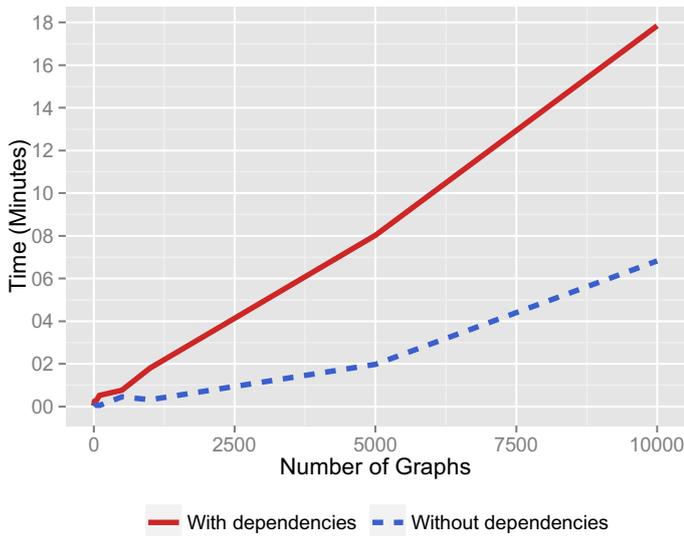


Fig. 5. Graph Pattern Detection Runtime

equipment parameters to the operators that ensure optimal production.

2) *Recommendation Model*: Again having a set of discovered positive execution graphs G^+ , current equipment parameters can be evaluated with respect to previously discovered patterns. If current parameters match one of the negative patterns, the task is to provide the minimum set of parameter corrections as recommendation that would yield a positive pattern. This can be formally defined as graph edit distance from current execution graph g to its closest positive pattern g' .

$$S = \arg \min_{g' \in G^+} \kappa(g, g') \quad (2)$$

where κ is a graph similarity measure. Such recommendations can reduce complexity by providing decision support to plant operators and workers.

VI. PRELIMINARY RESULTS

We evaluate our framework with a prototypical implementation based on *gSpan* algorithm for frequent graph pattern detection [17]. Simulated sample data was scaled from one operation execution graph to 10000, where the average graph has $|V| = 20$ entities and $|E| = 15$ edges. It can be seen that integrating dependency links significantly reduces runtime of graph pattern detection. There are other potential optimization possibilities in exploiting semantics of structure models in guiding pattern search, for example starting at entities that are known to have greater variation.

VII. CONCLUSION

In this paper we presented a new graph-based framework for Advanced Manufacturing Analytic. Referring to recent literature, we discussed requirements that are not yet or only partially met by current approaches. Our proposed architecture represents manufacturing data as a multi-graph with an semantic abstraction layer for flexible data integration. Applicability

of predictive and prescriptive analytic models was shown on the graph-based representation as well as considering special characteristics of manufacturing environments specified as background knowledge. Preliminary results look promising and we see further research possibilities in the usage of more extensive background knowledge, such as assigning constraints to entities and relations in the graph.

REFERENCES

- [1] F. Jovane, E. Westkämper, and D. Williams, *The ManuFuture road: Towards competitive and sustainable high-adding-value manufacturing*. Springer, 2009.
- [2] J. Jäger, A. Kluth, A. Schatz, and T. Bauernhansl, "Complexity patterns in the advanced complexity management of value networks," *Procedia CIRP*, vol. 17, pp. 645–650, 2014.
- [3] C. Gröger, "Advanced Manufacturing Analytics: Datengetriebene Optimierung von Fertigungsprozessen," Dissertation, Universität Stuttgart, 2015.
- [4] V. Jirkovsky, M. Obitko, P. Novak, and P. Kadera, "Big Data Analysis for Sensor Time-Series in Automation," in *Proc. of Emerging Technology and Factory Automation (ETFA)*, 2014, pp. 1–8.
- [5] E. Kharlamov, N. Solomakhina, O. L. Özçep, D. Zheleznyakov, T. Hubauer, S. Lamparter, M. Roshchin, A. Soyly, and S. Watson, "How semantic technologies can enhance data access at siemens energy," in *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference*, 2014, pp. 601–619.
- [6] T. Wuest, C. Irgens, and K.-d. Thoben, "An approach to monitoring quality in manufacturing using supervised machine learning on product state data," *Journal on Intelligent Manufacturing*, vol. 25, pp. 1167–1180, 2014.
- [7] a. R. Khan, H. Schioler, T. Knudsen, and M. Kulahci, "Statistical data mining for efficient quality control in manufacturing," *Emerging Technologies Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pp. 1–4, 2015.
- [8] C. Gröger, F. Niedermann, and B. Mitschang, "Data mining-driven manufacturing process optimization," *Proceedings of the World Congress on Engineering*, vol. III, pp. 0–6, 2012.
- [9] S. Strasser, J. Sheppard, M. Schuh, R. Angryk, and C. Izurieta, "Graph-based ontology-guided data mining for D-matrix model maturation," *IEEE Aerospace Conference Proceedings*, 2011.
- [10] D. Metz, S. Karadgi, U. Müller, and M. Grauer, "Self-Learning Monitoring and Control of Manufacturing Processes Based on Rule Induction and Event Processing," *Processing of the Fourth International Conference on Information, Process, and Knowledge Management (eKnow)*, pp. 88–92, 2012.
- [11] A. Vodencarevic, O. Niggemann, and A. Maier, "Using behavior models for anomaly detection in hybrid systems," in *23rd International Symposium on Information, Communication and Automation Technologies-ICAT 2011*, 2011.
- [12] P. Jiang, F. Jia, Y. Wang, and M. Zheng, "Real-time quality monitoring and predicting model based on error propagation networks for multistage machining processes," *Journal of Intelligent Manufacturing*, vol. 25, pp. 521–538, 2014.
- [13] D. Lechevalier, A. Narayanan, and S. Rachuri, "Towards a domain-specific framework for predictive analytics in manufacturing," *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, pp. 987–995, 2015.
- [14] T. Bauernhansl, *Industrie 4.0 in Produktion, Automatisierung und Logistik*, T. Bauernhansl, M. ten Hompel, and B. Vogel-Heuser, Eds. Wiesbaden: Springer, 2014.
- [15] M. Ringsquandl, S. Lamparter, and R. Lepratti, "Estimating Processing Times within Context-aware Manufacturing Systems," in *Proc. of the 2015 IFAC Symp. on Inf. Contr. Manuf. (INCOM 2015)*, 2015.
- [16] M. Ringsquandl, S. Lamparter, and S. Brandt, "Semantic-Guided Feature Selection For Industrial Automation Systems," in *Proc. of the 14th Int. Sem. Web Conf. (ISWC)*, 2015.
- [17] X. Yan and J. Han, "gSpan: Graph-Based Substructure Pattern Mining," *Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02)*, vol. 1, no. d, pp. 721–724, 2002.

Semantic-Guided Feature Selection for Industrial Automation Systems

Martin Ringsquandl¹(✉) , Steffen Lamparter² , Sebastian Brandt² ,
Thomas Hubauer², and Raffaello Lepratti³

¹ Ludwig-Maximilians University, Munich, Germany
martin.ringsquandl.ext@siemens.com

² Siemens AG Corporate Technology, Munich, Germany
{steffen.lamparter,sebastian-philipp.brandt,thomas.hubauer}@siemens.com

³ Siemens AG Digital Factory, Nuremberg, Germany
raffaello.lepratti@siemens.com

Abstract. Modern industrial automation systems incorporate a variety of interconnected sensors and actuators that contribute to the generation of vast amounts of data. Although valuable insights for plant operators and engineers can be gained from such data sets, they often remain undiscovered due to the problem of applying machine learning algorithms in high-dimensional feature spaces. Feature selection is concerned with obtaining subsets of the original data, e.g. by eliminating highly correlated features, in order to speed up processing time and increase model performance with less inclination to overfitting. In terms of high-dimensional data produced by automation systems, lots of dependencies between sensor measurements are already known to domain experts. By providing access to semantic data models for industrial data acquisition systems, we enable the explicit incorporation of such domain knowledge. In contrast to conventional techniques, this semantic feature selection approach can be carried out without looking at the actual data and facilitates an intuitive understanding of the learned models. In this paper we introduce two semantic-guided feature selection approaches for different data scenarios in industrial automation systems. We evaluate both approaches in a manufacturing use case and show competitive or even superior performance compared to conventional techniques.

Keywords: Semantic data models · Feature selection · Automation systems · Machine learning

1 Introduction

Processing and mining of large data sets in modern industrial automation systems is a major challenge due to the vast amount of measurements generated by several types of field devices (e.g. sensors, controllers, actuators). Deployment of machine learning models requires upfront feature selection in order to obtain a reduced feature set, thereby speeding up processing time and preventing overfitting, while still preserving inherent characteristics of the original data. Even

in the age of massively distributed data processing, feature selection remains one of the main problems in automation, since it is a highly domain expertise-intensive task [7]. On the other hand, data generated by engineered systems exhibits many structural dependencies that domain experts are well aware of. This holds especially for industrial automation systems that are systematically planned and simulated before going into production. For example, for a given electric motor, it is documented how torque, speed and power measurements relate to each other. Thus, there is no need to compute correlations between them for asserting statistical dependence.

These computations are common in most of today’s feature selection techniques, therefore they exhibit some major disadvantages when applied in high-dimensional data as observed in today’s automation systems [13]. By accessing huge proportions of the original data, they quickly become computationally expensive, plus they are prone to losing valuable information, especially when transforming the feature space to lower dimensions so that the remaining variables can no longer be intuitively interpreted. Motivated by the commonly faced difficulties of *a)* processing vast amounts of data and *b)* integrating domain knowledge into learning models, the semantic guidance approaches of this paper were developed in order to facilitate what remains the most expertise intensive task – feature selection.

In general, there are two different types of high-dimensional data that need to be considered in separation. The first case is present if we are given a huge number of both features and instances. In this scenario, we argue for an approach, in analogy to the notion of the usage of *OWL 2 QL* for ontology-based data access (OBDA), that makes it possible to perform feature selection on *TBox* level rather than instance (data) level [10]. The other case is given when there are fewer instances than features, also called *sparse* data (e.g. rare events such as device failures). For this type of data, embedded feature selection techniques like Lasso have shown to be very effective [5]. Therefore, we also introduce an embedded feature selection approach that leverages from engineering background knowledge in semantic data models. The subsequent sections will describe both approaches and their application in more detail.

2 Application: Manufacturing Use Case

As an application scenario, consider multiple assembly lines that are part of a car door production facility. The core of each assembly line is responsible for welding the window frame and inner door panel, which happens in a semi-automated fashion, as the actual welding is done by a human worker. The overall system consists of an automated frame loading station that is responsible for putting window frames on a conveyor kit. These kits are then routed through the core assembly process by an electrically-operated conveyor. After the products have been assembled, they must go through a final quality control that verifies integrity of certain product characteristics. If quality conforms to specification, the product is sent to an outgoing packaging station.



Fig. 1. Door assembly process and associated measurements

Plant operators are responsible for planning and scheduling of operations based on incoming orders. For this task, the operators have to assess uncertainties such as varying cycle times of each produced piece. Since production processes are of stochastic nature, it is non-trivial to get a solid estimate of the time when a certain product will be finished. In this case, decision support can be given by training advanced regression models that help to provide more robust and up-to-date time estimates. During production, all of the devices (e.g. light-barrier sensors, power meters, etc.) generate task-specific measurement data as shown in Figure 1. Today, data collection is agnostic of any machine learning tasks, as it is merely concerned with high-throughput historic data storage. As far as data analytics is concerned, any (sub-)set of the measurements taken could possibly be relevant for the time estimate regression task. A kind of brute-force approach would be to use all present data and try to train, for example, an ordinary least squares regression (OLS) model. However, this approach has some major shortcomings, since the OLS will include irrelevant and redundant information for fitting its coefficients and is very likely to overfit to particular patterns in the training data, therefore it is not going to generalize well in a live production scenario.

Consider the two regression models on the left-hand side of Figure 2 that try to predict cycle times. In this small example, employing five different predictor variables causes the model to overfit, while after p-value-based feature selection we obtain a more smoothed fit using only `ConveyorTime`. On the right-hand side it can be seen how constraining the five predictors by a regularization penalty reduces their coefficients until they are effectively set to zero. For example, `LoadingProductWeight` quickly gets eliminated due to its small effect on the regression task. Since it is known that product weight is part of the overall weight feature, it could have been removed beforehand without any computation. Throughout this paper, we will relate to this learning problem of forecasting product-specific cycle times in an automated manufacturing system given a huge number of different sensor measurements as a running example.

3 Preliminaries

In this section, we first introduce the notion of semantic data models in the manufacturing domain in 3.1 and how their graph representation is used to measure structural similarity of feature entities. This is followed by a description of the general problem of embedded feature selection in linear models in 3.2.

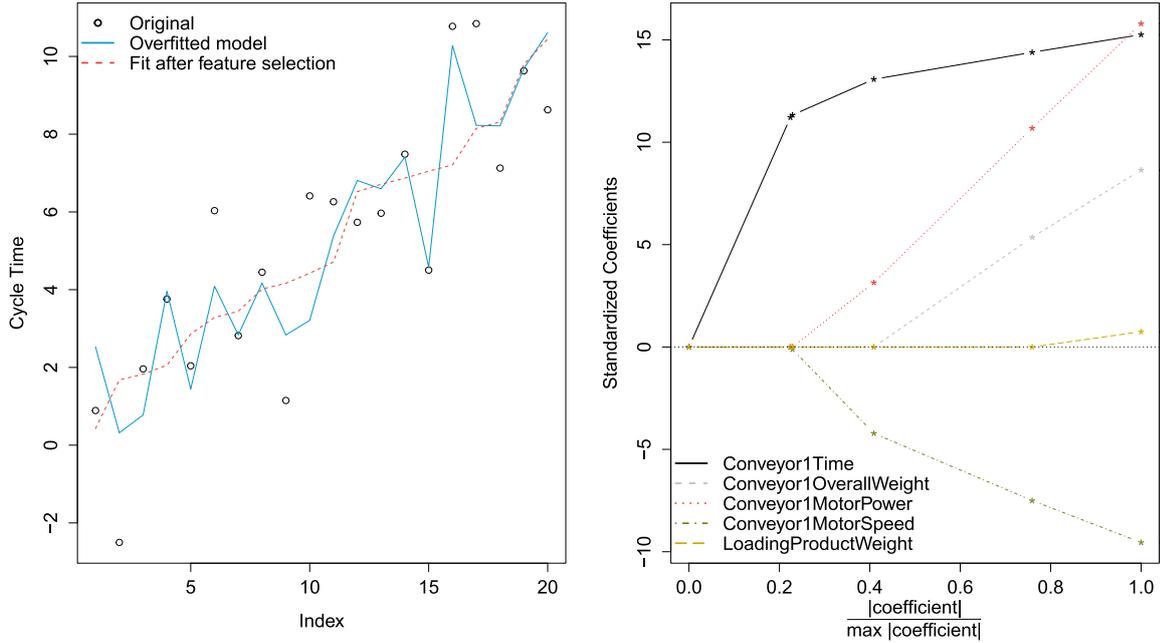


Fig. 2. Visualization of multiple linear regression model. Left: Overfitted model using all predictor variables vs. model after feature selection. Right: Coefficient values under decreasing regularization penalty

3.1 Semantic Representation of Manufacturing Data

Instead of having yet another information model language, we argue for the usage of the well-established Semantic Web standards to create, link, and share information models of automation systems in the manufacturing domain. For the purposes of feature selection, we augmented and tailored the Semantic Sensor Network (SSN) ontology¹ to meet the requirements of describing features in automation data, as can be seen in Figure 3. Here, the **Feature** concept is modeled as subclass of `ssn:InformationEntity`. Resorting to SSN is also beneficial for manufacturing systems, since devices and processes can naturally be integrated into its schema. Further details of this *feature ontology* are given in section 4.1. The graph representation of RDF-based² ontologies is a suitable property that we want to exploit for the description of dependencies between machine learning features. In formal terms, an RDF graph can be defined as a multi-graph.

Definition 1 (RDF Graph). *An RDF graph is a multi-graph $G = \langle V, E \rangle$ where each edge $e_i \in E$ is defined as triple (s, p, o) : $s, o \in V$ and p is the edge's label.*

This formal definition allows us to specify the degree of similarity between feature entities in the graph by means of common structural patterns. Graph kernel

¹ <http://www.w3.org/2005/Incubator/ssn/ssnx/ssn>

² <http://www.w3.org/RDF/>

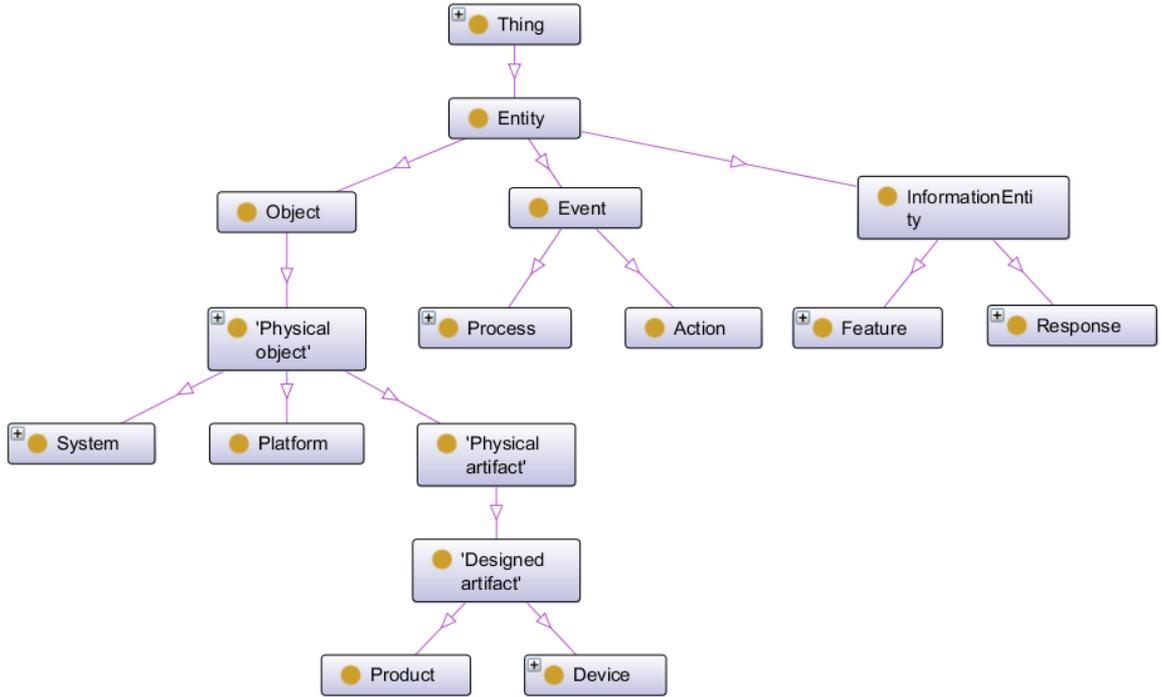


Fig. 3. Taxonomy of manufacturing feature ontology

functions have been shown to work well for capturing such patterns. The emerging field of machine learning in Linked Data has brought up a number of graph kernel functions particularly designed for RDF graph data.

Definition 2 (RDF Graph Kernel). *A graph kernel function is any function $\kappa : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ s.t. for all $G_i, G_j \in \mathcal{G}$ satisfies $\kappa(G_i, G_j) = \langle \phi(G_i), \phi(G_j) \rangle$ is a valid kernel, where \mathcal{G} is the space of RDF graphs and ϕ is a mapping to some inner product space.*

Given n entities, the graph kernel can be denoted as kernel matrix:

$$\kappa = \begin{bmatrix} \kappa(G_1, G_1) & \kappa(G_1, G_2) & \dots & \kappa(G_1, G_n) \\ \kappa(G_2, G_1) & \kappa(G_2, G_2) & \dots & \kappa(G_2, G_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(G_n, G_1) & \kappa(G_n, G_2) & \dots & \kappa(G_n, G_n) \end{bmatrix}$$

In order to get pairwise similarities between all feature entities in our feature ontology, we can resort to one of the state-of-the-art graph kernels [8]. The idea of graph kernels is that every entity can be represented as the graph that is spanned by its adjacent entities up to a certain depth d . Then, similarity between two graphs is given by some metric, e.g. size of the intersection graph or pairwise isomorphisms like in the popular family of Weisfeiler-Lehman graph kernels [3]. In Figure 4 a simplified example of the graph spanned by feature `Conveyor1Speed` is shown at different levels of depth d . Data properties of entities like RDF literals (formatted in *italic style*) are usually considered to belong

to their respective entity and therefore they do not span a new depth level. Clearly, quality of similarity calculations depends on the amount of knowledge put into the ontology creation process. Nevertheless, we expect that already a small number of annotations can support feature selection.

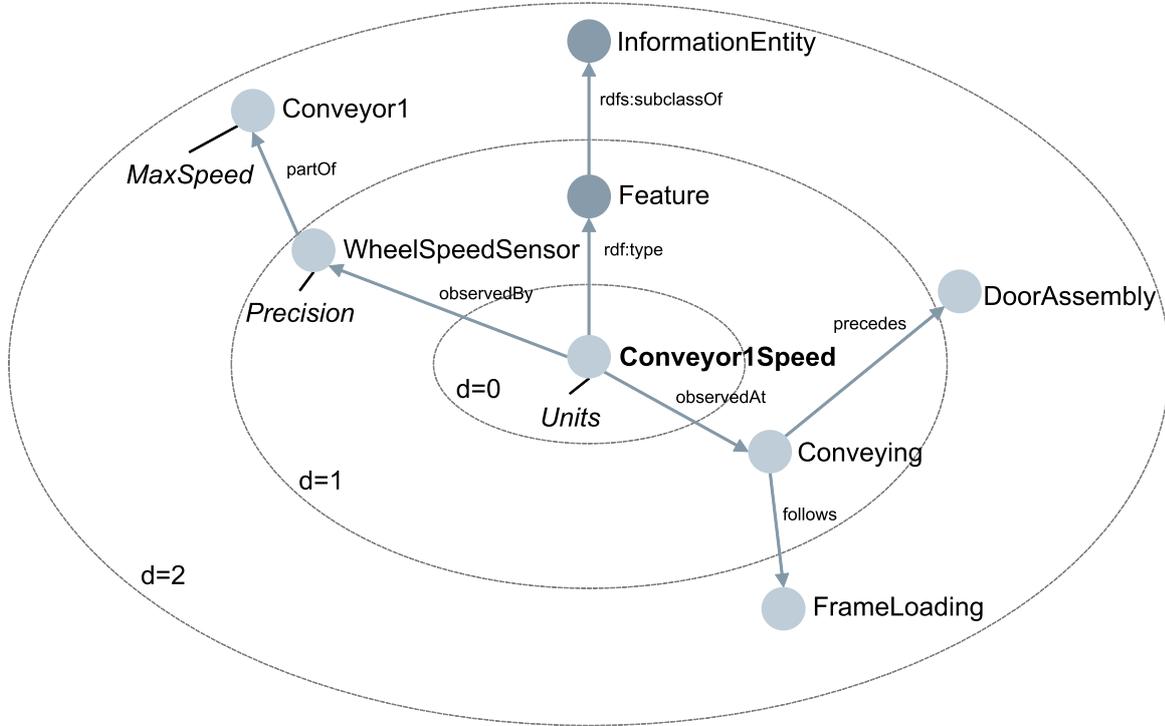


Fig. 4. Neighborhood graph of feature **Conveyor1Speed** at different depth values

Before describing how to exploit this notion of similarity between features in the training procedure of the manufacturing machine learning models, a general introduction to learning linear models is given.

3.2 Linear Model Embedded Feature Selection

Linear models are still one of the most popular machine learning models, especially in domains, where the emphasis lies on insights gained from looking at the model's coefficients. For example, the coefficients of the cycle time estimator regression can be interpreted for decision support in order to take action and reduce their influence on the overall cycle time. In case of sparse data, embedded feature selection techniques have shown to be very effective compared to other conventional feature selection. In the subsequent, we will introduce some standard formal notation of generalized linear models and their sparsity-inducing feature selection ability.

Given a training set $\{x_i, y_i\}_{i=1}^n$ where $x_i \in \mathcal{R}^p$ is a p -dimensional feature vector and $y_i \in \mathcal{R}$ is the response, i.e. for regression or classification. We consider

learning a linear model $h : R^p \rightarrow R$ with $h(\mathbf{w}) = \mathbf{w}^T \mathbf{x}$, where \mathbf{w} is a parameter vector. The general form of the regularized optimization problem is:

$$\operatorname{argmin}_{\mathbf{w}} l(y, h(\mathbf{w})) + \lambda \Omega(\mathbf{w}) \quad (1)$$

Here, $l(\cdot)$ denotes the loss function and $\Omega(\cdot)$ is the regularization term, also called penalty. The value of λ controls how much weight is given to the penalty, which is used to prevent overfitting of large parameter values. Setting $l(\cdot)$ to the square loss and $\Omega(\cdot)$ to the ℓ_1 -regularization results in the standard Lasso model:

$$\hat{\mathbf{w}}_{Lasso} = \operatorname{argmin}_{\mathbf{w}} (y - h(\mathbf{w}))^2 + \lambda \|\mathbf{w}\|_1 \quad (2)$$

The ℓ_1 -norm can be used for embedded feature selection by increasing the amount of shrinkage (λ) in the Lasso model, which effectively sets non-influencing components of \mathbf{w} to zero.

Due to their embedded feature selection ability, Lasso models have gained increasing attention for learning in sparse data sets, where the number of features is high, but many of them are irrelevant to the learning task [12]. Furthermore, in some applications, we want to include prior domain knowledge about relationships between features, for example if we know that motor speed and torque are depending on each other, they should also have similar influence (i.e. parameter weight) on the response variable. When features are represented in a graph structure, this quality is often called the smoothness property. The notion is as follows: If we specify relationships between features as *undirected* graph $G = \langle V, E \rangle$, the graph Lasso (Glasso) can be defined as

$$\begin{aligned} \hat{\mathbf{w}}_{GLasso} &= \operatorname{argmin}_{\mathbf{w}} l(y, h(\mathbf{w})) + \lambda (\alpha \|\mathbf{w}\|_1 + (1 - \alpha) \sum_{i,j \in E} (w_i - w_j)^2) \\ &= \operatorname{argmin}_{\mathbf{w}} l(y, h(\mathbf{w})) + \lambda (\alpha \|\mathbf{w}\|_1 + (1 - \alpha) \mathbf{w}^T L \mathbf{w}), \end{aligned} \quad (3)$$

where the second regularization term encourages connected features in the graph to have similar weights (smoothness). It can be preferably weighted against ℓ_1 by decreasing the α parameter. A more convenient formulation of the sum over squared weight differences is given by $\mathbf{w}^T L \mathbf{w}$, where L is the Laplacian matrix of the graph.

4 Approach

This section presents the main technical discussion of the developed semantic-guided feature selection approach. First, an introduction to concepts and axioms in the use case feature ontology is given, followed by a description of the semantic feature selection procedure. Ultimately, we present a custom linear model designed for embedded feature selection in RDF graphs.

4.1 Feature Ontology

There is a wide variety of modeling standards for manufacturing data that are used to facilitate interoperability of different systems, for example the exchange of material information between warehouse management and manufacturing execution systems. Recent developments of the OPC UA³ standard are concerned with a unified information model of field device descriptions, PLC programs, interfaces to enterprise levels such as ERP systems, and many more. Although these legacy data models describe several facets (e.g. device topologies, sensor measurements) of manufacturing systems, they are almost solely used for data exchange without taking advantage of their contained semantics.

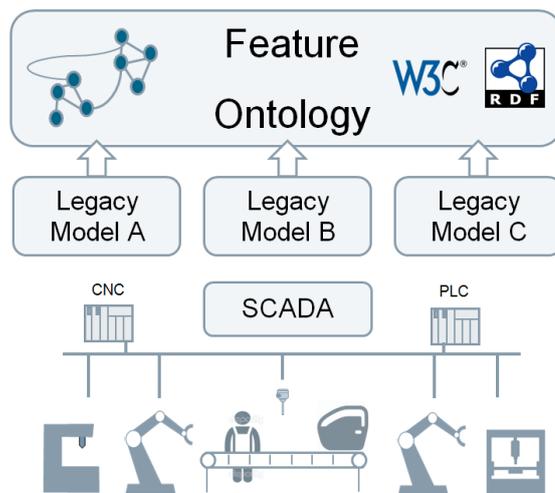


Fig. 5. Feature ontology on top of automation system legacy models

Instead of having another custom information model, our approach makes use of Semantic Web technologies that integrate existing semantics of legacy models into a unified ontology as shown in Figure 5. On top of the automation system and its supervisory control and data acquisition (SCADA), a feature ontology is deployed that represents domain concepts and relations between devices, events and information entities, such as taken sensor measurements. In this context feature means any piece of information that could be used as input to a learning algorithm. From an ontology engineering perspective, this is rather an ad-hoc modeling approach without strong axiomatization.

The result of what we call *Semantic Feature Selection*, i.e. inference about feature dependencies on a semantic level (rather than data level), is represented as RDF graph that contains a task-specific, reduced feature set that is tailored for consumption by machine learning models. These models are then able to perform preprocessing, training, and evaluation on the reduced feature set. One of the goals in development of the feature ontology was to keep the complexity of reasoning as small as possible so that modeling can be done with one of the *OWL 2* profiles that allow scalable reasoning as the number of features in

³ <https://opcfoundation.org/about/opc-technologies/opc-ua/>

the automation system grows. As discussed in section 3.1, the feature ontology references some concepts defined within the SSN ontology. In addition to that, we introduce some further relations concerning the connection between processes, devices and measurements in the manufacturing domain. Most importantly, we allow generic relations `dependsOn` and `independentOf` between features that subsume specific relations in existing engineering models, in case no further information is given.

Table 1. Main relations of the feature ontology

Relation	Description
<code>derivedFrom</code> \sqsubseteq <code>directlyDependsOn</code>	<i>Connects an event or measurement that is derived from an original source, e.g. threshold overshoot events like 'temperature too high'</i>
<code>follows</code> \equiv <code>precedes</code> ⁻¹	<i>Processes or Events that happen in a time-dependent order, e.g. packaging follows the assembly process</i>
<code>partOf</code>	<i>Partonomy describing device topologies, e.g. temperature sensor is part of a motor</i>
<code>directlyDependsOn</code> \sqsubseteq <code>dependsOn</code>	<i>A measurement is directly influencing another without further information, e.g. cycle time directly depends on failure events</i>
<code>physicalRelation</code> \sqsubseteq <code>directlyDependsOn</code>	<i>Measurements are connected by inherent physical laws, e.g. current and voltage</i>
<code>apartFrom</code>	<i>Events that happen at different locations, e.g. two assembly lines operating at separated shop floors</i>
<code>independentOf</code>	<i>A measurement is known to have no (direct) influence on the other, e.g. product ID does not influence conveyor motor temperature</i>
<code>observedBy</code>	<i>Measurement is sampled by a specific sensing device</i>
<code>observedAt</code>	<i>A measurement sampled during a specific process</i>

Table 1 gives an overview of important relations that bear semantics for feature selection purposes. The independence statement is of statistic nature, therefore it is also a symmetric relation based on the fundamental probability theorem $P(X|Y) = P(X) \Leftrightarrow P(Y|X) = P(Y)$.

The $\mathcal{R}\text{Box}$ of the feature ontology further specifies some axioms that propagate dependencies through the feature space, as described in Table 2.

4.2 Feature Selection Procedure

Consider that we are given a learning problem of the form $y = f(x)$, where y is part of the semantic model, such that $\mathcal{O} \models \text{Response}(y)$, then the feature space of x can be reduced by excluding everything that is known to be independent of y . Furthermore, a good choice of features has to include anything that is known

Table 2. $\mathcal{R}\text{Box}$ axioms of the feature ontology

Axiom	Description
$\text{symmetric}(\text{independentOf}),$ $\text{symmetric}(\text{apartFrom})$	<i>Independence and separation of processes are defined to be symmetric</i>
$\text{dependsOn} \cdot \text{observedBy} \cdot \text{observedBy}^{-1} \sqsubseteq$ dependsOn	<i>Dependencies propagate between measurements observed by the same sensing device</i>
$\text{independentOf} \cdot \text{dependsOn} \sqsubseteq$ independentOf	<i>If x is independent of y it is also independent of anything that y depends on</i>
$\text{observedAt} \cdot \text{apartFrom} \cdot \text{observedAt}^{-1} \sqsubseteq$ independentOf	<i>Assert independence of measurements taken at physically separated processes</i>
$\text{transitive}(\text{partOf}), \text{transitive}(\text{follows})$	<i>Process flows and device topologies are transitive by nature</i>

to directly influence the behavior of the response variable. Hence, the $\mathcal{T}\text{Box}$ is accordingly augmented with the following axioms:

$$\begin{aligned} \mathcal{T} = & \\ & \exists \text{independentOf}.\text{Response} \sqsubseteq \text{ExcludedFeature} \\ & \exists \text{directlyDependsOn}^{-1}.\text{Response} \sqsubseteq \text{MandatoryFeature} \\ & \text{ExcludedFeature} \sqcap \text{MandatoryFeature} \sqsubseteq \perp \end{aligned}$$

Overlap in excluded and mandatory features results in an inconsistent feature ontology that is most likely due to a feature modeling mistake, since there should not be independence and direct dependence for two information entities at the same time.

Algorithm 1 summarizes this schema-level feature selection procedure. First, the assertion of y as an individual of the class **Response** must be given. Further classification of individuals is done by a standard *OWL 2* reasoner (e.g. Hermit⁴). In case of an inconsistency, i.e. disjointness of mandatory and excluded features can not be satisfied, the procedure exits. Otherwise, the sets of excluded features and mandatory features are collected, respectively. Finally, the algorithm returns the set of mandatory features and the set of optional features for the learning task. Note that this can be done without looking at the data, but by relying on engineering domain knowledge. After applying *Semantic Feature Selection* learning tasks such as cycle time forecasting can be employed with the reduced set of mandatory and optional features.

The main advantages of our approach in comparison with common feature selection procedures are summarized in Table 3.

⁴ <http://hermit-reasoner.com/>

Algorithm 1. Semantic Feature Selection

```

Input : Learning problem  $y$ , feature ontology  $\mathcal{O}$ , feature space  $\mathcal{F}$ 
Output : Mandatory features  $\mathcal{S}_m$ , optional subset  $\mathcal{S}_o$ 
 $\mathcal{A} \leftarrow \text{Response}(y)$  ▷ Instantiate response variable
 $\mathcal{S}_o \leftarrow \mathcal{F}$ 
 $\mathcal{S}_m \leftarrow \emptyset$ 
if  $\mathcal{O} \models \text{Dis}(\text{ExcludedFeature}, \text{MandatoryFeature}) \sqsubseteq \perp$  then
    return ▷ Unsatisfiable disjointness, inconsistent ontology
end if
for  $x_i \in \{x \mid \mathcal{O} \models \text{ExcludedFeature}(x)\}$  do
     $\mathcal{S}_o \leftarrow \mathcal{S}_o \setminus x_i$ 
end for
for  $x_i \in \{x \mid \mathcal{O} \models \text{MandatoryFeature}(x)\}$  do
     $\mathcal{S}_m \leftarrow x_i$ 
end for
return  $\mathcal{S}_o, \mathcal{S}_m$ 

```

Table 3. Advantages of semantic feature selection

Criterion	Today	Our approach
Complexity	Grows with dimensionality and size of data sets	Grows only with dimensionality (i.e. new feature entities)
Re-usage	Need to be re-executed for every incoming instance	Needs only re-execution if new features are added
Intepretability	Reducing feature spaces often loses intuitive interpretability	Explicitly focuses on facilitated human interpretation

4.3 Graph Kernel Lasso

In some cases, data sets of automation systems are sparse. For example, if we want to forecast cycle times of rarely produced products, there will only be very few instances available for training. For better learning model performance it would be beneficial to further consider the semantics of the feature space during model training. In order to tackle this problem, we present a technique that integrates semantic dependencies into linear model learning and simultaneous feature selection.

In contrast to the standard graph Lasso defined in (3), where a relation between two features is either present or not, i.e. the graph’s adjacency matrix $A_{i,j} \in \{0, 1\}$, we want to use a more enhanced notion of dependencies that also takes semantics into account. In our application, we use RDF-graph kernels to capture similarities between entities in the manufacturing feature ontology. In reference to (3), we therefore define a graph kernel-weighted regularization term that encourages smoothness between similar entities in the RDF graph of the feature ontology.

$$\Omega(\mathbf{w}) = \alpha \|\mathbf{w}\|_1 + (1 - \alpha) \sum_{i,j \in V} \kappa(G_i, G_j) (w_i - w_j)^2 \quad (4)$$

where G_i and G_j are the spanned graphs of entities i, j in the vertex set of the whole RDF graph V and κ is some RDF graph kernel. It is easy to see that the second regularization term can be expressed as weighted Laplacian L_κ .

$$\sum_{i,j \in V} \kappa(G_i, G_j) (w_i - w_j)^2 = \mathbf{w}^T L_\kappa \mathbf{w} \quad (5)$$

with $L_\kappa = \text{diag}(r_1, r_2, \dots, r_p) - \kappa$, and r_i denoting the i th row sum of the kernel matrix κ . We refer to this model as the graph kernel Lasso (*GraKeLasso*). Note that if the kernel matrix is set to the identity matrix, this model is equivalent to the ElasticNet [14].

As mentioned above, the regularization penalty induces a smoothing, or grouping in a sense, that features that are similar with respect to the feature ontology graph have similar parameter values. Intuitively, if two features are closely related, e.g. torque and speed measurements of a conveyor motor, they should have a similar influence (i.e. signal) on the response variable. Additionally, if one feature turns out to be irrelevant, all its closely related features are also very likely to be irrelevant. The graph kernel Lasso model enforces both of these properties.

For our use case application, we can resort to a wide variety of graph kernels, such as the implementations of the *mustard* framework⁵.

A visual representation of the feature ontology graph kernels from our use case data set is given in Figure 6 6a for the full feature space on the left-hand side and 6b for the reduced feature space after semantic feature selection on the right-hand side. Every feature is represented by a segment on a circle the width of the chords connecting two features depicts the strength of similarity between the two. It can be seen that the full feature space contains some very dominant feature similarities, while the reduced feature space exhibits a more uniform distribution.

5 Evaluation

To show the value of semantic guidance in feature selection and the custom Lasso model, we evaluated the performance of forecasting cycle times, as sketched in the manufacturing use case scenario. The regression models are trained on different data sets generated by a discrete-event simulation model that conforms to the manufacturing process in section 2.

We compare five different regression models for the cycle time estimation task: Lasso, ElasticNet, Graph Lasso, *GraKeLasso*, and OLS. For *GraKeLasso*, we used a subtree-based variant of the Weisfeiler-Lehman graph kernel, whereas Graph Lasso incorporates only information about feature individuals connected via `dependsOn`, i.e. a simple dependency graph.

⁵ <https://github.com/Data2Semantics/mustard>

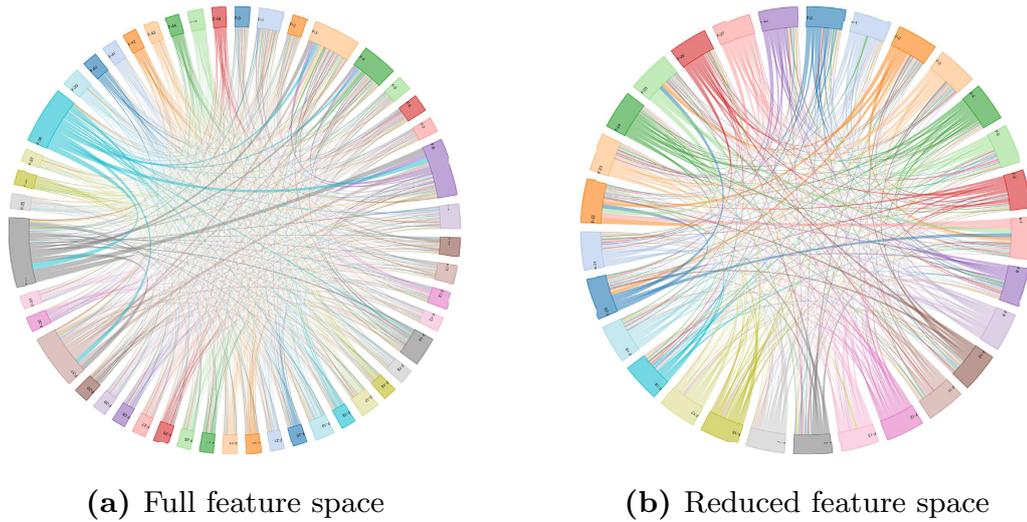


Fig. 6. Visualization of pairwise graph kernel weights between features. (a) Full feature space, (b) Reduced feature space after semantic feature selection

Set Up. Our simulation set up comprises of a source that generates two different product types at a specified time interval, each of which has a different distribution for weight and size. These products are sent to two separated assembly lines, where first a loading station measures product qualities using a balance and a barcode scanner. For the conveyor we monitor its electric motor (power, torque, speed, temperature) and some induced failure events. The simulated quality control again measures sizes of the products. Finally, the packaging station samples the cycle times we want to forecast. Each station further observes its current workload and operating timestamp (seen as soft sensors).

Overall, the final feature ontology consists of 6 processes (one additional for the separated assembly), 18 sensing devices that sample 47 different measurements, i.e. feature individuals. In addition to that there are 13 concrete instantiations of relations between two features.

Table 4. Original and reduced variants of product cycle time data set

Data Set	n	p	Reduction	OLS CV RMSE
Cycle time full	2000	47	-	$\approx 1.36 \times 10^{11}$
Cycle time semantic reduced	2000	29	38.3 %	0.08
Cycle time p-value reduced	2000	18	61.7 %	0.06
Cycle time sparse	40	47	-	9.49

Data Sets & Results. Starting from the original cycle time data set, we obtain three additional variants for evaluation purposes. Table 4 depicts their individual

Table 5. Embedded model performances on 10-fold cross validation

Data Set	Model	Reduction	CV RMSE
Cycle time full	Lasso	47.4 %	0.42
	ElasticNet	34.4 %	0.57
	Graph Lasso	4.5 %	0.32
	<i>GraKeLasso</i>	4.9 %	0.32
Cycle time sparse	Lasso	51.3 %	0.48
	ElasticNet	8.7 %	0.46
	Graph Lasso	8.9 %	0.54
	<i>GraKeLasso</i>	6.8 %	0.43

characteristics. The full data set consists of 47 dimensions and 2000 instances, while in the sparse case, the number of instances is kept to 40 so sparseness is preserved. After applying the reasoning procedure presented in Algorithm 1, the number of features p reduces to 29 – approximately 38 % reduction. On the other hand, a common p-value based selection reduces dimensionality to 18 (at 0.05 significance threshold). This means that there are many linear dependencies in the original data set which can be eliminated by pairwise correlation. The semantic approach does not eliminate dependencies by correlation, but excludes features that are inferred to be independent of the response variable by means of the feature ontology. For each of the data sets an OLS model is trained and evaluated with respect to the coefficient of variation of the root-mean-square error (CV RMSE) in cycle time seconds. It can be seen that best performance is given for the p-value reduced data set, however, the semantic reduced data set shows competitive results.

The embedded feature selection models are evaluated in a similar setting. Model performances shown in Table 5 correspond to the overall best value determined by a grid search over λ , whereas $\alpha \in]0, 1]$ was set to the best of an inner cross validation, respectively. Final performance results are again averaged over 10-fold cross validation. The reduction column also reports on each of the embedded model’s average feature selection capabilities, i.e. number of zero valued coefficients.

Discussion. Overall, our results indicate two main insights. First, compared to p-value based selection, which does a better job at reducing dimensionality, semantic feature selection shows competitive performance in a sense that it keeps the needed features in its original form without any data-intensive computations. Interestingly, after upfront feature selection the ordinary least squares model outperforms all the other approaches for this setting and yields the overall lowest error. Second, our *GraKeLasso* shows best performance on the full and the sparse data set, because it can take similarities of the whole feature space into account. In summary, it can be seen that both of our approaches effectively decrease prediction errors and show competitive or even superior performance compared to conventional techniques. Due to this limited simulation scenario, we could not

show that a combination of both approaches is beneficial. Further evaluations on large-scale systems, when upfront feature selection alone does not suffice, are necessary to investigate this.

6 Related Work

Due to the plethora of research concerned with feature selection, we will only present related works that are closely connected to the one in this paper. For a general overview of the field, we refer to the survey paper [4].

Coming from a semantic perspective on feature selection, the technique introduced by [6] describes a fuzzy approach to capture implicit semantics of data sets in order to reduce their dimensionality. They apply fuzzyfication on the degree of which features are dependent based on their co-occurring consistency with the decision variable. However, this technique does not rely on any explicit semantics defined in the data model and also needs preferably access to the full data set. In the field of biomedical machine learning applications, considering domain knowledge in feature selection has been studied and shown that feature spaces for association rules can be greatly reduced when medical domain knowledge about concepts is introduced, see [1]. The authors introduce dependencies between medical concepts, such as diseases and treatments, in order to learn more compact association rules. Another kind of related work that has been studied with increasing interest is the family of Lasso regularization models. Algorithms like the GOSCAR have been applied to consider dependency knowledge between genes in DNA sequences as penalty for group regularization in graph Lasso models. These models have shown to increase classification accuracy in several studies, e.g. [11]. Similarly, for image classification semantic dependencies between labeled images have been integrated into a Lasso approach [2].

In summary, including semantics into feature selection has been the concern of very few research studies outside of text and image processing domains, where semantics are mostly given through natural language. However, there are certain knowledge-based approaches that argue for the dynamic adaption of features to account for changes in the data generation process [9].

7 Conclusion and Future Work

In this work, we introduced the application of semantic feature selection for machine learning models in modern industrial automation systems. Only few works have been concerned with the usage of explicit semantics, in order to facilitate feature selection, which still remains one of the main issues. Especially in the manufacturing domain, there are many known dependencies between measurements that are cut out to guide this process. In this paper, we show how a small amount of semantic relations can be used to significantly reduce the size of feature spaces for exemplary learning problems in manufacturing systems and still yield good performance. Furthermore, we presented an embedded feature

selection for linear models that captures feature similarities within RDF graphs and outperforms conventional approaches when applied to sparse data sets.

In future work, we plan to implement the developed approach in a real-life automation system. A particular promising direction seems to be the conjunction of this approach within OBDA systems, where ontologies are used to retrieve instance data. By deploying machine learning on top of OBDA, a coupling of our approach and ontology-based queries could reveal some synergy effects.

References

1. Blake, C., Pratt, W.: Better rules, fewer features: a semantic approach to selecting features from text. In: Proc. of IEEE Int. Conf. on Data Mining, pp. 1–8 (2001)
2. Chen, X., Yuan, X., Yan, S., Tang, J., Rui, Y., Chua, T.S.: Towards multi-semantic image annotation with graph regularized exclusive group lasso. In: Proc. of 19th ACM Int. Conf. on Multimedia - MM 2011, pp. 263–272 (2011)
3. de Vries, G.K.D.: A fast approximation of the Weisfeiler-Lehman graph kernel for RDF data. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) ECML PKDD 2013, Part I. LNCS, vol. 8188, pp. 606–621. Springer, Heidelberg (2013)
4. Guyon, I.: An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res. (JMLR)* **3**, 1157–1182 (2003)
5. Jenatton, R., Audibert, J.Y., Bach, F.: Structured Variable Selection with Sparsity-Inducing Norms. *J. Mach. Learn. Res. (JMLR)* **12**, 2777–2824 (2011)
6. Jensen, R., Shen, Q.: Semantics-preserving dimensionality reduction: Rough and fuzzy-rough-based approaches. *IEEE Transactions on Knowledge and Data Engineering* **16**(12), 1457–1471 (2004)
7. Jirkovsky, V., Obitko, M., Novak, P., Kadera, P.: Big data analysis for sensor time-series in automation. In: Proc. of Emerging Technology and Factory Automation (ETFA), pp. 1–8 (2014)
8. Lösch, U., Bloehdorn, S., Rettinger, A.: Graph kernels for RDF data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 134–148. Springer, Heidelberg (2012)
9. Ringsquandl, M., Lamparter, S., Lepratti, R.: Context-aware analytics in MOM applications. In: Workshop Notes of the 6th International Workshop on Acquisition, Representation and Reasoning about Context with Logic (2014)
10. Rodríguez-Muro, M., Kontchakov, R., Zakharyashev, M.: Ontology-based data access: on top of databases. In: Proc. of the 12th Int. Sem. Web Conf. (2013)
11. Yang, S., Yuan, L., Lai, Y.c., Shen, X., Wonka, P., Ye, J.: Feature grouping and selection over an undirected graph. In: Proc. of the Int. Conf. on Knowledge Discovery and Data Mining (KDD), pp. 922–930 (2012)
12. Ye, J., Liu, J.: Sparse Methods for Biomedical Data. *SIGKDD explorations* **14**(1), 4–15 (2012)
13. Yu, L., Liu, H.: Feature selection for high-dimensional data: a fast correlation-based filter solution. In: Proc. of the 20th Int. Conf. on Mach. Learn., pp. 1–8 (2003)
14. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society* **67**, 301–320 (2005)

Knowledge Graph Constraints for Multi-label Graph Classification

Martin Ringsquandl*, Steffen Lamparter[†], Ingo Thon[†], Raffaello Lepratti[‡] and Peer Kröger*

*Ludwig-Maximilians Universität
Munich, Germany,

martin.ringsquandl@gmail.com, kroegerp@dbs.ifi.lmu.de

[†]Siemens AG, Corporate Technology
Munich, Germany

steffen.lamparter@siemens.com, ingo.thon@siemens.com

[‡]Siemens s.p.A, Digital Factory
Genoa, Italy

raffaello.lepratti@siemens.com

Abstract—Graph classification methods have gained increasing attention in different domains, such as classifying functions of molecules or detection of bugs in software programs. Similarly, predicting events in manufacturing operations data can be compactly modeled as graph classification problem. Feature representations of graphs are usually found by mining discriminative sub-graph patterns that are non-uniformly distributed across class labels. However, as these feature selection approaches are computationally expensive for multiple labels, prior knowledge about label correlations should be exploited as much as possible.

In this work, we introduce a new approach for mining discriminative sub-graph patterns with constraints that are extracted from links between labels in knowledge graphs which indicate label correlations. The incorporation of these constraints allows to prune the search space and ensures extraction of consistent patterns. Therefore, constraint checking remains efficient and more robust classification results can be obtained. We evaluate our approach on both, one public and one custom simulated data set. Evaluation confirms that incorporation of constraints still results in efficient pattern mining and can increase performance of state-of-the-art approaches.

I. INTRODUCTION

Real-world systems often naturally produce data with relational dependency structures, for example, atomic bonds of molecules, words in text documents, users and posts in social networks. These structures are commonly represented as graphs to encode corresponding logical dependencies between entities. The value of relations can be exploited for classification problems with graph-based approaches which have been shown to outperform traditional representations for tasks such as text categorization [1].

From a data mining perspective such graphs are treated as instances, i.e. *operational* data that is continuously generated. On the other hand, with the increasing availability of linked information contained in open knowledge graphs, *static* graph data is becoming an interesting objective to take into account for data mining and machine learning tasks [2]. For example, a movie classification task might benefit from links contained in the open Linked Movie Data Base¹ that stores facts about

movies, actors, locations connected via different semantic relations.

By combining these operational and static graph data, it is desired to use the static *logical links* in-between instances and labels in order to increase classification performance of data mining methods carried out on the operational graph data. Sometimes knowledge graphs are also referred to as heterogeneous information networks in the data mining literature, since they contain information about entities and relation of different logical types [3].

In case of multi-label classification, it is crucial to incorporate known correlations among labels, because of the high number of possible label combinations (exponential in size of number of labels) [4]. A direct consequence of this is also that some labels might only have a small number of positive examples in the dataset. In order to provide accurate multi-label classification models, numerous approaches have been introduced that try to model dependencies between labels by either considering statistics or by mere assumption. Mining graphs with respect to statistical correlation is computationally expensive and might also be incomplete, since some combinations are not observed in the training data. However, already the mere logical links between labels and instances can be used to estimate label correlations. This has been studied in the notion of collective classification [5], where additional features for drug disease classification are generated based on links in open knowledge graphs.

In this paper, we introduce a new approach for incorporating domain constraints based on knowledge graph links into existing sub-graph mining algorithms with the aim to increase classification performance. We motivate the approach by means of a manufacturing operations use case in Section II. After giving an overview of related work in this area in Section III, the graph classification problem is defined in Section IV. The contributions of this paper are presented in Section V and consequently evaluated in Section VI.

¹<http://www.linkedmdb.org/>

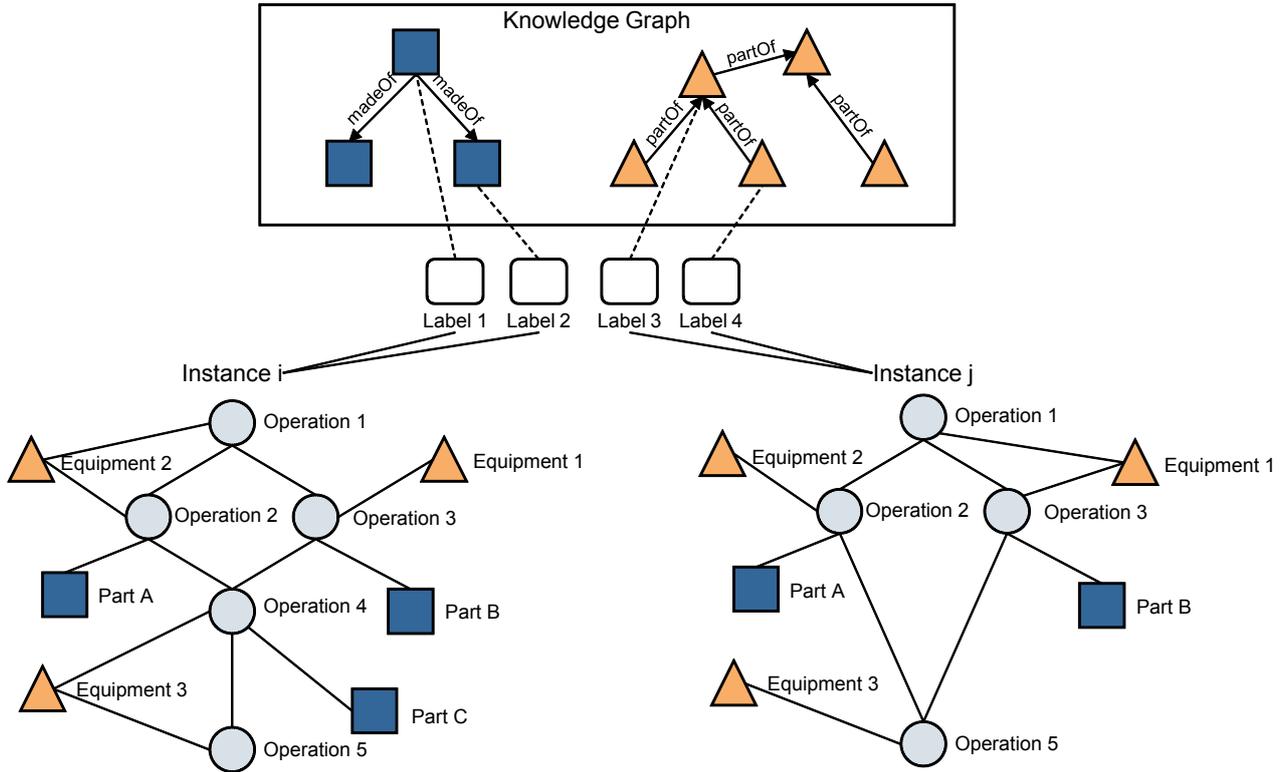


Fig. 1. Graph instances of manufacturing operations where labels are associated to knowledge graph entities

II. MOTIVATION: MANUFACTURING OPERATIONS

One application domain of multi-label graph classification is the area of manufacturing operations, where production equipment, product parts and operations can naturally be represented as graphs. Non-conformance events often share common root-causes in the configuration of operations, such as incompatible parts and equipments. Due to the engineering design of products and plants, there is an implicit correlation between non-conformances, e.g. failure propagation between connected parts in consecutive assembly processes.

These logical dependencies can be modeled as knowledge graphs consisting of hierarchical structures, such as part-of relations between equipment, products and processes.

Example: Consider the classification of quality issues in a manufacturing dataset, where the instances are represented as graphs of operation, equipment and product part entities, as shown in Figure 1. The manufacturing knowledge graph contains facts about product parts, e.g.:

$\langle \text{PartA}, \text{madeOf}, \text{PartB} \rangle$

This so-called triple notation says that entity PartA is linked to entity PartB via the named relation madeOf. Each quality issue (label) can be associated with an entity in the knowledge graph. Therefore, paths through the knowledge graph give

hints about correlations between labels. We want to incorporate this knowledge in form of Must-Link and Cannot-Link constraints for mining discriminative sub-graph patterns.

III. RELATED WORK

Early approaches from the inductive logic programming community were targeted at the direction of mining discriminative patterns with graph-theoretic representation [6]. Recently, a constraint-based graph pattern mining approach was considered in [7] with the application of classifying websites focused on bags of graphs with pairwise must- and cannot-link constraints. This work can be seen as a specialization of finding discriminative sub-graph patterns using branch-and-bound in a heuristic sense, which has also been studied in other works [8]. The same ideas can be transferred to the task of multi-label graph classification [9]. The authors suggest to use a form of the Hilbert-Schmidt Independence Criterion between the extracted sub-graph patterns and the respective labels. Since considering all correlations between each combination of features and labels is computationally expensive, other approaches have argued that links from heterogeneous information networks, i.e. semantic knowledge graphs, can be exploited in a collective classification sense [3], [5]. Here, the approach is to extract paths from these knowledge graphs and establish collective links between instances based on shared paths. Therefore, the extracted paths can be interpreted

as a form of soft linkage constraints derived from domain knowledge. Apart from that, there has been little work focusing on hard constraints specifically for graph classification with exceptions of the *gPrune* and *LEAP* frameworks, which define constraints for patterns on graph properties such as minimum density [10] or structural similarity [11]. However, as these constraints only evaluate structural properties of graphs, they are not suitable for representing more formal domain knowledge, i.e. constraints for specific relations between certain types of nodes.

Overall, the motivation of the work presented in this paper is similar to semantic feature selection [12], where dependencies between entities in a knowledge graph are used to guide feature selection process in technical systems.

IV. GRAPH CLASSIFICATION

The multi-label graph classification problem can be formalized by inducing a mapping $f(x) : \mathcal{X} \rightarrow \mathcal{Y}$, from a given training set $D = \{\langle x_i, \vec{y}_i \rangle\}$, where each $x_i \in \mathcal{X}$ is a graph and \vec{y}_i is an indicator vector of class labels from \mathcal{Y} .

Standard classification algorithms can only be used for graph classification if the graph instances are transformed into a numeric vector space. One popular choice is the transformation of every graph instance into an indicator vector of sub-graph patterns.

Definition 1 (Vector Representation): Given a set of graph patterns $\{g_1, g_2, \dots, g_m\}$, a graph instance x_i has vector representation $\vec{x}_i = [x_i^{g_1}, x_i^{g_2}, \dots, x_i^{g_m}]$, where every $x_i^{g_j} = 1$, if the graph $x_i \subseteq g_j$, otherwise $x_i^{g_j} = 0$

As a consequence any classification model that accepts numeric vectors as inputs $f(\vec{x})$ can be trained on the transformed graph data.

For the remainder of this paper, we focus on graph classification approaches that use this vector representation and therefore rely on the extraction of sub-graph patterns as features. Other graph classification methods, for example, graph kernels or latent space embeddings are not considered.

A. Pattern Mining for Classification

In order to find the best (optimal) set of sub-graph patterns that can be used as features, an exhaustive search through all graph instances in D would need to evaluate an objective function on every possible sub-graph. Since sub-graph matching itself is an NP-complete problem, it is commonly desirable to reduce the search space as much as possible by pruning paths that are known to be non-optimal. Classic pattern enumeration algorithms like *gSpan* [13] make use of the minimum support threshold of a pattern. This is not sufficient if the objective function of a pattern goes beyond simple frequency in the dataset.

In case of classification, the optimal patterns need to have different characteristics, i.e. they should distinguish positive from negative examples in the dataset. These patterns are

```

1: Input:  $D, minsup, \mathcal{ML}, \mathcal{CL}$ 
2: Output:  $\mathbf{g}$ 
3:  $S = \{1\text{-edge graph}\}$ 
4:  $\mathbf{g} = \emptyset$ 
5: while  $S \neq \emptyset$  do
6:   for  $g \in S$  do
7:      $S = S \setminus \{g\}$ 
8:     if  $support(g) < minsup$  then
9:       continue
10:    end if
11:    if  $bound(g) > threshold$  then
12:       $\mathbf{g} = \mathbf{g} \cup g$ 
13:    else
14:      continue
15:    end if
16:     $S = S \cup branch(g)$ 
17:  end for
18: end while
19: return  $\mathbf{g}$ 

```

Fig. 2. Basic Branch-and-Bound Search Framework

sometimes called *discriminative* patterns. Several objective functions have been proposed for finding discriminative graph patterns, which most effectively can be used in a branch-and-bound search.

B. Branch-and-Bound Search

In order to perform branch-and-bound during graph pattern enumeration, one has to consider an upper-bound as termination criterion. Let g be a pattern on the current search path and g' some possible extension to the current path, i.e. $g \subseteq g'$. An upper-bound $\hat{q}(g)$ of the objective function q is defined as follows:

$$\hat{q}(g) = \max_{g \subseteq g'} q(g') \quad (1)$$

That is if this criterion is met in the current search path, the whole branch can be discarded and does not have to be further explored. In the basic branch-and-bound framework, shown in Figure 2. First, the algorithm loops through all 1-edge graph patterns, then checks for minimum support. If this condition is not met, the current branch is closed. The same holds if the objective function criterion $bound(g)$ is not fulfilled. In case it is, the set of patterns \mathbf{g} is augmented with the current pattern g . Finally, the current pattern is extended (branched) with a new edge and added to the search space S . For the bounding criterion, there have been several proposals with the aim to give a high value to patterns that occur frequently in one class and infrequently in the other [8], [7]. Also for multiple labels [9]. However, these objective functions do not strictly follow anti-monotonicity, but are rather based on approximations or greedy algorithms.

C. Shortcomings of Discriminative Scores

For discriminative scores, it has been argued that especially for datasets with imbalanced classes they are sensitive to

outliers in the underrepresented class, since patterns occurring only in a small number of positive instances rarely occur in negative ones [14]. This is not however what is generally desired as output. Interesting patterns should cover the majority of positive cases and only a small fraction of negative ones (the dominant class). For example, in manufacturing data, there are often common root-causes to several problems and therefore we are more interested in finding generalizing patterns throughout all positive examples (failure cases) and not overly sensitive to individual symptoms, i.e. outlier structures.

In order to overcome these issues, domain knowledge can provide valuable hints about the nature of valid sub-graph patterns. This motivates the introduction of hard constraints that are derived from domain knowledge into pattern search.

V. GRAPH-PATTERN CONSTRAINTS

The incorporation of hard constraints during pattern search means that patterns that violate one of the constraints should not represent a valid sub-graph feature and therefore be discarded.

In this section we define – relating to terms in clustering literature – *Must-Link* and *Cannot-Link* constraints on pairs of graph instances in the following way:

Definition 2 (Must-Link Constraint): A Must-Link constraint of the form $\mathcal{ML}(x_i, x_j)$ restricts sub-structure patterns g such that $g \subseteq x_i \wedge g \subseteq x_j$ must be satisfied, denoted as $g \models \mathcal{ML}(x_i, x_j)$.

Let $\phi(g) = |\{x_i | g \subseteq x_i\}|$ denote the number of graph instances that contain pattern g . It can be seen that Must-Link Constraints are monotonic for pattern enumeration, since for each $g \subset g'$, if g is not contained in either x_i or x_j , i.e. violates the constraint, then g' cannot be contained. This directly follows from $\phi(g) \geq \phi(g')$.

Definition 3 (Cannot-Link Constraint): A Cannot-Link constraint of the form $\mathcal{CL}(x_i, x_j)$ specifies that each pattern g must satisfy $\neg(g \subseteq x_i \wedge g \subseteq x_j)$, denoted as $g \models \mathcal{CL}(x_i, x_j)$.

Cannot-Link Constraints are not monotonic, since a candidate pattern g that is contained in both x_i and x_j , can have a possible extension g' that is no longer contained in both graph instances. Therefore, pattern enumeration must continue to search for extensions that satisfy all Cannot-Link Constraints.

The augmented version of the branch-and-bound pattern search algorithm is specified in Figure 3. As basis for our pattern search, we used the well-known *gSpan* algorithm, which was developed to discover frequent subgraphs above a minimum frequency (support) threshold. We extended *gSpan* with state-of-the-art branch-and-bound scoring approaches and a component for handling Must-Link and Cannot-Link constraints search strategy.

The handling of constraints works as follows: As soon as a pattern g violates a Must-Link constraint ($g \not\models \theta_{ML}$),

```

1: Input:  $D, minsup, \mathcal{ML}, \mathcal{CL}$ 
2: Output:  $g$ 
3:  $S = \{1\text{-edge graph}\}$ 
4:  $g = \emptyset$ 
5: while  $S \neq \emptyset$  do
6:   for  $g \in S$  do
7:      $S = S \setminus \{g\}$ 
8:     for  $\theta_{ml} \in \mathcal{ML}$  do
9:       if  $g \not\models \theta_{ml}$  then
10:        continue
11:      end if
12:    end for
13:     $add = True$ 
14:    for  $\theta_{cl} \in \mathcal{CL}$  do
15:      if  $g \not\models \theta_{cl}$  then
16:         $add = False$ 
17:      end if
18:    end for
19:    if  $bound(g) > threshold \wedge add$  then
20:       $g = g \cup g$ 
21:    else
22:      continue
23:    end if
24:     $S = S \cup branch(g)$ 
25:  end for
26: end while
27: return  $g$ 

```

Fig. 3. Constraint-augmented Branch-and-Bound Search Framework

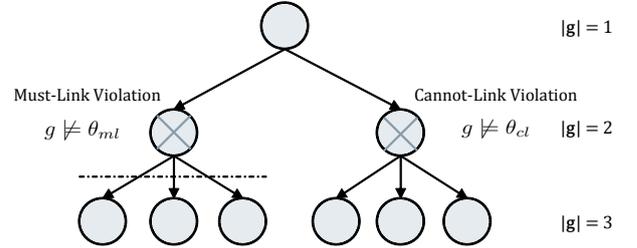


Fig. 4. Constrained Search Strategy

the current path is pruned from the search space, *continue* statement in line 10. Violations of Cannot-Link constraints on the other hand only disallow the current pattern to be added to the set of sub-structure features. In the pseudo-code in Figure 3, the flag *add* controls this behavior in line 19.

This constrained search strategy is exemplified in Figure 4. On level two where the current pattern size is $|g| = 2$, the violation of Must-Link constraint discards the current pattern and prunes every path under the current node, while a Cannot-Link constraint violation on the right node only discards the current node, but continues the search on its children.

A. Extraction of Constraints from Knowledge Graphs

Recently, Kong et al. introduced the concept of meta-paths in knowledge graphs for collective feature extraction [5]. The

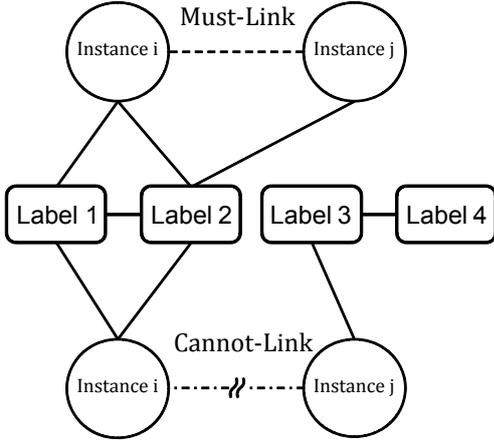


Fig. 5. Cannot-Link and Must-Link constraints based on label dependencies

idea is that instances and their respective labels are connected to entities in a knowledge graph. Then connections (paths) between instances are used to augment the feature space. In this work, instead of manipulating the feature space, we directly extract constraints from paths in knowledge graphs.

Definition 4 (Knowledge Graph): A knowledge graph is represented as 4-tuple $KG = (V, E, \mathcal{L}, l)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, \mathcal{L} is a set of labels, and $l : V \cup E \rightarrow \mathcal{L}$ is a mapping that assigns labels to vertices and edges.

Let v_i denote the knowledge graph entity associated to the i -th label. There is a dependency between label entity v_i and v_j if they share a direct link $e_l(v_i, v_j)$ or they have a common ancestor v_a , i.e. there is a link $e_l(v_i, v_a)$ and $e_l(v_j, v_a)$.

A Must-Link constraint between two instances $\mathcal{ML}(x_i, x_j)$ is specified if all associated labels of both instances are connected via dependencies. A Cannot-Link constraint between two instances $\mathcal{CL}(x_i, x_j)$ is specified if their labels do not share any dependency.

Example: Given the hierarchy induced by knowledge graph edges $\langle \text{Equipment2}, \text{partOf}, \text{Equipment1} \rangle$ and $\langle \text{Equipment3}, \text{partOf}, \text{Equipment1} \rangle$. Must-Link constraints will be extracted for all instances which have a label set that is associated to any subset of $\{\text{Equipment1}, \text{Equipment2}, \text{Equipment3}\}$, since it is assumed that these labels have a high correlation.

A graphical description of how these constraints can look like is depicted in Figure 5, where there is a dependency for $(\text{Label 1}, \text{Label 2})$, and $(\text{Label 3}, \text{Label 4})$, respectively.

VI. EXPERIMENTS

In order to demonstrate applicability and effectiveness of our approach, this section presents performance results for

TABLE I
CHARACTERISTICS OF GRAPH DATA SETS

Dataset	$ D $	$avg Nodes $	$avg Edges $	$ \mathcal{Y} $
Movie Summaries	500	32.3	43.3	10
Manufacturing	500	13.2	29.9	6

graph classification experiments with knowledge graph constraints in two different settings.

A. Movie Summaries

The first setting is based on the public CMU Movie Summary Corpus², which contains description (short text) and meta-data of movies. In this case, the classification task is to predict the genres of each movie based on the short description. We extract a keyword graph for each instance where two words are linked if they occur within a certain window. As knowledge graph, we used an excerpt of Wikidata³ that is concerned with facts about movie genre hierarchies, specifying for example that *Science fiction* is a sub-genre of *Fantasy*.

B. Manufacturing Operations

The second setting is concerned with a dataset from a simulated manufacturing system. A graph representation of the execution records is built consisting of interactions between equipment, parts and operations. The classification task to predict multiple non-conformance events, such as quality failures, for each execution. The knowledge graph in this case contains an equipment and a product part hierarchy conforming to plant layout and material specifications.

C. Evaluation

Both settings are used for evaluation of the presented approach via a structured comparison. For each dataset, classification is done by two *One-Vs-Rest* classification algorithms: linear Support-Vector-Machine (SVM) and Naive Bayes (NB). Both graph datasets were prepared to consist of 500 instances. Other characteristics, specifically average number of nodes (edges) and number of labels of both datasets are shown in Table I.

D. Feature Selection Comparison

As a baseline for graph feature selection we used a frequency-based *top-k* scoring as objective function. For state-of-the-art discriminative feature selection, the *greedy* search from [8] and the *multi-graph feature based learning (gMGFL)* algorithm with simple soft constraints on graphs with different labels [7] were considered.

The Cannot-Link and Must-Link constraints of this work can naturally be added to each of these algorithms. We denote the augmented feature selection approaches as *top-k+cons*, *greedy+cons*, and *gMGFL+cons*, respectively. Classification

²<http://www.cs.cmu.edu/~ark/personas/>

³<https://www.wikidata.org>

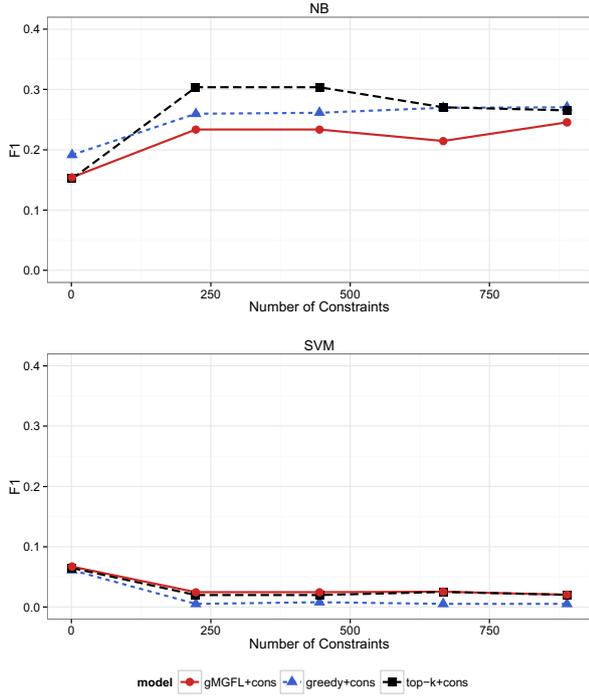


Fig. 6. Movies Dataset – Classification Results with different Feature Selection Models

performance is measured via a weighted F1 score that takes class label imbalance into account and averaged over a 5-fold cross-validation.

The graph in Figure 6 shows F1-scores for each feature selection method against the number of pairwise Must-Link and Cannot-Link constraints between instances for the movie summaries dataset. It can be seen that incorporation of constraints does improve classification performance for the Naive Bayes classifier, which also has much higher F1-scores compared to the SVM. In case of the SVM, the constraints are actually decreasing performance. We assume that the reason for the performance gain of Naive Bayes is due to increasing independence among features enforced by the constraints. The linear SVM does not benefit from this, but has simply less features for learning of its linear decision boundary.

From evaluation of the manufacturing dataset, shown in Figure 7, it can be seen that both Naive Bayes and the SVM benefit from the knowledge graph constraints. However, for the *gMGFL* approach, there is a drop in performance for some constraints, while the other approaches are not affected. As for the movies dataset, the biggest increase is observed in terms of the *top-k* approach. This is due to the relative high amount of frequent, but non-discriminative, sub-graph patterns occurring throughout all class labels.

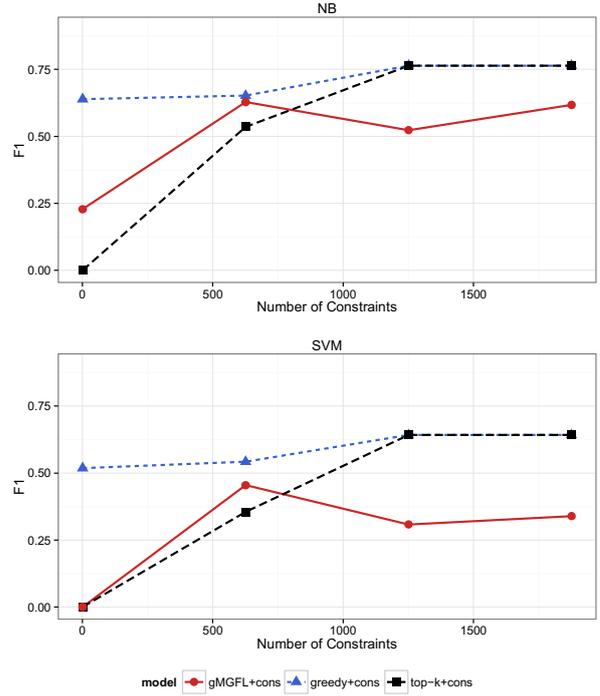


Fig. 7. Manufacturing Dataset – Classification Results with different Feature Selection Models

E. Runtime Comparison

In addition to classification performance, we also investigated the computational efficiency of constraint incorporation. By measuring runtime of the constraint-guided search strategies, the question is how expensive the constraint checking is compared to the reduction of search space via constraint-based path pruning.

For the movie summaries experiments, runtime results are shown in Figure 8. As the number of constraints grows, there seems to be no significant increase in runtime. In this setting, the extra work of constraint checking is compensated by the additional pruning capabilities of the constrained search.

Results obtained for the manufacturing dataset give a different indication, as shown in Figure 9. Due to the higher complexity of this dataset, i.e. stronger connectedness of graph instances, constraint checking becomes more expensive as the number of constraints grows. However, again there is a settling point where pruning weighs off these additional efforts.

VII. CONCLUSION

In this paper, we introduced a new approach to incorporate Must-Link and Cannot-Link constraints that are extracted from knowledge graphs, in order to guide the search for

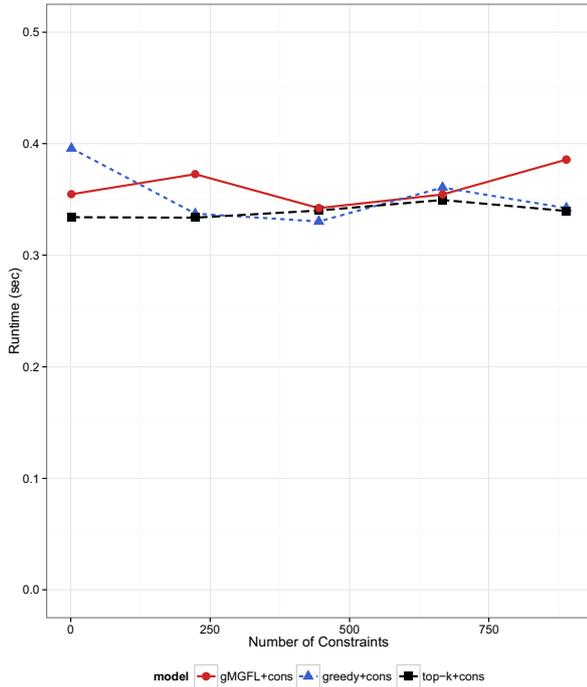


Fig. 8. Movie Dataset – Runtime Results with increasing number of constraints

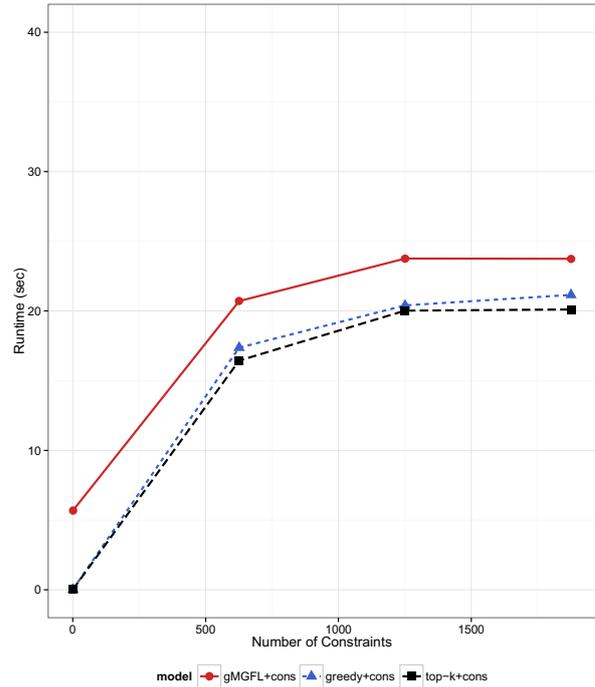


Fig. 9. Manufacturing Dataset – Runtime Results with increasing number of constraints

discriminative sub-graph features in multi-label classification. The constraints can readily be added to existing feature selection algorithms. We demonstrate the applicability in two experimental settings and show that classification performance can be increased by the extracted constraints. In terms of scalability, we see potential for more efficient constraint handling, especially for large graph instances.

Future work could be to analyze further possibilities of how links between instances can be used to derive constraints, instead of using hierarchical label dependencies only. We also plan to investigate learning of constraints from more complex paths through knowledge graphs.

REFERENCES

- [1] R. Angelova and G. Weikum, “Graph-based text classification: learn from your neighbors,” *SIGIR '06 Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 485–492, 2006.
- [2] Y. Sun, C. C. Aggarwal, and J. Han, “Relation strength-aware clustering of heterogeneous information networks with incomplete attributes,” *Proceedings of the VLDB Endowment*, vol. 5, pp. 394–405, 2012.
- [3] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild, “Meta path-based collective classification in heterogeneous information networks,” *Proceedings of the 21st ACM international conference on Information and knowledge management*, pp. 1567–1571, 2012.
- [4] J. T. Kwok, “Multilabel Classification with Label Correlations and Missing Labels,” *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 1680–1686, 2014.
- [5] X. Kong, B. Cao, and P. S. Yu, “Multi-Label Classification by Mining Label and Instance Correlations from Heterogeneous Information Networks Categories and Subject Descriptors,” *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 614–622, 2013.
- [6] L. B. Holder and D. J. Cook, “Graph-based relational learning: current and future directions,” *ACM SIGKDD Explorations Newsletter*, vol. 5, pp. 90–93, 2003.
- [7] J. Wu, S. Member, X. Zhu, and S. Member, “Bag Constrained Structure Pattern Mining for Multi-Graph Classification,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 10, pp. 2382–2396, 2014.
- [8] M. Thoma, H. Cheng, A. Gretton, J. Han, H.-P. Kriegel, A. Smola, L. Song, P. S. Yu, X. Yan, and K. Borgwardt, “Near-optimal supervised feature selection among frequent subgraphs,” *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 1076–1087, 2009.
- [9] X. Kong and P. S. Yu, “GMLC: A multi-label feature selection framework for graph classification,” *Knowledge and Information Systems*, vol. 31, pp. 281–305, 2012.
- [10] F. Zhu, J. Han, and P. S. Yu, “gPrune : A Constraint Pushing Framework for Graph Pattern Mining,” *Advances in Knowledge Discovery and Data Mining*, pp. 388–400, 2007.
- [11] X. Yan, H. Cheng, J. Han, and P. S. Yu, “Mining significant graph patterns by leap search,” *Proceedings of the 2008 ACM SIGMOD Int. conf. on Management of Data*, pp. 433–444, 2008.
- [12] M. Ringsquandl, S. Lamparter, S.-P. Brandt, and R. Lepratti, “Semantic-guided Feature Selection for Industrial Automation Systems,” in *Proc. of the 14th International Semantic Web Conference*. Springer, 2015.
- [13] X. Yan and J. Han, “gSpan: Graph-Based Substructure Pattern Mining,” *Proc. of the 2002 IEEE Int. Conf. on Data Mining (ICDM'02)*, vol. 1, no. d, pp. 721–724, 2002.
- [14] S. Pan and X. Zhu, “Graph classification with imbalanced class distributions and noise,” *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1586–1592, 2013.

Knowledge Fusion of Manufacturing Operations Data Using Representation Learning

Martin Ringsquandl¹(✉), Steffen Lamparter², Raffaello Lepratti³,
and Peer Kröger¹

¹ Ludwig-Maximilians Universität, Munich, Germany
martin.ringsquandl@gmail.com

² Siemens Ag, Corporate Technology, Munich, Germany

³ Digital Factory, Siemens Industry Software S.r.l., Genoa, Italy

Abstract. Due to increasingly required flexibility in manufacturing systems, adaptation of monitoring and control to changing context such as reconfiguration of devices becomes more important. Referring to the usage of structured information on the Web, digital twin models of manufacturing data can be seen as knowledge graphs that constantly need to be aligned with the physical environment. With a growing number of smart devices participating in production processes, handling these alignments manually is no longer feasible. Yet, the growing availability of data coming from operations (e.g. process events) and contextual sources (e.g. equipment configurations) enables machine learning to synchronize data models with physical reality. Common knowledge graph learning approaches, however, are not designed to deal with both, static and time-dependent data.

In order to overcome this, we introduce a representation learning model that shows promising results for the synchronization of semantics from existing manufacturing knowledge graphs and operational data.

Keywords: Representation learning · Digital twin · Knowledge fusion

1 Introduction

The ubiquitous availability of data empowers manufacturing companies to embrace advanced data analytic technologies that allow to monitor, predict, and optimize manufacturing operations. Still, ensuring semantic interoperability within hardware-software integrated cyber-physical systems (CPS) and management applications requires extensive manual data modeling effort, thus introducing and maintaining these technologies is challenging for manufacturers [7]. For example, today, deploying a new device for machine condition monitoring at a shop floor means manual effort to model this device and all of its signals throughout several software applications (e.g. SCADA, MES). Otherwise, physical reality

is not correctly reflected in existing models and there is no semantic interoperability between applications.

Recently, descriptive data models have been revitalized as part of a digital representation of physical systems, the so-called *digital twin*, which allows systems to discover, inherit, evaluate and share information across different subsystems [2]. From a data modeling perspective, structured information of digital twins can be represented as knowledge graph (KG), where relations and entities follow well-defined vocabularies and semantics.

Knowledge graphs are commonly understood as publicly-accessible Linked Data resources – prominent examples are Wikidata¹ and WordNet². Similarly, Manufacturing Execution Systems (MES) and engineering platforms that are built upon sizable relational databases can be seen as domain-specific knowledge graphs, when lifted to a semantic schema [5]. Such a *manufacturing knowledge graph* should be able to automatically acquire updated information based on different operational data sources (e.g. SCADA, PLCs, etc.), even if these data sources are not aware of their semantics.

Continuing the machine monitoring example: By observing data coming from the newly added device (e.g. events) the KG should automatically recognize the type of device, its location, or its capabilities and therefore allow other applications to adapt to this updated context.

Machine Learning in KGs has emerged recently with the goal to enable automated integration of new facts into KGs without manual modeling efforts [9]. When multiple data sources are used to extract information, the problem further extends to so-called *knowledge fusion* [3]. The same problems apply to models in manufacturing systems that need to be in-sync with physical reality reflected by multiple operational data sources [4]. In this paper, we present an approach to support fusion of information coming from operational data sources with manufacturing KGs by learning latent representations of entities. The goal is to offer automated recommendations on how to integrate unknown entities into the existing structure of the KG and thus keeping the digital twin in-sync without manual modeling effort. Ultimately, this is beneficial to monitoring and management applications that rely on a immediately aligned digital representation of the manufacturing system.

2 Motivation Scenario

In this section we present an example scenario that motivates the application of machine learning (knowledge fusion) to manufacturing KGs in conjunction with operational data sources.

Consider an automated production line at a discrete manufacturing facility, consisting of multiple production units that can be configured to produce several variants of a product. The manufacturing KG (e.g. provided by an MES) of this production line gives information about device topology and processes executed

¹ <http://wikidata.org>.

² <http://wordnet.princeton.edu>.

by each of the production units, whereas a SCADA system observes sequences of events during operation. As shown in Fig. 1, at the bottom, sequences of events are continuously generated and aligned to entities in the manufacturing KG. Entities and their relations are denoted as triples (**head-entity**, **relation**, **tail-entity**), in the middle of the Figure. The schema (classes and relations) of the KG is shown on top of the entities using a simplified class diagram notation. For example, the triple (**Event 1**, **occurs at**, **Conveyor**) in the KG states that entity *Event 1* occurs at entity *Conveyor*. Additionally the conveyor entity is modeled as device that is involved in the board assembly process.

Assuming a new device is deployed to the production line to monitor temperature measurements of the conveyor. As production resumes, events of this new device are continuously observed, but they are lacking semantic alignment to the existing KG. Figure 2 shows a new sequence of events, where unknown entities in the triples are denoted with question marks. Here, the class of the unaligned event *Event 2* and its source (device) are unknown, (**Event 2**, **is-a**, **?**), respectively (**Event 2**, **occurs at**, **?**).

However, the distribution of events in the sequence data should give an indication about which device is most likely to be hold responsible (in this case the conveyor). Since other conveyor events are assumed to co-occur in similar fashion as the new monitoring events, this information can be exploited to re-engineer semantics. Presuming one could obtain a vector representation of all involved entities (events, devices, etc.), it would be possible to calculate a *similarity* between *Event 1* and *Event 2* that would allow to infer that both are related to the conveyor entity in the KG. The representation learning approach in the following is motivated by learning latent entity embeddings that reflect such similarity.

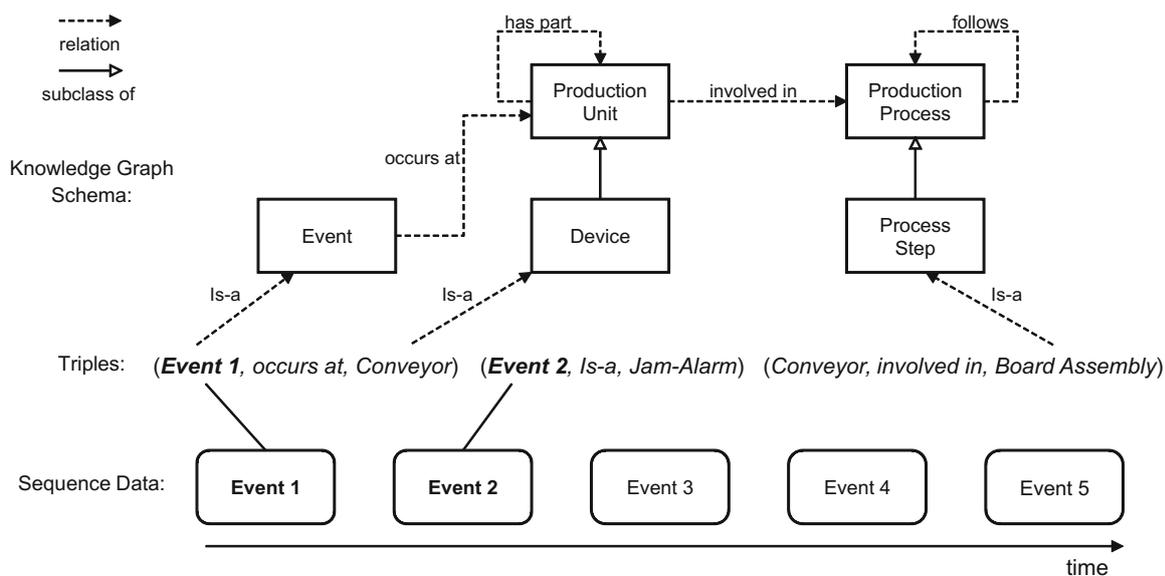


Fig. 1. Sequence data entities aligned to triples in the knowledge graph

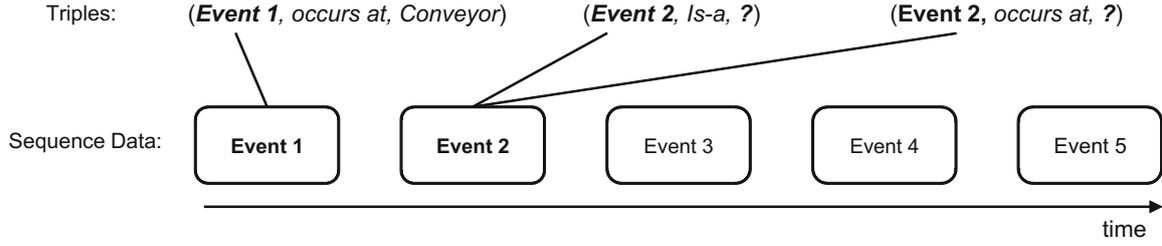


Fig. 2. Observing new events with unknown semantics

3 Problem Statement

In this section, we formally define the problem of learning joint representations of entities in KGs and operations data of manufacturing systems.

A **knowledge graph**, denoted as \mathcal{KG} is a directed graph with labeled edges. Each edge is represented in form of *triples* (h, r, t) that indicates the existence of a relationship between the head entity h and the tail entity t by the labeled relation r . Head and tail are contained in the set of entities $h, t \in \mathcal{E}$ and each relation respectively in the set of relations $r \in \mathcal{R}$.

A **sequence data set**, denoted as $\mathcal{D} = \{(x_1, \dots, x_i, \dots, x_m)_j^T\}$, is a set of sequences, where each sequence consists of an ordered set of event entities x_i . The length m of each sequence can vary depending on a sequence window size. It is implied that there exists a mapping of event entities x_i to entities in \mathcal{E} , i.e. event entities are also represented in the KG.

Knowledge Graph Embeddings. Given \mathcal{KG} , the problem of learning **knowledge graph embeddings** is to encode all entities in \mathcal{E} and relations in \mathcal{R} in a continuous low-dimensional vector space, i.e. $\mathbf{h}, \mathbf{t} \in \mathbb{R}^d$ and relation $\mathbf{r} \in \mathbb{R}^d$. In order to learn useful representations, a meaningful distance measure has to be employed, e.g. in the original TransE model [1], $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. This means that *translating* entity h with relation r should end up close at its tail entity t in the latent d -dimensional space. It has been shown that these *translation embeddings* can be effectively learned by using a ranking loss with the intuition that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ should be close for true triples and far apart for false/unknown ones. Formally, the learning objective is formulated as minimizing a margin-based ranking loss:

$$\mathcal{L}_{KG} = \sum_{(h,r,t) \in \mathcal{KG}} \sum_{(h',r,t') \in S'_{h,r,t}} \max(0, 1 + \text{dist}(\mathbf{h} + \mathbf{r}, \mathbf{t}) - \text{dist}(\mathbf{h}' + \mathbf{r}, \mathbf{t}')) \quad (1)$$

where $\text{dist}(\cdot)$ is some distance function (e.g. Euclidian) and $S'_{h,r,t}$ is a set of negative samples, i.e. artificially constructed false triples by replacing h or t with a random entity. This loss is minimized when the translation of correct triples is closer than that of unknown ones by a constant margin, here 1.

Sequential Data Embeddings. Given \mathcal{D} , the problem of learning **sequential embeddings** of entities x_i is similar to knowledge graphs, i.e. encode all entities in the same low-dimensional vector space, $\mathbf{x}_i \in \mathbb{R}^d$, where semantically

similar entities should end up close to each other in this latent space. Learning this kind of embeddings follows the distributional semantics hypothesis which states that similar entities occur in similar context. This has been one of the key ideas in the field of Natural Language Processing (NLP), since these embeddings tend to exhibit natural relations between words (e.g. capture synonymous meanings) [6]. Distributed representations are obtained by assuming that similarity between entities in the data can be modeled with a distribution, formally $P(x_i|W)$, i.e. the occurrence of entity x_i depends on and can be predicted from its surrounding window events W . Figure 3 displays how *Event 3* can be modeled from its surrounding events in a sliding time window of length m through the event sequences. It is assumed that events having similar causes and effects share similar semantics.

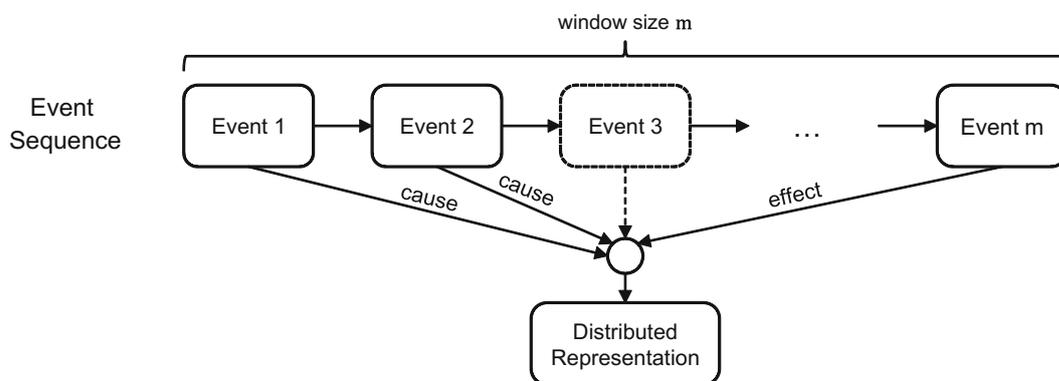


Fig. 3. Representations of event entities are learned from surrounding context

Mathematically, the probability distribution of predicting target entity x_i from its surrounding entities can be expressed by a categorical distribution, e.g. the Softmax function:

$$P(x_i|W_i) = \frac{\exp S(\mathbf{x}_i, \mathbf{W}_i)}{\sum_{j \neq i} \exp S(\mathbf{x}_j, \mathbf{W}_i)}, \quad (2)$$

where \mathbf{x}_i is the vector representation of entity x_i and $S(\cdot)$ is some similarity function between entities and their surrounding window entities represented as matrix \mathbf{W}_i . The objective function in terms of loss is given by the negative log likelihood:

$$\mathcal{L}_{Seq} = - \sum_i^n \log(P(e_i|W_i)) \quad (3)$$

Joint Embeddings. As the goal of this approach is to jointly model entities in the knowledge graph as well as in the sequential data, we propose a joint learning model that is trained by simply adding both loss terms:

$$\mathcal{L}_{Joint} = \mathcal{L}_{Seq} + \mathcal{L}_{KG} \quad (4)$$

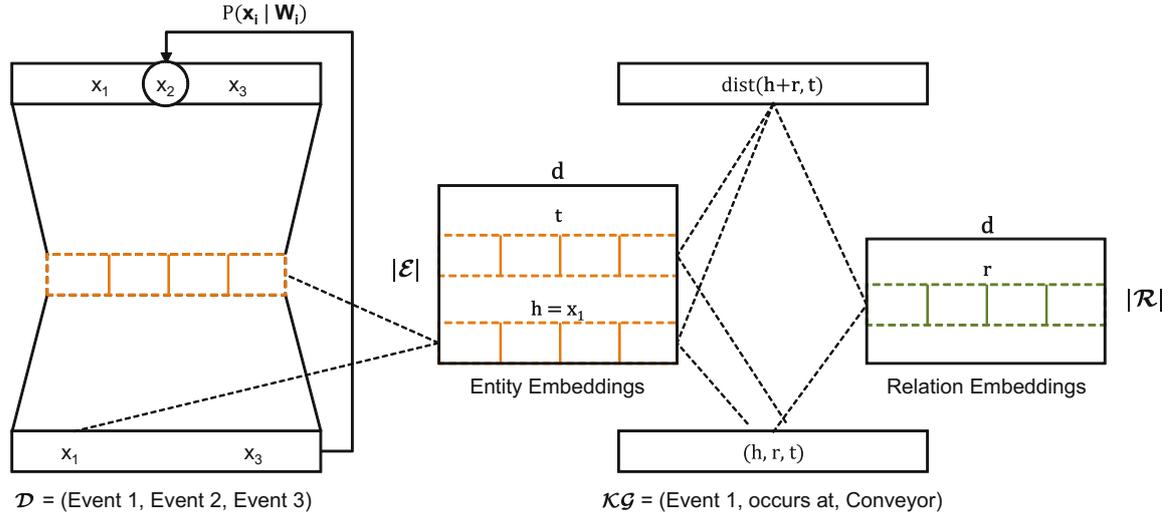


Fig. 4. Architecture of the joint embedding learning model

Minimizing the joint loss \mathcal{L}_{Joint} should result in solid embeddings of both, entities in the knowledge graph and the sequence data set. In reality, joint loss minimization is approximated using a state-of-the-art stochastic gradient descent optimizer. The key idea here is that entity embeddings are shared across both tasks and therefore the outcome should reflect co-occurrence of sequential data as well as the structure of the knowledge graph. The architecture of the joint embedding approach is shown in Fig. 4, where the $|\mathcal{E}|$ -by- d matrix of entity embeddings is located in the center. These embeddings are shared with the prediction model of entities in the sequential data on the left-hand side and the knowledge graph embedding model on the right-hand side. Note that in the depicted example this shared aspect is highlighted with *Event 1* having the same embedding (representation) in both models, i.e. $\mathbf{h} = \mathbf{x}_1$. The $|\mathcal{R}|$ -by- d matrix of relation embeddings on the right-hand side is solely used for the knowledge graph embeddings as it only influences distance calculation between triples.

4 Prototype Evaluation

We evaluated this approach on a real-world manufacturing KG data set coming from an automated assembly line. The event sequences are taken from a SCADA-level Alarms & Events database, whereas the initial KG was extracted from several spreadsheet files and CAD models. The final KG ended up with a size of about 3,700 triples about processes, equipments, and events, whereas the sequential data consisted of 57 thousand events occurrences. A prototypical implementation of the representation learning was implemented using the *TensorFlowTM* library. For performance evaluation, the usual criteria are (cf. [1]):

- Mean Rank: The average predicted rank of the head or tail entity that would have been the correct one (1 indicating perfect rank)
- Hits Top-10: The fraction of predicted ranks that were in the top 10

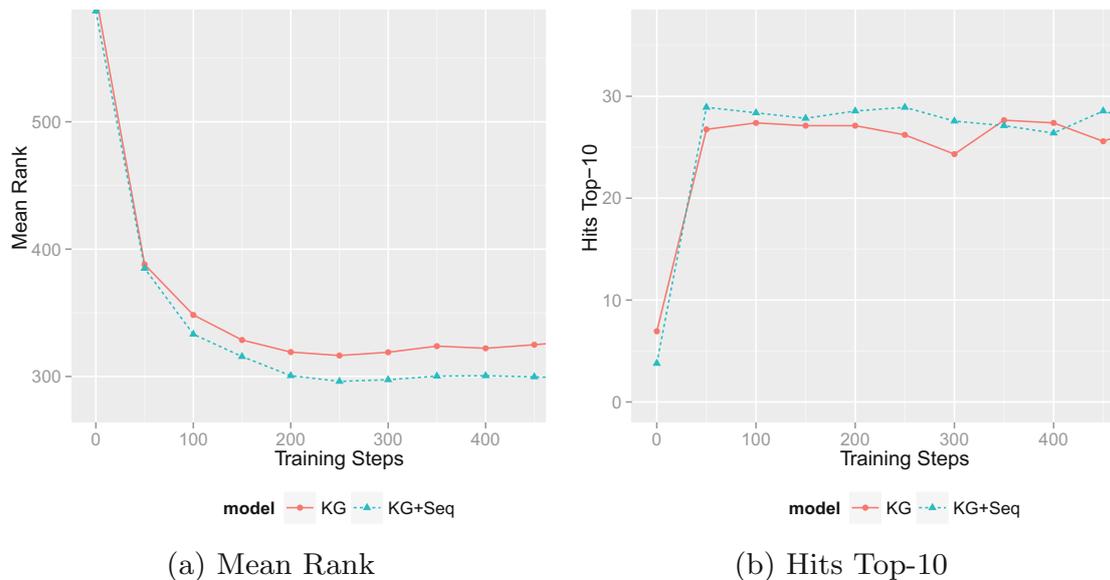


Fig. 5. Evaluation on hold-out test data (unknown triples) during training

Table 1. Models and data sets with and without sequential data

Model	$ \mathcal{KG} $	$ \mathcal{D} $	Test size	Mean rank	Hits top-10
KG	3.7k	-	3%	316.46	27.66
KG + Seq	3.7k	57k	3%	296.16	28.92

We compare two models, KG (knowledge graph embeddings only) and $KG+Seq$ (joint embeddings). In Fig. 5, the performance of KG and $KG+Seq$ are visualized during model training on a hold-out (unseen) test data set of incomplete triples, e.g. (Conveyor, involved in, ?). It can be seen that the joint model performs better in terms of lower mean rank and higher hits top-10 percentage (Table 1).

5 Related Work

We divide related work into two categories, limited to applications and techniques that are close to the one in this work.

Model Learning in Manufacturing. Machine learning has been used to discover influencing factors of manufacturing processes [14]. Other works of adapting to changing context have studied monitoring processing times in flexible production systems [10, 11] and more high-level architecture proposals for context extraction and self-adaption of production systems [12]. However, the authors do not specify a concrete methodology on how to extract context knowledge and align it with existing models.

Learning of Knowledge Graph Embeddings. Existing learning methods for KGs such as [1,9] have been extended to include many-to-many relationships [8] and to incorporate textual information to improve entity representation learning. Recently, word co-occurrences as sequential data were used in KG completion tasks [13]. In contrast to our approach, these works are focused on large-scale knowledge graphs containing noisy information.

6 Conclusion

An approach for automated recommendations for the alignment of semantics coming from operational data and manufacturing KGs was presented. Our model allows to predict missing relations introduced from changes in physical environments and unaligned event semantics, which can be detected and integrated into a global knowledge graph schema, thus lowering manual modeling effort. The joint representation of entities shows promising performance, which is vital for transition to fully automated synchronization, ensuring correct operation of monitoring and other management applications such as scheduling.

References

1. Bordes, A., Usunier, N., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. *Adv. NIPS* **26**, 2787–2795 (2013)
2. Datta, S.: Emergence of Digital Twins (2016). [arXiv:1610.06467](https://arxiv.org/abs/1610.06467)
3. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: *Proceedings of the 20th ACM SIGKDD 2014*, pp. 601–610 (2014)
4. Hirmer, P., Breitenb, U., Cristina, A.: Automating the provisioning and configuration of devices in the Internet of Things. *Complex Syst. Inf. Model. Q. (CSIMQ)* **9**(9), 28–43 (2017)
5. Kharlamov, E., et al.: Capturing industrial information models with ontologies and constraints. In: Groth, P., Simperl, E., Gray, A., Sabou, M., Krötzsch, M., Lecue, F., Flöck, F., Gil, Y. (eds.) *ISWC 2016 Part II*. LNCS, vol. 9982, pp. 325–343. Springer, Cham (2016). doi:[10.1007/978-3-319-46547-0_30](https://doi.org/10.1007/978-3-319-46547-0_30)
6. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: *International Conference on Machine Learning - ICML 2014*, vol. 32, pp. 1188–1196 (2014)
7. Lechevalier, D., Narayanan, A., Rachuri, S.: Towards a domain-specific framework for predictive analytics in manufacturing. In: *Proceedings - 2014 IEEE International Conference on Big Data, IEEE Big Data 2014*, pp. 987–995 (2015)
8. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence Learning*, pp. 2181–2187 (2015)
9. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graph. *Proc. IEEE* **28**, 1–23 (2015)

10. Ringsquandl, M., Lamparter, S., Brandt, S., Hubauer, T., Lepratti, R.: Semantic-guided feature selection for industrial automation systems. In: Arenas, M., et al. (eds.) ISWC 2015 Part II. LNCS, vol. 9367, pp. 225–240. Springer, Cham (2015). doi:[10.1007/978-3-319-25010-6_13](https://doi.org/10.1007/978-3-319-25010-6_13)
11. Ringsquandl, M., Lamparter, S., Lepratti, R.: Estimating processing times within context-aware manufacturing systems. In: Proceedings of the 2015 IFAC Symposium on Information Control Manufacturing (INCOM 2015) (2015)
12. Scholze, S., Stokic, D., Barata, J., Decker, C.: Context extraction for self-learning production systems. In: IEEE International Conference on India Information (INDIN), pp. 809–814 (2012)
13. Wang, Z., Li, J.Z.J.: Text-enhanced representation learning for knowledge graph. In: IJCAI, pp. 1293–1299 (2016)
14. Wuest, T., Irgens, C., Thoben, K.D.: An approach to monitoring quality in manufacturing using supervised machine learning on product state data. *J. Intell. Manuuf.* **25**, 1167–1180 (2014)

On Event-Driven Knowledge Graph Completion in Digital Factories

Martin Ringsquandl
Ludwig-Maximilians Universität
Munich, Germany
martin.ringsquandl@gmail.com

Evgeny Kharlamov
Oxford University
Oxford, United Kingdom
evgeny.kharlamov@cs.ox.ac.uk

Daria Stepanova
Max-Planck-Institut für Informatik
Saarbrücken, Germany
dstepano@mpi-inf.mpg.de

Steffen Lamparter
Siemens AG CT
Munich, Germany
steffen.lamparter@siemens.com

Raffaello Lepratti
Siemens PLM Software
Genoa, Italy
raffaello.lepratti@siemens.com

Ian Horrocks
Oxford University
Oxford, United Kingdom
ian.horrocks@cs.ox.ac.uk

Peer Kröger
Ludwig-Maximilians Universität
Munich, Germany
kroeger@dbs.ifi.lmu.de

Abstract—Smart factories are equipped with machines that can sense their manufacturing environments, interact with each other, and control production processes. Smooth operation of such factories requires that the machines and engineering personnel that conduct their monitoring and diagnostics share a detailed common industrial knowledge about the factory, e.g., in the form of knowledge graphs. Creation and maintenance of such knowledge is expensive and requires automation. In this work we show how machine learning that is specifically tailored towards industrial applications can help in knowledge graph completion. In particular, we show how knowledge completion can benefit from event logs that are common in smart factories. We evaluate this on the knowledge graph from a real world-inspired smart factory with encouraging results.

Keywords-Industrial Applications, Machine Learning, Knowledge Graphs, Events, Manufacturing

I. INTRODUCTION

A. Motivation

Digitalisation and automation are among the biggest trends in manufacturing [RVLB15]. Modern automated, or *smart*, factories are equipped with production and assembling machines that are not only capable of *sensing* their environments, e.g. reading RFID tags of products, but also of *interacting* with each other, e.g. raising a material shortage warning, and even performing *controlling* actions, e.g. turning on a cooling fan. Thus, it is common to distinguish in a smart factory its *physical part*, that is, machines and production lines, and its *digital representation*, referred to as the *digital twin* [Dat16].

The digital twin acts as an interface to the physical system, offering services such as automated monitoring, optimisation, and ultimately self-organisation without the need to interact with its actual physical representation [HBC17]. In Figure 1 we schematically visualise the separation between the physical part (in the bottom) and the digital twin (on top). The physical part consists of several machines for pre-assembly, assembly, and finishing of manufacturing. The digital twin consists of a specification of *Product1* that says

that the product has two components *PartA* and *PartB* and can be assembled with three operations, where the last two are conducted by robots *RobotA* and *RobotB* that in turn are located in a manufacturing line.

Development and maintenance of a digital twin poses significant challenges. In particular, one has to ensure that the relevant *industrial knowledge* about the plant is well captured and maintained. This knowledge representation is in the heart of the digital twin, upon which applications are built that rely and refer to it as backbone for communication. The knowledge should encompass both *master* and *operational data* (which are partially depicted in Figure 1). The former includes the catalog of plant’s equipment together with its technical documentation and the topology of its location in the manufacturing shop floor, personnel, warehouse data, and production blueprints. The latter includes log files of messages generated by individual pieces of equipment during manufacturing, flow of raw materials and products, and purchases.

Such industrial knowledge naturally satisfies the Big Data dimensions. Indeed, first, it is large in *volume*, e.g., at a mid size plant this knowledge may contain information about up to hundreds of machines, processes and materials, and hundreds of thousands of events. Second, the data *velocity* is high, e.g., a daily volume of transaction data generated only by shop-floor equipment can be up to hundreds of thousands of messages, master data is also dynamic in this regard: shop-floor devices may be added, moved, or removed due to maintenance, system configurations may change according to the respective production processes and products when, say, a new product variant requires an additional welding operation. Finally, the *variety* across various data sources is high, e.g., the transaction data is structured according to numerous relational schemata, technical documentation comes in flat files, and equipment capabilities are encoded in various proprietary formats.

Knowledge Graphs (KGs) are considered as a prominent approach to represent and share industrial knowl-

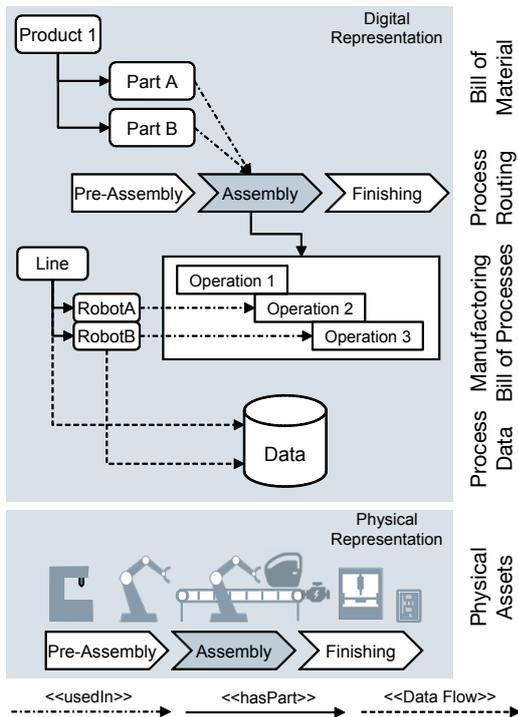


Figure 1: Schematic split in physical and digital representation of a factory

edge [KSÖ⁺14], [RLBL15], [KHS⁺17], [KMM⁺17], [HGKW16] since they offer a flexible mechanism for structuring both master and transaction data as an interconnected network of entities. Knowledge graphs are typically either available or can be exported as W3C standardised RDF datasets¹ that consist of *triples*, e.g., of the form $\langle \text{entity}, \text{predicate}, \text{entity} \rangle$. This format is well suited for both knowledge representation and exchange across applications over the network.

A typical KG in a digital factory consists of several logical parts that capture the main components of a digital twin in a smart factory [KGJ⁺16] and can be found in Figure 1: *Bill of Material* (i.e. a partonomy of products and materials), *Process Routings* (i.e. sequences of processes), the *Manufacturing Bill of Process* (i.e. assignment of machines to processes and more detailed operations), and *Process Data* (i.e. data collected from the machines during production). When equipped with rich semantic descriptions, these entities and their relationships are what constitutes the digital representation of the factory.

B. Challenges with Knowledge Graphs

Creating and maintaining a KG of good quality is a challenging task and an important bottleneck for the adoption of digital twins in industry [RLLK17]: due to the Big

Data dimensions of industrial knowledge, the corresponding KG cannot be manually created and curated. Thus, semi-automated techniques are needed to create and expand industrial KGs and a number of machine learning techniques have been proposed to address this challenge (see, e.g., [NMTG16] for overview). The main idea of these approaches is to convert entities and relations in a KG into a low-dimensional vector space and use it to infer missing information in the KG.

These approaches work better when the vector space is enhanced with some extra background knowledge, e.g., by also embedding textual documents attached to entities. We refer the reader to Section III for more details on existing machine learning approaches for KG expansion.

C. Contributions

In this work we show the effectiveness of machine learning for knowledge graph completion where the learning method is based on the vector space representation and accounts for background knowledge. In particular we develop an industrial scenario of a smart factory, a knowledge graph describing this factory, and how completion of this graph with the help of machine learning can be enhanced with a special type of background knowledge: log files of event messages generated by shop-floor manufacturing equipment.

The results of our evaluation are encouraging, where we apply the machine learning models to an industrial KG containing roughly 6,000 facts and create several use case scenarios of missing information. We show that our approach yields a boost in the quality of predicting missing links between digital twin entities and for certain relations we are able to restore missing information even in scenarios with highly incomplete relations.

II. USE CASE: INCOMPLETE INFORMATION IN A DYNAMIC MANUFACTURING SYSTEM

Our use case scenario is focused on synchronizing *Digital Twins* with their physical counterparts, more precisely we focus on a digital representation of automated factories that exhibit missing information. At the heart of such a digital factory representation is a data model describing the automation equipment, i.e. controllers and actuators, and other entities typically found in manufacturing environments, such as processes, products, and events.

The rest of the section is organised as follows. In Section II-A we describe the factory we study in our use case. In Section II-B we explain how we turn factory data in a KG. In Section II-C we describes event data we collected and why this data is relevant for our use case. Then, we present three scenarios of missing information in industrial knowledge graphs that we investigate in this work. All scenarios correspond to real-world situations in a factory that can be observed in smart factory environments. The scenarios are: change of factory equipment (Section II-D), introduction of

¹<http://www.w3.org/RDF/>

new processes in manufacturing (Section II-E), and update of equipments software that results in new events being emitted (Section II-F).

A. Factory Description

The factory we studied is a simplified simulation of a real-world smart factory, and it consists of four similarly structured production lines, each of which produces a particular variant of a common product family using a set of connected production equipment. The factory has 180 devices, 4 different products that consist of a total of 59 unique material parts, and 55 different manufacturing processes. Each device can perform some skills including drilling, welding, and assembling and operates by inputting and outputting some of the materials that are part of four different product variants. In total the devices emitted 728 unique event entities during the collection time period for this case study, more details on the event data are given in the subsequent section.

B. Manufacturing Data as a Knowledge Graph

Typically, manufacturing data is scattered throughout diverse data sources and formats (relational databases, spreadsheets, XML files). Since we rely on RDF-based knowledge graphs, we exploit an ontology driven ETL process, known as *Ontology-Based Data Access (OBDA)* in order to translate these heterogeneous data sources into RDF.

OBDA follows the classical data integration paradigm that requires the creation of a common ‘global’ schema that consolidates ‘local’ schemata of the integrated data sources, and mappings that define how the local and global schemata are related [DHI12]. In OBDA the global schema is an *ontology*: a formal conceptualisation of the domain of interest that consists of a *vocabulary*, i.e., names of classes, attributes and binary relations, such as *connectedTo*, *hasPart*, and *axioms* over the terms from the vocabulary that, e.g., assign attributes of classes, define relationships between classes, compose classes, class hierarchies, etc. The ontology we developed encodes generic specifications of manufacturing equipment, characteristics of sensors, materials, processes, descriptions of diagnostic tasks, etc. OBDA mappings relate each ontological term to a set of queries over the underlying data. OBDA mappings can be used in the same way as ETL rules for data transformation.

An overview of the KG that we obtained with the help of OBDA is shown in Table I, where $|\mathcal{E}_c|$ is the number of entities in the given class, $avg(In)$ represents the average number of incoming, and $avg(Out)$ the average number of outgoing links for each of the entity classes. Note that equipment entities are most densely connected in the KG, while events, although the largest class of entities in the KG, only have few outgoing links. In summary, the KG consists of 3,125 entities that are connected through 6,361 facts (triples) in 11 unique named relations (predicates).

Table I: Main entities in the digital twin knowledge graph

Entity Class	$ \mathcal{E}_c $	$avg(In)$	$avg(Out)$
Equipment	180	13.13	5.6
Process	55	4.89	7.0
Material	59	5.9	8.9
Event	728	0	2.17

C. Event Data

Factories, such as the one we simulated, are equipped with automation systems that continuously generate operational data, especially events, such as alarm codes or operator information messages. For a particular point in time, events from different locations in the factory are collected and later aggregated in log files. Observing these events enables the digital twin to trace some of the activity that is carried out by the physical equipment.

Due to the scattered layout of machines across factories, two consecutively emitted events are not necessarily correlated to each other, since their physical sources (i.e. production machines) may be completely independent. Nevertheless, as we will show in the following sections, co-occurrence patterns in event logs can be used to infer missing information contained in the factories KG.

For our use case study, we collected about 60,000 occurrences of events from the simulated factory.

D. Scenario: Changing Factory Equipment

The first scenario of completing missing information in the KG that we consider is related to equipment entities. More precisely, we study the effectiveness of KG completion by artificially removing certain links between equipment entities. Such missing information can naturally occur when additional devices are deployed at the factory, or existing devices need to be replaced due to maintenance. Having background knowledge in form of event sequences, this scenario studies how well such equipment connections can be automatically re-established through inference.

For our manufacturing KG, this scenario mainly affects two relations *hasPart* and *connectedTo*, as shown in Figure 2. Both sides of the figure show the same KG consisting of an assembly line that has two conveyors as parts. Also two exemplary event entities are related to their respective source locations. On the left-hand side of the figure the *connectedTo* relation is artificially removed, as indicated by the dashed arrow. On the right-hand side one of the *hasPart* relations is removed.

The KG completion task is to obtain a correct recommendation for both types of missing links, such that the missing information is restored via inference.

E. Scenario: Introduction of New Processes

In this scenario, we consider missing links between process entities that emerge when, e.g. new product variants

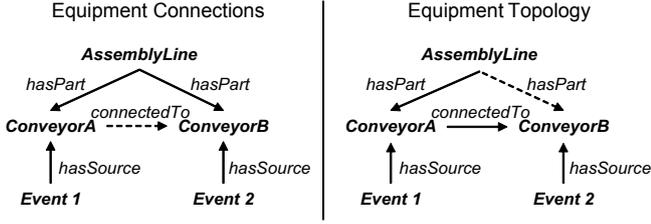


Figure 2: Equipment scenario, left: missing links that state facts about physical device connections. Right: Missing partonomy links of equipment

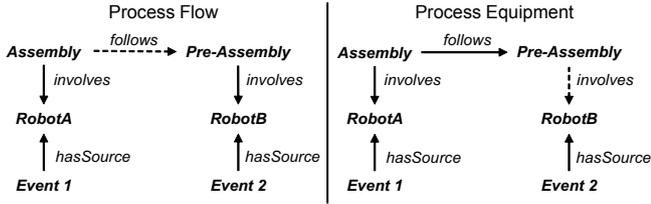


Figure 3: Process scenario, left: missing links that state facts the production process flow. Right: Missing links to involved equipment in the process

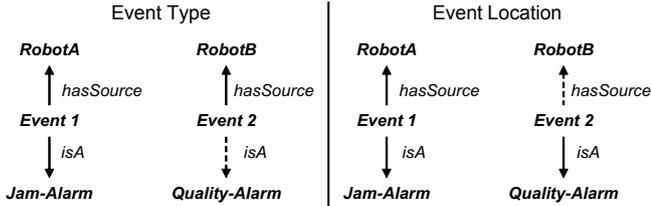


Figure 4: Event scenario, left: missing type information of event entities. Right: Missing source information of events

are introduced that require a different production process. Furthermore, in a separate study we consider missing links from processes to their involved equipment entities, as shown in Figure 3.

F. Scenario: Observing New Events

This scenario emerges frequently in automated factories in case the control logic that generates events of machines is modified, for example a new alarm message needs to be shown to the operator as soon as a certain oil pressure threshold is reached.

When new event entities are observed in the log that are missing annotations in the KG, the completion task can also be seen as a KG population task, since usually no previous information about a new event entity is present in the KG. Thus, predicting missing links means essentially introducing a completely new entity to the KG. This task corresponds to a well-known *zero-shot* learning.

III. MACHINE LEARNING APPROACH

We now briefly describe our machine learning approach.

A *knowledge graph* \mathcal{K} is a set of (positive) facts about a certain domain of interest represented as a set of triples of the form $\langle h, r, t \rangle$, where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. In our scenario the event-centric data is represented using a subset $\mathcal{X} \subseteq \mathcal{E}$ of entities from a given KG. A *sequence* is an ordered set of (event) entities $s_i = (e_1, \dots, e_{m_i})$, where $e_k \in \mathcal{X}$. A *sequence dataset* \mathcal{S} is a set of sequences $\mathcal{S} = \{s_1, \dots, s_n\}$.

Given a triple $\langle h, r, _ \rangle$ (resp. $\langle _, r, t \rangle$) with a missing entity, a KG \mathcal{K} and a sequence dataset \mathcal{S} , the *event-enhanced KG completion* is to utilize the background knowledge in \mathcal{S} to predict the missing t (resp. h) by retrieving a ranked list of possible candidate entities from a subset of all entities \mathcal{E} .

Following the common practice, we solve the KG completion problem by reducing it to a representation learning task, whose main goal is to represent entities h, t and relations r occurring in \mathcal{K} in a low dimensional, e.g., d -dimensional, vector space as vectors $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$, which are referred to as *embeddings*. In contrast to previously proposed extensions of the standard embedding approach, which improve the accuracy of the learned representations by taking into account additional knowledge in the textual form [WL16], we make use of the background knowledge represented as sequences of events. Unlike text, sequence datasets reveal the exact order of event occurrences, they do not follow any grammatical rules, e.g., miss stop words and reflect the structure of the process that produced these sequences.

In this work we are looking for *event-enhanced knowledge graph embeddings* to construct representations of $\mathbf{h}, \mathbf{t}, \mathbf{r}$ that leverage the sequential relationship between entities in \mathcal{S} . We note that despite the fact that \mathcal{S} is only directly connected to entities in \mathcal{X} , a collective learning effect is introduced by incorporating event sequence information into the learning of KG embeddings that propagates event entity representations to other parts of the KG.

Our joint model combines the objective of KG embeddings $\mathcal{L}_{\mathcal{K}}$ and the objective of event sequence data embeddings $\mathcal{L}_{\mathcal{S}}$ using the joint formulation proposed in [XHMZ17]:

$$\mathcal{L}_{joint} = \mathcal{L}_{\mathcal{K}} + \alpha \mathcal{L}_{\mathcal{S}}, \quad (1)$$

where, α is a hyper-parameter used as a weighting factor. Simultaneous training of both objective functions within an aggregated objective allows both models to influence each other through various parameter interconnections.

We consider three versions of \mathcal{L}_{joint} .

- EKL_{Skip} follows the intuition of the skipgram model [MCCD13], which relies on the distributed representation hypothesis that a word is defined by its surrounding context. The goal of this model is to predict a context event given a particular center event in the log.
- EKL_{Concat} accounts for the sequential dependencies among events: it deals with the characteristics of short event sequences and preserves the information

about their order when encoding entity embeddings; we achieve this by adapting a vector concatenation-based model that is similar to the paragraph-vector model [LM14] without the notion of a paragraph.

- EKL_{RNN} employs a many-to-one vanilla RNN and feeds the $m - 1$ predecessor events as inputs to make a prediction for the m -th event based on the last output state of the RNN.

IV. EVALUATION

A. Evaluation Protocol

We apply and evaluate the three novel approaches for event-enhanced KG completion, i.e., EKL_{Skip} , EKL_{Concat} and EKL_{RNN} , on each of the scenarios of Section II. In each of the experiments, the original KG is split into a training, validation (10 % of overall KG) and a test set that contains the artificially removed triples. Final model performance results are calculated based on the test set, for which we report a commonly-used evaluation metric:

- **Mean Rank:** the average rank of the entities (head and tail) that would have been the correct ones.

The mean rank in our experiments corresponds to the *filtered* version that has been originally proposed in [BUG⁺13], i.e. in the test set when ranking a particular triple $\langle h, r, t \rangle$, all $\langle h, r, t' \rangle$ triples with $t \neq t'$ are removed. Employing grid search through the hyper-parameters we determine the best configuration by mean rank on the validation set with early stopping over a maximum of 100 epochs.

As a baseline, we use the default TransE model. For the incorporation of event sequence data, the skipgram model is also already a strong baseline in terms of comparison to order-preserving embeddings models.

B. Results

In order to evaluate performance of our approach on the scenarios of Section II-D, we have designed dedicated test sets corresponding to the triples of the scenarios. For example, if triples with *connectedTo* relation are missing, then the test set contains a controlled proportion of all $\langle h, connectedTo, t \rangle$ triples in the KG. We call this proportion the *Out-of-KG-size* and varied it between 25%, 50% and 75% for each relevant predicate. This way we can simulate the degree of missing links and therefore can assess how well the models can handle different amounts of missing information. The performance results with respect to mean rank are shown in Figure 5. We now discuss them in details.

For the equipment connections scenario (top of Figure 5), it can be seen that incorporating events in the KG completion task does result in a lower mean rank compared to the standard TransE model as the proportion of missing connections is increased. On the other hand, this is not the case for the partonomy relation *hasPart*, since the event-enhanced models' prediction quality is decreasing with growing Out-of-KG-size.

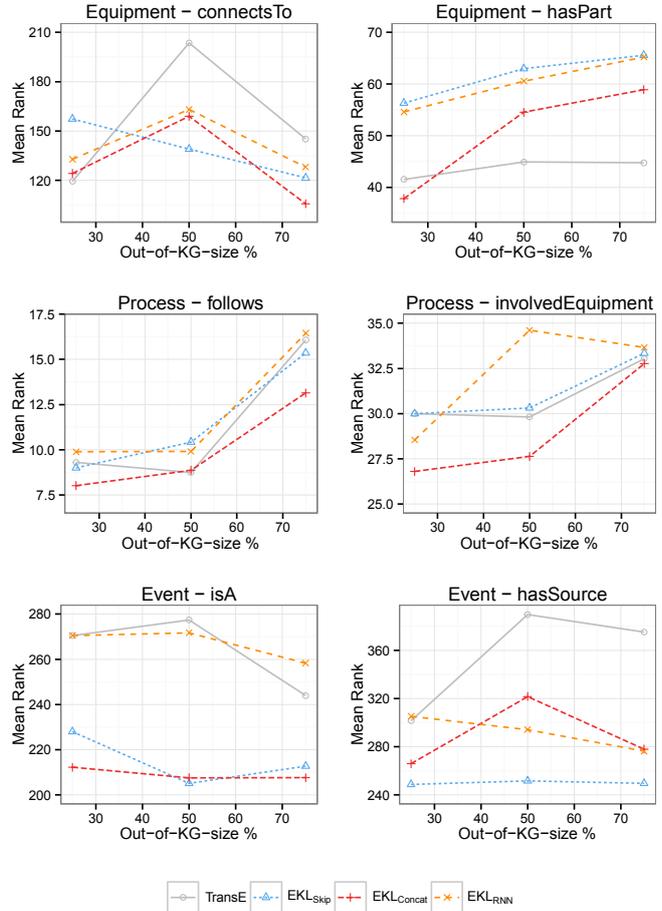


Figure 5: Mean rank statistics for KG completion task in each of the use case scenarios with increasing test set size

For the new process scenario (middle of Figure 5), and *follows* and *involvedEquipment* relations, especially EKL_{Concat} could robustly predict missing links with low mean rank. At the same time we observe that EKL_{RNN} and EKL_{Skip} in some parts perform worse than standard TransE. This supports our intuition for EKL_{Concat} that it can better capture process related relations.

For the new event scenario (bottom of Figure 5), one can see that, as expected, the event *isA* and *hasSource* relations benefit the most from incorporating their sequence as additional information. Most noticeably, EKL_{Skip} is robust to increasing size of missing links in both settings.

V. CONCLUSION AND FUTURE WORK

In this paper we presented a concrete industrial scenario of a Siemens digital twin based on a knowledge graph that models a physical factory, including its equipments, processes, and also operational data, such as events. We

showed how missing knowledge in this graph can be predicted by relying on machine learning that combines KGs with background data in the form of log files of events. In our scenarios the missing data corresponds to common changes in factories. We evaluated our machine learning method in these scenarios and showed that with the help of our event-enhanced learning model it can do a good quality KG completion and therefore synchronise the digital and the physical representations of a smart factory. The KG completion performance results in most of the scenarios are promising and our models outperform a state-of-the-art KG completion model. Our approach performs best for population of KGs with new event entities.

Acknowledgements

This work was partially supported by the EPSRC projects DBOnto, MaSI³, ED³, and VADA.

REFERENCES

- [BUG⁺13] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.
- [Dat16] Shoumen Datta. Emergence of Digital Twins. *arXiv:1610.06467*, 2016.
- [DHI12] AnHai Doan, Alon Y. Halevy, and Zachary G. Ives. *Principles of Data Integration*. Morgan Kaufmann, 2012.
- [HBC17] Pascal Hirmer, Uwe Breitenb, and Ana Cristina. Automating the Provisioning and Configuration of Devices in the Internet of Things. *CSIMQ*, 9:28–43, 2017.
- [HGKW16] Ian Horrocks, Martin Giese, Evgeny Kharlamov, and Arild Waaler. Using semantic technology to tame the data variety challenge. *IEEE Internet Computing*, 20(6):62–66, Nov 2016.
- [KGJ⁺16] Evgeny Kharlamov, Bernardo Cuenca Grau, Ernesto Jiménez-Ruiz, Steffen Lamparter, Gulnar Mehdi, Martin Ringsquandl, Yavor Nenov, Stephan Grimm, Mikhail Roshchin, and Ian Horrocks. Capturing industrial information models with ontologies and constraints. In *ISWC*, pages 325–343, 2016.
- [KHS⁺17] Evgeny Kharlamov, Dag Hovland, Martin G. Skjveland, Dimitris Bilidas, Ernesto Jimnez-Ruiz, Guohui Xiao, Ahmet Soylu, Davide Lanti, Martin Rezk, Dmitriy Zheleznyakov, Martin Giese, Hallstein Lie, Yannis Ioannidis, Yannis Kotidis, Manolis Koubarakis, and Arild Waaler. Ontology based data access in Statoil. *Journal of Web Semantics*, 44:3 – 36, 2017.
- [KMM⁺17] Evgeny Kharlamov, Theofilos Mailis, Gulnar Mehdi, Christian Neuenstadt, zgr zep, Mikhail Roshchin, Nina Solomakhina, Ahmet Soylu, Christoforos Svingos, Sebastian Brandt, Martin Giese, Yannis Ioannidis, Steffen Lamparter, Ralf Mller, Yannis Kotidis, and Arild Waaler. Semantic access to streaming and static data at Siemens. *Journal of Web Semantics*, 44:54 – 74, 2017.
- [KSÖ⁺14] Evgeny Kharlamov, Nina Solomakhina, Özgür Lütfü Özçep, Dmitriy Zheleznyakov, Thomas Hubauer, Steffen Lamparter, Mikhail Roshchin, Ahmet Soylu, and Stuart Watson. How semantic technologies can enhance data access at siemens energy. In *ISWC*, pages 601–619, 2014.
- [LM14] Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *ICML*, volume 32, pages 1188–1196, 2014.
- [MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 1–9, 2013.
- [NMTG16] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.
- [RLBL15] Martin Ringsquandl, Steffen Lamparter, Sebastian-Philipp Brandt, and Raffaello Lepratti. Semantic-guided Feature Selection for Industrial Automation Systems. In *ISWC*. Springer, 2015.
- [RLLK17] Martin Ringsquandl, Steffen Lamparter, Raffaello Lepratti, and Peer Kröger. Knowledge Fusion of Manufacturing Operations Data using Representation Learning. In *IFIP APMS*. Springer, 2017.
- [RVLB15] Roland Rosen, Georg Von Wichert, George Lo, and Kurt D. Bettenhausen. About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine*, 28(3):567–572, 2015.
- [WL16] Zhigang Wang and Juan-Zi Juanzi Li. Text-Enhanced Representation Learning for Knowledge Graph. *IJ-CAI*, pages 1293–1299, 2016.
- [XHMZ17] Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions. *AAAI*, pages 1–10, 2017.

Event-Enhanced Learning for KG Completion

Martin Ringsquandl^{1,2(✉)}, Evgeny Kharlamov³, Daria Stepanova⁴,
Marcel Hildebrandt¹, Steffen Lamparter², Raffaello Lepratti⁵, Ian Horrocks³ ,
and Peer Kröger¹

¹ Ludwig-Maximilians University, Munich, Germany

`martin.ringsquandl@siemens.com`

² Siemens AG CT, Munich, Germany

³ University of Oxford, Oxford, UK

⁴ Max-Planck Institut für Informatik, Saarbrücken, Germany

⁵ Digital Factory, Siemens PLM Software, Plano, USA

Abstract. Statistical learning of relations between entities is a popular approach to address the problem of missing data in Knowledge Graphs. In this work we study how relational learning can be enhanced with background of a special kind: event logs, that are sequences of entities that may occur in the graph. Events naturally appear in many important applications as background. We propose various embedding models that combine entities of a Knowledge Graph and event logs. Our evaluation shows that our approach outperforms state-of-the-art baselines on real-world manufacturing and road traffic Knowledge Graphs, as well as in a controlled scenario that mimics manufacturing processes.

1 Introduction

Knowledge Graphs (KGs) nowadays power many important applications including Web search¹, question answering [3], machine learning [19], data integration [10], entity disambiguation and linking [5, 8]. A KG is typically defined as a collection of triples $\langle \textit{entity}, \textit{predicate}, \textit{entity} \rangle$ that form a directed graph where nodes are entities and edges are labeled with binary predicates (relations). Examples of large-scale KGs range from general-purpose such as Yago [24] and DBPedia [12] to domain specific ones such as Siemens [10] and Statoil [9] corporate KGs.

Large-scale KGs are often automatically constructed and highly incomplete [6] in the sense that they are missing certain triples. Due to their size and the speed of growth, manual completion of such KGs is infeasible. In order to address this issue, a number of relational learning approaches for *automatic KG completion* have been recently proposed, see [6, 16] for an overview. Many of these approaches are based on learning representations, or *embeddings*, of entities and relations [4, 17, 22]. It was shown that the quality of embeddings can

¹ https://en.wikipedia.org/wiki/Knowledge_Graph.

conveyor C, i.e., $\langle CoilJam, hasSource, ConveyorC \rangle$. Event entities have further semantics described by the typing, e.g. *CoilJam* is of type *JamAlarm*, severity levels, and possible emitting source locations. At the same time, the KG misses the facts that the conveyors A and C are connected in the factory; that *BoardJam* is of type *JamAlarm*, and *HighEnergy (HE)* has the source *ConveyorA* and is of type *EnergyEvent*.

Additionally, in the example, we assume that an event log recorded during the operation of the factory consists of three following sequences over event entities:

$$(HE, LE, LE, BoardJam), (HE, LE, LE, CoilJam), (LE, HE, LE, HE).$$

Observe that the event log suggests that a jam typically occurs after a sequence of two consecutive low energy consumption (*LE*) events. \square

An event log gives external knowledge to the KG by specifying frequent sequential patterns on the KG's entities. These patterns capture some processes that the nodes of a KG can be involved in, i.e., manufacturing with machines described by the KG, traveling by a person mentioned in the KG, or traffic around traffic signals. This type of external knowledge has conceptual differences from text corpora where KG entities are typically described in a natural language and where occurrences of KG entities do not necessarily correspond to any process. Events are also different from rules or constraints that introduce formal restrictions on some relations.

The goal of this work is to understand how event logs can enhance relational learning for KGs. We address this problem by proposing an *Event-enhanced Knowledge Learning (EKL)* approach for KG completion that intuitively has two sub-steps:

1. *Event alignment*, where event entities are aligned in a low-dimensional vector space that reflects sequential similarity, and
2. *KG completion*, where the KG is extended with missing edges that can be either event-specific, e.g., such as the *type* edge between *BoardJam* and *JamAlarm* in Example 1, or not event-specific, e.g., such as *connectedTo* between *ConveyorA* and *ConveyorC* in Example 1.

Observe, the event logs *directly* influence the first step while also *indirectly* the second step of EKL. Hence, we expect a collective learning effect in a sense that the overall KG completion can benefit from event alignment, and vice versa.

Example 2. During the first step EKL will align *BoardJam* and *CoilJam* to be similar. In the second step EKL will accordingly adjust entities *ConveyorC* and *ConveyorB* and then predict that *ConveyorA* is likely to also be connected to *ConveyorC*. Intuitively the missing link between the conveyors can be inferred from the sequential pattern in the event log: the log tells us that both *BoardJam* and *CoilJam* occur as a consequence of two consecutive *LE* events and therefore exhibit similar semantics. This similarity is carried to conveyor entities B and C, which leads to an increased likelihood that they both follow the same entity *ConveyorA*. \square

Note that the prediction of event-specific missing links is not the standard task for relational learning since we are predicting links *within* the background. Moreover, our approach can address the *zero-shot scenario*, where some event entities only appear in the event log, but they are novel to the KG (it is marked with red in Fig. 1). E.g., *HE* in Example 1 corresponds to an entity that is missing in the KG, that has to be aligned during the first step of EKL and then linked to *ConveyorA* as well as to its type during the second step of EKL. Thus, EKL can also populate a KG with new (unseen) entities.

The contributions of our work are as follows:

- Several EKL approaches to KG completion that comprise
 - two model architectures that allow to combine (representations of) a KG and an event log for simultaneous training of both representations;
 - three models for event logs that reflect different notions of event context.
- An extensive evaluation of our approach and comparison to a state-of-the-art baseline on real-world data from a factory, on smart city traffic data, and controlled experiment data. Our results show that we significantly outperform two state-of-the-art baselines and the advantages are most visible for predicting links between entities that reflect the sequential process nature within the KG.

Finally, note that we presented a very preliminary version of this work as a short in-use paper [20]. Here we are significantly different from [20] and extend it, since [20] does not describe our EKL models and it only focusses on several simple KG extension scenarios that we do not study here.

2 Existing Methods for KG Completion

We now review the problem of KG completion and the existing methods to address it, following the standard problem definition, c.f. [4, 22, 26]. Let \mathcal{E} be a finite set of (all possible) entities and \mathcal{R} a finite set of relations. A KG \mathcal{K} is a set of triples $\langle h, r, t \rangle$, where $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. Given a KG \mathcal{K} and a triple $\langle h, r, _ \rangle$ (resp. $\langle _, r, t \rangle$) where ‘ $_$ ’ denotes a missing entity, the *KG completion* problem is to predict the missing t (resp. h) by computing for each $e \in \mathcal{E}$ a score $f(\langle h, r, e \rangle)$ (resp. $f(\langle e, r, t \rangle)$) that reflects the likelihood of the triple $\langle h, r, e \rangle$ (resp. $\langle e, r, t \rangle$) being in the KG.

Statistical Relational Learning for KG Completion. Statistical relational learning has gained a lot of attention by the research community, including translation-based models [4, 26], latent tensor factorization [18], neural tensor networks [23] and others (see e.g., [16] for an overview). In this work we focus on translation-based models. They address the KG completion problem by reducing it to a representation learning task where the main goal is to represent entities h , t and relations r occurring in \mathcal{K} in a low-dimensional, say d -dimensional, vector space $\mathbf{h}, \mathbf{t}, \mathbf{r} \in \mathbb{R}^d$, which define the model parameter matrices $\mathbf{W}_{\mathcal{E}} \in \mathbb{R}^{|\mathcal{E}| \times d}$ and

$\mathbf{W}_{\mathcal{R}} \in \mathbb{R}^{|\mathcal{R}| \times d}$ respectively. These parameters are typically referred to as *latent representations* or simply *embeddings*.

Such KG representations have been shown to be effectively learned by using a ranking loss with the objective that true triples should be ranked before false/unknown ones according to the scoring function. This learning objective is formulated as minimizing a margin-based ranking loss:

$$\mathcal{L}_{\mathcal{K}} = \sum_{(h,r,t) \in \mathcal{K}} \sum_{(h',r,t') \in N} \max(0, \gamma + f(\mathbf{h}, \mathbf{r}, \mathbf{t}) - f(\mathbf{h}', \mathbf{r}, \mathbf{t}')), \quad (1)$$

where $f(\cdot)$ is some scoring function that operates on the entity and relation embeddings $\mathbf{h}, \mathbf{r}, \mathbf{t}$ and $\mathbf{h}', \mathbf{r}, \mathbf{t}'$ from N , which is a set of negative examples, i.e. presumably false triples not contained in \mathcal{K} . This loss is minimized when the true triples outscore the false ones by a constant margin γ . In practice the training is done using mini-batches of \mathcal{K} , instead of iterating over all triples, together with stochastic-gradient descent (SGD), since this introduces more variance in the embedding parameter updates and can prevent early convergence in local optima.

For example, in the TransE model [4], given \mathcal{K} such f relies on distance or similarity between vectors. Intuitively, TransE follows the intuition that there is a linear relation for triples $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, hence the scoring function is defined as a dissimilarity measure (e.g. ℓ^2 -norm) $f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. This means that *translating* entity h with relation r should end up close to its tail entity t in the latent d -dimensional space. In order to prevent overfitting, the magnitudes of parameters in TransE are normalized after each mini-batch to unit-norm vectors, i.e. $\forall e \in \mathcal{E} : \|\mathbf{e}\| = 1$.

Enhancing KG Learning with Background Knowledge. It was shown that traditional representation learning can be improved using background knowledge. Most prominently, external text corpora can be used as background [27–29, 31]. The main approaches follow the idea of computing two separate representations of entities: a text-based and a KG-based one. There are two proposals to combine both representations:

1. include a linear combination layer to directly modify \mathbf{h} , \mathbf{r} , and \mathbf{t} in $\mathcal{L}_{\mathcal{K}}$ of Eq. 1, or
2. add a dedicated learning objective for text-based representations to $\mathcal{L}_{\mathcal{K}}$.

The TEKE model [27] follows the first proposal, by incorporating textual context of entities into a KG by exploiting a co-occurrence network of entities and words in the text thus defining a combination between pre-trained language model word embeddings and KG entities. It includes $\mathbf{n}(h)$ as the weight-averaged neighborhood word vector representation of an entity h and then applies a linear combination $\hat{\mathbf{h}} = \mathbf{A}\mathbf{n}(h) + \mathbf{h}$ to make up the final entity representation used in triple scoring. Furthermore, TEKE uses a weighted average of the merged neighborhood word embeddings for pairs of entities h, t in the text as $\mathbf{n}(h, t)$ to transform the relation embeddings $\hat{\mathbf{r}} = \mathbf{B}\mathbf{n}(h, t) + \mathbf{r}$.

The DKRL [29] follows the second proposal and adds three $\mathcal{L}_{\mathcal{K}}^i$ to the objective for text-based representations, where $\mathcal{L}_{\mathcal{K}}^1$ uses a translation objective within text-based representations, $\mathcal{L}_{\mathcal{K}}^2$ is a mixed translation from text-based to KG-based and $\mathcal{L}_{\mathcal{K}}^3$ from KG-based to text-based. Intuitively, DKRL considers correlation between entities within \mathcal{K} , between \mathcal{K} and the text and within the text. The text embeddings are learned using a continuous bag of words or a deep convolutional neural network. The KG and the text-based representations are then jointly optimized.

SSP [28, 31] also follow the second proposal and exploit the semantic information about KG’s entities given in the form of textual entity descriptions. The SSP method strengthens the effect of text descriptions by performing the embedding process in the semantic subspace and a topic model for entity text descriptions. They combine the objective of KG embeddings $\mathcal{L}_{\mathcal{K}}$ and of text embeddings $\mathcal{L}_{\mathcal{S}}$ using the joint formulation:

$$\mathcal{L}_{joint} = \mathcal{L}_{\mathcal{K}} + \alpha \mathcal{L}_{\mathcal{S}}, \quad (2)$$

where, α is a hyper-parameter used as a weighting factor. The main advantage of this approach is that the simultaneous training of both objective functions within an aggregated objective allows both models to influence each other.

3 Event-Enhanced KG Completion

In this section we present our approach to KG completion enhanced with event logs: we start with the problem definition, proceed to limitations of existing approaches to address the problem, propose our adaptation of the joint model formulation to the event-based setting based on $\mathcal{L}_{\mathcal{K}}$ and $\mathcal{L}_{\mathcal{S}}$, and define several ways to combine $\mathcal{L}_{\mathcal{K}}$ and $\mathcal{L}_{\mathcal{S}}$.

3.1 Problem Definition

Let $\mathcal{X} \subseteq \mathcal{E}$ be a set of *event* entities. An *event log* s is a finite sequence (e_1, \dots, e_m) of entities from \mathcal{X} and a sequence dataset \mathcal{S} a set $\{s_1, \dots, s_n\}$ of event logs. E.g., an event log $(LE, HE, BoardJam, ShutdownA)$ is generated by machines during operation.

The *event-enhanced KG completion* problem is, given a KG \mathcal{K} , a sequence dataset \mathcal{S} , and a triple $\langle h, r, _ \rangle$ (resp. $\langle _, r, t \rangle$), to predict the missing t (resp. h) by exploiting both \mathcal{K} and \mathcal{S} for the computation of a score $f(\langle h, r, e \rangle)$ (resp. $f(\langle e, r, t \rangle)$) for each $e \in \mathcal{E}$. We consider three variations of the event-enhanced completion problem with respect to the entities h, t in the given/predicted triple: the first setting corresponds to the standard KG completion problem, while the other two are specific for our scenario.

1. *non-event entities*: neither given nor predicted entities of $\langle h, r, - \rangle$ (resp. $\langle -, r, t \rangle$) are from \mathcal{X} , as in $\langle ConveyorA, connectedTo, ConveyorB \rangle$ from Example 1,
2. *event entities known by KG*: each given or predicted event entity of $\langle h, r, - \rangle$ (resp. $\langle -, r, t \rangle$) is in \mathcal{K} , as in $\langle BoardJam, type, JamAlarm \rangle$ from Example 1,
3. *event entities unknown by KG*: either given or predicted event entity of $\langle h, r, - \rangle$ (resp. $\langle -, r, t \rangle$), does not appear in \mathcal{K} , as for HE from Example 1. This corresponds to the problem of *zero-shot learning*.

3.2 Limitation of Existing Methods

We now discuss why the existing background-enhanced learning approaches are insufficient or cannot be naturally applied at all to our setting. Approaches of the first kind such as TEKE rely on the assumption that all entities of the KG should also appear in the background. This assumption is critical to TEKE’s enhancement of $\mathbf{Bn}(h, t)$, which is undefined if either h or t have no text representation. In our setting only the event entities occur in the background and in applications such as manufacturing only a small fragment of KG entities are actually event entities. Moreover, TEKE relies only on mere co-occurrences of words and KG entities in the text corpora and ignores the sequential correlation among entity occurrences. Approaches of the second kind including DKRL and SSP require that a dedicated piece of background, that is, a text corpus, is attached to each individual entity of the KG, while in our setting all event entities share a single event log, i.e. the same background is attached to every event entity. Although the convolutional version of DKRL is able to respect sequential correlation between the words in the entity description, it can not be directly applied to sequences of event entities, since this requires a different embedding objective. Instead of using the final output of the last convolutional layer as representation of the entity, we need to learn representations of the actual tokens in the sequence. The same limitations hold for the topic model of SSP, which also requires a dedicated background for each entity in the KG.

We note that that is a body of work on Business Process Mining that we see as relevant to our work since manufacturing can be seen as a business process. The exact nature of relation requires further study and it as a future work.

3.3 Adaptation of Joint Model Formulation

Observe that in our setting the KG embedding objective function is dominated by non-event entities, i.e., pre-trained event embeddings would be continuously marginalized during training. Thus, we see the joint training objective in Eq. 2 for SSP where there is an explicit definition of the background for learning as the most natural approach for our context and adapt to the idea of simultaneous training of both objective functions.

At the same time, adaptation of Eq. 2 to our setting is a non-trivial task that requires to carefully design three ingredients: (i) an embedding model $\mathcal{L}_{\mathcal{K}}$ (ii)

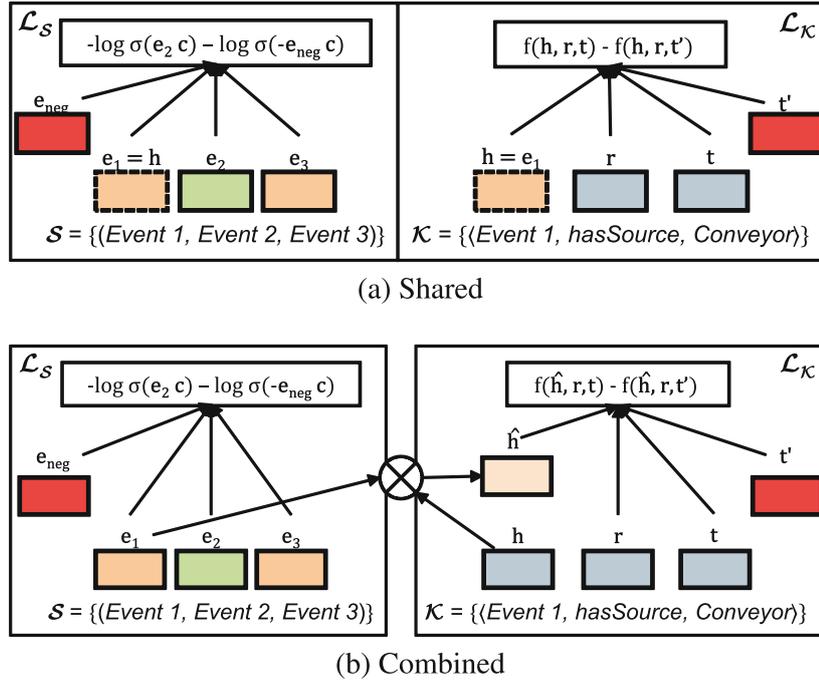


Fig. 2. Architectures: (a) shared entity embeddings (b) combination of separated entity embeddings.

an embedding model \mathcal{L}_S (iii) a way to connect the two models for simultaneous training. In our work we do not develop a new \mathcal{L}_K and exploit TransE as \mathcal{L}_K . The reason is that for our setting it is a good compromise between computational efficiency and quality of prediction.

In the remaining part of the section we start with our novel proposal on how to combine the embeddings of \mathcal{L}_K and \mathcal{L}_S , and then present our new models for event embeddings \mathcal{L}_S that we refer to as *event-enhanced knowledge learning (EKL)*.

3.4 Combining \mathcal{L}_K and \mathcal{L}_S

We now propose two architectures to combine \mathcal{L}_K and \mathcal{L}_S . The first, *shared*, is specific for our setting and different from what was used for text enhanced learning, while the second, *combined*, is inspired by TEKE.

Shared Architecture. In contrast to text-enhanced KG embeddings, in our setting both the KG and the background contain exclusively entities from \mathcal{E} . Hence, we can directly employ an architecture that uses a *single shared entity representation* for both \mathcal{L}_K and \mathcal{L}_S . In such a shared architecture one can define an identity connection between event entity embeddings \mathbf{e} and \mathbf{h}, \mathbf{t} , the entity embeddings in the KG, i.e. the event embedding matrix $\mathbf{W}_{\mathcal{X}}$ is the $|\mathcal{X}|$ -by- d submatrix of $\mathbf{W}_{\mathcal{E}}$. During training, the gradients of both objective functions are averaged and therefore simultaneously updated.

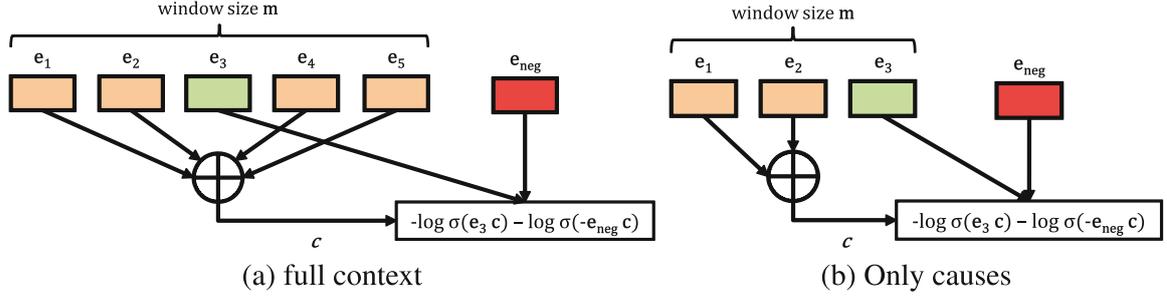


Fig. 3. Context models to predict target: (a) from causes and effects (b) from causes only

Figure 2a illustrates this approach on a simplistic example. Starting at the bottom, given an event sequence and a KG we simultaneously proceed with both objectives $\mathcal{L}_{\mathcal{K}}$ and $\mathcal{L}_{\mathcal{S}}$ as follows: The event entities of the event sequence are directed to the event embedding model using a shared entity representation. In the example the input event *Event1* uses the vector representation \mathbf{e}_1 . In parallel, head and tail entities h, t of the input triples from the KG on the right-hand side are also taken from the shared entity embedding matrix resulting in \mathbf{h} and \mathbf{t} respectively. The shared aspect is further indicated with the dashed representations stating that $\mathbf{h} = \mathbf{e}_1$. Note that the representation of the relation r in the input triple is not affected, since relation embeddings are used exclusively in the KG embedding model in $f(\mathbf{h}, \mathbf{r}, \mathbf{t})$. On top of each embedding model the calculation of the actual objective function $\mathcal{L}_{\mathcal{S}}$ and $\mathcal{L}_{\mathcal{K}}$ is carried out and combined according to \mathcal{L}_{joint} with negative event sample \mathbf{e}_{neg} and negative entity sample \mathbf{t}' .

Combined Architecture. The shared architecture proposes a very compact and efficient model with only few parameters. In correspondence to TEKE, we also propose a more flexible *combined* architecture that allows two separate representations of $\mathbf{W}_{\mathcal{X}}$ and $\mathbf{W}_{\mathcal{E}}$, however, without relying on the co-occurrence of head and tail entity. More precisely, given \mathbf{e} from $\mathbf{W}_{\mathcal{X}}$ and \mathbf{h} from $\mathbf{W}_{\mathcal{E}}$, we define $\hat{\mathbf{h}}$ with a custom combination operator denoted as $\mathbf{h} \otimes \mathbf{e}$ as follows:

$$\hat{\mathbf{h}} = \mathbf{h} \otimes \mathbf{e} := \mathbf{h} \odot \mathbf{a}_r + \mathbf{e} \odot \mathbf{b}_r,$$

where \mathbf{a}_r and \mathbf{b}_r are trainable weighting vectors for each relation r and \odot is the Hadamard product. Intuitively, this allows the model to adjust the influence of event embeddings on the KG entity embeddings specifically for each relation in a weighted average fashion. Figure 2b shows this combination leading to $\hat{\mathbf{h}}$ being fed into the triple scoring functions.

Note that both of the above proposed architectures are very general, as in fact any KG embedding model, e.g. factorization-based, can be ‘plugged’ in them.

3.5 Defining \mathcal{L}_S via Negative Sampling

As for the event embeddings, we first propose to learn event embedding parameters that are given by $\mathbf{W}_\mathcal{X}$ using the following negative sampling adapted softmax loss objective:

$$\mathcal{L}_S = \sum_{s \in \mathcal{S}} \sum_{e_i \in s} -\log \sigma(\mathbf{e}_i^\top \mathbf{c}_i) - \log \sigma(-\mathbf{e}_{\text{neg}}^\top \mathbf{c}_i), \quad (3)$$

where σ is the sigmoid function, \mathbf{e} , \mathbf{e}_{neg} are the vector representations of the target event and a negative sample event of an additional parameter matrix for the softmax prediction respectively, whereas \mathbf{c} is the representation of the *context* of the target event, i.e. some of its predecessors and successors in the event log. Intuitively, this loss is minimized when the context representations consistently have higher similarity to the actual target compared to the negative sample. The notion of context is critical for such definition of \mathcal{L}_S and we propose two models of context where we assume that the further an event appears from a given one, the lower is their dependency. We model this assumption by selecting a sliding window of size m , which stores only those events that potentially can have effect on each other.

EKL_{Full} Model. In the full model, we define the context as the target’s predecessors and successors in a sliding window. This is conceptually shown in Fig. 3a, where a sliding window contains a target event entity in the center and its neighboring events (i.e., *context*) are the center event’s possible causes and effects. The goal is to predict the target event based on the representations of its causes and effects. Following this intuition, we define the context operation for a given context target e_i in a window of size m :

$$\mathbf{c}_i = \bigoplus_{j=1}^{\lfloor \frac{m}{2} \rfloor} \mathbf{e}_{i-j} \oplus \bigoplus_{j=1}^{\lfloor \frac{m}{2} \rfloor} \mathbf{e}_{i+j},$$

where \oplus represents the vector concatenation operation. Since the resulting vector from the concatenation is of size $\mathbb{R}^{(m-1) \cdot d}$, the size of the vector that represents the target entity \mathbf{e} for classification in the softmax has to be adapted as well, i.e., $\mathbf{e} \in \mathbb{R}^{(m-1) \cdot d}$. Note that the actual entity representations preserve their original size in the d -dimensional space, hence the only additional parameters are needed for the prediction of each \mathbf{e} . Therefore, we can still ensure the efficient training of event sequence embeddings. The window size m is added as an additional hyper-parameter.

EKL_{Cause} Model. Further we also address the case where only the causing events may have influence on the target. In order to preserve the predictive information that a sequence of events inherently possesses, we propose to concatenate the representation of the $m - 1$ event predecessors to predict a given target event, i.e. the m -th event in the window. Formally this can be denoted as

$\mathbf{c}_i = \bigoplus_{j=1}^{m-1} \mathbf{e}_{i-j}$. In Fig. 3b we illustrate this idea again for a generic sequence window of event entities. Observe that the target event here is always the last one in the window.

Note that negative sampling in EKL_{Full} and EKL_{Cause} should be done with care. In order to avoid an accidental inclusion of dependent events as negatives, we make sure that the negative sample is always taken from outside of the complete sequence, i.e. after a certain time threshold before and after the target event.

3.6 Defining \mathcal{L}_S via Autoencoders: EKL_{Auto} Model

Sequence modeling is usually closely related to Recurrent Neural Networks (RNNs). Based on previous experiments it was known that RNNs do not yield good representations of the events for our datasets. Another natural way to learn representations of event sequences is to resort to the family of autoencoder models. The goal here is to encode the sequence s to a latent representation and then to try decoding this back, hence, the latent encoding needs to conserve the sequential information within a low-dimensional vector. Formally, our event embedding objective is the mean-squared error of the original sequence and its decoding:

$$\mathcal{L}_S = \sum_{s \in \mathcal{S}} (\mathbf{s} - \phi(\omega(\mathbf{s})))^2, \quad (4)$$

where $\mathbf{s} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m]$ is the stacked vector representation of s and $\phi \circ \omega$ is the encoder-decoder function chain. In this work we use a convolutional autoencoder [13] with different filter sizes (adapted to the window size) on the stacked vectors of the sequence, i.e. for one filter \mathbf{W}_f we have: $\omega(\mathbf{s}) = \sigma(\mathbf{s} \star \mathbf{W}_f + b)$ and $\phi(\mathbf{h}) = \sigma(\mathbf{h} \star \tilde{\mathbf{W}}_f + c)$, where \star is the convolution operator, σ is the sigmoid activation function, $\tilde{\mathbf{W}}_f$ is the flipped filter matrix, and b and c are bias terms.

4 Evaluation

We have implemented both of our architectures for event-enhanced KG completion into a system prototype² employing the TransE model with the original max-margin objective and negative sampling techniques in TensorFlow [1]. In contrast to the original implementation provided by the authors, our TensorFlow models use a more efficient training technique from [15] exploiting the AdaGrad variant of stochastic gradient-descent [7], which has been shown to yield good quality of prediction for other relational learning models, due to its frequency-adaptive weighting of updates [17].

The AdaGrad technique is beneficial for parameter updates of entities that are only sparsely connected in the KG, and on the other hand, it prevents overfitting for densely connected entities. As suggested for TransE [4], all embeddings

² Source code of EKL: <http://github.com/NetherNova/event-kge>.

Table 1. Characteristics of two domain-specific datasets

Dataset	$ \mathcal{K} $	$ \mathcal{S} $	$ \mathcal{E} $	$ \mathcal{R} $
Manufacturing	6,791	56,000	3,180 (728)	10
Traffic	11,000	157,000	13,113 (4,000)	5

were initialized by sampling from a uniform distribution $U[-\frac{6}{\sqrt{d}}, \frac{6}{\sqrt{d}}]$. In terms of negative sampling we employ the random replacement of head or tail entities relying on the closed-world assumption and the Bernoulli sampling proposed in [26].

4.1 Evaluation Protocol

We evaluated our three novel approaches for event representations, i.e., EKL_{Auto} , EKL_{Full} and EKL_{Cause} , together with the two architectures (shared and combined) for KG completion by comparing them to plain TransE and the state-of-the-art TEKE model, precisely the TransE-based TEKE_E version, as a baseline for incorporating events, which are in the TEKE case treated as common text corpus and pre-trained using the word2vec skipgram model [14]. In each of the experiments, the original KG is split into a training (70% of the original KG), validation (10%) and test (20%) sets. Final results on quality of prediction are calculated based on the test set, for which we report two commonly-used evaluation metrics:

- **Mean Rank:** the average rank of the entities (head and tail) that would have been the correct ones;
- **Hits@N:** the portion of ranks within N highest ranked entities for $N \in \{1, 3, 10\}$.

The mean rank in our experiments corresponds to the *filtered* version that has been originally proposed in [4], i.e. in the test set when ranking a particular triple $\langle h, r, t \rangle$, all $\langle h, r, t' \rangle$ triples with $t \neq t'$ are removed. Employing grid search through the hyper-parameters we determine the best configuration by mean rank on the validation set with early stopping over a maximum of 100 epochs.

4.2 Dataset Descriptions

Our experiments were performed on two real-world datasets enriched with event sequences: *manufacturing* and *traffic*. The statistical details of these datasets are presented in Table 1, where we report the total number of triples $|\mathcal{K}|$, the number of sequences $|\mathcal{S}|$, the total number of entities $|\mathcal{E}|$ with the number of event entities stated in brackets and the number of relations $|\mathcal{R}|$. We made both datasets and the corresponding sequences available online, see the Github link above.

Manufacturing Dataset. The first dataset is an excerpt of a real-world manufacturing KG from an automated factory that stores data about production equipment, product-part descriptions, and production processes. It models several automated production lines and contains entities corresponding to equipments, products, material, and processes connected via different domain-specific relations, e.g. *connectedTo*, *madeOf*, *follows*. The events are messages collected from a subset of the production machines, i.e. machine event logs. These are mostly alarms and warnings reporting about critical states of the production process, e.g. alarms about jams at the material intake of a machine. Some event sequences were known not to influence each other; these bring noise to the embeddings. To avoid this situation, we pre-processed the raw sequences of events by splitting them into multiple disjoint ones based on a maximum time gap that was given by domain experts. This does not bias the embeddings to any of the models, since it just removes spurious correlations. Then, we set the following parameters: mini-batch size to 32 samples; $d \in \{40, 60, 80\}$ as embedding size; and $\eta \in \{0.01, 0.05, 0.1\}$ as learning rate. For the event embeddings, we set the context window size $m \in \{2, 3, 4, 5\}$ for EKL_{Cause} and EKL_{Auto} , $m \in \{3, 5, 7, 9\}$ for EKL_{Full} and $\alpha \in \{0.1, 0.5, 1.0\}$. The number of negative samples for all event embedding models was empirically set to 8.

Traffic Dataset. Here we took a fragment of the *CityPulse* data collection³ that was used in the smart city applications [2] for monitoring traffic with sensors placed on several locations in the area of Aarhus in Denmark. From this dataset we engineered a KG consisting of the sensor locations, streets, routes, and point of interest locations with typing information crawled from the Google places API⁴. The event data is based on the observed vehicle counts for different routes, e.g. *IncreasedTrafficEvent* between two streets. Since connected streets as well as similar localities (e.g., schools) should intuitively exhibit a similar traffic pattern, the events may be used to complete the data about street connections and locality information. This dataset is particularly challenging and interesting, as the number of entities is *higher* than the number of overall facts, witnessing the KG sparsity. To cope with the large number of entities and triples in the KG, we set the mini-batch size to 64 samples and the embedding size d from $\{60, 80, 100\}$, while keeping the rest of the hyper-parameters the same as in the previous scenario.

4.3 Evaluation Results

Overall KG Completion. In Table 2 we report the results for the variations (1) and (2) of the event-enhanced KG completion problem from Sect. 3.1. As expected, our EKL models significantly outperform TEKE and TransE in both settings. We report that EKL_{Cause} in the shared architecture shows the best

³ iot.ee.surrey.ac.uk:8080/datasets.html.

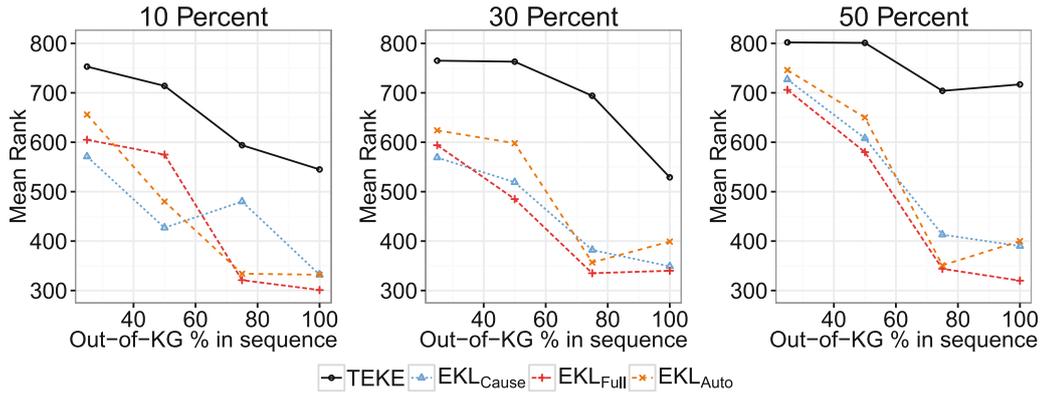
⁴ developers.google.com/maps/documentation/javascript/places.

Table 2. KG completion results for EKL and baselines, where m/n denotes completion results for shared/combined architecture.

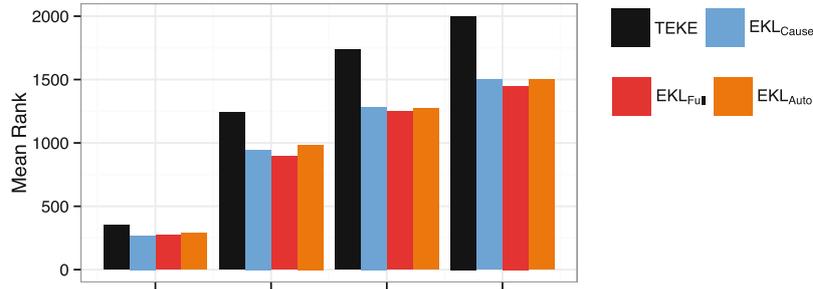
Model	Mean rank	Hits@10	Hits@3	Hits@1
Dataset: Manufacturing				
TransE	317	36.1	23.2	7.5
TEKE_E	596	24.5	10.8	3.6
EKL _{Full}	285/663	37.9/23.5	25.0/12.3	8.0/4.8
EKL _{Cause}	280 /691	38.1 /21.4	25.8 /11.5	7.4/5.2
EKL _{Auto}	302/692	34.5/22.5	23.6/10.1	9.6 /2.7
Dataset: Traffic				
TransE	4126	26.8	24.6	9.5
TEKE_E	897	25.3	22.6	18.9
EKL _{Full}	1118/ 758	27.0/27.3	25.3 /24.5	21.1/20.6
EKL _{Cause}	999/783	27.2/27.0	24.7/24.4	20.0/20.5
EKL _{Auto}	944/840	27.5/ 27.7	24.8/24.8	22.2 /20.6

results in terms of mean rank and the first two hits metrics in the manufacturing case. The shared entity embeddings are highly beneficial for the other EKL models as well and show significant improvements compared to the TEKE_E model. In this case, TEKE_E performs even worse than default TransE, confirming that mere co-occurrence between events does not contribute to the completion. However, in the traffic scenario TEKE_E shows much stronger results compared to TransE, but using our combined architecture achieves consistently better results, as in terms of mean rank EKL_{Full} outperforms the rest. On the other hand, the EKL_{Auto} model has highest hits@1 for both datasets using the shared embeddings, therefore it is the most specific model, but cannot generalize as well as EKL_{Full} and EKL_{Cause}.

Impact on Relations. Table 3 (top right and left) contains the mean rank for every KG relation achieved by the best EKL and TEKE_E model on the manufacturing and traffic data. For the manufacturing scenario the relations that are semantically closer to events benefit from the sequential embeddings more than others and we expected this effect. E.g., the improvement for the *connectedTo* relation that links equipments to each other is more evident than for other relations like materials partonomy *isPartOf*. The additional knowledge given by the sequences also propagates to the process-oriented *follows* relation, for which the significant improvement over TEKE is observed. Similar conclusions can be made about the traffic scenario. Here, again the EKL_{Cause} model performs exceptionally well on the *hasStartPoint* relation, while for less event-dependent relations such as *locatedAt* the difference of EKL compared to TEKE is less apparent.



(a) Mean rank on zero shot test sets with increasing portion of out-of-KG entities



(b) Mean rank with increasing branching factor

Fig. 4. Meanrank evaluation: (a) zero shot test sets, (b) with increasing branching factor

Further evaluation is focused on the manufacturing scenario, since this dataset has richer semantics in terms of relations and typing of event entities.

Impact of Window Size. In Table 3 (bottom, left) we examine the impact of the window size on the overall mean rank performance of our models. One can observe that capturing the sequential nature of the event entities is sensitive to the window size parameter. In our manufacturing scenario the EKL_{Cause} model performs very well for small window sizes, and the results deteriorate after the window size of 4. In the preparation of EKL_{Full} the window size must always be an odd number, therefore the window size here is increased by two. It can be observed that EKL_{Full} needs a slightly larger window to capture the context, and shows best performance at a window size of 7. In case of EKL_{Auto} we see a deterioration after window size 3.

Zero-Shot Learning. The zero-shot learning (variation (3) of the event-enhanced KG completion problem in Sect. 3.1) addresses the case when triples about event entities present in the test set are not in the training KG, hence their links to known entities in the KG can only be inferred through their sequential occurrence.

Table 3. Evaluation results and controlled experiment statistics

Traffic Rel			TEKE_E	EKL _{Full}	Manufacturing Rel			TEKE_E	EKL _{Cause}
<i>hasSource</i>			868	837	<i>connectedTo</i>			674.0	28.5
<i>hasStartingPoint</i>			970	350	<i>type</i>			807.9	163.9
<i>hasEndPoint</i>			1,627	1,104	<i>follows</i>			16.1	7.8
<i>locatedAt</i>			214	291	<i>isMadeOf</i>			29.5	12.5
<i>type</i>			788	829	<i>hasSource</i>			289.7	169.9
					<i>involvedEquipment</i>			37.15	17.75
Window	EKL _{Full}	EKL _{Cause}	EKL _{Auto}	<i>hasComponent</i>			234.1	47.3	
2	-	280	361	<i>isPartOf</i>			15.4	64.8	
3	293	322	302	<i>observedBy</i>			525.8	769.9	
4	-	306	328	<i>hasSkill</i>			22.2	14.4	
5	301	356	337						
7	285	-	-	BF	 K 	 E 			
9	287	-	-	2	3,459	2,005 (549)			
				3	11,925	7,102 (2,180)			
				4	16,578	9,097 (2,155)			
				5	24,835	15,252 (4,809)			

To evaluate the effectiveness of our EKL models in a zero-shot learning setting, we have accordingly designed tailored KG test sets, by varying the percentage of the *out-of-KG* event entities in the test set (10%, 30%, and 50% of the overall set of event entity triples). Furthermore, we also vary the percentage of out-of-KG event entities in the event log, where 100% indicates that every out-of-KG entity of the test set has been observed *at least once* in one of the sequences. The results of our experiments are reported in Fig. 4a. Note that in the setting on the right, when 50% of all event entities are solely present in the test set, the EKL_{Full} model consistently outperforms all other approaches w.r.t. mean rank. In other settings, as the sequence dataset proportion is increased, EKL_{Full} shows best improvement and ends up with the lowest mean rank eventually.

It appears that EKL_{Full} is more stable at capturing typing and location of events, due to its incorporation of future context, compared to EKL_{Cause} and EKL_{Auto}. On the other hand, all EKL models significantly outperform TEKE in all zero-shot settings and seem to converge with less data. We argue that this is due to their ability to capture sequential correlations inside the event logs and the joint optimization.

Controlled Topology of Processes. In our real-world manufacturing scenario the process entities reflect the *topology* of physical equipment in the factory. Since our experiments witness that EKL does the best predictions for relations that reflect this topology, we designed a controlled scenario where we can validate how the changes in topology affect quality of prediction.

To this end we chose six relations that naturally determine manufacturing topology: *follows*, *isA*, *hasSource*, *involvedEquipment*, *hasComponent*, and *connectsTo*. We scaled the topology in two dimensions: *structure of the processes* and *number of events*. This gave us four KGs, each describing a complex tree-shaped manufacturing process, where one concrete piece of manufacturing equipment is attached to each node of the tree and multiple events are attached to each

piece of equipment. These KGs are described in Table 3 (bottom right), where **BF** stands for branching factor.

We now describe the concrete procedure that we followed to generate these KGs. First, we set the branching factor of the process varying from 2 to 5 and the depth of the process. The intuition behind the branching factor is that, starting from a root process step, the successor steps can be partly executed in parallel and the branching controls this degree of parallelism, which is a common characteristic of real-world processes in manufacturing, road traffic, etc. E.g., in a manufacturing case a particular preparation process may have a fixed set of three successor process steps (branching factor three) each performing a different operation in parallel to the others. Second, for each process we iteratively constructed the process-tree starting from the root, until the process depth, by randomly selecting the number of children in each node to be at most the branching factor. Using the process-tree, we generate corresponding equipment entities participating in the process. Then, using multiple random walks through the process-tree (with restart) we simulated 50,000 ($|\mathcal{S}| = 50,000$) event occurrences that are linked to the equipment of the process. Each random walk follows successor process entities from the root to the end with uniform probability for all successors and a small probability of staying in the current process. Hence, we end up with multiple cause and effect event patterns in the sequence data instead of having a purely linear chaining. Note that the relative amount of event entities to overall entities in the KG stays roughly at 30%.

Again, we compared our three event-enhanced KG embedding models to TEKE_E. The results are presented in Fig. 4b. Observe that, as the branching increases, the more all EKL models outperform TEKE_E, since the alignment of process and machine entities no longer follows a linear sequence, which is hard to capture without considering sequential correlations. In general, EKL_{Full} seems to be the most effective event embedding model in terms of adapting to the non-linear process chains, while we conjecture that EKL_{Auto} might be more prone to overfit to linear sequences.

5 Conclusions

We proposed EKL, a novel method for event-enhanced learning of knowledge graph embeddings by jointly modeling representations of KGs and event logs consisting of sequences of event entities. Our approach has many applications across domains such as manufacturing, smart cities, and social networks. More specifically, we proposed two different architectures, using a single shared entity embedding layer and another one using combined embeddings for joint optimization. Furthermore, we presented several event embedding models with various notions of context concatenation and an event sequence Autoencoder model. Evaluation on two real-world scenarios and a controlled experiment showed the effectiveness of our approach compared to the state-of-the-art TEKE model. Especially process-oriented parts of KGs exhibit significantly improved completion performance when provided with event embeddings. Our EKL models are

also capable of zero-shot learning, in which event entities are not linked to the KG. The scaled zero-shot experiments showed that EKL models significantly improve handling of zero-shot event entities in the KG completion.

Acknowledgements. This work was partially supported by the EPSRC projects DBOnto, MaSI³ and ED³.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., et al.: TensorFlow : a system for large-scale machine learning. In: OSDI (2016)
2. Ali, M.I., Gao, F., Mileo, A.: CityBench: a configurable benchmark to evaluate RSP engines using smart city datasets. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 374–389. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25010-6_25
3. Bordes, A., Chopra, S., Weston, J.: Question answering with subgraph embeddings. In: EMNLP, pp. 615–620 (2014)
4. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
5. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: EMNLP-CoNLL, pp. 708–716 (2007)
6. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: ACM SIGKDD, pp. 601–610 (2014)
7. Duchi, J.C., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **12**, 2121–2159 (2011)
8. Hachey, B., Radford, W., Nothman, J., Honnibal, M., Curran, J.R.: Evaluating entity linking with Wikipedia. *Artif. Intell.* **194**, 130–150 (2013)
9. Kharlamov, E., Hovland, D., Skjæveland, M.G., Bilidas, D., Jiménez-Ruiz, E., Xiao, G., Soylu, A., Lanti, D., Rezk, M., Zheleznyakov, D., Giese, M., Lie, H., Ioannidis, Y.E., Kotidis, Y., Koubarakis, M., Waaler, A.: Ontology based data access in Statoil. *JWS* **44**, 3–36 (2017)
10. Kharlamov, E., Mailis, T., Mehdi, G., Neuenstadt, C., Özçep, Ö.L., Roshchin, M., Solomakhina, N., Soylu, A., Svingos, C., Brandt, S., Giese, M., Ioannidis, Y.E., Lamparter, S., Möller, R., Kotidis, Y., Waaler, A.: Semantic access to streaming and static data at Siemens. *JWS* **44**, 54–74 (2017)
11. Krompaß, D., Baier, S., Tresp, V.: Type-constrained representation learning in knowledge graphs. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9366, pp. 640–655. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25007-6_37
12. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: Dbpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Web* **6**(2), 167–195 (2015)
13. Masci, J., Meier, U., Cireşan, D., Schmidhuber, J.: Stacked convolutional auto-encoders for hierarchical feature extraction. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) ICANN 2011. LNCS, vol. 6791, pp. 52–59. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21735-7_7
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 1–9 (2013)

15. Minervini, P., Fanizzi, N., D'Amato, C., Esposito, F.: Scalable learning of entity and predicate embeddings for knowledge graph completion. In: ICMLA, pp. 162–167 (2015)
16. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. *Proc. IEEE* **104**(1), 11–33 (2016)
17. Nickel, M., Rosasco, L., Poggio, T.A.: Holographic embeddings of knowledge graphs. In: AAAI, pp. 1955–1961 (2016)
18. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: ICML (2011)
19. Ringsquandl, M., Lamparter, S., Brandt, S., Hubauer, T., Lepratti, R.: Semantic-guided feature selection for industrial automation systems. In: Arenas, M., et al. (eds.) ISWC 2015. LNCS, vol. 9367, pp. 225–240. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25010-6_13
20. Ringsquandl, M., Lamparter, S., Kharlamov, E., Lepratti, R., Stepanova, D., Kroegeer, P., Horrocks, I.: On event-driven learning of knowledge in smart factories: the case of siemens. In: *IEEE Big Data* (2017)
21. Santos, H., Dantas, V., Furtado, V., Pinheiro, P., McGuinness, D.L.: From data to city indicators: a knowledge graph for supporting automatic generation of dashboards. In: Blomqvist, E., Maynard, D., Gangemi, A., Hoekstra, R., Hitzler, P., Hartig, O. (eds.) ESWC 2017. LNCS, vol. 10250, pp. 94–108. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58451-5_7
22. Shi, B., Weninger, T.: ProjE : embedding projection for knowledge graph completion. In: AAAI 2017, pp. 1–14 (2017)
23. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: NIPS, pp. 1–10 (2013)
24. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge. In: *Proceedings of WWW*, pp. 697–706 (2007)
25. Wang, Q., Wang, B., Guo, L.: Knowledge base completion using embeddings and rules. In: IJCAI, pp. 1859–1866 (2015)
26. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI, pp. 1112–1119 (2014)
27. Wang, Z., Li, J.Z.J.: Text-enhanced representation learning for knowledge graph. In: IJCAI, pp. 1293–1299 (2016)
28. Xiao, H., Huang, M., Meng, L., Zhu, X.: SSP: semantic space projection for knowledge graph embedding with text descriptions. In: AAAI, pp. 1–10 (2017)
29. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of knowledge graphs with entity descriptions. In: IJCAI, pp. 2659–2665 (2016)
30. Yang, Z., Tang, J., Cohen, W.W.: Multi-modal bayesian embeddings for learning social knowledge graphs. In: IJCAI, pp. 2287–2293 (2016)
31. Zhong, H., Zhang, J., Wang, Z., Wan, H., Chen, Z.: Aligning knowledge and text embeddings by entity descriptions. In: EMNLP, pp. 267–272 (2015)