Enhancement Semantic Prediction Big Data Method for COVID-19: Onto-NoSQL

K. ElDahshan, E. K. Elsayed, H. Mancy

Abstract— **Nowadays, the world suffers a Coronavirus mutation in 2019 (COVID-19). The COVID-19 data sources possess three main characteristics: big volume, velocity, and variety. These challenges have compelled the authors to employ Big Data technology, data mining techniques, and Ontology-based approaches instead of using a statistics hypothesis. Big Data Frameworks are involved in most data-related activities, such as storing, processing, analyzing, and sharing. Nevertheless, most of them miss having a semantic layer that is required for meaning-related activities, such as decision support, reasoning, and event detection. A semantic layer is based on Ontology, Semantic Query Engine, Association Rule Mining, and Fuzzy Logic. Therefore, this research aims to build Ontology and transform Big Data from Big Data Frameworks to semantic environments. This paper presents Onto-NoSQL, a Protégé plug-in that supports the creation of Ontology and transformation of a column-oriented NoSQL datastore like Hbase into Protégé. Besides, the whole process of transformation is carried out automatically without any external intervention. It demonstrates the proposed plug-in through a case study of air pollution and weather phenomena's data. The proposed plug-in is utilized to predict COVID-19 prevalence and the relationship between COVID-19 prevalence and weather factors. Moreover, the plug-in handles many challenges due to Big Data size and time processing. The time consumption to import up to 64 GB data is 17 minutes. The data prediction accuracy is 96.9% applying association rules discovery on Ontology creation.**

Keywords— ***Big Data, COVID-19, NoSQL Database, Ontology***

Manuscript received May 03, 2020; revised July 20, 2020.

This research is funded by the Academy of Scientific Research and Technology (ASRT), Cairo, Egypt, project titled "Coronavirus Prevalence Prediction Model" (Project ID: 6641).

Kamal ElDahshan is a Professor of Computer Science and Information Systems at Al-Azhar University in Cairo, Egypt; e-mail: dahshan@gmail.com.

Eman K. Elsayed is a Professor of Computer Science; she obtained a Ph.D. in Computer Science in 2005, Al-Azhar University, and a Master degree of Computer Science and Information Systems at Al-Azhar University in Cairo, Egypt; e-mail: emankaran10@azhar.edu.eg.

Hend Mancy is an Assistant Lecturer at the Computer Science Department, Science Faculty, Al-Azhar University, Cairo, Egypt; e-mail: dr.hendfathi@azhar.edu.

I. Introduction

Computer scientists are monitoring the deadly Coronavirus outbreak. They work diligently to track the next movements of the virus. Around 367,153 people have died from the novel virus, which triggers a respiratory disease dubbed COVID-19. It first appeared in the Chinese town of Wuhan in December and has since infected more than 5,934,195 people till the end of May 2020 all over the world. The World Health Organization has classified this as a pandemic [1].

Some scientists used the classic Susceptible-Exposed-Infected-Removed (SEIR) model to predict the spreading of Coronavirus Disease 2019 (COVID-19) in China [2], [3]. Others used three kinds of mathematical models, i.e., Logistic model, Bertalanffy model, and Gompertz model, to predict and analyze Coronavirus Disease 2019. The fitting effect of the Logistic model may be the best among the three models [4].

All previous studies depend only on historical and observation data. These studies do prediction using statistical and mathematical models. The data size in these studies is marginal. These studies include one or two dimensions of the data only. The world needs a more efficient and accurate prediction method that allows studying historical and real-time data of the world. The new method should be more flexible to import all the data related to the study issue and easy to use. The new method should allow linking the types of diseases with the reasons for their spread. The *new method* should be able to *study* factors affecting Coronavirus spread like weather and migratory birds.

Ontologies and Ontology-based solutions have a wide range of applications, including semantic integration, support for decisions, search, and annotation. They support the process of handling heterogeneous data, as well as access to relevant data, because of increasing data. Besides, they offer a broad spectrum of applications in the context of Big Data [5].

Not only SQL (NoSQL) databases usually lack a clear, unambiguous system, making it more difficult to grasp their contents and harder to implement and evaluate them. The use of semantic web technologies is needed to enrich the contents of the NoSQL database by adding meaning and context [6].

It is often important to transform Big Data from NoSQL databases into ontologies, particularly when Ontology is used to define the data used by a software application on a semantic basis. Another category of applications requires Big Data-Ontology integration and/or interoperation, where

the main focus is on mapping Big Data with less schema structure or schema to Ontology concepts.

Eldahshan et al. developed a Semantic Smart World Framework (SSWF). It is a general semantic Big Data Framework for a smart world. SSWF added millions of records of Resource Description Framework (RDF) triples in the Big Data framework. SSWF provided a universal knowledge base for data generated by different data sources. SSWF can run semantic queries on the Big Data Framework [7]. This research aims to take the opposite direction of SSWF. The authors develop Onto-NoSQL as a Protégé plug-in that allows the user to import NoSQL Big Data into Protégé. The plug-in also supports creation Ontology from a schema-less database. The plug-in supports Ontology based data access, conversion, and mapping Big Data.

To achieve the research objectives, this paper is structured as follows: Section II reviews the background. Section III presents the previous related works. Section IV illuminates the proposed approach for plug-in architecture. Section V describes the implementation of the authors' proposed approach. Section VI provides multiple case studies: one for air pollution and weather phenomena's Big Data and another for COVID-19 data and global weather data. Section VII explains the results' analysis of applying the proposed approach. Section VIII discusses the significant contribution and limitations of this research and concludes the paper.

II. Background

The term Big Data refers to a concept nowadays generally used to classify very large amounts of data. Big Data can be defined as "the amounts of data that are beyond the ability of technology to effectively store, manage, and process." Big Data can be characterized as data that exceeds the processing capacity of traditional database systems according to Gupta and colleagues. It means that the data count is too high, and/or data values change too quickly and/or do not obey traditional database management systems guidelines [8].

The relational databases are not supposed to address the issues of size and agility relevant to Big Data applications. NoSQL (Not only SQL) databases not only meet these requirements but also take advantage of commodity hardware and immense processing power [9].

NoSQL databases allow large quantities of structured, semistructured, and unstructured data to be stored. These also provide high-speed and very versatile access to semistructured and unstructured data. There are four types of NoSQL databases: key/value stores, family column stores, document-oriented databases, and graph databases [9], [10].

Followed by Apache Cassandra, Redis, and Hbase, MongoDB is the most common NoSQL database. Neo4j and Amazon DynamoDB, respectively, are the most common NoSQL graph database and cloud database [11].

Hbase [12] is a repository of structured data that is scalable, distributed, and column-oriented. Hbase is a Hadoop ecosystem component. Hbase data is modeled as a multidimensional map with four keys indexing values. The four keys are TableName, RowKey, ColumnKey, and Timestamp. ColumnKey is a combination of column family name and column name. Each column family may have an arbitrary number of columns but during table creation, the column families are fixed. Therefore applications can create new columns on the fly dynamically.

Ontology is characterized as a mutual conceptualization formal, an explicit specification. Where formal means that the Ontology is readable by machine explicit means that the category of concepts should be specified, and conceptualization refers to an abstract model for any real-world phenomenon.

One of the Ontology benefits is that it can cope with the complexity of the real-world and adapt to the changes. It can also be easily expanded, and it promotes sharing and reuse of knowledge.

According to [13] there are *common* components of ontologies:

- *Individuals*. They are instances or objects which refer to the basic components of the Ontology and may be a concert or abstract individuals.
- *Classes*. They are the elements of Ontology that denote the collection of instances, and they are also called concepts, entity types. Classes may be for classifying individuals, other classes, or a combination of both.
- *Attributes*. They may be aspects, properties, features, characteristics, or parameters, which objects and classes can have.
- *Relations*. They are ways in which classes and individuals can be related to one another. They are also called associations, roles, relationship types, object properties.
- *Axioms*. They are assertions including rules that are in the logical form and together comprise the overall theory that the Ontology describes in its domain of application.

Ontology has three types of properties, which are the following:

- *Object Properties*. They relate individuals to other individuals.
- *Datatype Properties*. They relate individuals to literal data such as strings, numbers.
- *Annotation Properties*. They are used to add information to classes, individuals, and other type properties.

There are many Ontology languages such as Ontology Web Language (OWL), Resource Description Framework (RDF) (S), and DAML+OIL, but they are all based on First Order Logic (FOL) [13], [14].

The RDF is used to provide any real-world entity with a digital representation [14]. Each entity of the real world will have certain attributes, whose values will distinguish it from other entities of the same group. These attributes are referred to as properties, and the property's value is referred to as the property value. The properties and values of the property are treated again as things. RDF uses features provided by XML to represent resources, properties, and relationships. The "thing," its "property," and the "value of the property" are called the triple RDF.

There are several editors of open-source commercial Ontology in science and practice. For example, Protégé, TopBraid Composer, and OntoStudio3 are well-known and

widely distributed editors. Protégé is probably the most widely known and used Ontology editor. It offers different plug-ins. It offers extensive features including a visual representation of the Ontology created [15].

III. Related works

In this section, the authors present and analyze the existing literature focusing on importing and integration of Big Data to semantic view. Csongor Nyulas, Martin O'Connor, and Samson Tu presented DataMaster, a Protégé plug-in that supports the import into Protégé of schema structure and relational database data. The plug-in supports ontologies based on both OWL and frames and can be used with any JDBC/ODBC driver relational database [16]. The authors presented the four ontologies that can be used to represent the database structure and table data.

Hanen Abbe and Faiez Gargouri Proposed an automated Ontology training approach that takes OWL Ontology from a document-oriented MongoDB database. The authors defined a tool that applies the rules of transformation [17].

Craig A. Knoblock and Pedro Szekely described how we use semantics to address the Big Data variety issue. The authors also explained Karma, a system that implements their approach, and demonstrated how Karma can be used to integrate data into the domain of cultural heritage. In this use case, although the data sets from various museums are extremely heterogeneous, Karma incorporates information across many museums [18].

Antonio M. Rinaldi and Cristiano Russo presented a comprehensive approach to integrating very extensive multimedia knowledge bases using knowledge representation formalisms as ontologies. The proposed solution is assisted by a computer system capable of creating a global Ontology through the incorporation of existing ones that automate the entire Ontology construction process [19]. There is a comparative analysis of Protégé plug-ins performances which converts databases into ontologies [20].

SungYoung Lee et al. presented the DTD2OWL framework that allows the automatic mapping of Document Type Definition (DTD) to OWL domain knowledge and transforming XML instances into OWL individuals [21].

Yuangang Yao et al. proposed a method that can transform the associated JSON data sets into OWL ontology automatically. Using semantic reasoning, this method can extend the data semantics and combine multiple related resources into a unique Ontology [22].

All of them except [18] do not handle Big Data. Although [17, 18] accept NoSQL databases, [17] converts schema only and [18] integrates data only without schema conversion. A new approach handles Big Data, accepts NoSQL databases, and allows schema conversion is needed.

Table I shows the comparison among our approach and related works through input and output of different approaches and data size.

TABLE I. The comparison among our approach and related works.

Approach	Input	Output	Data size
[16]	Relational database data	OWL schema and data	Small (6400kb)
[17]	MongoDB	OWL Ontology	(6400kb)
[18]	Relational database data/NoSQL/ OWL Ontology	Integration of data	Big Data (up to 20 GB)
[19]	Light-weight ontologies	Integration schema	NA
[21]	DTD/XML Schema	OWL	NA
[22]	JSON	OWL	NA
Onto-NoSQL	NoSQL DB Hbase	OWL Schema/data	Big Data (up to 64 GB)

IV. The proposed Onto-NoSQL plug-in architecture

Although Ontology creation is not a new area of research, Big Data faces new challenges because of its characteristics (velocity, variety, and volume). Therefore, our approach to Big Data Ontology creation includes taking into consideration Big Data characteristics.

The key idea to support the Ontology creation from huge numbers of heterogeneous sources is to create a schema from schema-less and import Big Data to it. A user may need to automatically generate Ontology and import Big Data, but the authors' aim is to simplify this step as much as possible with an easy interface and good visualization. The other important contributions of this research are (1) the ability to support Ontology based Big Data access layer, (2) the approach to learning transformations rules to build Ontology, (3) the ability to import different sources of Big Data using NoSQL, (4) the capability to easily build a semantic query over Big Data, and (5) capability to merge different ontologies and reengineering them.

Fig. 1 shows the proposed Onto-NoSQL plug-in structure against SSWF.

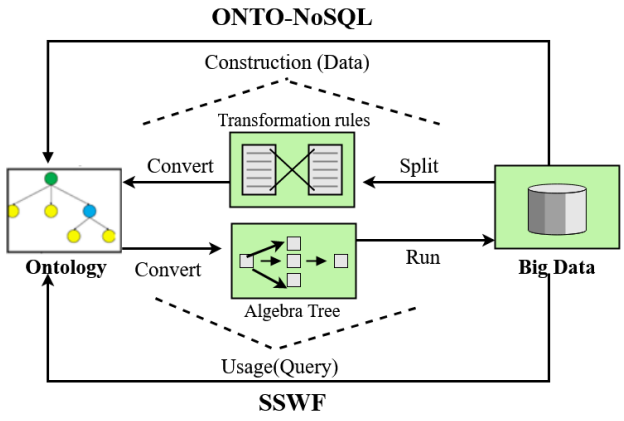

Fig. 1. Onto-NoSQL plug-in structure.

The proposed Onto-NoSQL plug-in is a structure like the following phases:

A. *Preprocessing Phase*

In this phase, the authors build Ontology based Big Data access layer that allows users to connect to Big Data environment through the Ontology environment. A user can query datasets, regardless of whether those datasets are integrated and stored on a Big Data environment.

Also, this phase generates a schema of Big Datasets. No explicit schema holding data structures exist in NoSQL datastores such as Hbase. Therefore, the schema must be generated for datastores given that there are millions of individuals in the store and there is no restriction on the number attributes for each individual.

The information on the metadata table is converted to suitable OWL primitives and mapped to OWL Ontology. Once the NoSQL datastore schema has been obtained, the following rules can be applied to convert the obtained schema to OWL Ontology.

- The table name is converted to the OWL class.
- Column family and column are converted to OWL subclass.
- Column Information. Column value is converted to a data type property named "family column _column".
- The individuals in the datastore are considered as instances of class formed by table name.
- The data types of the column are considered as domain and range.
- The column constraints are mapped to property cardinalities.
- The relation is converted automatically to is-a or has relation.

Table II shows the mapping objects from the NoSQL database to Ontology.

TABLE II. THE MAPPING OBJECTS FROM NOSQL TO ONTOLOGY.

NoSQL Objects	Ontology Objects
Table name	OWL class
Column family/column	OWL subclass
Column value	Datatype property
Individuals	Instances
Data types	Domain and range
Column constraints	Property cardinalities
Relation	Relation automatic is-a or has

B. Processing Phase

In this phase, the authors extended HDFS (Hadoop Distributed File System) and MapReduce interface for Big Data by the semantic layer. The authors depend on an efficient algorithm, namely, Semantic Ontology Retrieval (SOR), which used a Hadoop platform to retrieve multimedia Ontology from a variety of Big Data domains [23]. The authors use a Semantic Map-Reduce bulk extraction technique to extract data from Hbase and SHDFS (Semantic Hadoop Distributed File System) to store data. Like HDFS has Name Node and Data Node, SHDFS has OWL schema and RDFs. The data is huge and dynamic in datastores such as Hbase. Thus, it is not prudent to move an entire Hbase table to one RDF because it involves multiple large tables. Due to the restriction of RDF size which opens on the knowledge base system, the authors create a Semantic Map-Reduce bulk load to import some individuals with the user criteria specification. The created Ontology can easily merge with other ontologies on the plug-in. Fig. 2 shows the proposed Onto-NoSQL plug-in phases.

V. THE ONTO-NOSQL PLUG-IN IMPLEMENTATION AND INTERFACE

The plug-in is implemented as a tab in Protégé version 3.4. This work is directed using the JAVA programming language. The authors use Cloudera CDH as a Big Data environment. They use ten eight-core compute nodes with 64 GB memory from the HPC (High Performance Computing) System of Bibliotheca Alexandrina which has a SUN cluster of peak performance of 11.8 Tflops, 130 eight-core compute nodes, 2 quad-core sockets per node, each is Intel Quad Xeon E5440 @ 2.83GHz, 8 GB memory per node, Total memory 1.05 Tera Bytes, 36 TB shared scratch, node-node interconnect, Ethernet & 4x SDR Infiniband network for MPI, 4x SDR Infiniband network for I/O to the global Lustre filesystems. Fig. 3 explains the workflow diagram of Onto-NoSQL plug-in.

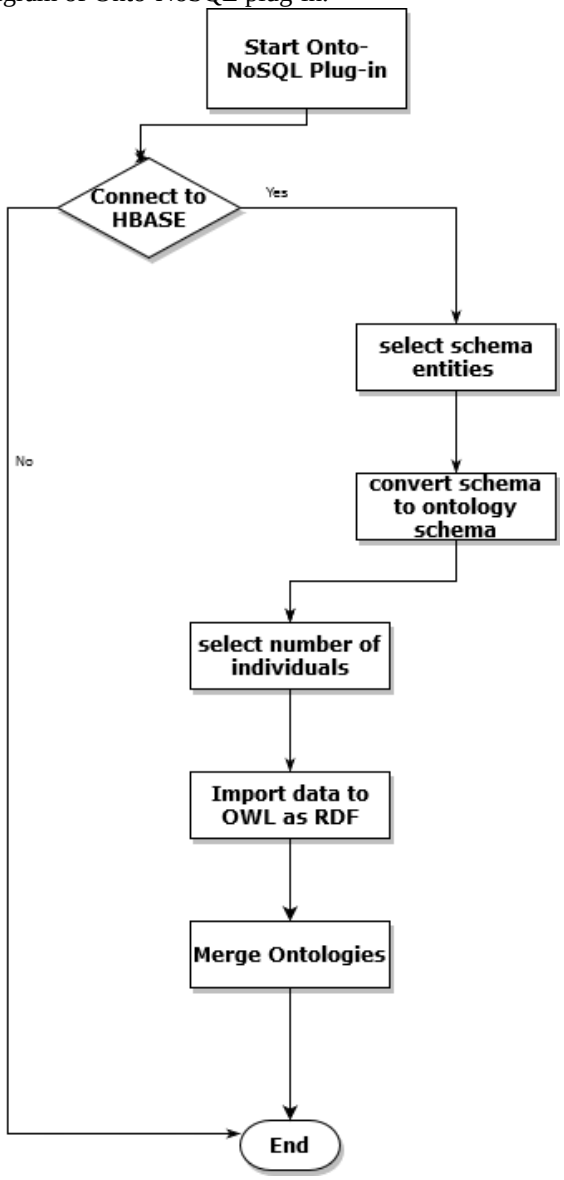

Fig. 3. Workflow diagram for Onto-NoSQL plug-in.

The phases of the proposed Onto-NoSQL plug-in are implemented as follows:

A. Preprocessing Phase

To access the Hbase database, the authors used the Hbase-client version 1.1.1 and phoenix-core version 4.4.0. They use the following code to connect with Hbase. In the code, they read the path of the Hbase configuration file and read the Hbase configuration file (hbase-site.xml).

```
Configuration configuration = HBaseConfiguration.create();
String path = this.getClass()
.getClassLoader()
.getResource("hbase-site.xml")
.getPath();
config.addResource(new Path(path));
```

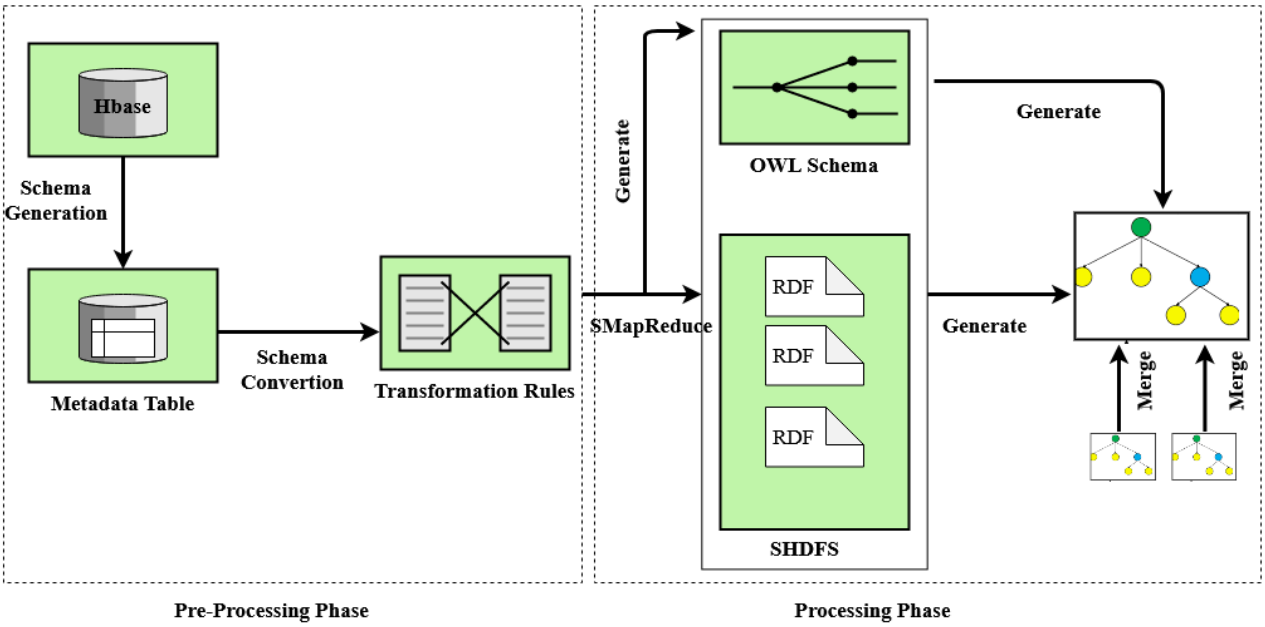

Fig. 2. Onto-NoSQL plug-in phases.

The following code is used to scan all family columns in the qualifier variable and columns in the Hbase database and create a Metadata table.

```
Scan sc = new Scan();
scan.setFilter(new FilterList(Operator.MUST_PASS_ALL,
qualifier));
try (ResultScanner scanner = table.getScanner(sc)) {
   for (Result result : scanner) {
      System.out.println("row: " + result);
   }
}
```

The authors' plug-in allows users to select some family columns and columns before Ontology creation and press the Convert button to create Ontology.

B. Processing Phase

To perform the Ontology creation, the authors used the OWL-API version 4.0.1. They use the following code to create Ontology and its hierarchy.

```
String reasonerFactoryClassName = null;
      OWLOntologyManager manager =
OWLManager.createOWLOntologyManager();
      IRI documentIRI = IRI.create(args[0]);
            OWLOntology Ontology = manager
        .loadOntologyFromOntologyDocument(documentIRI);
      System.out.println("Ontology created...");
         SimpleHierarchyExample simpleHierarchy = new
SimpleHierarchyExample(
            (OWLReasonerFactory)
Class.forName(reasonerFactoryClassName)
                  .newInstance(), Ontology);
      OWLClass class1 =
manager.getOWLDataFactory().getOWLThing();
      simpleHierarchy.printHierarchy(class1);
```

There are limitations in importing Big Data in one Ontology file. So, the authors' plug-in allows users to determine number triples to be in a separate file before the press import button. The maximum file size is 200MB. After schema generation and mapping data, the plug-in allows users to merge another Ontology with the new one. They use the following code to merge ontologies:

```
public Merger(ArrayList<Ontology> ontologies,
OWLOntologyManager man, String filename){
      this.ontologies = ontologies;
      this.man = man;
      this.fileName = filename;
      merger = new OWLOntologyMerger(man);
      mergeOntologies();}
private void mergeOntologies(){
      IRI mergedOntologyIRI = IRI.create(filepath + fileName);
      for(Ontology ontology : ontologies){
         try{
man.loadOntologyFromOntologyDocument(ontology.getFile());}
catch (OWLOntologyCreationException e) {
            e.printStackTrace();}}
            try {
         mergedOntology = merger.createMergedOntology(man,
mergedOntologyIRI);
      } catch (OWLOntologyCreationException e) {
         e.printStackTrace();}}
```

Fig. 4 shows this sequence diagram of Onto-NoSQL plug-in.

A screenshot of the Onto-NoSQL plug-in is shown in Figure 5. The graphical interface consists of several components. “Connection panel” in the upper left allows users to connect on the Hbase database. Pressing the “Connect” button will open a connection to the database and activate the “schema selection panel.” Schema entities in the “schema selection panel” come from the schema generation phase. The user can select the column families and columns to be imported into Protégé. The user can select “Convert” to run Conversion schema to the Ontology phase. Finally, the user can write the number of triples to import in separate RDF files and press the import button.

VI. Case Studies

A. Weather and Air Pollution

The authors apply plug-in on air quality [24] and weather forecasting [25] monitoring data for 40 European countries from 1969 to 2012. The size of the data per year is 1.5GB. Air pollutants included particulate matter with an aerodynamic diameter ≤ of 10 μm (PM10), PM2.5, nitrogen

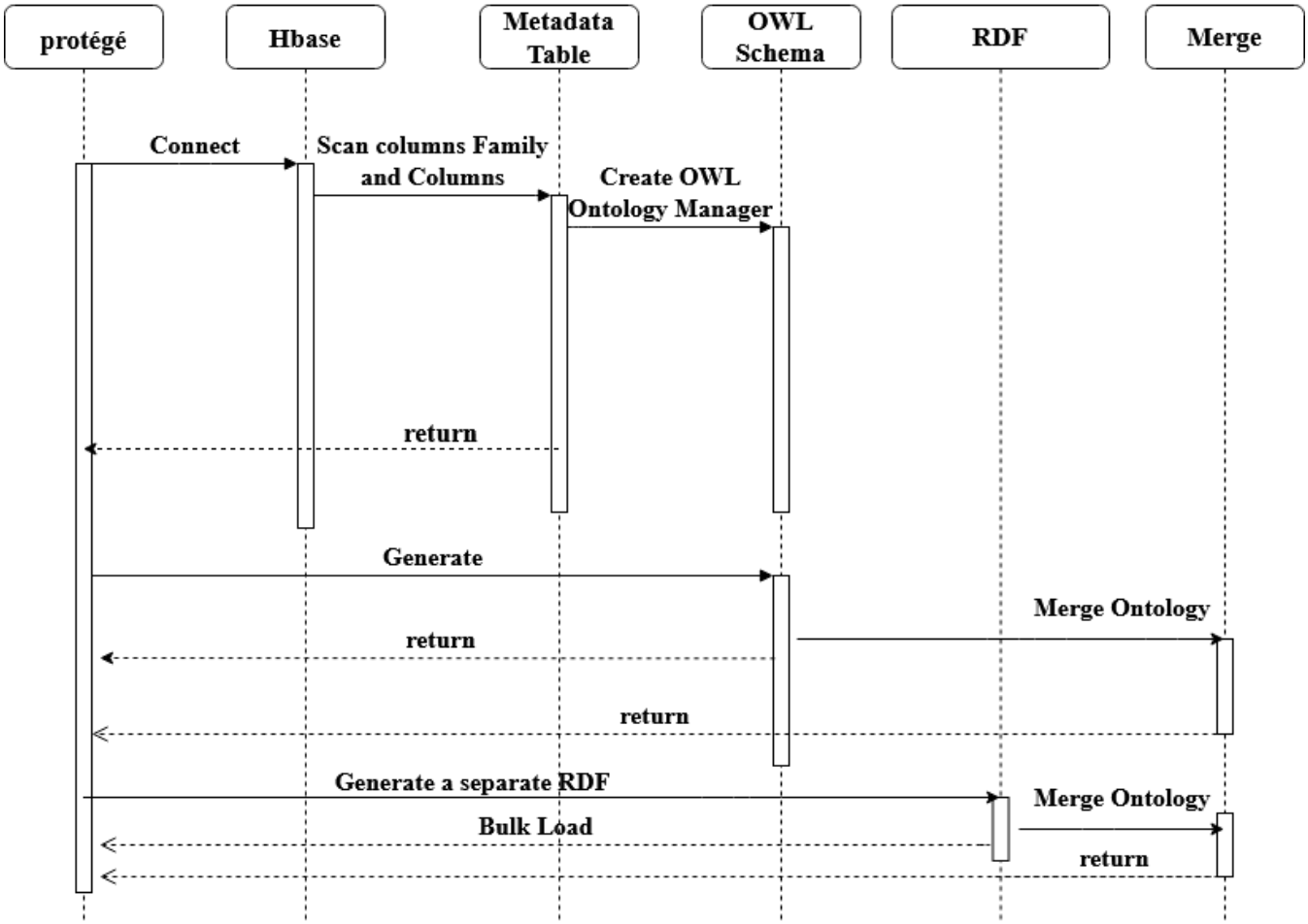

Fig. 4. Sequence diagram for Onto-NoSQL plug-in.

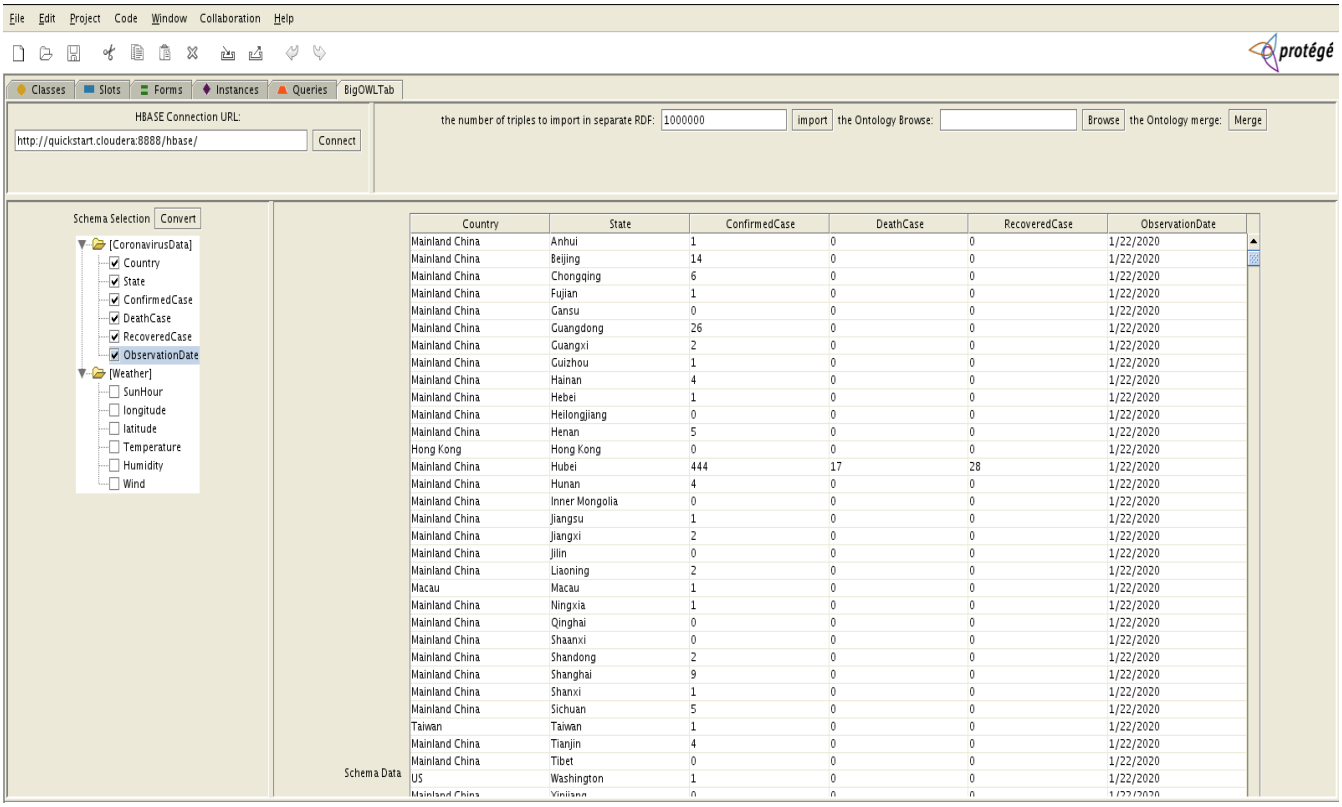

Fig. 5. The Onto-NoSQL plug-in screenshot.

dioxide (NO2), sulfur dioxide (SO2), carbon monoxide (CO), and Ozone (O3). Multiple weather factors (temperature, wind speed, humidity, rainfall, etc.) are taken into consideration based on hourly monitoring. The datastore in Hbase database serves as the environmental information domain. It has *air* quality column family and *air* pollution parameters in columns value. It has weather column family and weather parameters in columns value. The logical view of the air quality column family is shown in Table III. The logical view of the weather column family is shown in Table IV. After connection to Hbase database, the authors use the number of triples to import in separate RDF file as 1000000 triples, which is the maximum number to import in the separate RDF file. The time consumption to import 64 GB is 17 minutes in 330 separate RDF files. The OWL Class Hierarchy of environmental Ontology is presented in Fig. 6.

TABLE III. HBASE TABLE STRUCTURE AND DESIGN FOR AIR QUALITY COLUMN FAMILY.

Column family	Columns
CF: air quality	CO O3 NO2 SO2 PM10 PM2.5 CO2 CH4

TABLE IV. HBASE TABLE STRUCTURE AND DESIGN FOR THE WEATHER COLUMN FAMILY.

Column family	Columns
CF: weather	Temperature Humidity Rainfall Precipitation Wind Solar-irradiance

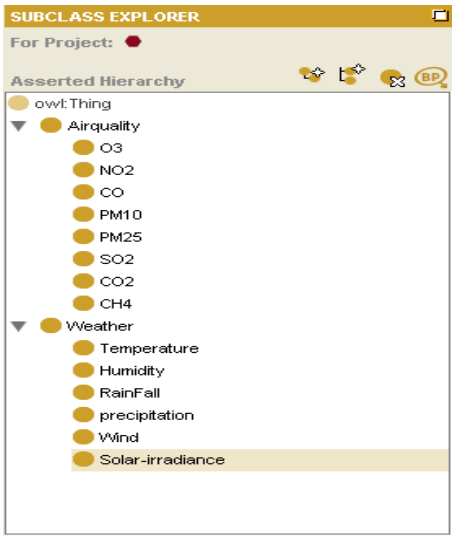

Fig. 6. The OWL Class Hierarchy of environmental Ontology.

Fig. 7 visually describes the mapping of columns in Hbase to data type properties.

B. *Weather and COVID-19*

The authors apply plug-in on COVID-19 data [1] and weather forecasting [26] monitoring data all over the world. They merge them with the Coronavirus Infectious Disease Ontology (CIDO) [27] that contains etiology, transmission, pathogenesis, diagnosis, prevention, and treatment for Coronavirus. COVID-19 data included confirmed, deaths, and recovered cases per country and state on daily basis. Multiple weather factors (temperature, wind speed, humidity, etc.) are taken into consideration based on daily monitoring. The datastore in Hbase database serves as the environmental information domain. It has CoronavirusData column family and COVID-19 parameters in columns value. The logical view of the CoronavirusData column family is shown in Table V. The logical view of the weather column family is shown in Table VI. The OWL Class Hierarchy of COVID-19 Ontology is presented in Fig. 8. Fig. 9 visually describes the mapping of columns in Hbase to data type properties.

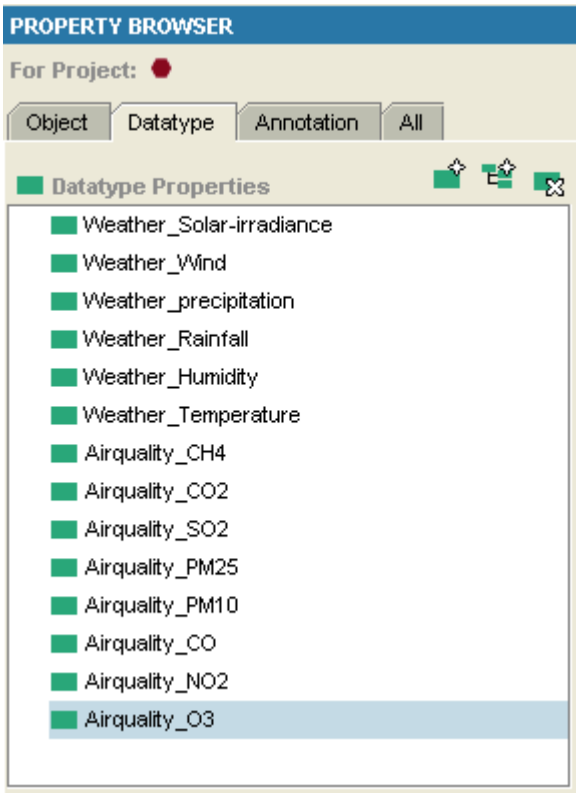

Fig. 7. The data type properties in environmental Ontology.

TABLE V. HBASE TABLE STRUCTURE AND DESIGN FOR CORONAVIRUSDATA COLUMN FAMILY.

Column family	Columns
CF: CoronavirusData	Country State Confirmed case Death case Recovered case Observation date

TABLE VI. HBASE TABLE STRUCTURE AND DESIGN FOR THE WEATHER COLUMN FAMILY.

Column family	Columns
CF: weather	Longitude Latitude Temperature Humidity Wind Sun-hour

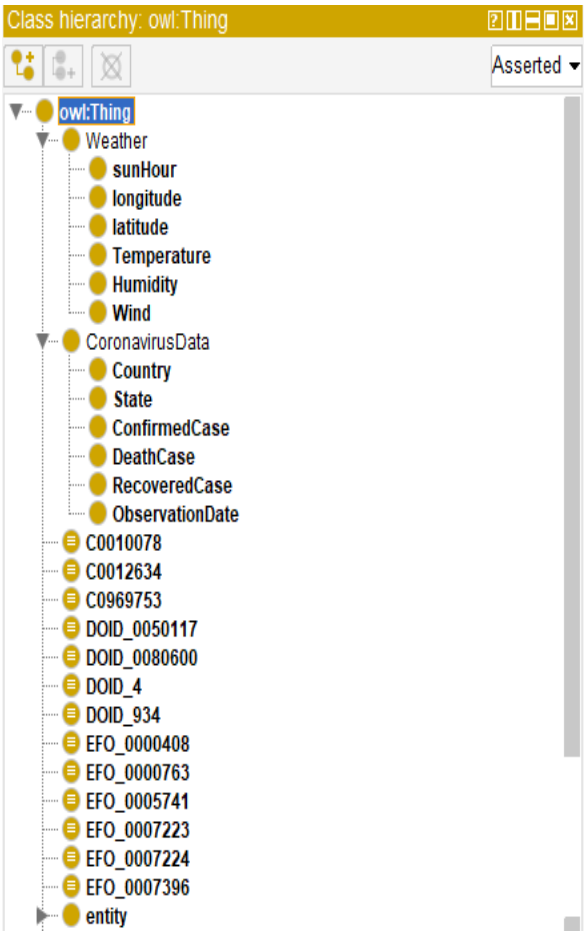

Fig. 8. The OWL Class Hierarchy of COVID-19 Ontology.

VII. RESULTS' ANALYSIS

A. Weather and Air Pollution

For the evaluation of the proposed plug-in, the authors measure the time processing of Onto-NoSQL plug-in against SSWF. Table VII shows the comparison between processing periods of time of the data (seconds) for SSWF and Onto-NoSQL plug-in and a different period.

TABLE VII. THE COMPARISON BETWEEN PROCESSING PERIODS OF TIME OF THE DATA (SECONDS) FOR SSWF AND ONTO-NOSQL PLUG-IN AND A DIFFERENT PERIOD OF TIME.

Time	SSWF Time of processing (seconds)	Onto-NoSQL plug-in Time of processing (seconds)
1 Year	14.1	23.3
10 years	30.5	100.1
20 years	45.3	400.4
30 years	62.4	700.6
40 years	75.2	1020

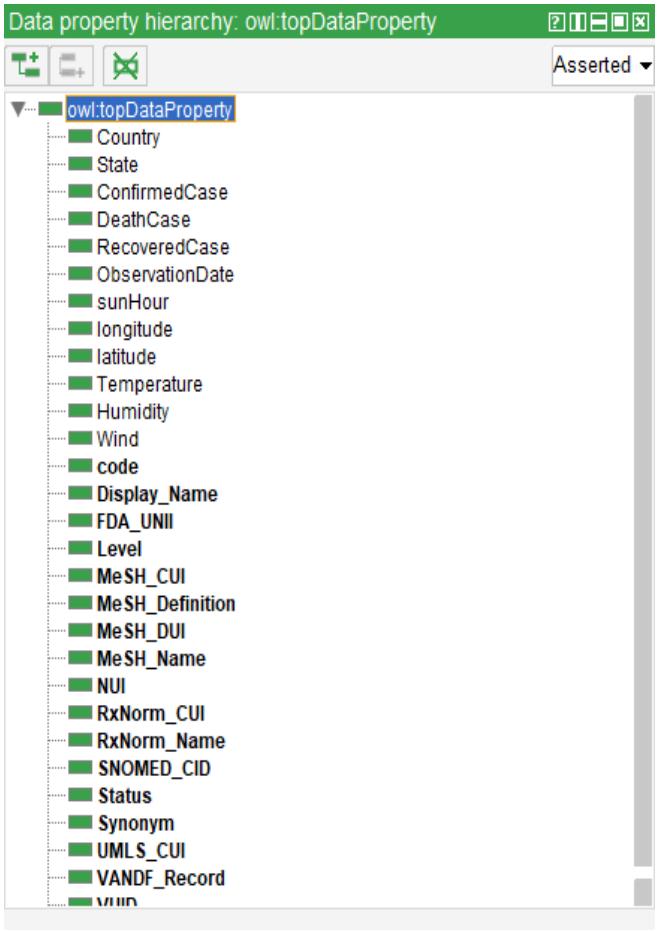

Fig. 9. The data type properties in COVID-19 Ontology.

Despite the lack of time-processing in SSWF compared to the Onto-NoSQL plug-in, SSWF needs a good knowledge of Big Data technologies and frameworks.

Also, the authors calculate the Gamma association coefficient for the daily average of humidity and PM10 by using association rules discovery for imported data [21]. The Semantic Web Rule Language (SWRL) rule is shown in the following code. This rule is counting 'a', where 'a' is the number of PM10 that satisfy the two classes at the same time, 'e' is the total number of PM10 in each class of the second class, and T is the total number of the samples.

Events(?e) ^ E_PM10(?e,?m) ^ humidity (?ws)^ swrlb:lessThan(?m, 70) ^ swrlb:greaterThan(?m, 10) ^ swrlb:lessThan(?ws, 20.0) -> swrlq:select(?e) ^ swrlq:orderBy(?m)

For example, the authors count several PM10 (events) with range low (10-20 (µg m–3)) and humidity low type (less than 70).

Table VIII visualizes the outcome of the Gamma association coefficient for the daily average of humidity and PM10. There is a strong relation between PM10 class low (10-20 (µg m–3)) and humidity low type (less than 70) and weak relation between low PM10 and not low humidity.

TABLE VIII. GAMMA ASSOCIATION COEFFICIENT BETWEEN THE DAILY AVERAGE OF HUMIDITY AND PM10.

No. of events	Humidity	PM10	Ɣ (Gamma association coefficient)
278	Low	Low	76.16%
62	Not low	Low	16.98%

Fig. 10 shows the relation curve among humidity, temperature, and PM10 that confirms the above knowledge discovery rule.

A user can query datasets in Onto-NoSQL plug-in, regardless of whether or not those datasets are incorporated and stored in a Big Data environment. Algebra tree has a limitation in the conversion of semantics query. Onto-NoSQL plug-in allows any semantic query and association rules to be run by the user. Onto-NoSQL plug-in is easy to generate a knowledge base of various data sources. Regardless of the Big Data and semantic environment, the user can handle the semantic Big Data D structure.

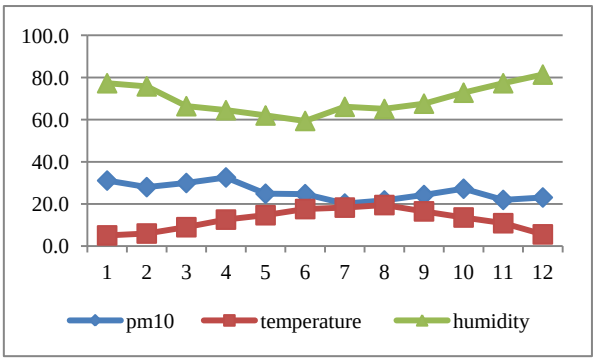

Fig. 8. Curve relations among humidity, temperature, and PM2.5.

The authors convert schema for air quality parameters and import data in Cyprus area. They build association rules discovery functions to predict the annual average value of NO2 over Egypt. The main challenge is that the data is unavailable in some countries or cities like Egypt. The nearest area to Egypt is Cyprus that is far away from Egypt with 956KM. According to the Egyptian Environmental Affairs Agency (EEAA) (http://www.eeaa.gov.eg), the annual average of NO2 over Egypt in 2011 is 58 (μg m–3). After importing Cyprus data and applying association rule, the prediction of the annual average of NO2 is 56.25 (μg m–3) with an accuracy percentage of 96.9%.

B. Weather and COVID-19

The proposed plug-in solved the problem of Big Data Ontology creation by dividing individuals into multiple RDF. Merged Ontology leads to the dynamic knowledge base with different facts and patterns.

The proposed plug-in allowed users to import Big Data to the semantic environment and apply semantic operations like decision support, reasoning, and event detection. To evaluate the proposed plug-in, the authors run SPARQL queries on the new COVID-19 Ontology to determine the most countries infected with Coronavirus (COVID-19) and its position. That is shown in Query 1. The result of the query shows that the United States of America and China are the most countries infected with Coronavirus (COVID-19) till the end of May 2020 and these countries are found in latitude range [40-43].

Query 1: Most countries infected with Coronavirus (COVID-19).

```
Select ?ObservationDate ?Country max(?ConfirmedCase)
Where
{
?Coronavirus: Country ?Country.
?Coronavirus: ConfirmedCase ?ConfirmedCase.
?Coronavirus:ObservationDate ?ObservationDate

} Group by ?Country ? ObservationDate
LIMIT 2
```

The authors measure the temperature range for most countries infected with Coronavirus (COVID-19). That is shown in Query 2. The temperature middle range is [6-20] ℃.

Query 2: The temperature range for most countries infected with Coronavirus (COVID-19).

```
Select min(?Temperature) max(?Temperature) Where
{
?CoronavirusData: Country ?Country.
?Weather: Temperature ? Temperature.
FILTER(?Country = " Italy "^^xsd:string
|| ?Country = " Spain"^^xsd:string)
}
```

VIII. CONCLUSION

The world is in bad need of a big dynamic knowledge base for Coronavirus (COVID-19) that is combined with all data and facts related to this virus. This data takes the nature of Big Data. The Big Data provides a huge wave of data storage, management, analysis, knowledge extraction, security, and visualization opportunities and challenges. Thus, this research aimed to build a big dynamic knowledge base from Big Data and extract patterns from this knowledge. The main goal is Ontology creation and transforming Big Data from Big Data Frameworks to the semantic environment. This paper has presented the Onto-NoSQL Protégé plug-in, which allows a user to import schema and data from Hbase. The proposed plug-in architecture has been explored. A short overview of the different options available in the Onto-NoSQL plug-in has been presented. Finally, the plug-in has been implemented on air quality and weather Big Datastore in Hbase. The plug-in allows the user to import the schema and the data. The time consumption to import 64 GB is 17 minutes. Imported data has predicted new data values employing association rules discovery with an accuracy percentage of 96.9%. In addition, the plug-in has been implemented on Coronavirus (COVID-19) and weather Datastore in Hbase. The plug-in enables building COVID-19 Ontology. This Ontology maintains CIDO ontology, Coronavirus data, and weather data around the world. After building Ontology, the authors explore the most countries infected with Coronavirus (COVID-19) at the end of May. Furthermore,

the authors study these countries in regard to temperature and latitude range. In future work, they will apply the Onto-NoSQL plug-in in all types of NoSQL databases. Also, they will combine COVID-19 Ontology with other ontologies to extract more knowledge and patterns with regard to this virus.

REFERENCES

[1] Coronavirus disease (COVID-2019) situation reports, Available :https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports/,Accessed 31 05 2020.

[2] Zhan, Choujun and Tse, Chi and Fu, Yuxia and Lai, Zhikang and Zhang, Haijun, Modelling and Prediction of the 2019 Coronavirus Disease Spreading in China Incorporating Human Migration Data (2/25/2020). Available at SSRN: https://ssrn.com/abstract=3546051 or http://dx.doi.org/10.2139/ssrn.3546051

[3] Zhou T, Liu Q, Yang Z, et al. Preliminary prediction of the basic reproduction number of the Wuhan novel coronavirus 2019-nCoV. J Evid Based Med. 2020;13:3–7.

[4] Jia, L., Li, K., Jiang, Y., Guo, X., and Zhao, T. (2020), "Prediction and analysis of Coronavirus Disease 2019," Working Paper: https://arxiv.org/ftp/arxiv/papers/2003/2003.05447.pdf

[5] K.Agnieszka, "Ontology-Based Approaches to Big Data Analytics." ACS (2016).

[6] M. Ptiček, B. Vrdoljak, M, Gulić, "The potential of semantic paradigm in warehousing of Big Data", Automatika, 60:4, 393403, DOI:10.1080/00051144.2019.1630582,2019.

[7] K. ElDahshan, E. K. Elsayed, and H. Mancy," Semantic Smart World Framework", Applied Computational Intelligence and Soft Computing, Vol. 2020, pp.12, https://doi.org/10.1155/2020/8081578,2020

[8] D. Gupta, R. Rani, "A study of Big Data evolution and research challenges", Journal of Information Science, Vol. 45, No. 3, pp.322–340, https://doi.org/10.1177/0165551518789880, 2019.

[9] A. Nayak, A. Poriya, D. Poojary, "Type of nosql databases and its comparison with relational databases", International Journal of Applied Information Systems. vol.5, pp.16-19, 2013.

[10] M. Ptiček and B. Vrdoljak, "Semantic web technologies and Big Data warehousing," 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, 2018, pp. 1214-1219.

[11] DB-engines ranking, Available in [http://db-engines.com/en/ranking] (Accessed 1 January 2020).

[12] Talks about HBase storage architecture, Computer Science (FCS), New York, USA, ISSN 2249-0868, Available in http://hbase.apache.org/book/architecture.html#arch.overview. (Accessed 15 January 2020).

[13] Budi Susanto, Hake Y. Situmorang, and Gloria Virginia, "RDFS-Based Information Representation of Indonesian Traditional Jamu," Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2019, 13-15 March, 2019, Hong Kong, pp352-357.

[14] S. Sakr, Albert Y. Zomaya," Ontologies for Big Data", Encyclopedia of Big Data Technologies. Springer 2019, ISBN 978-3-319-63962-8.

[15] M.A. Musen, The protégé project: a look back and a look forward, AI Matters, Vol. 1, No. 4, pp. 4–12, 2015.

[16] C. Nyulas, M. O'Connor, S. Tu, "DataMaster - a Plug-in for Importing Schemas and Data from Relational Databases into Protégé", semanticscholar, 2007.

[17] H. Abbes, F. Gargouri, "M2Onto: an approach and a tool to learn OWL Ontology from MongoDB database", In 16th international conference on intelligent systems design and applications (ISDA), 2016.

[18] C. Knoblock, P. Szekely, "Exploiting Semantics for Big Data Integration", AI Magazine, Vol. 36, pp. 25-38, 2015.

[19] A. M. Rinaldi and C. Russo, "A Matching Framework for Multimedia Data Integration Using Semantics and Ontologies," IEEE 12th International Conference on Semantic Computing (ICSC), Laguna Hills, CA, pp. 363-368, 2018.

[20] K. Desmond, J. Vincent, "Automatic conversion of relational databases into ontologies: a comparative analysis of protégé plug-ins performances", International Journal of Web & Semantic Technology (IJWesT) Vol.7, No.3/4, October 2016

[21] P. Thuy and L. Sungyoung, "DTD2OWL: automatic transforming XML documents into OWL ontology", Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture, and Human, November 2009, pp. 125–131.

[22] Y. Yoa, R. Wu, H. Liu," JTOWL: A JSON to OWL Converto", Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning, November 2014 pp. 13–14.

[23] K. Guo, Z. Liang, Y. Tang, and T. Chi, ''SOR: An optimized semantic ontology retrieval algorithm for heterogeneous multimedia big data,''J.Comput. Sci., vol. 28, pp. 455–465, Sep. 20

[24] European Environment Agency. (n.d.), Available: https://www.eea.europa.eu, Accessed 7 15 2019

[25] European Climate Assessment & Dataset. (n.d.), Available: https://www.ecad.eu., Accessed 6 30, 2019

[26] Dongyang Li, Dan Yang, Jing Zhang, and Xuedong Zhang, "AR-ANN: Incorporating Association Rule Mining in Artificial Neural Network for Thyroid Disease Knowledge Discovery and Diagnosis," IAENG International Journal of Computer Science, vol. 47, no.1, pp25-36, 2020.

[27] COVID19 Global Weather Data., Available: https://www.kaggle.com/winterpierre91/covid19-global-weather-data, Accessed 01 05 2020

[28] Coronavirus Infectious Disease Ontology. Available : http://bioportal.bioontology.org/ontologies/CIDO, Accessed 15 05 2020

Kamal Abdelraouf ElDahshan is a Professor of Computer Science and Information Systems at Al-Azhar University in Cairo, Egypt. An Egyptian national and graduate of Cairo University, he obtained his doctoral degree from the Universitè de Technologie de Compiègne in France, where he also taught for several years. During his extended stay in France, he also worked at the prestigious Institute National de Tèlècommunications in Paris. Professor ElDahshanâAZs extensive international research, teaching, and consulting experiences have spanned four continents and include academic institutions as well as government and private organizations. He taught at Virginia Tech as a Visiting Professor; he was a Consultant to the Egyptian Cabinet Information and Decision Support Centre (IDSC), and he was a Senior Advisor to the Ministry of Education and Deputy Director of the National Technology Development Centre. Professor ElDahshan is a professional Fellow on Open Educational Resources as recognized by the United States Department of State and an Expert at ALECSO as recognized by the League of Arab States.

Eman Karam Elsayed is a Professor of Computer Science; she obtained a Ph.D. in computer science in 2005, Al-Azhar University, Master of Computer Science, Cairo University, in 1999, and Bachelor of Science, Mathematics and Computer Science Department, Cairo University, in 1994. She published thirty-three national and international papers until 2017 on data mining, Ontology engineering, e-learning, image processing, and software engineering. She also published two books on informal methods and event B on the Amazon database. Moreover, she is a Member of the Egyptian Mathematical Society and Intelligent Computer and Information Systems Society.

Hend Mancy received her B.Sc. from Science Faculty, Al-Azhar University, in 2006, Master of Computer Science, Al-Azhar University, in 2016, in "Handling Uncertainty in Big Data." Currently, she is a Working Assistant Lecturer in the Computer Science Department, Science Faculty, Al-Azhar University, and she is a Ph.D. Student at Al-Azhar University in the Smart City Frameworks under the supervision of Dr. Kamal Eldahshan. Her research interests include Big Data, data science, and Smart City architectures and frameworks.