Scaling ML Analytics with Knowledge Graphs: A Bosch Welding Case

Baifan Zhou
baifanz@ifi.uio.no
SIRIUS Centre, University of Oslo
Oslo, Norway

Dongzhuoran Zhou
dongzhuoran.zhou@de.bosch.com
Bosch Center for AI, Germany
SIRIUS Centre, University of Oslo, NO

Jieying Chen
jieyingc@ifi.uio.no
SIRIUS Centre, University of Oslo
Oslo, Norway

Yulia Svetachova
yulia.s@causaly.com
Causaly
London, UK

Gong Cheng
gcheng@nju.edu.cn
State Key Laboratory for Novel Software Technology, Nanjing University, China

Evgeny Kharlamov
evgeny.kharlamov@de.bosch.com
Bosch Center for AI, Germany
SIRIUS Centre, University of Oslo

ABSTRACT

Automated welding is heavily used in automotive industry to produce car bodies by connecting metal parts with welding spots. Modern welding solutions and manufacturing environments produce high volume of heterogeneous data. Analytics of these data with machine learning (ML) can help to ensure high quality of welding operations. However, due to heterogeneity of data and application scenarios, scaling such ML-based analytics is challenging. We address this challenge by relying on knowledge graphs (KG) that not only conveniently allow to integrate welding data, but also to serve as the bases for layering ML-based analytical applications, thus enabling quality monitoring of welding operations. In this work we focus on construction of a KG for welding that is tailored towards further use for ML applications. Furthermore, we demonstrate how selected ML analytical tasks are supported by this KG.

KEYWORDS

knowledge graph, quality monitoring, manufacturing, machine learning, industrial application, analytics

1 INTRODUCTION

Industry 4.0 [14] and technologies of the Internet of Things (IoT) [12] behind it lead to unprecedented growth of data generated during manufacturing processes [3, 26]. Indeed, modern manufacturing machines and production lines are equipped with sensors that constantly collect and send data and with control units that monitor and process these data, coordinate machines and manufacturing environment and send messages, notifications, requests. Availability of these voluminous data has led to a large growth of interest in applying Machine Learning (ML) approaches for *monitoring* manufacturing processes, machines, and products, e.g., by predicting machines' down-times or the quality of manufactured products [27].

Consider an example of *welding quality monitoring* at Bosch, where welding is performed with machines as shown in Fig. 1 to connect pieces of metal together by pressing them and passing high current electricity through them [4]. The high current generates a huge amount of heat due to resistance in the small area between the two welding electrodes. The metal materials in that area will melt and congeal after cooling down, creating a welding spot that effectively connects the two metal worksheets. Hence, this type of welding is named as resistance spot welding (RSW).

In automotive industry, such welding is essential for producing high-quality cars, where the worksheets are car body parts in the car factories. Indeed, RSW processes are fully automated, introduce up to 6000 spots [29] in each car, and each spot comes with thousands of sensor measurements, welding configurations, status, quality indicators, etc. resulting in millions of data records generated by RSW only from one car. The quality failure of a single spot can halt an entire car production line, which means the loss of several cars, production down-time, and cost to bring the production line back to running. Thinking about the number of cars produced everyday, it reveals the huge economic benefit behind improving quality monitoring of RSW. Effective monitoring and quality control of welding spots thus essentially impacts production efficiency and cost. Furthermore, if the technology developed for improving RSW can be generalised over a large amount of data and applications, the industrial impact behind the research endeavour to improve RSW will be tremendous.

However, it is very difficult to monitor the welding quality reliably and in a scalable way. The common practice is to tear the welded car body apart and measure the spot diameters [5, 11], which is extremely expensive and time-consuming. Bosch's data are from hundreds of Bosch plants worldwide and many Bosch's renowned customers. This huge amount of highly heterogeneous data come from production lines in real-time, and are collected and stored with highly diversified sensors settings, formats, databases, software versions, customer individualisation, etc. Furthermore, the ML solutions are typically tailored to the datasets and process where the data scientists have gained a sufficiently deep understanding of the welding process and data, after time-consuming communication between data scientists and welding experts. Considering that data scientists usually have a background distinct from the welding experts, if they need to develop ML solutions to other processes or datasets, the same time-consuming communication needs to repeat again, which makes it difficult to scale the developed ML solutions to other datasets and processes. Therefore, it is demanding to: (1) manage heterogeneous data from a large variety of sources with a unified mechanism; (2) scale and reuse developed ML solutions across different datasets.

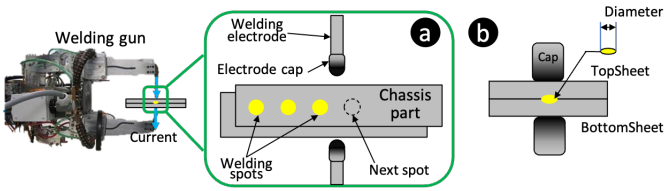

Figure 1: Schematic illustration of the Resistance Spot Welding (RSW) process. (a) The machine produces welding spots continuously on a chassis (car body) part by passing a high current through the car body part. (b) The welding worksheets are a sheet combination, including a top sheet and a bottom sheet (sometimes also a middle sheet), between which a welding spot is generated to connect the two metal worksheets.

In this work we address these challenges by relying on knowledge graphs (KG) that provide an efficient foundation for quality monitoring applications like machine learning analysis. In particular, we present an *application* and introduce our system for generating the Welding-ML KG in a semi-automated fashion. Our KG solution can unify heterogeneous data into Welding-ML KG, which disentangles the generality of ML solutions from the data specificities, easing data retrieval and reuse of ML solutions, thus making ML solutions scalable across datasets and processes.

The system consists of a set of semantic artefacts (including core ontology, domain ontologies, ML ontology, and mappings), semantic modules (including a mapping reasoner/annotator, a data integration module, and a KG generation module), and an ML analysis module. The core ontology is a upper level ontology that encodes the general knowledge of manufacturing process. The domain ontology *rsw* reflects specific knowledge of the RSW welding processes, and follows patterns in core ontology. We summarise the complex domain ontology *rsw* into a smaller domain ontology *rsw-kg*, which reflects more specifities in particular datasets that the KG should be constructed from.

The data to domain ontology mapping (Data-to-DO Mapping) is created by the users, who annotate raw welding data with terms from the domain ontology *rsw*. The mapping is used by the data integration module to transform heterogeneous welding raw data into uniform data formats, e.g. csv tables. From the Data-to-DO Mapping, a Data-to-ML mapping is automatically reasoned. The latter one combined with an ML ontology is used for generation of the Welding-ML KG from uniform data formats (e.g. csv tables). The Welding-ML KG provides a representation that considers the perspective of ML analysis, thus making the data suitable for data analysis/ML analysis. This representation includes entities of different feature groups, which are syntactically and semantically well defined abstract representation of data. The Welding-ML KG is used by the ML analysis module to generate ML results.

To summarise, our contributions are:

- We present an industrial application of KG generation for ML analytics in welding quality monitoring. The application is deployed for industrial scenarios and uses data collected from welding production plants.
- We propose a practical system architecture for our KG solution for ML analytics in automated welding, in which semantic technological components like semantic artefacts (core ontology, domain ontology, ML ontology) and reasoning are adequately organised to achieve the application.
- We introduce a novel concept of KG-based data management with specialised support for ML analytics, and provide proof-of-concept examples of ML pipelines for quality monitoring.

This paper is organised as follows. Section 2 introduces the use case of ML-based welding quality monitoring, and derives the requirements from the use case and challenges. Section 3 presents our solution of KG generation. Section 4 describes the ML analytics application with three ML tasks and two example pipelines that our KG-solution are deployed on. Section 5 briefly discusses some related works. Section 6 concludes the paper, summarises lessons learned and previews future directions.

2 USE CASE: BOSCH WELDING MONITORING

This section gives an introduction to our use case of ML-based welding quality monitoring at Bosch.

2.1 The Resistance Spot Welding (RSW) Process

Resistance Spot Welding at Bosch is a type of fully automated manufacturing process widely applied in automotive industry. We illustrate RSW with Figure 1, in which the two electrode caps of the welding gun press two or three worksheets between the electrodes with force. A high electric current then flows from one electrode, through the worksheets, to the other electrode, generating a substantial amount of heat as a result of electric resistance. The material in a small area between the worksheets will melt, and form a welding nugget connecting the worksheets, known as the welding spot. The quality of welding operations is typically quantified by quality indicators like spot diameters, as prescribed in international and German standards [5, 11]. To obtain the spot diameters or tensile shear strength precisely, the common practice is to tear the welded car body apart and measure these two quality indicators [5], which essentially destroys the welded cars and is extremely expensive. Nevertheless, the expensive practice to repeated to ensure the welded spots have good quality, because the quality of each welding spot has a great importance. Consider a scenario in the car factory, where cars are continuously produced in several production lines. RSW production lines usually have a sequential structure, where multiple types of car body parts go through a sequence of welding machines. Each machine performs welding operation for a number of welding spots on each car body part in a fixed order, and each car body part has with a large number of welding spots (up to 6000 [29]). If a quality failure happens on one single spot, the welding machine that works on that spot needs to stop, and the entire production lines need to stop until the quality failure is resolved and necessary maintenance measure is undertaken. This causes a huge amount of loss in time and cost.

2.2 Analytic Tasks for Quality Monitoring

Since Bosch RSW solutions are fully automated and produce large volumes of heterogeneous data, we rely on ML approaches [30] for quality monitoring for RSW. ML approaches have proven their great potential for quality monitoring and thus they have received an increasing attention in industry [27]. The reasons are that ML allows to predict the quality by relying on statistical theory in

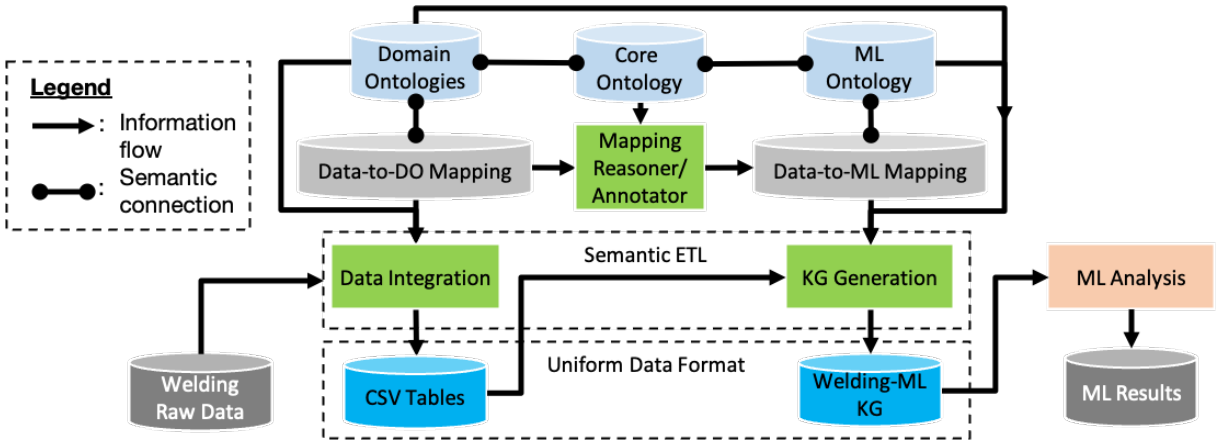

Figure 2: An architectural overview of our KG solution. Semantic connection means that they share some common classes or some of their classes are connected by properties.

building mathematical models, thus enabling computers to make inference from data without being explicitly programmed [1, 21]. ML analysis is an important practice to generate data-driven models or insights for quality monitoring in RSW. Informally, ML analysis is to mathematically transform the representation of input data and then to model on the input data to predict output data with statistic (e.g. linear regression) or biologically inspired (e.g. neural networks) methods.

Our quality monitoring tasks can be categorised into three groups:

(1) Quality estimation, where the quality of a finished welding operation needs to be reliably estimated. For example, the welding electrode has just finished three welding spots in Figure 1, and the quality of the finished three spots need to be assessed by e.g. spot diameters.
(2) Quality prediction, where the quality of a welding operation that has not happened needs to be predicted. For example, the quality of the next spot (the fourth spot) in Figure 1 needs to be predicted.
(3) Feature importance evaluation, where the welding experts would like to understand what features in the datasets are the most important factors influencing the welding quality. This task is essential for transparent ML analytics and explainable AI, extremely desired in industrial applications.

2.3 Our Semantics-Enhanced ML Workflow

For the use case, we have been working on enhancing ML analysis [23, 30] with an ML workflow consisting of six iterative steps: (1) data collection, (2) task negotiation, (3) data integration, (4) ML modelling, (5) ML interpretation, (6) ML deployment. For the task negotiation, we developed a core ontology and some domain ontologies[1] to enable a common understanding basis in Step 1 and Step 5 for the users from distinct knowledge background, including welding experts, measurement experts, data scientists, data managers, managers, etc. For data preparation, the users annotate the collected data with domain ontology terms so that data from different sources can be integrated into uniform data formats. The ML modelling was enhanced by an ML ontology and a set of ML solutions in the form of ML pipelines, which ease the construction of ML solutions and their explainability.

[1]The domain ontologies are created in a semi-automated fashion based on *core* and ontology templates [22]. The exact way it is created is beyond the scope of this paper. Interested readers can refer to [23].

2.4 Requirements for the KG Solution

Based on the use case, tasks and challenges, we derive the requirements for our KG solution as follows.

- *R1 Completeness.* The knowledge graphs should be able to completely represent all different raw datasets, namely that the generated KG should cover all attributes in datasets from all sources..
- *R2 Uniform data access.* The knowledge graphs should integrate all datasets into a uniform data format, and renaming the attributed names to unified property names.
- *R3 User-friendliness.* The KG-schema should not be over-complicated for the users to write queries. It should be easy to understand and use. The generated KG based on the schema should not have too many blank nodes, ideally zero blank node.
- *R4 ML analytics support.* The knowledge graphs should support ML analytics, for example, easing data retrieval or reuse of ML analytic pipelines.

3 OUR SOLUTION: KG GENERATION FOR ML ANALYTICS

Now we present our solution of KG generation for ML analytics.

3.1 Overview

Our solution transforms the Welding Raw Data into Welding-ML KG via Semantic ETL (extract-transform-load). The system consists of several semantic artefacts and modules. The Welding-ML KG can support ML analysis, easing the data retrieval and reuse of ML pipelines. We now walk the readers through the system with the architectural overview illustrated in Figure 2.

We start with the bottom left, where data are collected constantly from the welding machines in production. These data are called welding raw data and depicted in a grey barrel. The welding raw data are highly heterogeneous data. They are stored in various data formats: SQL database, text files, csv files, Bosch rui files,

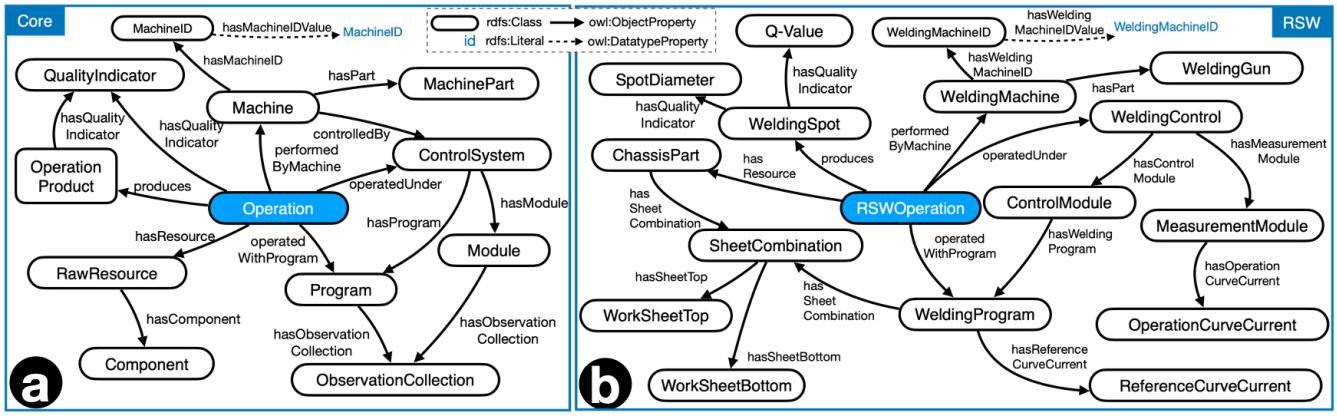

Figure 3: Schematic illustration of part of the Core ontology O_{core} (a) and the domain ontology, Resistance Spot Welding (RSW) ontology O_{rsw} (b). Important classes are coloured with blue for readers' convenience.

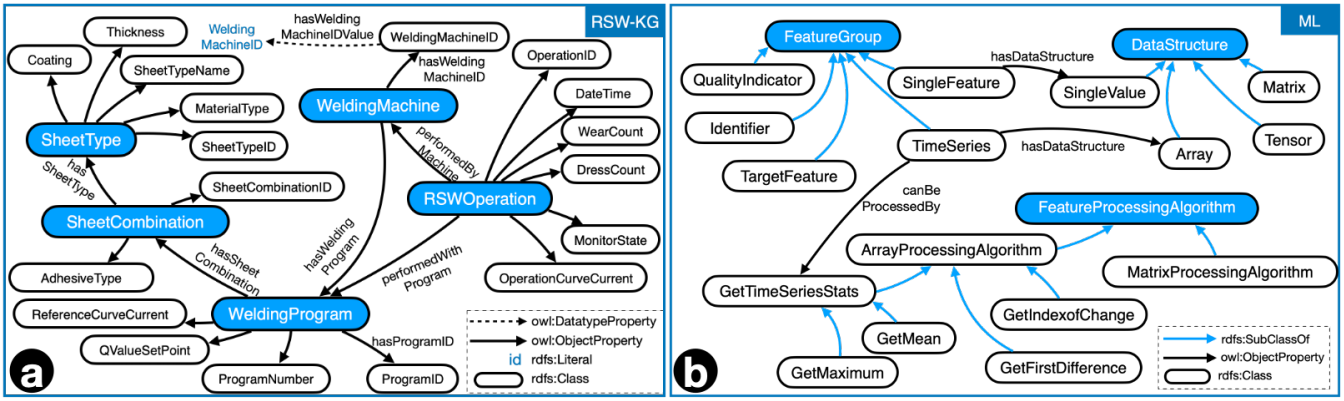

Figure 4: Schematic illustration of part of the KG ontology O_{rsw-kg} (a) and the ML ontology O_{ml} (b). Object properties and datatype properties are partially shown for simplicity. Important classes are coloured with blue for readers' convenience.

json, etc. They go through the data integration module and are transformed into uniform data formats, which are a series of data formats with well-defined structures such as csv, json, KGs, etc. The data integration takes two semantic artefacts as inputs, the domain ontologies and the Data-to-DO Mapping.

The data integration module is part of the semantic ETL process, which includes another module, the KG generation module. The latter one takes the uniform data formats, the domain ontologies, the ML ontology and Data-to-ML Mapping as inputs, and generates the Welding-ML KG. The ML analysis module takes the Welding-ML KG as inputs and executes ML pipelines to deliver ML results.

In the system, the semantic artefacts include the core ontology O_{core}, two domain ontologies O_{rsw} and *rsw-kg*, the ML ontology *ml* and two mappings. The O_{core} is an upper level ontology for manufacturing processes. The O_{rsw} reflects the domain understanding of the RSW process, and all classes in O_{rsw} are sub-classes of classes in O_{core}. It is used by the welding experts users to annotate the welding raw data, to create the Data-to-DO Mapping. This mapping goes through the Mapping Reasoner/Annotator, which automatically generates the Data-to-ML Mapping through reasoning. The latter mapping connects data with classes in *ml* via annotations. The Mapping Reasoner/Annotator also allows the users to manually modify the annotations. The *rsw-kg* is a smaller domain ontology that reflects specifities of datasets. The *ml* is an application ontology that encodes general knowledge of ML. Between the domain ontologies, the core ontology and the ML ontology there exist semantic connections, namely that they share some classes or their classes can be connected by properties, which will be explained in detail in Section 3.2.

3.2 Semantic Artefacts and Mapping Reasoner

Ontologies. The three types of ontologies are O_{core}, the domain ontologies (e.g. O_{rsw} and O_{rsw-kg}) and the ML ontology O_{ml}. The core ontology is an OWL 2 ontology and can be expressed in Description Logics $\mathcal{SHI}(\mathcal{D})$. With its 1741 axioms, which contain 221 classes, 197 object properties, and 114 datatype properties, it models the general knowledge of discrete manufacturing processes that produce distinct, countable item [23, 31], e.g. a welding spot on a car-body. Products of such manufacturing can be easily identified and differ greatly from continuous process manufacturing whose products are undifferentiated, e.g. petroleum. The core ontology has an operation-centric view (Figure 3a), where an operation is a process that produces an atomic product. The operation takes raw resource in and outputs operation products. The operations are performed by machines under some control systems. Quality

indicators are essential for operations since they need to be monitored to track the product quality. The core ontology is important to make the patterns of different domain ontologies consistent across different domain ontologies (explained in the next paragraph), thus allowing the KG solution to scale to other manufacturing processes. The core ontology is also important to allow semantic connections of all domain ontologies of different manufacturing processes to O_{ml} in a standardised way (explained in the Mapping Reasoner/Annotator paragraph). All classes in O_{core} can be grouped into two types: Type 1 classes correspond to the entities in the later KGs. Type 2 classes correspond to data properties in the KGs. All Type 1 classes are connected to Type 2 classes with object properties. All Type 2 classes are connected with datatype properties with a similar name. These Type 2 classes correspond to features in manufacturing datasets. For example, the triples:

core:Machine core:hasMachineID, core:MachienID
core:MachienID core:hasMachienIDValue xsd:string

where *core:Machine* is a Type 1 class corresponding to an entity in KGs. *core:MachienID* is a Type 2 class, corresponding to a property of the entity *Machine* (and a feature name in manufacturing datasets).

The domain ontologies follow the patterns in O_{core}, namely all the classes/properties in domain ontologies are sub-classes/sub-properties of classes/properties in O_{core}. We constrain the creation of domain ontologies in such a way to ensure the consistent patterns and unified understanding across different domain ontologies. We can see in Figure 3b that the O_{rsw} follows the patterns of O_{core} by instantiating the generic terms like operation, machine, control system to RSW-specific terms, like RSW operation, welding machine, welding control, and adding a lot more detailed knowledge of the RSW process, such as control module, spot diameter, etc. The RSW domain ontology O_{rsw} (Figure 3b) helps to reflect the understanding of users for the domain so that they can use it as a discussion basis and annotate data from difference sources with RSW domain ontology terms, which correspond to classes in O_{rsw}. The O_{rsw} follows the structure in O_{core} that the datatype properties are connected with classes with the same name, so that these classes can be used to annotate feature names in raw datasets. The data annotation then helps to generate the Data-To-DO Mapping.

The KG ontology O_{rsw-kg} is a simpler domain ontology (Figure 4a). It is used to generate the upper level schema of the Welding-ML KG. The O_{rsw-kg} differs from the domain ontology O_{rsw} in that O_{rsw-kg} is created from a bottom-up and data-driven approach, and should reflect the the lower level projection of O_{rsw} on specific datasets. The introduction of O_{rsw-kg} is necessary because O_{rsw} is more close to the domain understanding and cannot meet the specificities of vaiours datasets. If O_{rsw} is directly applied for KG-generation, the resulting KGs will have many blank nodes, since many classes in O_{rsw} cannot find their correspondence in the data, leading to inconvenience of the applications layered on top of the KGs. The O_{rsw} follows the structure in O_{rsw} that the datatype properties are connected with classes with the same name, so that the Data-to-DO Mapping also works for O_{rsw-kg}, e.g. *rsw-kg:WeldingMachine rsw-kg:hasWeldingMachineID*, *rsw-kg:WeldingMachienID*, and *rsw-kg:WeldingMachienID rsw-kg:hasWeldingMachienID*, *xsd:string*. The O_{rsw-kg} can be generated manually or automatically. In this work, we assume it is generated manually. The automation of generation of O_{rsw-kg} through e.g. aggregating O_{rsw} remains as a future research direction.

The ML ontology O_{ml} (Figure 4b) is a task ontology that encodes the general knowledge of machine learning analysis. It contains 353 axioms, 86 classes, 25 object properties and 5 datatype properties; it can be expressed using Description Logic $\mathcal{ALCH}(\mathcal{D})$. The O_{ml} enumerates the possible features groups that the domain ontology terms should be assigned to, like single features, identifiers, time series, etc., which reflect the semantic aspects of the features. The feature groups are also syntactically well-defined by the object property *ml:hasDataStructure*, which links the semantic feature groups to data structure classes like *ml:SingleValue*, *ml:Array* or *ml:Matrix*, syntactically defining the dimension of data. Then O_{ml} also prescribes algorithms that are applicable to these feature groups, such as preprocessing algorithms, feature processing algorithms, and ml algorithms, thus defining the reachability between the feature groups and algorithms. In this way, we categorise all input data into feature groups in O_{ml}. This helps the users to select a series of feature processing algorithms, thus creating a chain of feature processing modules, and generating ML solutions in a semi-automated way [30].

Mappings. The KG solution system has two types of mappings: *Data-to-DO Mapping* and *Data-to-ML Mapping*. The *Data-to-DO Mapping* is generated manually by users (typically welding experts). It maps the raw data to the domain ontology terms. In particular, the users inspect the raw data and the domain ontology O_{rsw}, and create links between the raw feature names and the domain ontology terms, i.e. classes in O_{rsw}. The class labels then serve as unified feature names of the features in uniform data formats. For example, the raw feature names *CurrenAmp*, *Current*, *Strom* come from production datasets and simulation datasets. They are all mapped to the Class *rsw:OperationCurveCurrentArrayValue*.

The *Data-to-ML Mapping* is either generated automatically by the Mapping Reasoner, or modified manually by the users via Mapping Annotator. It maps the features in uniform data formats to the ML feature groups. In particular, it maps the unified feature names (class labels) to the feature group classes in the O_{ml}. For example, the unified feature name *ObservationCollectionArrayValue* is mapped to the class *ml:TimeSeries*, which is a feature group class that means series of numeric values with time stamps. Since all features of type *ml:TimeSeries* are one dimensional arrays, the class *ml:TimeSeries* is linked to the class *ml:Array*.

Mapping Reasoner/Annotator. The Mapping Reasoner/Annotator takes the Data-to-DO Mapping and O_{core} as input and automatically generates the Data-to-Mapping. It also allows users to manually annotate the data with ML feature groups. We now illustrate how the automatic mapping generation works using OWL 2 axioms in the Manchester Syntax [9], where classes and properties have prefixes *rsw-kg:*, *core:*, *ml:* that indicate the ontologies they belong to.

(1) Class: *rsw-kg:OperationCurveCurrentArrayValue*
(2) SubClassOf: *core:ObservationCollectionArrayValue*
(3) Class: *core:ObservationCollectionArrayValue*
(4) SubClassOf: *ml:TimeSeries*
(5) Class: *ml:TimeSeries*
(6) SubClassOf: *ml:hasDataStructure* only *ml:Array*

Algorithm 1: Welding-ML KG Generation

Input: O_{rsw-kg}, O_{ml}, M, D
Output: KG

1 **Initialisation:** $S \leftarrow O_{rsw-kg}$, $KG \leftarrow \{\}$
2 **foreach** $(A, B, L, r_1, r_2) \in O_{rsw-kg}$, $C \in O_{ml}$ **do**
3 **if** $r_1(A, B), r_2(B, L)$ **then**
4 S := $S \cup \{r_1(A, L)\}$
5 S := S $\setminus \{r_1(A, B), r_2(B, L)\}$
6 **if** $\{B \subseteq C\} \in M$ **then**
7 S := S $\cup \{a(r_2, C)\}$
8 **end**
9 **end**
10 **end**
11 E $\leftarrow$ extractEntities(S)
12 **foreach** $(E_i, D_{sub}) \subseteq (E, D)$ **do**
13 **foreach** $attribute \in D_{sub}$ **do**
14 o $\leftarrow$ identifyEntities($attribute$)
15 **if** $o \in E$ **then**
16 KG := KG $\cup \{r_o(E_i,\ o)\}$
17 **else**
18 KG := KG $\cup \{r_d(E_i,\ o)\}$
19 **end**
20 **end**
21 **end**

We continue the example of the feature with the name *Current* in the raw data. It is annotated by the users with the class *rsw-kg:OperationCurveCurrentArrayValue* (Line 1), which is a sub-class of *core:ObservationCollectionArrayValue* (Line 2). The class *core:ObservationCollectionArrayValue* is linked to *ml:TimeSeries* via *rdfs:SubClassOf*, thus connecting O_{core} and O_{ml}. Here domain ontologies are connected to O_{ml} through O_{core}, so that we do not need to create semantic connections for specific domain ontologies. Instead, there exists a standardised way of semantic connections between O_{ml} and all domain ontologies. To define that the class *ml:TimeSeries* is a one dimensional array, it is linked to the data structure class *ml:Array*. Through reasoning, the feature *Current* is mapped to *rsw-kg:OperationCurveCurrentArrayValue* in domain ontologies, and *ml:TimeSeries* and *ml:Array* in the ML ontology, namely:

(7) Class: *rsw-kg:OperationCurveCurrentArrayValue*
(8) SubClassOf: *ml:TimeSeries*
(9) SubClassOf: *ml:hasDataStructure* only *ml:Array*

3.3 KG Generation

We illustrate and explain the procedure of generation of the Welding-ML KG in Algorithm 1. Our Welding-ML KG Generation algorithm takes four inputs: the KG ontology O_{rsw-kg}, the ML ontology O_{ml}, the Data-to-ML (data to ML ontology) mapping M, and the integrated Data D in relational tables. It takes two steps to generate the KG. Step 1: generate KG schema S; Step 2: transfer integrated data D to KG.

We start our Algorithm 1 with initialisation (Line 1), where the KG schema S is initialised with O_{rsw-kg} and the KG with an empty set. In step 1 (Line 2 to 11), we simplify the KG ontology O_{rsw-kg} to the KG schema S in order to make the KG more user-friendly by directly connecting entities with datatype properties in S. We first find all classes in O_{rsw-kg} that are connected to datatype properties in a structure of $r_1(A, B), r_2(B, L)$ (Line 3), where A and B are classes, L is a literal, r_1 is the object property connecting A and B, r_2 is the datatype property connecting B and L. The structure is then simplified to $r_2(A, L)$ in the KG schema (Line 4 and 5). It makes the hierarchy in the KG becomes shallow and more user-friendly to be queried. Further more, we find the class C in the O_{ml}, which is linked to class B through the Data-to-ML Mapping M (Line 6), and create an annotation $a(r_2, C)$ (e.g. *ml:hasMLAnnotation*) that links the simplified datatype property $r_2(A, L)$ to class C (Line 7). In Step 2 (Line 11 to 21), we first extract the entities E from the Schema S (Line 11). Then we transform the relational csv tables D to Welding-ML KG. Each sub-table D_{sub} corresponds to a class E_i. We enumerate all attributes in each sub-table D_{sub}, and identify whether the attribute is an entity (Line 14), using the attribute name (unified feature names). If the identified attribute is an entity (Line 14), the KG is extended by adding a triple $r_o(E_i,\ o)$ that connects the entity E_i in the sub-table D_{sub} to the object o with an object property (Line 16). Otherwise, the KG is expanded with another triple $r_d(E_i,\ o)$ that connects the entity E_i to the object o with an datatype property (Line 18).

4 ML ANALYTICS APPLICATIONS

We now introduce the ML analytics applications for solving the three tasks in Section 2.2.

4.1 ML Analysis for Quality Estimation

Question Definition. In this application, the ML analysis will process and model the welding data to estimate the welding quality indicator, spot diameters, after each welding operation. Therefore, the spot diameter will be the output feature of ML models. We analyse a most common scenario simple scenario consisting of one welding machine, one type of car body worksheets with identical nominal sheet thickness and material, and three welding programs for three different target spot diameters.

Data Description. The data are collected from welding simulation process generated by a verified Finite Element Method (FEM) model [29]. A total of 13,952 welding spots with diameter measurements were collected. Two types of data exist for each welding spot: (1) 20 process curves, including input curves such as electric current, voltage, resistance, and process feedback curves, such as welding electrode force of, electrode displacement, temperature of certain measurement positions, etc. (2) 235 Single features, such as nominal and measured geometry or material properties of the welding electrodes and worksheets, simulation setting parameters, welding programs etc.

ML Pipeline. To solve the quality estimation task, we follow the ML pipeline depicted in Figure 4a, which requires three types of feature groups: Single Features (SF), Time Series (TS), and Quality Indicator (QI). This is easily achieved by querying the Welding-ML KG since all data in the KG are annotated with ML feature groups.

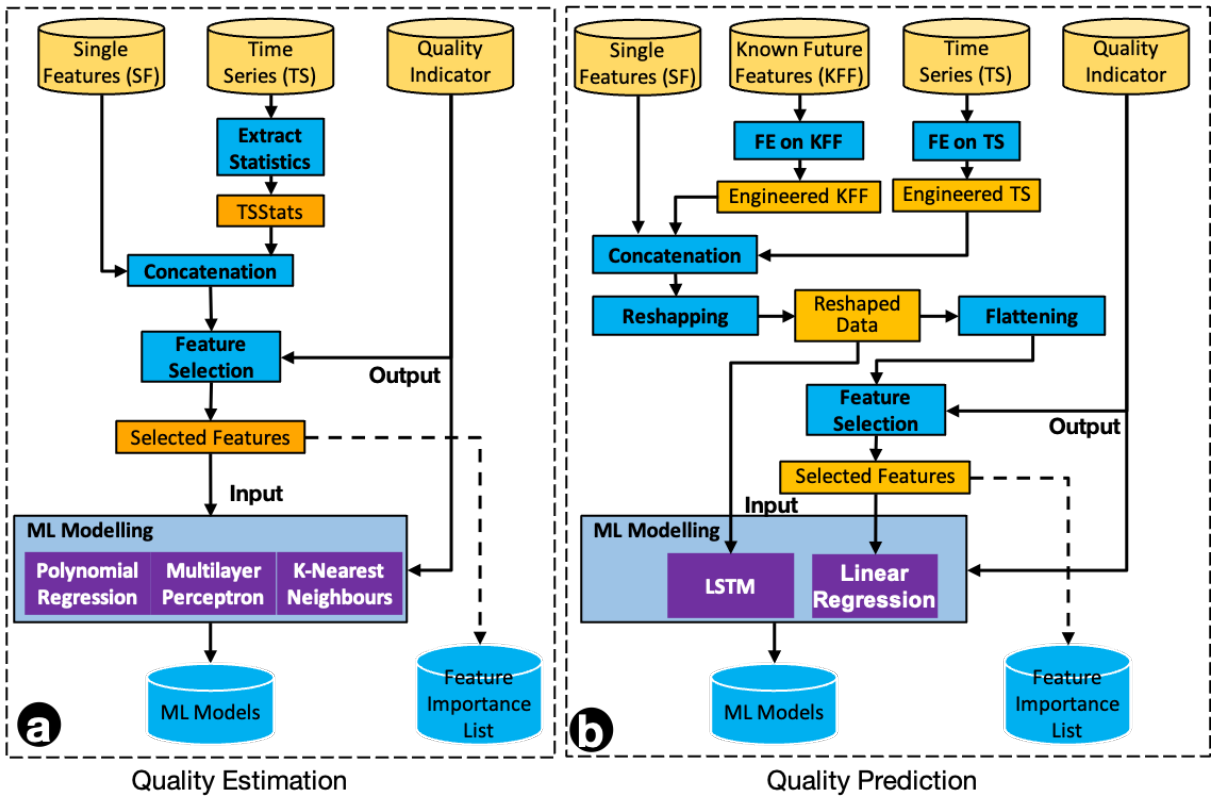

Figure 5: Schematic illustration of the ML pipelines. FE: feature engineering. The ML pipelines generated a number of ML models for quality estimation and prediction, and Feature Importance Lists, which is ranking lists of the most influential features.

We illustrate this with the Example 1, where we retrieve all single features.

Example 1 Retrieving single features

```
SELECT ?feature
WHERE { ?feature ml:hasMLAnnotation ml:SingleFeature . }
```

After retrieving all of the single features, we enumerate them with SPQRQL queries and serialise the data as some formats that are amicable for data analysis, e.g. numpy arrays. Then these data are fed into ML pipelines. In Figure 4a, the Single Features of the data structure *ml:SingleValue*, namely single values (i.e. there exists only one value for one operation entity). They are concatenated with time series statistics (TSStats), which are generated from Time Series. The feature group TS are of the data structure *ml:Array*, namely one dimensional vectors. From the time series, statistic features (such as mean, maximum, maximum position, etc.) are extracted and named as TSStats. The TSStats are of the data structure *ml:SingleValue*, the same as SF, and thus can be concatenated with SF. The concatenated features go to the algorithm Feature Selection, which also takes in the output feature, the quality indicator, since it follows a wrapper method [18]. After feature selection, the Selected Features will serve as input, and Quality Indicator as the output, for ML modelling with three methods: Polynomial Regression, Multilayer Perceptron, and K-Nearest Neighbours, generating a number of ML models. The Selected Features with its ranking of importance will be stored as the Feature Importance List, which reveals what features are influential for estimating the spot diameters.

Benefits of the Solution. This example demonstrates that our solution disentangles the specificities of heterogeneous data from the generalities of ML pipelines. Both in the query and the ML pipelines, no specific information of attribute names, data formats, etc. are mentioned at all. Instead, the ML pipelines work directly with the feature groups (e.g. single feature, time series). This is due to: (1) unification of all data formats to the KG; (2) annotation of properties in the KG with ML classes. Without the KG solution, this disentanglement would not be so convenient and efficient.

4.2 ML Analysis for Quality Prediction

Question Definition. In this application, the ML analysis will process and model the welding data to predict the welding quality indicator, Q-Value, in the future before the actual welding operation happens. Therefore, the Q-Value will be the output feature of ML models.

Data Description. The data are collected from two welding machines in a running production line, with a total of 5994 welding operations. The input features also contains two types: (1) 4 process curves, including current, resistance, voltage and pulse width modulation; (2) single features, such as production setting parameters, monitoring statuses, electrode wearing status, maintenance status,

etc. The input feature will be single features and time series. One more feature group added is a subset of the single features, named as *known future features*, which include the welding program number, wear count, and dress count. These features for future welding operations are already known before the operations happen since the operations are performed according to a pre-designed scheme.

ML Pipeline. To solve the quality estimation task, we follow the ML pipeline depicted in Figure 4b, which requires four types of feature groups: Single Features (SF), Known Future Features (KFF), Time Series (TS), and Quality Indicator (QI). The KFF are features for future welding operations that are already known before the operations happen since the operations are performed according to a pre-designed scheme, e.g. welding program number, wear count, and dress count.. Another complication is that here we need to predict quality indicators in the future, which means the temporal order of welding operations is important. This is different from the quality estimation, where each welding is treated as an independent event and the temporal order of data is largely ignored. The data need to be retrieved and ordered by the temporal order to assure that the data are arrange in such a way that they can attain the correct temporal order. This is easily achieved by querying the Welding-ML KG and is illustrated by Example 2.

Example 2 Retrieving time series and ordering by date time

```
SELECT ?operation ?feature
WHERE { ?operation rdf:type rsw-kg:RSWOperation .
    ?operation ?p ?feature .
    ?operation rsw-kgs:hasDataTime ?datetime.
    ?p ml:hasMLAnnotation ml:TimeSeries . }
ORDER BY ?datetime.
```

After data retrieval and preparation, the data go through the ML pipeline for predicting the future quality indicators. Three different types of input features, SF, KFF and TS go through different modules of Feature Engineering (FE). In this example, the FE on KFF is to perform mathematical transformation like deriving the first difference, getting the index of change etc. The FE on TS is to extract statistic features (such as mean, maximum, etc., similar to the previous example) as well as geometric features, like slope, drop, etc. The resulting engineered features are of the data structure *ml:SingleValue*, and can be concatenated. The concatenated features go to the algorithm Reshaping, which will reshape the data in such a way that in each row of the reshaped data, a matrix of data of the previous welding operations is created to predict the quality indicator (Q-Value). The Reshaped Data contain the temporal order of the input data. They can be directly fed as input into LSTM (long short-term memory), which is a type of neural networks that are powerful for handling temporal data. The Reshaped Data can also be flattened and then go through Feature Selection, to be modelled by classic ML methods like Linear Regression (LR). The output feature is the quality indicator, Q-Value. The ML modelling will generate a number of ML models. Similarly, the Selected Features with its ranking of importance will be stored as the Feature Importance List, revealing what features are influential for predicting the Q-Values.

Benefits of the Solution. In addition to the disentanglement mentioned in the previous example, this example shows more advanced manipulation to the data, that the ordering by date time becomes simple by slightly modifying the SPARQL query. Without which, the users would need to rely on programming languages (e.g. Python) to process the huge volume of data, which our users would need excessive time to learn and adapt on.

5 RELATED WORK

Knowledge graphs have been widely used in industries [7, 10, 19, 28]. The methods for KG generation have also been studied in many works [8, 13, 17]. An extensive survey [20] covers semantic technologies for data mining and knowledge discovery, in particular in the facilitation of ML workflows. There exist other ontologies for manufacturing (e.g., [2], [24], [16], [15], [6], [25]). Still, to the best of our knowledge, existent ontologies and system solutions only partially meet our R1-R4 requirements, and do not address the challenges of handling heterogeneous data and scaling ML methods.

Thus we had to develop our own KG solution and ontologies as well as KG-based, highly customised and configurable application, integrated into the workflow to support ML analytics for quality monitoring in manufacturing.

6 CONCLUSION AND OUTLOOK

Conclusion. In this paper we introduced our Bosch use case of ML-based quality monitoring in a highly-automated manufacturing process, the resistance spot welding. We summarised the challenges of the use case and derived requirements for KG solutions. To address the challenges we proposed our KG solution that can generate the Welding-ML KG from welding raw data. Our KG solution takes semantic artefacts such as ontologies and mappings and inputs, and generates the KG with reasoning and an algorithm. We demonstrated the usage of the KG solution in tasks of ML analytics for quality estimation and prediction, with two example ML pipelines. The proposed KG solution is used in our system with real industrial production data of two production lines and 27 welding machines.

Lessons Learned. The Welding-ML KG provides an efficient foundation for the data retrieval, since the users do not need to dive into the data level any more. They can simply pick a ML pipeline and then query the KG to prepare the data. Further more, the ML solutions encoded in such ML pipelines are highly reusable, since they disentangle the ML generalities (feature groups and ML pipelines) from data specificities (various feature names from different raw datasets) by defining feature groups, which are abstract representation of data in the ML ontology.

Outlook. The KG solution is deployed in our evaluation environment, and we consider to push it further into a more advanced and strict evaluation phase of production that runs in real-time. To show the benefits, we also plan to demonstrate our KG solution with more users and more use cases. In the future research, we plan to improve the our KG solution in many directions: to enhance the KG generation modules to improve the compatibility of the KG schema to the domain ontologies; to extend the KG solution for more applications, e.g. question answering, visualisation, statistic analysis; to improve the semantic artefacts, e.g. to compare the domain ontologies and core ontology with generic upper ontologies.

REFERENCES

[1] Ethem Alpaydin. 2009. *Introduction to Machine Learning*. MIT Press, Massachusetts.

[2] Stefano Borgo and Paulo Leitão. 2004. The Role of Foundational Ontologies in Manufacturing Domain Applications. In *OTM*.

[3] Sujeet Chand and Jim Davis. 2010. What is Smart Manufacturing. *Time Magazine Wrapper* 7 (2010), 28–33.

[4] DIN. 2004. EN14610:2004 Welding and Allied Processes - Definitions of Metal Welding Processes; Trilingual Version. *German Standards (Deutsche Norm)* 14610 (2004).

[5] DVS. 2016. *Widerstandspunktschweißen von Stählen bis 3 mm Einzeldicke – Konstruktion und Berechnung*. Standard. Deutscher Verband für Schweißen und Verwandte Verfahren e. V., Düsseldorf, DE.

[6] Xenia Fiorentini et al. 2007. *An Ontology for Assembly Representation*. Technical Report. NIST.

[7] Martina Garofalo, Maria Angela Pellegrino, Abdulrahman Altabba, and Michael Cochez. 2018. Leveraging Knowledge Graph Embedding Techniques for Industry 4.0 Use Cases. In *Cyber Defence in Industry 4.0 Systems and Related Logistics and IT Infrastructures*. IOS Press, 10–26.

[8] Travis Goodwin and Sanda M Harabagiu. 2013. Automatic Generation of a Qualified Medical Knowledge Graph and Its Usage for Retrieving Patient Cohorts From Electronic Medical Records. In *2013 IEEE Seventh International Conference on Semantic Computing*. IEEE, 363–370.

[9] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F Patel-Schneider, and Sebastian Rudolph. 2009. OWL 2 Web Ontology Language Primer. *W3C Recommendation* 27, 1 (2009), 123.

[10] Thomas Hubauer, Steffen Lamparter, Peter Haase, and Daniel Markus Herzig. 2018. Use Cases of the Industrial Knowledge Graph at Siemens.. In *International Semantic Web Conference (P&D/Industry/BlueSky)*.

[11] ISO. 2004. *Resistance Welding – Procedures for Determining the Weldability Lobe for Resistance Spot, Projection and Seam Welding*. Standard. International Organization for Standardization, Geneva, CH.

[12] ITU. 2012. *Recommendation ITU – T Y.2060: Overview of the Internet of Things*. Technical Report. International Telecommunication Union.

[13] Nitisha Jain. 2020. Domain-Specific Knowledge Graph Construction for Semantic Analysis. In *European Semantic Web Conference*. Springer, 250–260.

[14] Henning Kagermann. 2015. Change Through Digitization – Value Creation in the Age of Industry 4.0. In *Management of Permanent Change*. Springer, 23–45.

[15] Sylvere Krima et al. 2009. *OntoSTEP: OWL-DL Ontology for STEP*. Technical Report. NIST.

[16] S. Lemaignan et al. 2006. MASON: a Proposal for an Ontology of Manufacturing Domain. In *IEEE DIS*.

[17] Thorsten Liebig, Andreas Maisenbacher, Michael Opitz, Jan R Seyler, Gunther Sudra, and Jens Wissmann. 2019. Building a Knowledge Graph for Products and Solutions in the Automation Industry. (2019).

[18] Ralf Mikut et al. 2006. Data Mining in Medical Time Series. *Biomedizinische Technik* 51 (2006).

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

[20] Petar Ristoski and Heiko Paulheim. 2016. Semantic Web in data mining and knowledge discovery: A comprehensive survey. *Journal of Web Semantics* 36 (2016), 1–22.

[21] Arthur L. Samuel. 2000. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development* 44, 1.2 (2000), 206–226.

[22] Martin G. Skjæveland, Daniel P. Lupp, et al. 2018. Practical Ontology Pattern Instantiation, Discovery, and Maintenance With Reasonable Ontology Templates. In *ISWC*.

[23] Yulia Svetashova, Baifan Zhou, Tim Pychynski, Stefan Schmidt, York Sure-Vetter, Ralf Mikut, and Evgeny Kharlamov. 2020. Ontology-Enhanced Machine Learning: a Bosch Use Case of Welding Quality Monitoring. In *International Semantic Web Conference*.

[24] Zahid Usman, Robert Ian Marr Young, et al. 2011. A Manufacturing Core Concepts Ontology for Product Lifecycle Interoperability. In *Enterprise Interoperability*.

[25] Dušan Šormaz and Arkopaul Sarkar. 2019. SIMPM – Upper-level Ontology for Manufacturing Process Plan Network Generation. *Robotics and Computer-Integrated Manufacturing* 55 (2019).

[26] Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. 2016. Machine Learning in Manufacturing: Advantages, Challenges, and Applications. *Production & Manufacturing Research* 4, 1 (2016), 23–45.

[27] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X. Gao. 2019. Deep Learning and Its Applications to Machine Health Monitoring. *Mechanical Systems and Signal Processing* 115 (2019), 213–237.

[28] Pai Zheng, Liqiao Xia, Chengxi Li, Xinyu Li, and Bufan Liu. 2021. Towards Self-X Cognitive Manufacturing Network: An Industrial Knowledge Graph-Based Multi-Agent Reinforcement Learning Approach. *Journal of Manufacturing Systems* 61 (2021), 16–26.

[29] Baifan Zhou, Tim Pychynski, Markus Reischl, and Ralf Mikut. 2018. Comparison of ML Approaches for Time-series-based Quality Monitoring of Resistance Spot Welding. *AoDS, Series A* (2018).

[30] Baifan Zhou, Yulia Svetashova, Seongsu Byeon, Tim Pychynski, Ralf Mikut, and Evgeny Kharlamov. 2020. Predicting Quality of Automated Welding with Machine Learning and Semantics: a Bosch Case Study. In *CIKM*.

[31] Baifan Zhou, Yulia Svetashova, Tim Pychynski, and Evgeny Kharlamov. 2020. SemFE: Facilitating ML Pipeline Development with Semantics. In *CIKM*.