Published in: Industrial Knowledge Graphs, co-located with the 9th International ACM Web Science Conference 2017. June 25, 2017 in Troy, NY, USA

Generating Digital Twin models using Knowledge Graphs for Industrial Production Lines

Agniva Banerjee
University Of Maryland, Baltimore County
Baltimore, MD 21250, USA
agniv1@umbc.edu

Raka Dalal
University Of Maryland, Baltimore County
Baltimore, MD 21250, USA
rak2@umbc.edu

Sudip Mittal
University Of Maryland, Baltimore County
Baltimore, MD 21250, USA
smittal1@umbc.edu

Karuna Pande Joshi
University Of Maryland, Baltimore County
Baltimore, MD 21250, USA
kjoshi1@umbc.edu

ABSTRACT

Digital Twin models are computerized clones of physical assets that can be used for in-depth analysis. Industrial production lines tend to have multiple sensors to generate near real-time status information for production. Industrial Internet of Things datasets are difficult to analyze and infer valuable insights such as points of failure, estimated overhead. etc. In this paper we introduce a simple way of formalizing knowledge as digital twin models coming from sensors in industrial production lines. We present a way on to extract and infer knowledge from large scale production line data, and enhance manufacturing process management with reasoning capabilities, by introducing a semantic query mechanism. Our system primarily utilizes a graph-based query language equivalent to conjunctive queries and has been enriched with inference rules.

KEYWORDS

Digital Twin, Knowledge Graph, Big Data, Industrial Internet of Things, Semantic Web

1 INTRODUCTION

The manufacturing industry depends on a very wide range of technical fields which include precision and ultra-precision machining, non-traditional machining, micro-nano manufacturing, composite forming, assembly technique, digital design and manufacturing. Every such system consumes or generates huge datasets. This data is extremely valuable for enterprises as it aids in making crucial production line decisions. Therefore, it's important to integrate, reuse, create and manage knowledge derived from this data to maximize benefits.

Digital Twin is quickly becoming the cornerstone for manufacturing process management. The concept of creating a virtual, digital equivalent to a physical product or a 'Digital Twin' was introduced in 2003 at University of Michigan Executive Course on Product Life-cycle Management (PLM) [13]. It contains three main parts: a) physical products in real space, b) virtual products in virtual space, and c) the connections of data and information that ties the virtual and real products together. Since this model was introduced, there has been tremendous increases in the amount, richness, and fidelity of information of both the physical and virtual products. With the advent of Industrial Internet of Things, even more data is being generated that further highlights the importance of Digital Twinning.

In this paper, on the virtual side, we have added numerous reasoning characteristics so that an entire production line can be tested for performance capabilities. Our lightweight model allows manufacturers to reason about complex system, including their physical behaviors, in real-time and with acceptable computational costs. With this, our model shifts the need of having domain experts who help create rules for production line management, to an artificially intelligent reasoning system which anybody with minimal technical know-how can use. We have created a pipeline to automate the process of extracting semantic relation from sensor input. Our pipeline has four distinct stages, starting from extracting features, manipulating the extracted features based on the ontology, generating the knowledge graph [14] and lastly, inferring relations from the graph. Since we sourced our data from Bosch though Kaggle, majority of our work had to modeled around the data. But our overall research contributions are generic, and can easily be ported to any such manufacturing production line management system. The ontology we have defined enables reusing domain knowledge, structure and vocabularies, which makes our approach generic and suitable to other sensor input data. The main contributions of this paper are:

(1) Feature Extraction: We normalized the Bosch Production Line Performance data from three types (*Categorical*, *Date and Numeric*) to a *Numerical* type. Our analysis of the normalized data revealed correlated patterns. We extracted features which best explained the variance in the data, for example: *lead/lag rate*, *featureMeasurement*, *turnAroundTime*, etc.
(2) Ontology Creation: Based on the features extracted and our analysis of the data, we came up with an ontology tailored towards manufacturing production line data. Our ontology has the following main classes: *Facility* (resource, tools, other assets integral to the operation), *Process* (an activity/action to be performed in a manufacturing enterprise), *Object* (parent class for holding concepts about

Industrial Knowledge Graphs, Troy, United States

DOI:

materials and items), *Operation* (generic concept, holds common entities such as timeStamp, location, isFlagged, etc.) and *Organizational Unit* (holds organization structure of an enterprise).

(3) Knowledge Graph Generation: Based on the ontology mentioned above, we generate a knowledge graph corresponding to the features described. This helps facilitate semantic relation extraction.

(4) *Semantic Relation Extraction*: We extracted inferred relations (relations which does not pre-exist) from the knowledge graph by utilizing Path Ranking Algorithm. For example, given queries:
   - "*What is the average time variance of feature X?*": We correlate time variance of every station feature X passes through and calculate the average based on the time variance values.
   - "*What is the average turnAround time of staion Y?*": We infer average turn around time for stations by correlating turnaround times of features with respect to Line.
   - "*Which Line has minimum response rate?*": We calculate minimum response rate per line by correlating between feature response of every feature, every station a feature passes through, and every individual station in every Line.

The rest of the paper is organized as – In Section 2 we discuss the related work. We discuss our system in Section 3 and our experimental results in Section 4. Section 5 includes a discussion on the topic and our future work.

2 RELATED WORK

Industries depend on the vast amounts of data collected by embedded sensors, data generated from the production line, etc. We need means to extract and infer knowledge from the data. Li et al. [10] were one of the first ones to come up with a dynamic bayesian network approach for digital twin, wherein they utilized the concept of digital twin for tracking the evolution of time-dependent variables to monitor aircraft structure. Fei Tao et al. [17] proposed a digital-twin driven product design, manufacturing and service with big data, but their work has been mostly investigative in nature. The Internet of things (IoT) is a dynamic smart networked system with physical devices and connected sensors that enables collection and data exchange among themselves. Nowadays, digital twins are rapidly used in the Industrial Internet or Industrial Internet of Things and certainly engineering and manufacturing. Canedo et al. discussed how IoT devices and IoT systems of systems can be managed and optimized throughout their lifecycle using the mechanism of digital twins [2]. Hong-guang et al. [1] proposed a novel intelligent process knowledge discovery strategy based on rough set attribute reduction. X Wang et al. [18] developed the web based process knowledge management base and an application system. Building knowledge graphs is one of the techniques to formalize knowledge. However, exploiting these data to build knowledge graphs is difficult due to the heterogeneity of the sources, scale of the amount of data, and noise in the data. Szekely et al. proposed to build knowledge graphs by exploiting semantic technologies to reconcile the data continuously crawled from various sources, to scale the data extracted from the crawled content, and to support interactive queries on that data [16]. Dong et al. [4] proposed an approach that combines extractions from Web content with prior knowledge derived from existing knowledge repositories. They employed supervised machine learning methods for fusing these distinct information sources.

Although a lot of techniques have been invented for creating knowledge graphs to represent huge amount of data which proved beneficial for enriching search results, answering factoid questions, and training semantic parsers and relation extractors, little progress has been made in the way of reasoning with these knowledge bases or using them to improve machine reading. They are mostly treated as simple lookup tables, a place to find a factoid answer given a structured query, or to determine whether a sentence should be a positive or negative training example for a relation extraction model. Gardner et al. proposed an approach to extract characteristics of the knowledge graph and construct a feature matrix for use in machine learning models. The extracted characteristics corresponded to Horn clauses and other logic statements over knowledge base predicates and entities [7]. Franconi et al. [6] proposed a common methodology for investigating the research problem of combining ontology and rule languages.

One of the more promising approach was taken by Song et al. [15] wherein they proposed a novel way of defining and organizing manufacturing process knowledge. From an industrial point view, Open innovation within an Enterprise 2.0 context [3] is one of the most popular paradigms for improving the innovation processes of enterprises, based on the collaborative creation and development of ideas and products. The key feature of this paradigm is that knowledge is exploited in a collaborative way flowing not only between internal sources, e.g. R&D departments, but also between external ones such as employees, customers, partners, etc. But, a couple of key limitations of this approach is that it requires support by an advanced technological infrastructure.

Another important approach is miKrow [12], a lightweight framework for knowledge management. It is composed of two layers: micro-blogging layer that simplifies how users interact with the whole system and two, a semantic engine that performs all the intelligent heavy lifting by combining semantic indexing and search of microblogs and users. But it does not support any manufacturing process management knowledge acquisition and reasoning.

3 METHODOLOGY

The Bosch Production Line performance data got published in Kaggle[1]. We are using the data for our research purposes. It contains 4266 features, spread accross three different types (Categorical, Date and Numeric). Based on our preliminary analysis of the data, we found that the numerical features are the most descriptive of the three types, although Categorical and Date types can be converted to numerical type. The data mainly contains information about stations production line and a number of combinations of related features such as turn around time, feature time variance, measurement, etc.

[1] https://www.kaggle.com/c/bosch-production-line-performance/data

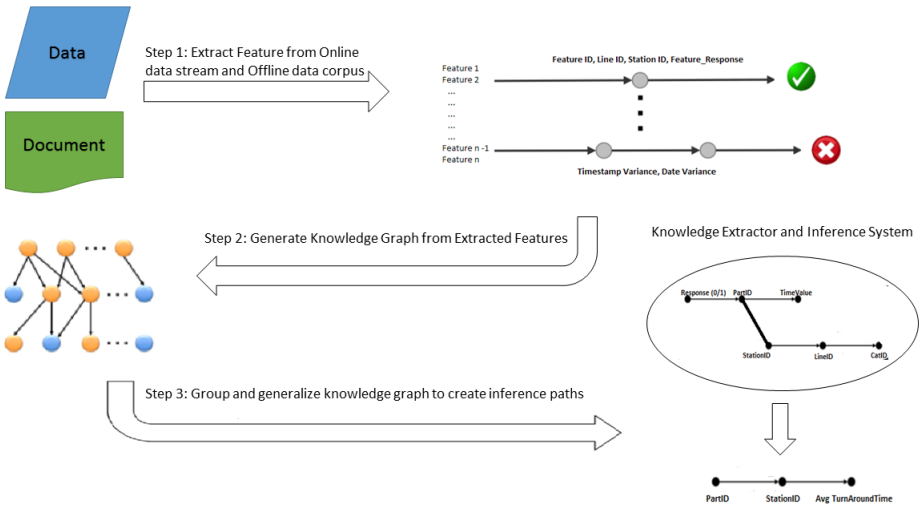

Figure 1: System Architecture Diagram

As shown in Figure 1, our system architecture has three distinct stages: data processiong and feature extraction, generating knowledge graph from the extracted features and lastly, semantic relation extraction from the knowledge graph. In the first stage, we process the data and extract features that help define the underlying variance in the data. Our analysis of the data had revealed that the categorical and date features could be normalized to mimic numeric featureset, and valuable insights such as Feature breakdown, delayed turnaround time and such can be inferred from the data. To maximize the probability of inferring such insights, we created an ontology and generated knowledge graph based on that ontology. For extracting semantic relation from the knowledge graph, we utilized Path Ranking Algorithm. In essence, the algorithm considers a generalized version of the knowledge graph and tries to infer relations based on paths it can traverse.

For example, lets consider Figure 4 with respect to the query "What is the average time variance of feature X?"To infer the relation between time variance of stations with respect to lines and an individual feature X, the PRA considers every station s_i and traverses individual lines l_i for the feature X. Since there exists no direct relation between time variances of stations and lines with respect to features, PRA generates a feature vector for each (s_i, l_i) pair. Since in this case there exists an edge sequence between the source and target nodes $(T_1 - StationID - FeatureID - LineID - T_1)$, the value of this feature vector will be non-zero.

Moreover, the approach at the core of our semantic inference and reasoning system is based on Digital Twin. We process the data based keeping in consideration its manufacturing, maintenance, operations and operating environments, and use this data to create a unique model of each asset, system or process, while focusing on a key behavior, such as life, efficiency or turn-around time.

3.1 Data Description

We collected Bosch Production Line Performance data from Kaggle[2] for our research. Bosch data represents measurements of parts as they move through Bosch's production lines. Because Bosch records data at every step along its assembly lines, they have the ability to apply advanced analytics to improve these manufacturing processes. However, the intricacies of the data and complexities of the production line pose problems for current methods.

In our case, the data that were made available had been obfuscated and reported in 3 distinct types, categorical, date and numeric [11]. Each part has a unique Id. The data represents measurements of parts as they move through Bosch's production lines. The goal is to predict which parts will fail quality control (represented by a 'Response' = 1). The training data consists of 1,183,747 samples with 969 numeric features, 2,141 categorical features and 1,156 date features, totaling 14.3GB of raw data. Hence, one of the biggest challenges of this dataset is to process these features into something meaningful so they can be used to make a predictive model.

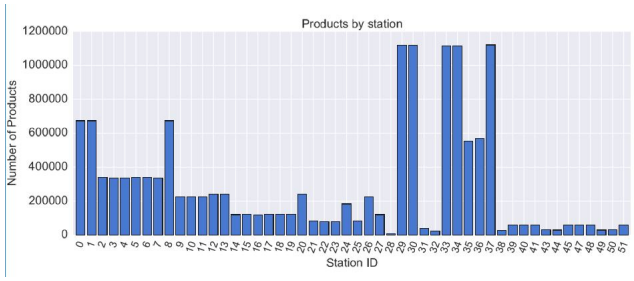

Figure 2: Number of Products/Station

[2]https://www.kaggle.com/c/bosch-production-line-performance/data

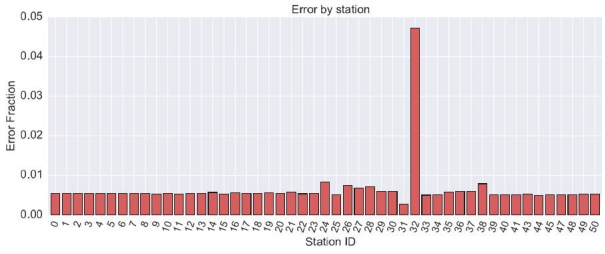

Figure 3: Errors/Station

The feature set included product timestamps along with Line and Station, Lead/Lag rate, product characteristics (mentioned as Feature in the dataset) separated as Numeric and Category. The numerical features contain information about stations production line and a test number combination. The value of a numerical feature is the corresponding measurement. For example, a feature named L3_S50_F4243 for a component indicates that the part went through production line 3, station 50, and the feature value corresponds to a test number 4243. This way, each product coming out of the manufacturing line can be segregated according to the production ffow. We observed that there exists 51 stations distributed between 4 production lines. Figures 2 and 3 visualizes number of individual poducts moving per production line, and numbes of false Responses (or Errors) per station. The categorical features has 500 multivalued, 1490 single valued and rest are empty (which we did not consider for our model). We have converted these features to numerical features by using one-hot encoding technique where every class has been represented by an integer [8]. The date features names are labeled by production line, station id and date id. For example, L3_S50_D4242, would mean the product went through production line 3, station 50, and the feature value corresponds to date id 4242. There are a total of 1157 date features, with a lot of missing values. We also observed that the train and test has the same time period, where the dates are transformed to 0 - 1718 with granularity of 0.01. We converted them to numerical features the same way we converted the categorical features, i.e, by one-hot encoding technique.

Since the size of the data is large, we applied Online Learning [5] to handle scalability issues. Online learning is a technique which is used when its computationally expensive to train the entire dataset in a batch or the algorithm needs to dynamically adapt to new patterns in the data. The training data becomes available sequentially and the model is updated each time a new data point becomes available.

3.2 Ontology Description

We for the purpose of this research, have constructed an ontology framework tailored towards manufacturing. The main entities are four in the ontology, namely Process, Organizational Unit, Object and Operations. Main entities may be used to hold data to non-specific, generic to all manufacturing. Since ontologies only carry general concepts, the two entities called Operations and Process are specific to enterprise. The following briefly describes the entities in the domain.

(1) *Facility* is the resources, tools, or other assets to perform the Operations. For our purpose, we have maintained Facility Type to cover *ProductionLine, MachineTools, Station,* etc. Facilities also organized in a hierarchy based on the attribute Parent Facility depending on the type of layouts such as *LineBatch, StationBatch* or *Mass Production.*
(2) *Process* is any kind of activity or action to be performed in the manufacturing enterprise. They are the basic descriptions of Operations. Parent Process is similarly defines the hierarchy of all processes.
(3) *Organizational Unit* is simply organizational structure of the enterprise. It may contain information about the whole enterprise itself or specific units under manufacturing department.
(4) *Object* entity holds concepts about materials and information necessary to perform manufacturing Operations. They flow through the Facility and undergoes a specific Process. A specific object always has a unique ID associated with it, along with the Facility and Process it goes through. It also has FeatureSets describing the Object (such as *Measurement, TurnAroundTimePerFacility, ObjectLocationInFacility,* etc.)
(5) *Operation* is not a generic concept in our ontology due to its *TimeStamp* and *Location* attributes. It is specific to our data. Operation data may be used for data mining purposes. Operations are common examples of first-level production activities.

The above entities apart from Operation are generic and more attributes to each entity may be added later.

3.3 Knowledge Graph and Semantic Relationship

We have defined the system methodology in 3 concurrent steps that correspond to the agents in the system framework. Each step is supported by relevant procedural and declarative knowledge. Our second and third step corresponds to the job of creating a knowledge graph based on the ontologies, and then consuming it through Semantic Relationship Extractor. Relation extraction is the task of translating some relationship between entities expressed in text into the formal language of a given knowledge base. For example, the sentence "*Average TurnAround time for a non-defective product X is Y*"might be translated into *AverageTurnAroundTime* (\"X") ,*IsResponse*(1, \"X") →(*Y*), where the relation *AverageTurnAroundTime* is an inferred relation not pre-existing in the knowledge base. Knowledge graph reasoning is similar to such relation extraction and link prediction task in social network analysis. To infer from a knowledge graph, is equivalent to the task of filling in facts that are missing from a knowledge base. That is, assuming that there exists a true knowledge base K with set triples (E_s, R, E_o), with this true knowledge base not being visible, and that there is a partial knowledge base K with missing facts (prepared from the data at hand), the task of inferring missing knowledge can be written as F = K - K. It is substantially more challenging because of 2 main reasons: 1. nodes in our knowledge graph are entities with different types and attributes, and 2. edges in knowledge graph are relations of different types. We have utilized a graph based method called PRA

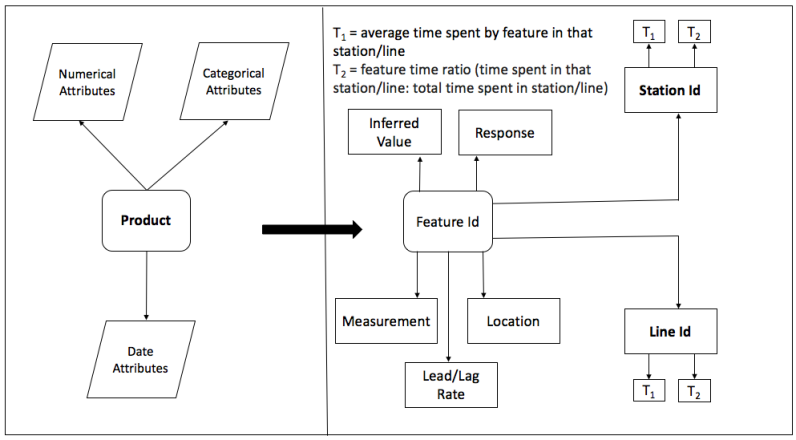

Figure 4: Data Format and Brief Feature Expansion

(Path Ranking Algorithm) [9] for performing link prediction in our graph.

The assumption made by PRA is that there is common substructure around node pairs that share the same edge label. PRA tries to model this substructure for each relation in the KB by extracting features of this substructure that correspond to paths between the node pair. PRA has a strong connection to logical inference, as each of the features used by PRA can be viewed as a particular kind of Horn clause. Consider a graph G with nodes N, edges E, and edge labels R, and a set of node pairs $(s_j, t_j) \in$ D that are instances of a target relation r. PRA will generate a feature vector for each (s_j, t_j) pair, where each feature is some sequence of edge labels $-e_1 - e_2 - \ldots - e_l-$. If the edge sequence corresponding to the feature exists between the source and target nodes in the graph, the value of that feature in the feature vector will be non-zero.

For computing the probability of a path existing between two sets of Feature vector, *StationID* and *Response* is found by using random walks to approximate the probability via rejection sampling: for each path type and source node, a number of random walks are performed, attempting to follow the edge sequence corresponding to the path type. If a node is reached where it is no longer possible to follow the path type, the random walk is restarted. This does not reduce the time necessary to get an arbitrarily good approximation, but it does allow us to decrease computation time, even getting a fixed complexity, at the cost of accepting some error in our probability estimates. Also, Lao [9] showed that when the target node of a query is known, the exponent can be cut in half by using a two-sided BFS. We have utilized this extensively, since in our case the path to traverse can be inferred directly from the question posed to the system, provided the question falls under the domain knowledge. We will elaborate further on this in Experimental Results section.

For example, let us consider the query "What is the average time variance of feature X?"again. In order to infer the co-relation between time variance of stations with respect to lines and an individual feature X, the PRA traverses the generated knowledge graph, starting its exploration from the source node extracted from the query, which in this case is *TimeVariance* (T_1). It traverses the graph based on reducing degrees of nodes and path length. In this case, PRA considers every station s_i and traverses individual lines l_i for the feature X. Since it could not find any direct correlation between *TimeVariance, Feature and Line*, PRA generates a feature vector for each (s_i, l_i) pair, starting from the source node of *TimeVariance* with respect to s_i / l_i. In this case, there only exists an indirect edge sequence between the source and target nodes $(T_1 - StationID - FeatureID - LineID - T_1)$, the value of this feature vector gets evaluated to a non-zero value. We utilize the probability of such path sequences generated by PRA to infer semantic relations.

Since there exists a close relation between product, process and resources, i.e., given a product, its manufacturing process and resources allocated towards manufacturing that product, we can exploit this domain specific knowledge to a great extent. To that end, given the domain specific knowledge, we were able to identify and infer from the knowledge graph machining operations and sequences, machining costs, average turnaround time for a non-defective product (Response = Yes) by the process aforementioned.

Id	Line	Station	Feature	Value	Response
4	0	0	0	0.03	0
7	0	0	3	0.088	1
9	0	2	3	0.002	0
11	0	2	3	0.016	0
15	1	4	5	0.015	0
18	1	4	6	-0.016	0
26	1	4	7	0.016	1
27	1	5	7	-0.062	1
28	2	6	8	-0.075	1
31	2	6	8	-0.003	0

Figure 5: Result Visualization

4 EXPERIMENTAL RESULTS

In our research, we have mined the multidimensional relationships between entities, and solved the information conflicts generated by multi-source information fusion. Since an ontology is an explicit

specification of a concept that provides a comprehensive specification of knowledge for a domain, it provides a basis for semantic relation mining for systems such as ours that process information and infer knowledge. Although production line data usually bear mnemonic names, their only actual connection to natural language is by the labels that are attached to them. These labels often provide a canonical way to refer to the data. Figure 5 shows a snippet of our result. Here, the value column holds measurement, and the Response clumn shows Errors per station and Production Line.

Based on the production line labels and the relations that those label exemplifies, we queried our system about *Response Rate* (pass/-fail of products passing through the production line), *Time Variance* (difference between expected duration of a product to individually pass through stations and lines during a unique process and the actual time taken) and *Average Turnaround Time* (average time taken by a product to finish) for a non-defective product. We made a total of 120 different queries to our model - 40 about response rate, 40 about time variance and 40 about average turnaround time for different products. Next we manually evaluated the accuracy of our model. It answered 34 questions correctly on response rate with respect to individual features, 37 on time variance with respect to station and line corresponding to an individual feature and 33 on average turnaround time for different products. Our overall accuracy is 88.33%.

To present a working example, we fed a query to our system "What is the turn around time of product X?"Our system extracts the required knowledge from the knowledge graph and feeds it to the semantic relationship extractor and finally we get the answer "Turn around time of product X is 6.32".

5 DISCUSSION AND FUTURE WORK

In this paper, we describe a mechanism to extract and infer knowledge from large scale production line data. Our system utilizes a graph-based query language to extract features from data for reasoning. For our research, we have done feature extraction, ontology creation, knowledge graph generation and semantic relation extraction. We had to normalize and analyze the data which revealed existence of correlated patterns. We extracted features which could explain the variance and correlation of data based on the product vectors. We found that some features, such as *lead/lag rate, featureMeasurement, turnAroundTime*, etc. contributed towards describing the data variance and predicting Pass/Fail response the best. We came up with an ontology tailored towards manufacturing production line data with the following main classes: *Facility, Process, Object, Operation* and *Organizational Unit.* To help facilitate semantic relation extraction, we generated a knowledge graph based on the ontology described. Furthermore, we successfully showed that it is possible to extract inferred relations (relations which does not pre-exist) from the knowledge graph by utilizing Path Ranking Algorithm.

Though we are are able to extract relations and answer complex queries using this model, there is a lot of scope of improving this model in future. The ontology we developed has been explained with great clarity, and it also provides us with the basis for useful inferential services, but it is not exhaustive and robust. We plan to take this ontology forward with the inclusion of more domain specific classes, and imposing more ontological constraints for a more robust, structured and inferentially useful semantic relationship extractor.

ACKNOWLEDGEMENT

The research in this paper was supported partially by the grants from GE Global Research Center and partially by the grants from International Business Machines Corporation (commonly referred to as IBM).

REFERENCES

[1] Bo, H.-g., Liu, X.-b., Ma, Y., and MENG, Q.-n. Rough-set-based process knowledge discovery approach in iron and steel industry. *Comput. Integr. Manuf. Syst 15* (2009), 135–141.

[2] Canedo, A. Industrial iot lifecycle via digital twins. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis* (2016), ACM, p. 29.

[3] Carbone, F., Contreras, J., and Hernández, J. Enterprise 2.0 and semantic technologies for open innovation support. *Trends in Applied Intelligent Systems* (2010), 18–27.

[4] Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (2014), ACM, pp. 601–610.

[5] Fontenla-Romero, Ó., Guijarro-Berdiñas, B., Martinez-Rego, D., Pérez-Sánchez, B., and Peteiro-Barral, D. Online machine learning. *Efficiency and Scalability Methods for Computational Intellect 27* (2013).

[6] Franconi, E., and Tessaris, S. Rules and queries with ontologies: a unified logical framework. In *International Workshop on Principles and Practice of Semantic Web Reasoning* (2004), Springer, pp. 50–60.

[7] Gardner, M. *Reading and Reasoning with Knowledge Graphs.* PhD thesis, Carnegie Mellon University, 2015.

[8] Kung, S. Y. *Kernel methods and machine learning.* Cambridge University Press, 2014.

[9] Lao, N., and Cohen, W. W. Fast query execution for retrieval models based on path-constrained random walks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), ACM, pp. 881–888.

[10] Li, C., Mahadevan, S., Ling, Y., Wang, L., and Choze, S. A dynamic bayesian network approach for digital twin. In *19th AIAA Non-Deterministic Approaches Conference* (2017), p. 1566.

[11] Mangal, A., and Kumar, N. Using big data to enhance the bosch production line performance: A kaggle challenge. *arXiv preprint arXiv:1701.00705* (2016).

[12] Penela, V., Álvaro, G., Ruiz, C., Córdoba, C., Carbone, F., Castagnone, M., Gómez-Pérez, J., and Contreras, J. mikrow: Semantic intra-enterprise microknowledge management system. *The Semanic Web: Research and Applications* (2011), 154–168.

[13] Rangan, R. M., Rohde, S. M., Peak, R., Chadha, B., and Bliznakov, P. Streamlining product lifecycle processes: a survey of product lifecycle management implementations, directions, and challenges. *Journal of computing and information Science in Engineering 5*, 3 (2005), 227–237.

[14] Singhal, A. Introducing the knowledge graph: things, not strings. *Official google blog* (2012).

[15] Song, H., Wang, H., Liu, T., Zhang, Q., and Gao, B. The design and development of manufacturing process knowledge base system based on ontology. In *International Conference on Cooperative Design, Visualization and Engineering* (2016), Springer, pp. 9–16.

[16] Szekely, P., Knoblock, C. A., Slepicka, J., Philpot, A., Singh, A., Yin, C., Kapoor, D., Natarajan, P., Marcu, D., Knight, K., et al. Building and using a knowledge graph to combat human trafficking. In *International Semantic Web Conference* (2015), Springer, pp. 205–221.

[17] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., and Sui, F. Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology* (2017), 1–14.

[18] Wang, X., Liu, C., and Wang, J. The manufacturing-oriented process knowledge representation and application for the rubber pad forming. *Mech. Sci. Technol. Aerosp. Eng 1* (2012), 10–14.