A Semantic Question Answering for Smart Factories

Orcun Oruc, Adrian Singer, Ken Wenzel, *Fraunhofer IWU, IEEE*

Abstract: The industrial production characterized by the increasing interconnection of machines and devices (e.g. Industrial Internet of Things, IIoT). Smart Factories evolved complex linked data aspect of manufacturing devices over the past few years. Leveraging Industry 4.0, smart factories require a machine-to-human or machine-to-machine cooperation in order to present information to experts, users or human operators. Semantically created data by production systems, mostly used by the Human Machine Interface (HMI) Devices. Nowadays, smart factories produce a large amount of data that need to be apprehensible by HMI Devices. OPC UA is a de facto standard to tackle the data source problem; several proposals introduced relevant to the question answering. Our proposal is to develop a question answering that serves as an assistance industrial process and assess the efficiency and accuracy of it. Consequently, a semantic question answering placed in a smart factory can use streaming data generated by production systems, statically generated data by OPC UA Servers or Semantic Tools such as eniLINK[1]. Nature of the semantic question answering might have different properties from open domain or closed domain question answering so that we will examine a restricted domain question answering system with linked data through our main architecture. Moreover, we present a semantic question answering with data based on eniLINK and generated data from an OPC UA Server known as Dynamic Server. In addition, we will evaluate our question answering with different measurements according to the requirements of a restricted-domain question answering.

***Index Terms*— Semantic Web, Web 2.0, Web Services, Information Retrieval**

I. INTRODUCTION

THIS work introduces a new concept for smart factories in terms of linked data processing integrated into a semantic question answering. The Semantic Web is a new research area that handles the semantical understanding data between human to machines or machine to machines. In a smart factory, connected sensors, actuators or manufacturing devices creates a massive amount of data.

As a result, a great amount of unlabeled data could not be used by applications, so W3C (World Wide Web Consortium) decided to create a standard for Semantic Web in order to apply linked open data concept. Fraunhofer IWU started to desägn its smart factories that are capable of creating structured linked data. A semantic question answering used for information retrieval to provide answers from questions through linked data. Our semantic question answering is able to understand complex natural language inputs and it can respond back to the user by answers. Mainly, a question answering system employs unstructured data or structured data. We take linked data generated by an OPC UA Server and eniLINK streaming data. The empirical analysis indicates the answer return rate and precision it evaluates the usability for a human operator, experts or an end-user of a web application.

The goal of this research is to show a model of semantic question answering for a smart factory utilizes the natural language inputs as sentences, questions or keywords to give a precise answer to human operators or experts. The paper organized as follows: Chapter 2 will give a brief overview of previous studies about algorithms for the question answering and evaluation criterion. Chapter 3 summarizes the requirements of semantic question answering aspect of the smart factory constructed by Fraunhofer IWU. Chapter 4 explains the status of Industry 4.0 and Smart Factories. Chapter 5 introduces the serialization process from the Information Model and streaming data to linked data. Chapter 6 and 7 introduces theoretical background and practical implementation respectively. As for Section 8, we will explain the test environment; accordingly, we give the results of the semantic question answering. Last two chapters will be relevant to Discussion, Conclusion and Future Works.

II. RELATED WORKS

[Diego Molla et. al.] reviewed a main characteristic of question answering in restricted domains is the integration

of domain-specific information that is either developed for question answering or disclosed for other purposes [2]. [Diego Molla, Jose Luis Vicedo] defined main characteristics of question answering system over limited domains, e.g. circumscription of question answering, the complexity of question answering, and practical usage of question answering[2].

The authors have compared between open-domain and restricted-domain question answering by figuring out key points. [Diego Molla, Jose Luis Vicedo] offers four clear-cut subjects such as the size of data, domain context, resources, and use of domain-specific resources.

[Sebastian Ferre] has published one of the detailed reports that express common pitfalls of natural language processing, essential points while consolidating SPARQL query and morphological definitions [3]. SQUALL is a solution for querying and updating RDF graphs by exploiting a controlled natural language that restricts grammar structures of a sentence in order to diminish complexities [3]. It has grouped all substantial features of a morphological language and pointed out what type of features in a natural language harnesses with regarding priorities and orders. The main contribution of SQUALL is categorizing ambiguities of natural languages and how turned out an advantage when using a controlled natural language [3].

[Payas Biswas et. al.] proposed an architecture that extracts precise answer for the given question [4]. They described the module distinctly and defined the types of questions that can be asked against the question answering.

The authors sketched a translation from their intermediate language to SPARQL to gain more accuracy with their system [3]. Template based solutions have been commented for restricted domain and open domain question answering systems. [Lehnmann Et al.] proposed a template based solution that produce a SPARQL template which directly mirrors the internal structure of the question [5]

Evaluation of a semantic question answering is still cumbersome and hard problem. Lack of test question that belongs to specific domain is one of the major problems. [Diekerma, Yılmazel & D. Liddy][6] offers different methodology from open-domain question answering while evaluating the restricted domain question answerıng. The authors specify the evaluation methodology as below [6]:

System Performance: Speed and Availability

Answers: Accuracy, Completeness

Display User Interface: Querying style, NL query, Keywords, Browsing, and a Question Formulation Assistance (Spell Checker, Abbreviation Solver)

The authors state that the TREC style Question Answering evaluation does not suit their restricted domain system so that user-based evaluation can be more viable in order to evaluate the system [6].

III. REQUIREMENT AND APPROACHES

Research Questions:

RQ 1) Can a semantic question answering utilize heterogeneous linked data source (e.g. OPC UA Information Model, streaming data, static data) in the domain of smart factory?
RQ 2) What are the requirements of the Semantic Question Answering for smart factories?
RQ 3) What are the main features associated with the methods of the Semantic Question Answering
RQ 4) Can we generalize our approach to other plants of and how did the research contribute to the research area?

RQ 1: Today, a smart factory creates massive amount of data by leveraging big data analysis technology. However, the data source suffers from comprehensible by applications. The question related to the implementation of a serialization process into linked data. This question evaluates the types of the data source by implementing solutions

RQ 2: The question related to algorithm design and domain-specific requirements to fulfill information retrieval and natural language understanding. This question has to evaluate the practical application.

RQ 3: This question assesses the pros and cons of our approach and gives the list of features of a semantic question answering in the domain of smart factory.

RQ 4: The question examines the viability of the proposal aspect of division of a plant or a smart factory. Generated new test parameters set to evaluate our semantic question answering. In the test phase, we will see how to generate the questions.

The rest of this paper is organized as follows: Section 2 describes requirement and approaches, Section 3 defines how we can prepare real-time data. In Section 4, we clearly explain the prerequisite methods in natural language processing. Thereafter, Section 5 clearly examines the proposed architecture. As a result, we conclude in Section 9.

We will answer the following research questions throughout the research in order to clarify key points.

At the end of this section, we might write our research questions.

IV. SMART FACTORIES AND INDUSTRY 4.0

The definition of smart factories has evolved over the past few years. In the present study, a smart factory has defined an aspect of boosted technologies named Industry 4.0 and Human-Machine Interface. Impact of manufacturing development affected economic growth over the last few decades in Germany. Continuously improvement of Industry 4.0 brought the researchers to find cutting-edge technologies such as Question Answering System, Manufacturing Augmented Reality etc.

A smart factory is a highly digitized and connected production facility that relies on smart manufacturing [7]. This concept one of the key outcome of Industry 4.0, which intelligently changes manufacturing technologies. Smart manufacturing is a term coined by a set of departments of the United States [8]. The central power of the smart factory is making data collection possible. Additionally, sensors enable the monitoring of specific processes throughout the factory that increases awareness of what is happening on multiple levels [9].

The development of Industry 4.0 has a big influence on the manufacturing industry. In the era of smart manufacturing systems, Industry 4.0 is a necessity that needs to standardize all communication structures in smart factories. The primary objective of Industry 4.0 makes the manufacturing technologies of factories more intelligent, optimizing the chain of processes and enhancing capabilities of communication one to another. Moreover, Industry 4.0 enforces end-to-end digital integration of engineering throughout the value chain to facilitate highly customized products, thus reducing internal operating costs [10].

V.LINKED DATA SERIALIZATION

Our main data sources are structured semantic data source. All data source has linked triples regardless of the type of semantic data such as Turtle, RDF or OWL.

A. The Meaning of data for OPC Unified Information Model

OPC UA was developed for devices of industrial internet of things to remedy problems about service orientation, loose coupling, and object-orientation paradigm. It evolved starting from OPC to OPC UA over the past few decades and architectural design entirely was changed. OPC was dependent on Component Object Model that should work with only Microsoft documents. The fundamental restriction of OPC is restricting devices to connect only a Windows-based operating system and there was no service orientation. After developing Distributed OPC and OPC UA idea, there has been constructed a viable concept for object-oriented, loose coupling and service orientation in a manufacturing system.

Aside OPC UA is being a complex protocol; OPC UA is one of the ubiquitous industrial communication protocol that can be used in the various stage of the manufacturing. Thanks to the OPC Client-Server architecture, any devices can connect to the protocol in a production system. A PLC, a sensor or actuator can connect to the same server and they can assign their values into different folder organization in order to represent data in an address space. The address space is a major data plane for an OPC UA Server, hence it should coordinate variable, methods, objects, and nodes respectively. An end-user can identify primitive and user-defined types so that the complex structure of devices can be represented as a whole in a big data plane. However, this data plane only provides definitions and types.

The Information Model support object-oriented paradigm such as abstraction and inheritance between References and Objects. It is well known that an object can live as a Node Class in the address space. The objects may have relationships with other objects in the information model. By means of References, a user can browse in address space to reach all level of nodes and variables. Nevertheless, neither the address space or nor the information model is far apart from understanding the meaning of data. Semantic understanding of the Information Model has a vital role to build up a question answering system. The Information model holds all device-specific information such as device type, data changes of the device, vendor type or relationship among devices. These information sources would be helpful to a human operator or expert who works with manufacturing systems.

B. Mapping an OPC UA Data into a Semantic Data

Our main data sources are semantically parsed data from eniLINK [11] and OPC UA Server in Fraunhofer IWU named Dynamic Server. In the phase of OPC UA Server generated data, we used an SDK which is published by FreeOPCUA [12][13]. We contributed to [12][13] with extra conversion steps such as XSLT and triple store processing.

OPC UA Protocol utilizes an information model and the information model can be used with metadata to simulate in other OPC UA Server with languages such as XML (Extensible Markup Language). Due to the nature of XML, it is a language depends on strong hierarchical elements and hardly extendable. However, semantic data such as Resource Description Framework (RDF) can employ triples with SPARQL. Different RDF Graphs stored in eniLINK[11] are uniquely identified.

Algorithm 1 Node Extraction

```
function MainFunction()                                   ▷ Starting point
    export = ServerExport(serverurl, filename)
    export.IMPORT NODES(serverurl)
    export.EXPORT FILE(outputFile, namespaces)
    export.XSLTCaller()
function Build Node Tree(nodes)                           ▷ Node Formatting
    client ← GetEndpoint()
    client ← Client(serverurl)
    nodecumulated ←None
    nodeID ←0
    for node < nodes do
        nodecumulated = node.nodeid.Namespaceindex
        for ref < node.getreferences() do
            nodecumulated.extend( ref.nodeid.Namespaceindex)
        nodecumulated = list(set(nodecumulated))          ▷ Clear duplicates
    return nodeID                                         ▷ Return node id list
function Import Nodes(serverurl)                          ▷ Traverse Node
    client = Client(serverurl)
    client.connect()
    for ns < client.getNamespaces() do
        namespaces[client.getNamespaceIndex(ns)] = ns
    root = client.getRootNode()
    child = client.iterateChildNodes()
function Export File(outputFile, namespaces = None)       ▷ Export into XML
    if namespaces != None then
        for node != nodes do
            if node.nodeid.namespaceindex is namespaces
                nodes = [node]
            else
                nodes = list(nodes)

        export = XmlExport(client) then
        export.BUILD NODE(nodes)
        export.appendXML(outputFile)
```

Figure V-1. OPC UA Information Model Serialization

The algorithm identifies tree elements of a node by taking the namespace index. Namespace index contains node ids. Once a user desired to browse from a node to another, he or she needs to know node identification number. If a user did not range the total number of reference, we should get all nodes that have references until we reach all elements of the mesh network. Accumulated nodes were inserted into a list to export as an XML format. After we obtained XML structures, we can convert the elements into linked data such as Turtle through Extensible Stylesheet Language Transformations. XSLT can transform from XML to RDF by minimizing the blank nodes. Once we converted to RDF/XML format, Graph libraries can deal with the conversion process into triple formats. The conversion application only take cares the uniform locator identifier to do so, and then we should consider it different from 'example.org'.

A. Real Time Data Mapping with KVIN

Real-time data has intricacies to use as linked data taken from sensors, actuators or software logs. In the aspect of Smart Factories, sensors and actuators that are the underlying structure of manufacturing machines mostly create continuous streamed data. Fraunhofer IWU collects the real data source by saving into a time series database. The major drawback is that when a time series data taken, semantic query endpoint cannot use the pure data without annotating. Such annotation could be triple creation, adding of predicates or serialization from one formal language to another.

Our architecture is providing a real-time semantic data annotator KVIN that utilize to extract triples from time-series data in a database.

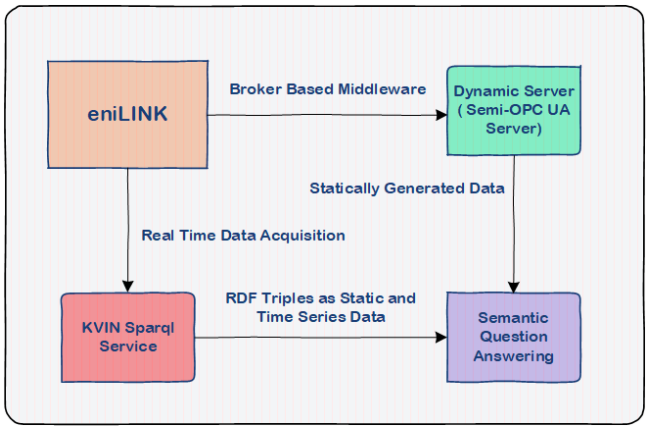

Figure V-2. KVIN Continuous SPARQL Mapping

This work proposes a service named KVIN to perform a SPARQL request into a specified endpoint. This service is based on a combination of the triple store through Level DB which is a key-value storage library written by Google [1]. It has been used an RDF4J's extension to create a SPARQL Service. After obtaining time series data, the data are mapping the SPARQL graphs. These graphs contain mapped triples with their time-stamped and value in order that we can employ values with some complex process with SPARQL Language. Moreover, a federated service replies to the queries determined by users, which is reducing the answer return time of a question answering system. KVIN does not create instant hard-coded triples or a new language such as C-SPARQL. It only arranges the size of the time window and put the graphs into service to present to the end user.

VI. THEORY OF THE NATURAL LANGUAGE UNDERSTANDING

In Natural Language Processing, we need to identify the sentence's structure in order to reach the step of Query Formulation. The following methods that we used in the practical application are concisely gıven.

Semantic Query Language: SPARQL was designed using for semantically structured triples, not for relational datasets. PREFIX, SELECT and WHERE are three basic operators of SPARQL Protocols. PREFIX makes the serialization steps easier referencing Uniform Locator. UNION statement can help at federating multiple triples into a single query. The OPTIONAL statement used for allocating a particular portion of SPARQL into triples. As can be seen, the main goal of the query language federating information among heterogeneous systems.

[1] https://github.com/google/leveldb

Preprocessing and Tokenization: Chiefly, all natural language tasks start with preprocessing which means cleaning data for specific tasks. This could be the reduction of undervalue data, reduction of discrepancies between the values or removing non-related morphological properties. A question answering system should parse all input as tokens. Tokenization is an initial step for Part of Speech tagging to parse into a verb, noun, cardinal numbers, adjective etc.

Lemmatization and Stemming: Lemmatization and Stemming are similar steps to each other with one difference. While stemming used to find syntactical structures, lemmatization looks for a semantic structure. Stemming clear of the structure of suffix and prefixes. In our system, we are supposed to use a lemmatizer and stemmer in order to reduce lexical complexities. A lemmatizer has been used to consider the morphological analysis of verbs, e.g. from “contains” or “contained” to “contain”. Then we use this verb mapping into a predicate to construct a SPARQL Query. The lemmatization and stemming are part of the normalization process in terms of linguistic properties.

Part of Speech Tagging: It is a preprocess step for parse tree to identify item taggers such as verbs, adjectives or nouns. A sentence consists of a couple of structure, including words like noun, verb, pronoun, preposition, adverb, conjunction, participle and article that are main categories of part of speech processing [14]. Part of Speech Tagger mostly uses a Markov Model that is a part of statistical natural language understanding. Markov Model stands for a state can depend on a previous step, but there is no dependency on states of historical steps more than one. For instance, a noun or a verb defines its neighbors, e.g. nouns are preceded by determiners, adjectives, verbs [14]. For example, a chess player makes a movement according to the last movement of a rival rather than guessing from the first movement of the rival. In this step, pre-saved corpora that have a massive amount of words have to be tagged by POS Taggers.

Penn Treebank: One of the common list that has an identifier for POS denominated as Penn Treebank. A Treebank used for annotating syntactic and semantic structure of a sentence with a million words of part-of-speech tagged text. When a natural query is given, a question answering system should understand the grammar behind it. POS tagger is not enough to identify the grammatical structure for complex natural queries. Relationships among noun phrases, adjective phrases, adverb phrases, and verb phrases should be examined in order to map correctly subject-predicate-object triples in linked data.

Parsing: The approach of parsing divided into two main sections, which are the rule-based approach and the probabilistic approach [15]. The rule-based approach is a top-down approach to solve problems via predefined rules such as Regex-parsing. Therefore, a question answering system should define rules precisely to get the correct answer.

Open-domain question answering systems use this approach because of the complexity of the bottom-up approach and broadened question types. Nevertheless, the rule-based approach could give undesirable results in restricted-domain question answering or semantic question answering so that this could be a time-wasting and an error-prone approach.

A dependency parser analyzes the grammatical structure of a sentence and it gives the relationship among them. The dependency parser also gives the relationship between general words and root words. Thus, we can identify the center verbs or nouns of complex sentences. This parser utilizes a dependency treebank file and word embedding files. Chiefly, a dependency parser applies the supervised machine learning method to reach a syntactical result. For example, with dependency treebank, data is broken into test and training set, however, word embedding used for the training phase.

A constituency (phrase) parser likely known as a phrase parser. The objective of this parser is to check the grammatical structure of sentences by parsing the chunks of morphological structure. The constituency parser may not handle the relationship among language items. Dependency parser analyzes the grammatical structure of natural input to define the relationship between the root word and the rest of them.

Named Entity Recognition: Named-Entity Recognition is a subtask of information extraction to locate and classify named entities with pre-classified labels such as names of people, organizations, locations, etc. Named-entity recognition is a method that identifies the item of a sentence as a domain-specific. It identifies all structures mainly as a person, a location, an organization, and an entity. It solves the problem of recognition in the same way that the chunking method does. However, named entity recognition may be trained with labeled data.

Similarity Analysis: Sentence similarity used for comparing two string inputs in order to achieve indicative questions like ‘Is the system health good?’. Mainly, this similarity method leverages averaging word vectors such as word2vec and glove that implements Euclidian, Manhattan Distances or Cosine Similarity. Three similarity methods we analyzed, which shown as below:

Levenshtein is that the calculation time could be O(|s1| x |s2|) using O(min(|s1|, |s2|))space. After calculating distance among s1 and s2, the result may be divided into maximum length of string [16]. Jaro Winkler has a transposition matrix t with common characters that are calculated together to reach similarity value [16]. Jaccard Similarity algorithm takes into consideration the size of intersection

divided by the size of the union of two sets [16]. Under the same test data and methods, similarity level of Jaccard, Jaro Winkler, and Levenshtein are 0.5652, 0.6699, and 0.51162 respectively. The higher score shows a better performance for similarity measurement.

In order to calculate word-based similarity, we are using WordNet with glove vectors. Such vectors are pre-calculated synset values stored into a file can be downloadable. These synset values show the similarity value with cosine similarity algorithm. WordNet can calculate the similarity of acronym and hypernym except for synonym. Calculation a semantic similarity is a hard and complex process. As we explained in the following scenario, two phrases such as 'Internet of Things' and 'Mesh Network' are semantically similar. One of them implies 'th e network of physical objects with electronics, software, sensors, and connectivity' and the latter implies 'the topology of a network whose components are all connected directly to every other component'. We cannot easily calculates this semantic similarity. Instead of calculating semantic similarity, we can calculate word vectors of verbs and nouns that relate to similarity synset. If a calculated synset value is over than the threshold value, a question answering can accept these two string similarly constructed. In the practical implementation, we have used verb synonym similarity to map onto <IRI: predicate> sets.

Question Classification: Questions should be categorized to get the correct answer. It is a part of question processing that can parse the question input and assign into the correct labels. Derivation of an expected answer can be defined by machine learning methods. This paper utilized Logistic Regression and Support Vector Machine for question classification phase. While the support vector machine was classifying the question with TREC Dataset (reference), the logistic regression examined the type of question at the Github repository[17]. Questions are grouped with coarse-grained labels, which are Abbreviation, Entity, Description, Human, Location, and Numeric. On the other hand, another dataset that we have trained with Logistic Regression that comprises of 'what', 'quantity', 'who', 'unknown', and 'why' labels. Logistic Regression and Linear Support Vector Classifications have supervised machine-learning methods by identifying coarse-grained question indicators with pre-trained labels. Logistic Regression estimates the parameter with a logistic function. The type of regression allows classifying multi-labels the afore-mentioned labels. Support Vector Machine aims to improve quality of hyperplane that separates multi-class labels. Linear SVC is such a method that implements a linear kernel function through Support Vector Machine. The Newton-cg has a gradient descent function that reduces the error rate each iteration to find out global minimum. The Limited BFGS is an optimization method over the Newton-cg. Logistic Regression Cross Validation (CV) applies a cross validation to training and test set by splitting at particular percentages between them.

Our result has been listed as in Listing VI-1.

Parameters	Precision	F1	Recall
Newton-cg	%95.55	%95.56	%95.57
Linear SVC	%92.75	%92.76	%92.77
Limited BFGS	%94.21	%94.22	%94.23
Logistic Regression CV	%95.63	%95.63	%95.64
Linear SVC for Li&Roth Taxonomy	%65	%45.5	%35

Listing VI-2: The evaluation of the Question Classification

VII. PROPOSED ARCHITECTURE OF THE SEMANTIC QUESTION ANSWERING

We are implementing a mixed parsing based approach in order to define essential elements of a natural query. Our priority is to detect <subject-predicate-object> triples and then mapping the verbs and nouns onto template SPARQL. This template created according to requirements of a smart factory. For instance, dynamic queries that fetch information from streaming data possibly need SUM, AVG, or MIN filter statement of SPARQL language.
As for static queries, we have hierarchical data and semantical data of the Information Model. Listing VII-1 shows an example hierarchical triple of the smart factory of Fraunhofer IWU. Such predicates <factory: contains> should parse and they need to be matched with verbs that we have parsed from natural input. However, this could lead us to a misconception to match synonym verb of predicates. Therefore, as illustrated in Figure VII-1, we have an extra step to identify the synonym of verbs.

```
<http://linkedfactory.iwu.fraunhofer.de/linkedfactory/linkedfactory/demofactory/machine10>
factory:contains
<http://linkedfactory.iwu.fraunhofer.de/linkedfactory/demofactory/machine10/sensor1>
```

Listing VII-2: Sample triples of eniLINK

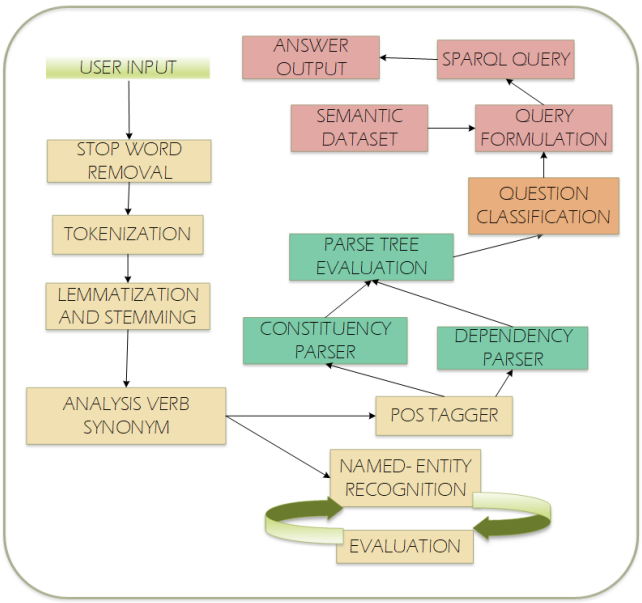

Figure VII-2. Natural Language Processing Steps for Question Answering

After taking input from any user, stop-word preprocessing stage start to filter unnecessary characters such as question mark, exclamation point, comma, dot or determiners. Tokenization is the next step in order to reduce the size of characters to provide optimization in natural language processing and it reduces the complexity of instances of sequence characters. Lemmatization and Stemming are fundamental steps before WordNet Verb analysis because our main target is to extract verb, nouns and related chunking in order to formulate a SPARQL query that can give an answer.

There is a control step for named-entity recognition after finding synonyms of verb. As previously explained, it is a way of extracting most common entities such as location or names. Names, locations or organization can face a problem about identifying domain-specific objects. For instance, linkedfactory can be comprehensible for Franhofer IWU's smartfactory, but another smart factory or different domain may not know what kind of entity is. Therefore, if we catch the entity-relationship pair as shown in Figure VII-3, we put them into a shallow and deep parsing steps.

For dynamic queries, the question answering system applies a similarity measurement. In Figure VII-4, similarity flag employs a sentence similarity in the following case. 'Is the system trouble' is a reasoning query. This query should be interpreted by the system and the system need to know exactly the semantic meaning of the sentence. However our approach the above-statement is a similarity-based identification. When a user a question like 'Is the system trouble for sensor1 in machine1?', the semantic question answering can interpret a reasoning question without understanding underlying semantic meaning.

Algorithm 2 Query Formulation

```
function QUERY FORMULATION(naturalinput)
    query ← QueryWithPrefixes
    r ← contituent.parse.tree
    indirectdependency ← dependency.parse.tree
    while nodes ≠ leafs.terminal do          ▷ Until leaf nodes(Terminals)
        verbs ← PARSER(nodes)
        nouns ← PARSER(nodes)
        similarityflag ← WORDLATENANALYSIS(verbs)
        if StaticInformation is True then
            indirectdependencyFlag ← DEPENDENCYPARSER(nodes)
            if similarityflag and IndirectDependency is true then
                object ←nouns
                predicate ←verbs
                query += object + predicate + ?subject
            else
                subject ←nouns
                predicate ←verbs
                query += ?object + predicate + subject
        if DynamicInformation is True then
            predicate ← PARSER(nodes)
            object ← PARSER(nodes)
            similarityflag ← SENTENCESIMILARITY(input)
            query += object + predicate + ?subject
    return query
```

Figure VII-5. Natural Language Processing Steps for Question Answering

The general information about our architecture is in Figure 1.1 below: The RDF data from eniLINK and OPC UA Server (right-hand side). A SPARQL Endpoint has been provided by our architecture for local static data and KVIN presents a SPARQL Endpoint for time-series data.

We are using different techniques for different question types. For instance, the natural query is the following:

'Could you tell me which one contains fofab?'

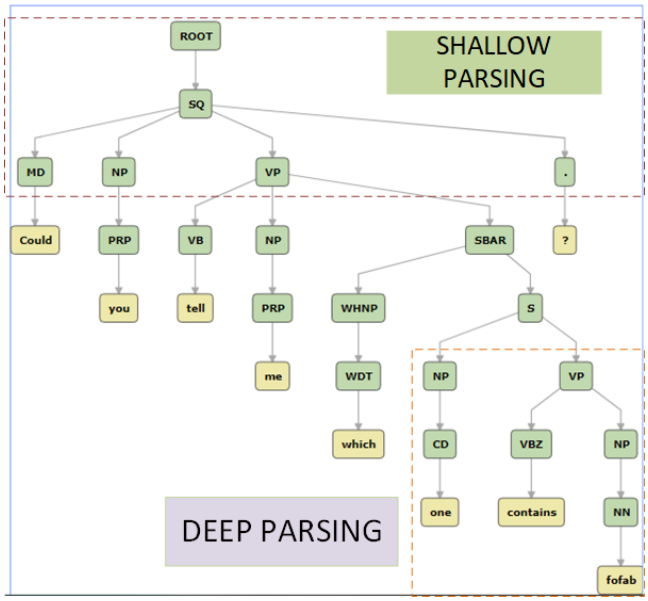

Figure VII-6: An example sentence from Stanford CoreNLP [18]

To identify noun and verb phrases at a basic level, a shallow parsing can make the constituency-parsing step easier. If we catch the right verb-noun pairs, we should eliminate phrases to reach the origin of the noun or verb. Such phrases may

represent a determiner, adjective or pronoun. As shown in Figure VII-7, we have two verbs that we need to map onto the predicate of triple in Turtle. If we may find out the similarity level of 'contains' and 'tell', the question answering could say the essential verb to be evaluated. However, the order of a verb is important for direct and indirect questions. As shown in Figure VII-8 and Figure VII-9, multiple objects have relationships with the head verbs 'tell' and 'contains'. Listing VII-3 triples can occur exact opposite. Subject and object can inverse the order of SPARQL query. In this case, we need to identify universal dependencies[2]. A named entity recognition can show these types of relationships as illustrated in Figure VII-10 and Figure VII-11. A drawback about this identification is being special keyword may perplex of the identifier, noun etc. In fact, a question answering system needs deeper analyzes to solve the perplexities of special keywords and open-domain words.

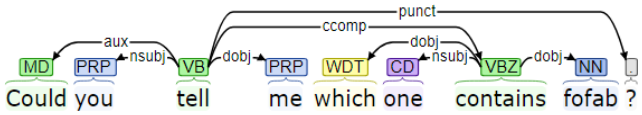

Figure VII-12: An indirect Query Dependency Parser[18]

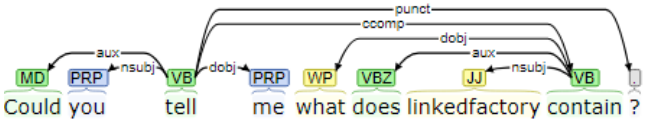

Figure VII-13: A direct query with Dependency Parser[18]

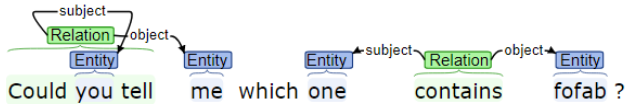

Figure VII-14: Named-Entity Recognition Result [18]

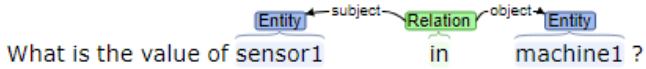

Figure VII-15: Named Entity Recognition with Open IE[18]

VIII.EVALUATION

A. Test Environment

In the evaluation phase, our main data sources are semantic data from OPC UA, eniLINK Hierarchical Data that consists of elements under the linkedfactory [19], and streamed data that resides in eniLINK. As previously detailed the process of serialization, we have a heterogeneous data source for the semantic question answering. OPC UA Generated Data has not specific namespace definition unless we define. However, the user-defined IRIs definition has drawbacks such as collision or non-extendibility. Longlisted linked data makes the structure complex so that two subjects of the list can be overlapped because of same-defined IRIs. In our case, all namespaces are generated with http://www.example.org" or "<unknown_namespace>".

The size of dataset that we generate from OPC UA Server has 19, 687, which is 2 MB sized Turtle File. The Linkedfactory triples relate to hierarchical structures has 70 triples as Turtle format.

Data Sets have been specified previously, so we are using the question answering with data set in a machine powered by Intel® Core™ i7-2720QM CPU @ 2.20 GHz, 2201 MHz, and x64 based Windows 10 Pro.

B. Result

Evaluation criteria exhibit recall; accuracy, precision, and F1 score of answers against semantic question answering system. General evaluation parameters for a restricted domain question answering is not only limited with answering of questions but also we can assess with speed, user interaction, querying style (keywords, browsing, spell checker, abbreviation recognition)

Precision = True Positives / (True Positives + False Positives)
Recall = True Positives / (True Positives + False Negatives)
F1-Score = 2 x Precision x Recall / (Precision + Recall)
Accuracy of the Model = (True Positive + True Negative) / (True Positive + False Negative + False Positive + True Negative)

Precision represents expected answers has been correctly predicted to the total answers. F1 Score is a balanced weight average between Recall and Precision. The recall is the proportion of correctly answered questions to the total amount of questions. Accuracy shows us the model that we have created has a ratio of correctly predicted observation to the total observation.

Test questions have been created with a combination of keywords and elements of sentences. Due to the domain restriction, the generation goal was answering precisely the questions ranging from keywords to complex natural input. The target data source was a mixed source that combines static and streaming data. In the appendix, one can observe combinations of test question to use further improvements

[2] https://universaldependencies.org/

Evaluation Parameters	Properties
Answer Return Rate	Generated Data from OPC UA – 39.88 second Consecutive Query of Generated Data 12.31 second Static query from RDF file of eniLINK – 19.33 second Template-Based Open-Domain Question Answering Query – 20.55 second
Querying Style	Keyword-Based Search and Semantic Question Search
Coverage	eniLINK data, streaming data, generated data from OPC UA
Size	Static data relatively small size Streaming data relatively large size
Up-to-dateness	No update statement provided by SPARQL
Query Formulation Assistance	Voice Input Recognition, Spell Checker

Listing VIII-1: The semantic question answering evaluation criterion

Question Answering Parameters	Total Questions
True Positive	34
False Negative	13
False Positive	3
Precision	%94.44
Recall	%72.34
F1 Score	%81.92
The accuracy of the Model	%68.00

Listing VIII-2: The Evaluation of the Question Answering

IX. DISCUSSION

In this chapter, we will discuss the significance of our findings that relevant to research problem being investigated. Taking into considering the findings, we will summarize insights about the problem

As a result, requirements 1, 2 addressed distinct architectures for the use of semantic question answering. Our proposal is implementing a service called KVIN that employs key-value mapping with windowed time series data. Windowing size can define the size of data that we can range. Although the information structure is limited to map onto Turtle triples, but it can be useful for rapid prototyping. There is no cost like designing a new language onto SPARQL or overhead of instant linked data creation from streamed data. Generating test data set still is a problematic topic for restricted domain question answering systems. For instance, atest data for IT domain is not valuable for a manufacturing domain. This restricts the testability however we have used the parameters of referenced research [6]. Our finding the answer return rate is similar to template-based open domain question answering [20]. If we want to get answer relevant to node id, parent id, references, and the connected devices to OPC UA Server, we should convert the Information Model to linked data. Broadly speaking, converting from the root node to leaf node with namespaces of nodes would be enough to map onto subject-predicate-object triples. The Semantic Question answering should give precise answers for dynamic data and list the results of answer against static data. Previous studies tried to solve the restricted domain question-answering problem with template based solution and implement a generic solution. Whereas, we implemented a domain based deep parsing without template-based to a particular domain.

By showing, the test results of the question answering and question classification, this study guide for the researchers of Industry 4.0 how to develop an advanced system. RQ-3 defines the main features of the semantic question answering in the smart factory domain, which are being short-listed answering, deep and shallow parsing based, and the ability to use of heterogeneous data source. Display interface may reduce the time that a human operator spends while typing and correcting spelling mistakes so that the efficiency of query processing may increase.

For a conclusion, generalization a question answering that belongs to a smart factory to other one is not logical solution. Algorithm and architecture generalization are possible, however the main drawbacks the special keywords in unstructured data and streamed data. This research contributed to research circle with algorithms, test set generation and features of a semantic question answering to be used against heterogeneous sources.

X.CONCLUSION AND FUTURE WORKS

Operator Assistance System increases the productivity of human operators and experts in smart factories. In this paper, we have proposed an application for restricted domain question answering that utilizes generated data from OPC Unified Architecture and streamed data. This application can reduce the total amount of time for searching through a large number of triples. The major findings are that the proposed novel approach can be used effectively to create a supervisor tool for manufacturing technologıes and synthesized theory caters a robust architecture for the aimed platform. Proposed model reduced the complexity of the normalization process and employs state-of-the-art natural language understandıng toolkits. The major problem of this proposal is question answering strictly depends on the predicates of data set that defined by the smart factory. In order to solve this problem, subject-predicate-object pairs can be recognized by deep learning methods with unstructured data. First finding is that

the named-entity recognition has shown poor performance than the parsing method aspect of identifying noun phrases and verb phrases. Second finding complex paragraph needs a complex mechanism such as coreference resolution to detect an object. Speed is another factor that we should infer when the point comes to customizable. Accordingly, a technical operator or expert cannot get an answer against streaming data in the constraints of a mission-critical system. The third finding is serializing OPC UA is a time-consuming task; moreover, there must be a control script to detect unchanged hierarchical part. Our proposal is that one can detect simulation data in OPC UA Server with a script to stave off the repercussion while serializing.

The last finding is creating a generalized algorithm could degrade the precision of answers but increase the scalability at the various department at a smart factory.

APPENDIX

Sample Question ID	Sample Questions
1	Provide me a combined result for IWU and e3sim
2	There is a member named fofab. Please give me all of its members
3	What POWERMETER holds?
4	I need to learn parent node id in generated data
5	Give me all data blocks
6	What is the value of average for the sensor1 in machine1?
7	I need to learn the maximum value of sensor5 in machine7
8	Give me all registered node id
9	Could you tell me the system health for sensor2 in machine7?
10	Could you browse in generated data?

ACKNOWLEDGMENT

Fraunhofer IWU supports this work and we would like to thank the group of HMMI on the account of financial support.

REFERENCES

[1] F. IWU, ‘eniLink’, 2015. [Online]. Available: http://platform.enilink.net/. [Accessed: 23-Nov-2018].

[2] D. Mollá and J. L. Vicedo, ‘Question answering in restricted domains: An overview’, *Comput. Linguist.*, vol. 33, no. 1, pp. 41–61, 2007.

[3] S. Ferré, ‘SQUALL: A controlled natural language for querying and updating RDF graphs’, *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7427 LNAI, pp. 11–25, 2012.

[4] P. Biswas, A. Sharan, and N. Malik, ‘A framework for restricted domain Question Answering System’, *Proc. 2014 Int. Conf. Issues Challenges Intell. Comput. Tech. ICICT 2014*, pp. 613–620, 2014.

[5] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano, ‘Template-based question answering over RDF data’, *Proc. 21st Int. Conf. World Wide Web - WWW ’12*, p. 639, 2012.

[6] A. R. Diekerma and E. D. Liddy, ‘Evaluation of restricted domain Question- Answering systems’, *Cent. Nat. Lang. Process.*, pp. 12–16, 2004.

[7] R. Margaret and D. Daniel, ‘Definition of Smart Factory’. [Online]. Available: https://searcherp.techtarget.com/definition/smart-factory. [Accessed: 05-Dec-2018].

[8] K. D. Thoben, S. A. Wiesner, and T. Wuest, ‘“Industrie 4.0” and smart manufacturing-a review of research issues and application examples’, *Int. J. Autom. Technol.*, vol. 11, no. 1, pp. 4–16, 2017.

[9] C. Team, ‘What is the smart factory and its impact on manufacturing?’, *13 June 2018*. [Online]. Available: https://ottomotors.com/blog/what-is-the-smart-factory-manufacturing. [Accessed: 05-Dec-2018].

[10] T. D. Oesterreich and F. Teuteberg, ‘Understanding the implications of digitisation and automation in the context of Industry 4.0: A triangulation approach and elements of a research agenda for the construction industry’, *Comput. Ind.*, vol. 83, pp. 121–139, 2016.

[11] L. D. Platform and F. IWU, ‘eniLink’. [Online]. Available: http://platform.enilink.net/. [Accessed: 23-Nov-2018].

[12] Pure Python OPC-UA Client and Server, ‘Free OPC-UA Library’. [Online]. Available: https://github.com/FreeOpcUa/python-opcua. [Accessed: 22-Nov-2018].

[13] TU Dresden, ‘Plt-TUD’. [Online]. Available: https://github.com/plt-tud/opc_ua_xml_export_client. [Accessed: 22-Nov-2018].

[14] D. Jurafsky and J. H. Martin, ‘Speech and Language Processing’, *Speech Lang. Process. An Introd. to Nat. Lang. Process. Comput. Linguist. Speech Recognit.*, vol. 21, pp. 0–934, 2009.

[15] J. Perkins, D. Chopra, and N. Hardeniya, *Natural Language Processing : Python and NLTK*. 2016.

[16] P. Christen, ‘A Comparison of Personal Name Matching: Techniques and Practical Issues’, *Sixth IEEE Int. Conf. Data Min. - Work.*, no. September, pp. 290–294, 2006.

[17] S. Khare, ‘Question Classification Implementation’. [Online]. Available: https://github.com/swapkh91/Question-Classification. [Accessed: 26-Feb-2019].

[18] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, ‘The Stanford CoreNLP Natural Language Processing Toolkit’, *Proc. 52nd Annu. Meet. Assoc. Comput. Linguist. Syst. Demonstr.*, pp. 55–60, 2014.

[19] Fraunhofer IWU, ‘Linkedfactory Intro Page’, 2018. [Online]. Available: http://linkedfactory.iwu.fraunhofer.de/linkedfactory/view. [Accessed: 19-Feb-2019].

[20] Machinalis Group, ‘Quepy Question Answering’. [Online]. Available: http://quepy.machinalis.com/. [Accessed: 27-Feb-2019].