

BUILDING A KNOWLEDGE GRAPH FOR FOOD, ENERGY,
AND WATER SYSTEMS

A THESIS IN
Computer Science

Presented to the Faculty of the University
Of Missouri-Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

By
Mohamed Gharibi

B.S., Jazan University, 2015

Kansas City, Missouri
2017

© 2017
Mohamed Gharibi
ALL RIGHTS RESERVED

BUILDING A KNOWLEDGE GRAPH FOR FOOD, ENERGY, AND WATER SYSTEMS

Mohamed Gharibi, Candidate for the Master of Science Degree
University of Missouri-Kansas City, 2017

ABSTRACT

A knowledge graph represents millions of facts and reliable information about people, places, and things. Several companies like Microsoft, Amazon, and Google have developed knowledge graphs to better customer experience. These knowledge graphs have proven their reliability and their usage for providing better search results; answering ambiguous questions regarding entities; and training semantic parsers to enhance the semantic relationships over the Semantic Web. Motivated by these reasons, in this thesis, we develop an approach to build a knowledge graph for the Food, Energy, and Water (FEW) systems given the vast amount of data that is available from federal agencies like the United States Department of Agriculture (USDA), the National Oceanic and Atmospheric Administration (NOAA), the U.S. Geological Survey (USGS), and the National Drought Mitigation Center (NDMC). Our goal is to facilitate better analytics for FEW and enable domain experts to conduct data-driven research. To construct the knowledge graph, we employ Semantic Web technologies, namely, the Resource Description Framework (RDF), the Web Ontology Language (OWL), and SPARQL. Starting with raw data (e.g., CSV files), we construct entities and relationships and extend them semantically using a tool called Karma. We enhance this initial knowledge graph by adding new relationships across entities by extracting information from ConceptNet via an efficient similarity searching algorithm. We show initial performance results and discuss the quality of the knowledge graph on several datasets from the USDA.

APPROVAL PAGE

The faculty listed below have examined a thesis titled “Building a Knowledge Graph for Food, Energy, and Water Systems,” presented by Mohamed Gharibi. Student, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Praveen R. Rao, Ph.D., Committee Chair
School of Computing and Engineering

Yugyung Lee, Ph.D., Committee Member
School of Computing and Engineering

Deep Medh, Ph.D., Committee Member
School of Computing and Engineering

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF ILLUSTRATIONS	vi
LIST OF TABLES	vii
ACKNOWLEDGMENTS	viii
Chapter	
1. INTRODUCTION	1
1.1 Overview	1
2. CHALLENGES	6
3. BACKGROUND AND RELATED WORK	7
3.1 Converting databases to RDF model	7
3.2 Enriching a dataset with extra triples based on the existing ones	17
4. APPROACH	27
4.1 Overview	27
4.2 Converting a database table into RDF triples	28
4.3 Enriching RDF data triples	29
4.4 Architecture	34
5. EVALUATION	35
4.5 Implementation	35
4.6 Work load	36
4.7 Results	36
6. CONCLUSION AND FUTURE WORK	40
REFERENCES	41
VITA	45

LIST OF ILLUSTRATIONS

Figure	Page
1. RDF model.....	2
2. RDF model for the second book	4
3. D2RQ-ML syntax	9
4. Comparison between SML and R2RML	9
5. Any23, list of extractors.....	10
6. BioDSL, mapping syntax	12
7. DBpedia semantic query	18
8. DBpedia results for running "Lemon"	19
9. Dandelion semantic results for comparing "Lemon" and "Lime"	20
10. Dandelion results for comparing two phrases.....	20
11. Results of ParallelDots for comparing two phrases	21
12. Results of ParallelDots when comparing two terms	21
13. WordNet hierarchy for nouns and verbs	22
14. Results returned from WordNet.....	23
15. WordNet results when running a general term	23
16. WordNet result when running a phrase contains dictionary-based term	23
17. ConceptNet example for a relationship.....	25
18. FEW ontology while using Karma	28
19. First level of the searching tree	31
20. Third level of the searching tree	32
21. The searching tree for the term "Flower".....	33
22. The searching tree for the term "Fire"	33

LIST OF TABLES

Table	Page
1. Example of a book database	3
2. A single row from employees database	7
3. Example of dealer database	11
4. The original input table for Open Refine	13
5. The input table after sorting	14
6. The input table after deleting all blank cells	14
7. Time needed for different RDF datasets	36
8. FEW systems input experiments	37
9. An input example (Ingredients)	37

ACKNOWLEDGMENTS

This work would have never seen the sunlight without the help and support of many people. Thank you all!

I would like to express my gratitude to my father who supports me all the time and who taught me to be who I am today. I'm also grateful to my mother who motivates me and wants me to be the best, and to my brother who taught me a lot and was always there for me. I appreciate having you in my life! My sincere thanks go to my grandparents who believe in me!

I would like to express my sincere gratitude to my advisor, Dr. Praveen Rao, for his unlimited guidance, support, and motivation. Thank you for teaching me how to be a better student and a better person. I appreciate all the time and the efforts you invested in me. It is an honor to complete my studies under your supervision.

Prof. Ghulam Chaudhry, thank you for your great support and for all the opportunities you offered me.

Dr. Lee, Dr. Deep, and Dr. Zheng, thank you for your continuous support and guidance. I appreciate your time in teaching me.

Cuong Cu, thank you for your help and your time. I appreciate your continuous help.

My lab friends, thank you for your help and for all the nice times we spent together.

Finally, my sincere thanks go to the School of Computing and Engineering faculty, staff, and friends. Thank you all.

CHAPTER 1

INTRODUCTION

In this chapter, we briefly introduce the area of my research, the research problem that we address in the area, the objectives of the work, the tools and the web services that have been used, and my work contribution.

1.1 Overview

Winding back the clock to 10 years ago where anyone could hardly believe they would own a mobile phone much less a laptop, where nowadays most cars have more powerful computing microprocessors than the ones used in the space vehicles that were utilized as transportation to send men to the moon [17]. The huge jump of technology innovates new lifestyles and the way we communicate in many different aspects. It even modifies the priority in our way of thinking, transforming the agricultural revolution to industrial revolution and resulting in a huge information revolution. Nowadays, anything can be accomplished via technology including online meetings, online degrees, online jobs, social communication, and etc. Furthermore, entertainment and communicating with friends and family can be done online through social networking websites. This significant information revolution generates a huge amount of data every day, called Big Data (BD) [6]. The Big Data concept refers to a complex and large volume for both structured and unstructured data where the traditional data processing applications software are inadequate to deal with the huge amount of data that is generated every day [4].

Big Data Science (BDS) is the science that studies managing, storing, analyzing, and retrieving huge amount of data. One of the challenges for BDS is that data on the internet does not follow a particular format. Different social media websites use different ways to store and

manipulate online data [29]. For instance, Youtube website stated that 400 hours worth of videos are being uploaded per minute and one billion hours is the amount of content being watched on Youtube daily [13]. Youtube stores these hours of videos in a structured format whereas Facebook –which has more users than China’s population- stores its data in graphs [14]. These different formats create new challenges for users who want to analyze and process such data. The essential part of BDS is enabling users to analyze and process big data with different formats. Structured data, also known as Relational Databases (RDB), includes tables, spreadsheets, and databases that use Structured Query Language (SQL) for processing. Although SQL is a common and powerful language, there are still many challenges for joining structured and unstructured data such as texts, videos, images, emails, and audio files.

Fortunately, there is a universal data model that is considered a solution for all the aforementioned challenges. Resource Description Framework (RDF) is a World Wide Web Consortium (W3C) data model. RDF represents data in three parts: subject, predicate, and object which are known as RDF triple, <subject> <predicate> “object”, figure(1).

A new value can be added to describe the context of the triple, which is called the <context> and that becomes RDF quads instead of triples [1][2].

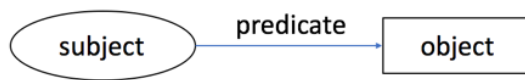


Figure 1: RDF model

RDF triples represent semantic information and facts between entities and concepts for both humans and computers [38]. Subjects within RDF data model provided with a Universal Resource Identifier (URI) to present unique information and facts. This allows both humans and computers to trace back the origin of a word, related terms, and in what context it was mentioned [36]. The following line illustrates a quad model after engaging the URIs.

`<http://example.com/subject> <http://example.com/predicate> "object" <http://example.com/quad_shape>`

Furthermore, one of the most important uses of the RDF model is joining and merging data from different formats. Without using RDF model, it can be complex to merge two different databases. The level of complexity increases by increasing the number of databases. When using RDF model, the process begins by converting tables to RDF model, then joining these triples. The advantage of joining RDF data model is usable for different amounts of data with various formats. Converting a database into RDF model is one of the challenges that many users face since there is no specific tool that can be used automatically without a human contribution. Converting a database into RDF model requires a special structure of mapping the data from a database into RDF model. Different databases require different structures; these structures are called ontologies. For each database, a user is required to provide an ontology.

Few ontologies exist on the internet, but they do not cover different users' purposes. Therefore, we developed a new ontology, based on DBpedia ontologies, that can be used to serve users who are working with FEW knowledge base, called FEW ontology. FEW ontology contains tens of the relationships that can be used to specify the relationship between two entities while converting to RDF model. For instance, the following table contains books' titles, authors, publishers, etc.

Table 1: Example of a book database

Isbn	Title	Author	publishedID	Pages
0596002637	Partial RDF	Shelley Powers	7642	350
0596000480	JavaScript	David Flanagan	3556	936

The relationship that links the second book with "JavaScript" is "title." There are few ontologies that define such simple relationships, but for another column the relationship might be "number of pages." In this case, a user has to search for an ontology that defines the

relationship “number of pages” or creates his own. After converting the previous table to RDF model, data will be presented as in figure (2).

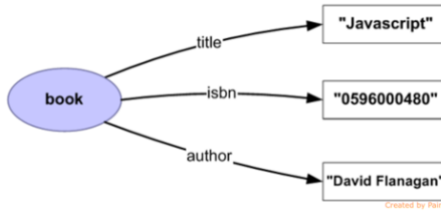


Figure 2: RDF model for the second book [19]

Another advantage of RDF data model is that a user can simply understand all the information presented using these RDF triples. A user may add extra information such as a link to the author personal website, how many children he has, and what other books he had written.

The second part of our thesis is enhancing the mapped RDF dataset by adding extra information based on the semantic similarity between the entities in a given dataset. Our program starts by comparing semantically two entities at a time. Based on the relationships between these entities, extra triples will be added to the dataset containing the compared entities with the semantic similarity score and the relationship between them. In a dataset, multiple entities may have a relationship other than the mentioned ones. For instance, a dataset may contain names such as “David Flanagan” and “Java in a Nutshell” which can be confusing to users. In this case, adding extra information based on the semantic similarity between the first and second name such as “author” or “owned_by” will enrich the dataset and provide valuable information for users to understand the exact relationships between names and entities. Moreover, enriching a dataset with extra information will minimize the searching time. For example, adding the relationship “author” between “David Flanagan” and “Java in a Nutshell” will save time and effort for users who want to search for the relationship between

these names. For this purpose, we utilize ConceptNet web service to provide us with all the semantically related concepts for a given word in order to use them to conduct our calculations.

CHAPTER 2

CHALLENGES

1. Most of the technology nowadays is concerned with computer-related projects such as the social media, banks, advertisements, education, and etc. Food, Water, and Energy systems are not having the same technology interests as the other majors. Hence, our project aims to build a knowledge base for FEW systems to shed light on these areas in a way that enhances these systems and enables users to analyze databases in a better way [26].
2. The lack of the existing ontologies while converting databases to RDF model, obligated us to create a new ontology based on DBpedia ontologies to be used with FEW systems.
3. Analyzing data is not a new concept, but enriching a dataset by adding extra RDF quads related to the existing ones based on the semantic similarity between these quads is a real challenge, that will enrich a dataset and provide users with more helpful information and facts about concepts exist in that particular dataset.

CHAPTER 3

BACKGROUND AND RELATED WORK

In the first part of this chapter, we present a brief introduction to several approaches and tools that are used to convert various data formats to RDF data model and the reasons why we chose Karma integration tool in our project.

In the second part of the chapter, we present the most reliable semantic networks and the reasons behind choosing ConceptNet to work with in our project.

3.1 Converting databases to RDF model (Part I)

3.1.1 R2RML

R2RML stands for RDB 2 RDF Mapping Language. R2RML is a language that provides Direct Mapping (DR) from relational databases to RDF model in a customized way [22]. Direct mapping enables users to express vocabularies (relationships) and structures based on users' choice. One of the best features that R2RML provides is to allow SPARQL Protocol and RDF Query Language (SPARQL) end point queries over the mapped relational data. For instance, given the following table that needs to be converted to RDF model:

Table 2: A single row from employees' database

EMPNO Integer Primary Key	ENAME Characters (100)	JOB Characters(20)	DEPTNO Integer DEPT. NO. (DEPTNO)
7369	SMITH	CLERK	10

Using R2RML, a user may write a query as the following:

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.
<#TriplesMap1>
  rr:logicalTable [ rr:tableName "EMP" ].
  rr:subjectMap [
    rr:template "http://data.example.com/employee/{EMPNO}";
    rr:class ex:Employee;
  ];
  rr:predicateObjectMap [
    rr:predicate ex:name;
    rr:objectMap [ rr:column "ENAME" ]; ].
```

Using this snippet of code, R2RML will map the given table to RDF model to be in this format after the mapping process:

```
<http://data.example.com/employee/7369> rdf:type ex:Employee .  
<http://data.example.com/employee/7369> ex:name "SMITH" .  
<http://data.example.com/employee/7369> ex:department <http://data.example.com/department/10> .
```

R2RML provides a mapping language where many other languages and platforms rely on it while mapping. Users who want to use R2RML are required an advanced knowledge of R2RML, and extra knowledge about RDF models in order to use R2RML. Moreover, it does not provide a user interface, which makes it harder to read and understand.

3.1.2 D2RQ-ML

D2RQ-ML is a Declarative Mapping Language that maps Relational databases to RDF model. It provides SPARQL access since D2RQ is written in RDF Turtle syntax. D2RQ provides virtual access to graphs and databases, such as the views in SQL, to process users' queries [24][37]. D2RQ requires users to write a template (which can be considered as the mapping structure) in order to use D2RQ. Writing a template for a single database consumes a lot of time and efforts. The complexity level of this procedure increases gradually with the number of columns in a database and the number of databases. Furthermore, users are required to have an advanced knowledge in writing templates and being familiar with D2RQ syntax because it does not provide a Graphical User Interface (GUI). The following code snippet shows the syntax of D2RQ-ML.


```

map:Database1 a d2rq:Database;
d2rq:jdbcDSN "jdbc:mysql://localhost/iswc";
d2rq:jdbcDriver "com.mysql.jdbc.Driver";
d2rq:username "user";
d2rq:password "password";
.
map:Conference a d2rq:ClassMap;
d2rq:dataStorage map:Database1.
d2rq:class :Conference;
d2rq:uriPattern "http://conferences.org/comp/
confno@@Conferences.ConfID@@";
.
map:eventTitle a d2rq:PropertyBridge;
d2rq:belongsToClassMap map:Conference;
d2rq:property :eventTitle;
d2rq:column "Conferences.Name";
d2rq:datatype xsd:string;

```

Figure 3: D2RQ-ML syntax [33]

3.1.3 SML

Sparqlification Mapping Language (SML) is a mapping language that maps RDB to RDF model. SML offers a better syntax of mapping RDB to RDF than R2RML [15]. The following snippet of code illustrates how SML syntax is easier and more understandable for users.

SML	R2RML
<pre> Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> Prefix xsd: <http://www.w3.org/2001/XMLSchema#> Prefix ex: <http://ex.org/> Create View hotels As Construct { ?s a ex:Hotel ; rdfs:label ?l ; ex:vacancy ?v } With ?s = uri(?website) ?l = plainLiteral(?name,'en') ?v = typedLiteral(?vacancy, xsd:boolean) Constrain ?s prefix "http://ex.org/" From ""SELECT website, name, vacancy FROM hotels"" </pre>	<pre> @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> . @prefix xsd: <http://www.w3.org/2001/XMLSchema#> . @prefix ex: <http://ex.org/> . <HotelTriplesMap> rr:logicalTable [rr:sqlQuery ""SELECT website, name, vacancy FROM hotels""]; rr:subjectMap [rr:column "website"; rr:class ex:Hotel]; rr:predicateObjectMap [rr:predicate rdfs:label; rr:objectMap [rr:column "name"; rr:language "en"];]; rr:predicateObjectMap [rr:predicate ex:vacancy; rr:objectMap [rr:column "vacancy"; rr:datatype xsd:boolean];]. </pre>

Figure 4: Comparison between SML and R2RML [33]

SML made the syntax easier for users but it still does not provide a GUI. Likewise, a user is still required to have an advanced knowledge in SML syntax before using it which is not an easy task for most of the users who just want to convert their data to RDF data model. In addition, it is time consuming since a user has to map each column manually.

3.1.4 Apache Any23

Apache Any23 is a web service, library, and a command line that extracts and produces RDF triples format from various web documents [25]. The name Any was derived from Anything to triples. To be more specific the current supported formats are:

- RDF/XML, Turtle, Notation 3.
- RDFa with RDFa1.1 prefix mechanism.
- Microsoft formats: Adr, Geo, hCalendar, hCard, hListing, hResume, XFN and Species.
- HTML5 microdata such as schema.org.
- JSON-LD: JSON for linked data.
- CSV: Comma Separated Values with separator auto detection.

Apache Any23 is written in Java and it can be accessed using the command line. In this case, a user is required to be familiar with all the commands that Any23 permits. It does not provide any GUI as well. Consequently, a user cannot see the changes that have been made on the database during the mapping process. The following figure illustrates the types that Any23 able to extract RDFs from:

```
cli$ any23 extractor --list
csv [org.apache.any23.extractor.csv.CSVExtractorFactory]
html-embedded-jsonld [org.apache.any23.extractor.html.EmbeddedJSONLDExtractorFactory]
html-head-icbm [org.apache.any23.extractor.html.ICBMEExtractorFactory]
html-head-links [org.apache.any23.extractor.html.HeadLinkExtractorFactory]
html-head-meta [org.apache.any23.extractor.html.HTMLMetaExtractorFactory]
html-head-title [org.apache.any23.extractor.html.TitleExtractorFactory]
html-mf-adr [org.apache.any23.extractor.html.AdrExtractorFactory]
html-mf-geo [org.apache.any23.extractor.html.GeoExtractorFactory]
html-mf-hcalendar [org.apache.any23.extractor.html.HCalendarExtractorFactory]
html-mf-hcard [org.apache.any23.extractor.html.HCardExtractorFactory]
html-mf-hlisting [org.apache.any23.extractor.html.HListingExtractorFactory]
html-mf-hrecipe [org.apache.any23.extractor.html.HRecipeExtractorFactory]
html-mf-hresume [org.apache.any23.extractor.html.HResumeExtractorFactory]
html-mf-hreview [org.apache.any23.extractor.html.HReviewExtractorFactory]
html-mf-hreview-aggregate [org.apache.any23.extractor.html.HReviewAggregateExtractorFactory]
html-mf-license [org.apache.any23.extractor.html.LicenseExtractorFactory]
html-mf-species [org.apache.any23.extractor.html.SpeciesExtractorFactory]
html-mf-xfn [org.apache.any23.extractor.html.XFNExtractorFactory]
html-microdata [org.apache.any23.extractor.microdata.MicrodataExtractorFactory]
html-rdfa11 [org.apache.any23.extractor.rdfa.RDFA11ExtractorFactory]
html-xpath [org.apache.any23.extractor.xpath.XPathExtractorFactory]
rdf-jsonld [org.apache.any23.extractor.rdf.JSONLDExtractorFactory]
rdf-na [org.apache.any23.extractor.rdf.NQuadsExtractorFactory]
rdf-nt [org.apache.any23.extractor.rdf.NTriplesExtractorFactory]
rdf-trix [org.apache.any23.extractor.rdf.TriXExtractorFactory]
rdf-turtle [org.apache.any23.extractor.rdf.TurtleExtractorFactory]
rdf-xml [org.apache.any23.extractor.rdf.RDFXMLExtractorFactory]
yaml [org.apache.any23.extractor.yaml.YAMLEExtractorFactory]
```

Figure 5: Any23, list of extractors [9]

3.1.5 CSV2RDF

CSV2RDF is a simple but powerful library that allows mapping from Comma separated Values (CSV) to RDF. CSV2RDF is similar to D2RQ-ML, because both of them requires a template written by users to provide the mapping structure. We present a simple table with the required template to be mapped to RDF model [30].

Table 3: Example of dealer database

Year	Make	Model	Description	Price
1997	Ford	E350	ac, abs, moon	3000
1997	Chevy	Venture "Extended Edition"	-	4900
1999	Chevy	Venture "Extended Edition, Very Large"	-	5000
1996	Jeep	Grand Cherokee	MUST SELL! air, moon roof, loaded	4799

The required template to convert table (3) to RDF model [30].

```
:Manufacturer-#{Make} a gr:BusinessEntity ;
  rdfs:label "#{Make}" .

:Model-#{Model} a gr:ProductOrServiceModel ;
  rdfs:label "#{Make} #{Model}" ;
  gr:hasManufacturer :Manufacturer-#{Make} .

:Car-#{_ROW_} a vso:Automobile, gr:ActualProductOrServiceInstance ;
  rdfs:label "#{Make} #{Model} ({Year})" ;
  gr:hasManufacturer :Manufacturer-#{Make} ;
  gr:hasMakeAndModel :Model-#{Model} ;
  vso:modelDate "#{Year}-01-01"^^xsd:date .

:Offer-#{_UUID_} a gr:Offering ;
  rdfs:comment "#{Description}" ;
  gr:includes :Car-#{_ROW_} ;
  gr:hasBusinessFunction gr:Sell ;
  gr:hasPriceSpecification _:price .

_:price a gr:UnitPriceSpecification ;
  gr:hasCurrency "USD"^^xsd:string ;
  gr:hasCurrencyValue "#{Price}"^^xsd:float .
```

A user has to write his own code to parse the data and then use this template within that code. This may take a lot of time especially when a user is dealing with large databases that contain many columns or dealing with many databases.

3.1.6 BioDSL

BioDSL is a new approach in mapping CSV files to RDF using a Domain Specific Language(DSL) called BioDSL. BioDSL allows users to write different programs to map biodiversity data to RDF format then link these RDFs to Linked Data. BioDSL uses Groovy

Programming Language where its syntax is based on the objects and functions [28]. A CSV table represents its entities as columns, BioDSL has an object called “csv” to represent each column name, for instance (csv.ename) indicates the csv object that represents the “ename” columns in a table. BioDSL functions provide a function called “Map” that maps the repressed column to RDF along with its relationship to other columns in that particular table. Other functions define how the URIs will be generated for each RDF Triple based on the table classes. BioDSL has two main parts, the first part is loading ontologies and the second part is mapping data to RDF model. The below figure illustrates these two parts in a single example.

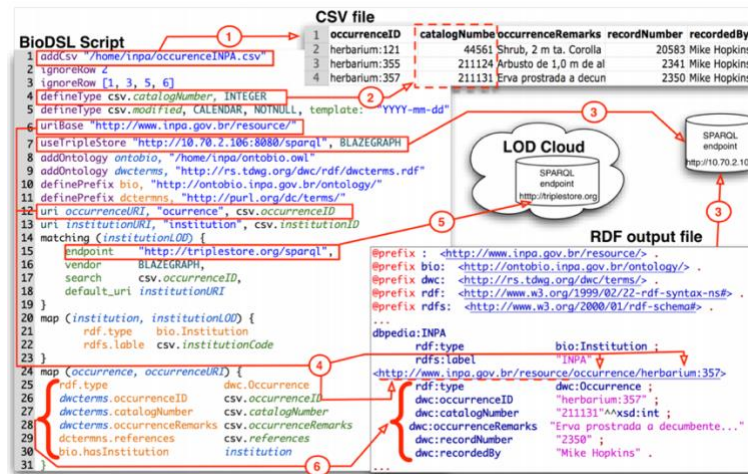


Figure 6: BioDSL, mapping syntax [33]

Similar to the previous approaches, BioDSL does not provide a GUI. Although BioDSL provides better features than the previous approaches, but it still consumes both time and efforts from users.

3.1.7 Open Refine

The formal name of Open Refine was Google Refine. Eventually, Google stopped supporting this project since October 2012. Since that time, the name has changed from Google Refine to Open Refine.

Open Refine is the name of the tool which contains many plugins that perform many different tasks. RDF Refine is the name of the plugin that is used by Open Refine to map data from a CSV database to RDF model. Open Refine is an effective tool for working with messy data. It allows user to clean data, transfer it from one format to many different formats. Furthermore, Open Refine offers a neat GUI for users. Open Refine has many contributions in the world of semantic web [39]. A user may use Open Refine without having an advanced knowledge in writing lines of code nor any experience with writing templates or providing URIs for RDF graphs. On top of that Open Refine provides many other features such as:

1. Importing data to Open Refine in various formats.
2. Explore the whole dataset within seconds.
3. Applying basic and advanced cell transformations.
4. Provide features to handle the cell which contains multiple values.
5. Create instantaneous links within the dataset.
6. Filtering and joining datasets with regular expressions.
7. Providing automatic identifying for the named-entity-extraction.
8. Performing advanced operation on datasets.
9. Changing values and links of the cells directly.
10. Displaying results to the user after each operation instantaneously.

Additionally, Open Refine delivers powerful features for minimizing and simplifying databases. For instance, given the following table:

Table 4: The original input table for Open Refine

Speed	Car	payLoad	Image Link
-	Mercedes	-	Link1
200 mph	Mercedes	2200 pounds	Link2
-	Mercedes	-	Link3

160 mph	Toyota	1620 pounds	Link1
-	Toyota	-	Link2
-	Toyota	-	Link3

Open Refine starts by exploring the given database then start sorting columns based on the relationships. After sorting the given table, a new table will be generated that look like this:

Table 5: The input table after sorting

Car	Speed	payLoad	Image Link
Mercedes	-	-	Link1
Mercedes	200 mph	2200 pounds	Link2
Mercedes	-	-	Link3
Toyota	160 mph	1620 pounds	Link1
Toyota	-	-	Link2
Toyota	-	-	Link3

The next step is deleting the empty cells and combining the values for each entity. The generated table shown below:

Table 6: The input table after deleting all blank cells

Car	Speed	payLoad	Image Link
Mercedes	200 mph	2200 pounds	Link1, Link2, Link3
Toyota	160 mph	1620 pounds	Link1, Link2, Link3

On top of that, it is easy to install on many different operating system. Open Refine is compatible with Mac, Windows, and Linux.

Open Refine enables users to group together all the identical cells. Using the text facet, a user may change one of these identical cells and the change will be applied to the other cells. It provides a cluster to group the groups based on special characteristics. Using a cluster allows users to further control the group of groups such as sorting, replacing, formatting, etc. At any time, a user may undo any process and the changes will be displayed instantly. Using Open Refine, a user may import any ontology to be used. Open Refine also allows SPARQL end

point queries to be applied on the dataset and to choose what type of reconciliation to use. A user is not required to write any code, in contrast, a user may choose the vocabularies from a list to be applied to a table. Users provided with a graph to illustrate the structure model for a table before start mapping.

In the latest version of Open Refine, while we were mapping a CSV file to RDF data model, it was unable to finish the reconciliation part completely. Without finishing the reconciliation part, no RDF triples will be generated. We tried many different datasets on different machines and we repeatedly received the same outcomes. As a result, we did not use Open Refine in our project.

3.1.8 Karma Integration Tool

Karma is an information integration tool that enables users to integrate data faster and easier from different data sources such as databases, delimiters, spreadsheets, XML, KML, JSON, and Application Programming Interface (API). Users integrate data based on their choice of ontologies. One of the best features in Karma is learning the mapping process to classes and proposing a model that ties together these classes. Karma generates models based on what it learnt from users previous mapping [21]. A user may simply adjust and normalize the suggested model. Once the model is complete, a user may publish the integrated data as RDF or store it in a dataset. Karma provides many valuable features, we discuss these features one by one:

1. **Ease of use:** Karma offers a very simple user interface, which allows users to easily perform all the desired tasks. Karma uses programming-by-example to learn the mapping models and algorithms optimization to automate the process as much as possible. This feature is one of the best features for users, since all the tools requires users to go through the columns

one by one to set the relationships between them. Using Karma, a user has to map the columns for the first time only, then Karma will propose to users what it required. This learning technique by Karma saves a lot of time, efforts, and prevents error that might occur from a user side.

2. Hierarchal Sources: We have discussed many tools that map databases to RDF. Karma is the first tool that supports hierarchal data sources such as XML, JSON, and KML. This feature makes Karma a unique tool.
3. Web APIs: Karma supports importing data from both the static sources such as databases and file and from the APIs that contain thousands of data sources.
4. Semantic models: Karma provides few ontologies to use by default. In addition, Karma allows a user to upload any ontology to use. Karma recognizes ontologies in different type and different extensions.
5. Scalable processing: A user may work on a subset of a table to set the desired RDF model to be used. During this process, Karma will learn the structure of the user's model. After importing a larger dataset, Karma will propose a modeling structure for the user.
6. Data transformation: Karma enables users to transform data expressed in various formats into a common format. Karma is a capable tool when it comes to mapping CSV tables into RDF triples.
7. Mapping visualization: This is another unique feature. Karma displays a simple visualization graph for users which makes it easier to read the relationships between the columns and their relationships.
8. Editing a dataset: After importing data to Karma, a user may easily delete, add, swap, move, and change the values of columns and cells.

Karma provides semantic labeling which is the process of mapping columns in a database to the classes in an ontology [31]. Semantic labeling is a very challenging task when it comes to homogeneous data types and variety of data formats. Other techniques use machine learning to extract features that are related to the data from a domain, which means the data has to be re-trained for each new domain. Whereas Karma uses machine learning and it also uses similarity metrics to return correct semantic labels for data in a faster technique. Karma assigns 2 gigabytes of space to users, a user may increase this space up to 16 gigabytes. All these features can be applied using a simple and easy user interface. Therefore, based on all the aforementioned features and powerful services, we decided to use Karma in our project for mapping databases to RDF models.

3.2 Enriching a dataset with extra triples based on the existing ones (Part II)

This is the heart of our project, considering that there are no tools or web services that add extra triples to a dataset based on the semantic similarity between the existing triples. For this purpose, we utilize ConceptNet web service that provides us with related terms of a word with the weight for each edge that we can use in our calculations. In this section, we briefly introduce the most common and powerful semantic networks and the reasons why we chose ConceptNet.

3.2.1 DBpedia

DBpedia is a project available on the World Wide Web (WWW) that extracts knowledge and facts from Wikipedia. DBpedia is one of the largest stores for semantic relationships on the web. It allows its users to semantically query the extracted information from Wikipedia. DBpedia is known as one of the most famous linked data networks, as Tim Berners-Lee described it. Its articles are based on the infobox, which is a structured box of

information generated based on Wikipedia [32]. DBpedia contains almost 4.58 million entities. Out of the total entities, 4.22 million entities were classified under consistent ontologies. These entities include persons, places, games, films, albums, organizations, and many other aspects in more than 125 different languages [11]. Moreover, DBpedia uses SPARQL to query Wikipedia factual information. For instance, let us say a user was interested in the Shojo manga Japanese series Tokyo Mew Mew, and wanted to get some information about it. With a simple query, DBpedia allows a user to combine information about Tokyo Mew Mew from Wikipedia.

```
PREFIX dbprop: <http://dbpedia.org/property/>
PREFIX db: <http://dbpedia.org/resource/>
SELECT ?who, ?WORK, ?genre WHERE {
  db:Tokyo_Mew_Mew dbprop:author ?who .
  ?WORK dbprop:author ?who .
  OPTIONAL { ?WORK dbprop:genre ?genre } .
}
```

Figure 7: DBpedia semantic query [10]

This query lists the related genres of Tokyo Mew Mew from Wikipedia. In addition, DBpedia dataset is interlinked with various open source datasets on the internet to enrich Dbpedia knowledge. There are more than 45 interlinks between DBpedia and external datasets including OpenCyc, Freebase, CIA Free Fact Book, GeoNames, Bio2RDF, and MusicBrainz. These interlinks with DBpedia provide a substantial amount of information and facts. This combination of substantial amount of data set make DBpedia a powerful semantic network provides a huge amount of semantic relationships between different entities.

Despite all of the powerful features DBpedia provides, most of DBpedia information is based on Wikipedia. Wikipedia was criticized for presenting truths, half-truths, and falsehoods. On top of that, Wikipedia's articles can be edited by any user who has an internet connection which make many of Wikipedia topics controversial topics and subjects to spin and manipulation [25]. Therefore, all the facts on DBpedia that were extracted from inaccurate information are controversial facts. Furthermore, the results of DBpedia are messy as it returns

all the pages where the word was mentioned in. The returned result could be a synonym, related word, meaning in different languages or just a random article where the concept of the word was mentioned. For instance, we ran the word “Lemon” on DBpedia and it returned the following results.

Entity	Title
wikidata:Q18166039	Lemon Tree Passage
dbr:You_Are_Forgiven	You Are Forgiven
dbr:Lemon-throated_white-eye	Lemon-throated white-eye
dbr:Lemon_squash	Lemon squash
dbr:Lemon,_Lime_&_Bitters	Lemon, Lime & Bitters
dbr:Lemon_butterfly	Lemon butterfly
dbr:Lemon_Migrant	Lemon Migrant
wikidata:Q6521324	Lemon Love

Figure 8: DBpedia results for running "Lemon" [15]

These results are considered the most related terms and synonyms to the word “Lemon” where a user could barely understand in what context these results were mentioned. Hence, for the mentioned reasons we did not use DBpedia in our project for comparing the semantic similarity between concepts.

3.2.2 Dandelion

Dandelion is a web service that provides text processing to performs different operations on texts. These operations include text extraction, text similarity, text classification, and sentiment analysis. All these operations on text are helpful for users to analyze texts [5]. In our project, we are interested in text similarity and the related terms. Hence, we ran a few of different test cases to test how Dandelion behaves. The first test case was running two related terms of “Lemon” and “lime” to check the similarity between these terms. Dandelion returned a result of 0 which means that there is no semantic similarity at all between these terms. Similarly, we ran many other similar terms such as “Flower” and “Rose,” in all the experiments, Dandelion return 0 as the result. Figure (9) illustrates the first test case.

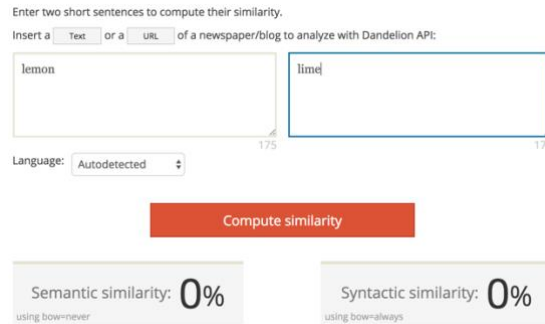


Figure 9: Dandelion semantic results for comparing "Lemon" and "Lime" [5]

Dandelion cannot be used to compare single terms, instead it is used to compare phrases. Consequently, we ran two related phrases “Lemon is healthy” and “I do not like lime”. The returned result was 0. We ran many other semantically related sentences and we got the same result each time.

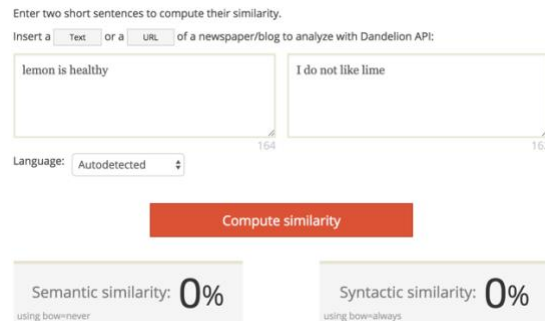


Figure 10: Dandelion results for comparing two phrases [5]

As illustrated, Dandelion does not provide accurate results, unless the exact term was repeated in both texts. Therefore, we did not use Dandelion services in our project.

3.3.3 ParallelDots

ParallelDots is another web service that uses Artificial Intelligence (AI) to analyze texts and detect the semantic similarity between two given texts. ParallelDots is very similar to Dandelion based on the functionality and the relatedness to our project. ParallelDots analyzes the relatedness between texts by eliminating the redundancy. ParallelDots can be helpful for

publisher, bloggers, researchers, and engineers who want to check the relatedness between two texts. Analyzing text using ParallelDots compares the structure and the meaning between these texts [23]. It also extracts the similar sentences and similar ideas from the corpus. ParallelDots provides a number starts from 0 to 5 where 0 has no similarity at all and 5 is almost the same. ParallelDots is an effective tool when running entire texts as shown in the figure (11).



The screenshot shows a web interface titled "TRY OUR SEMANTIC ANALYSIS DEMO". It features two input text boxes. The first box contains the text "I like lemon" and the second box contains "lime is sour". To the right of the input boxes, the word "Relatedness" is displayed with a small circular icon. Below the input boxes is a rounded rectangular button labeled "COMPARE". To the right of the "COMPARE" button, the numerical result "4.93" is shown in a green font.

Figure 11: Results of ParallelDots for comparing two phrases [23]

ParallelDots does not support comparing two words. Figure (12) illustrates how it returns 0 as a result when comparing two terms.



The screenshot shows the same web interface as Figure 11, titled "TRY OUR SEMANTIC ANALYSIS DEMO". The two input text boxes now contain the words "lemon" and "lime". To the right of the input boxes, the word "Relatedness" is displayed with a small circular icon. Below the input boxes is a rounded rectangular button labeled "COMPARE". To the right of the "COMPARE" button, a red error message is displayed: "Please enter complete sentences".

Figure 12: Results of ParallelDots when comparing two terms [23]

3.3.4 WordNet

WordNet is a lexical database in English language that present short definitions, examples on how to use these terms and the relations among synsets. Synsets is a group of all the synonyms that WordNet describes for each term. WordNet can be considered as a dictionary that provides extra features for users. WordNet includes a lexical dictionary that contains nouns, verbs, adverbs, and adjectives but ignores the rest of function words including

propositions [12]. The knowledge structure of WordNet organizes both nouns and verbs into hierarchies defines by relationships such as “Is_A”. For instance, the word “dog” is represented by the following hierarchy. Each level of the hierarchy represents a different synsets and each synset has a unique index as shown in the following figure.



Figure 13: WordNet hierarchy for nouns and verbs [16]

Each hierarchy contains 25 levels for nouns and 15 levels for verbs. Adjectives and adverbs ordered in the next levels. The main goal of WordNet is to build a lexical database that is consistent with semantic human theories. WordNet is known for its relationships between concepts. Therefore, it is known as an ontology that used as a lexical ontology in Computer Science field. WordNet as an ontology is helpful for users to provide them with nouns, verbs and the relationship between them. This feature saves a significant amount of time and efforts for users when building new ontologies.

Despite all of the powerful features that WordNet provides, but there are still some limitations. For instance, WordNet does not include etymology within its data and it does not provide much information on usage. Moreover, the goal of WordNet is to include everyday English concepts without including much about domain specific terminology. WordNet is used to provide lexical comparison of English words limited by dictionary words. Hence, it does not provide brand names, organizations, food, places and other concepts. The purpose of our

project is to enhance FEW databases that might contains information on organization names, countries, food ingredients, and etc. In this case, WordNet will not be efficient in our project to enrich such databases. The following test cases show how WordNet is effective when using dictionary words and its poor performance when using other general terms.

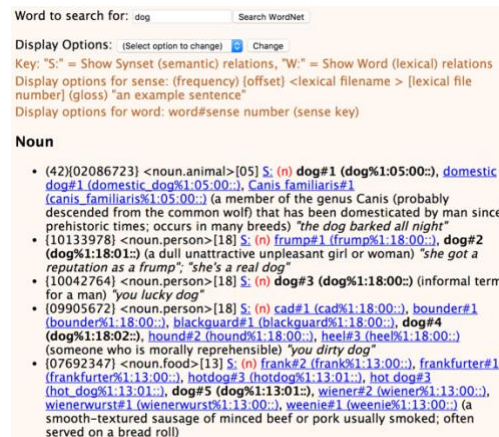


Figure 14: Results returned from WordNet [35]

Based on the above figure, we can see that WordNet provides reliable information when running a dictionary term like “dog”. But when a user runs a general name of a car; for example, “Ferrari” WordNet does not provide any information.

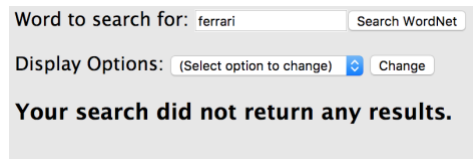


Figure 15: WordNet results when running a general term [35]

Even when the user adds another dictionary term, “car,” to the name “Ferrari” WordNet will still does not provide any information.

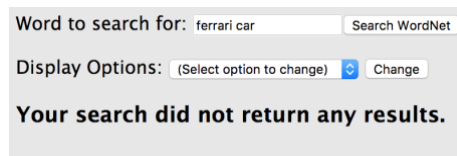


Figure 16: WordNet result when running a phrase contains dictionary-based term [35]

Therefore, based on the previous examples, we did not use WordNet to compare the semantic relationships between concepts in our datasets.

3.3.5 Document Similarity

Document Similarity (DS) is another web service that provides the semantic similarity between two concepts or two texts. DS is powerful when running related concepts since its calculations are based on advanced Natural Programming Language (NLP) [18]. NLP is a field of Computer Science (CS) where it provides artificial intelligence and computational linguistics to improve the interaction between humans and computers [3]. NLP provides powerful approach to extract concepts from a text and to compare two different texts. One of the best features provided by NLP is natural understanding of texts for computers side, based on semantic analyzing of texts. This feature allows NLP to demonstrate powerful semantic relationships between concepts. On top of that, NLP provides many other features in the world of semantics such as machine translation, Named Entity Recognition (NER), natural language generation, textual entailment recognition, relationship extraction, and semantic analysis. Based on these features, NLP is known as one of the best resources for semantic operations[7].

DS is one of the services provided by NLP. The results of DS are accurate and reliable. The only reason why we did not use DS in our project is because it does not provide users with similar concepts. NLP provides this service as a separate feature. Therefore, to be able to check the similarity between two concepts and the related concepts we have to merge these services of NLP. Instead, we chose to utilize ConceptNet that provides all of the mentioned services combined.

3.3.6 ConceptNet

ConceptNet is a semantic network that provides reliable information and facts about entities and concepts for users [8][20]. ConceptNet uses transitive interference between ideas and concepts to enable even the dissimilar entities to share indirect and similar relationship [21]. For instance, when running an entity like “Lemon” on ConceptNet, it provides the weight of the entity and hundreds of synonyms, related entities, similar words in different languages, types of the entity, prosperities of the entity, and many other information and facts. ConceptNet also displays the origin of the entity and where it was derived from. This feature interlinks ConceptNet with many other semantic networks to enhance and enrich ConceptNet knowledge. The weight being used on ConceptNet is based on Markov model. Hence, it is accurate and reliable weight that can be used in our calculations. Another important feature is that ConceptNet provides the conditional probabilities for changing an edge type. For instance, the relationship “is_a” is an edge for a concept X and the probability for this edge is 0.13. In contrast, reversing this relationship to “has_a” the probability drops to 0.5 which means reversing relationships affects the weight of a particular entity [27][34].

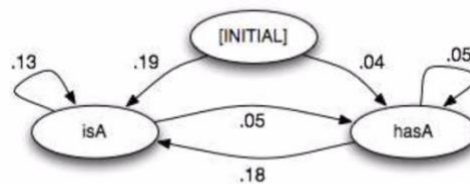


Figure 17: ConceptNet example for a relationship [27]

ConceptNet latest version contains data from different resources to improve the state of knowledge and humans’ knowledge. ConceptNet resources:

1. ConceptNet 5
2. Trusted information and facts from DBpedia.

3. Much of ConceptNet knowledge comes from Dictionary that provides synonyms, antonyms, translations of terms to hundreds of languages, and multiples labeled words.
4. Open Multilingual WordNet that provides dictionary style knowledge.
5. UMBEL that connects ConceptNet to OpenCyc ontology.
6. knowledge on people's intuitive word associations.

As we can see, that ConceptNet delivers trusted knowledge based on the most common networks such as Dictionary, DBpedia, and WordNet. Therefore, ConceptNet is the best service to be used in our project due to its trustworthy information. Furthermore, ConceptNet provides information on dictionary-based terms and other concepts including names, organizations, persons, etc. On top of that, ConceptNet also accepts single terms and phrases to analyze.

CHAPTER 4

APPROACH

In this chapter, we present the method of enriching an RDF triples dataset with new triples related to the existing ones based on the relationships of the existing triples. Our project is separated into two parts: converting databases like CSV format to RDF triples. The second part, which is the main part, is enhancing the converted RDF triples by adding extra triples to enrich the original contents in order to provide stronger and more reliable features.

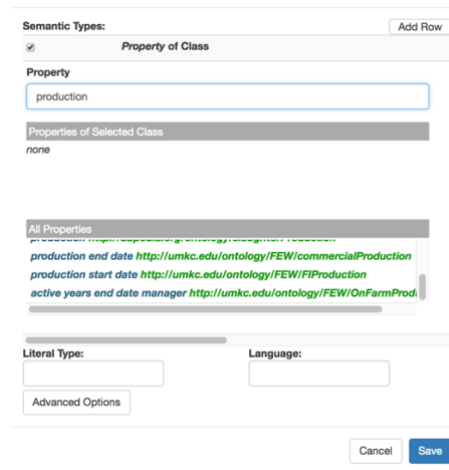
4.1 Overview

We have developed a powerful program and a new approach to enhance the contents of a database based on its contents. Our approach can be separated into two parts. The first part is converting a database table (CSV format) into RDF triples. For this step, we had to use Karma tool since it is a capable tool that delivers helpful services for users to integrate and convert data from various formats such as SML, JSON, and KML to RDF model. One of Karma's best features is that it allows various operations for users such as converting databases into RDF triples in fast and easy way using a simple interface. We used Karma tool for the converting purpose, but due to the lack of the existing ontologies, we developed a new ontology called FEW ontology. FEW ontology provides the most common terms used in FEW databases based on our observations.

The second part of the project is to enrich the outcome RDF dataset with new and related triples. The outcome RDF triples contain the exact same data of the original database but in different format. Entity extraction has to be done first. Then based on the semantic comparison between these entities, new triples will be added containing the most related entities to the existing entities.

4.2 Converting a database table into RDF triples

In order to be able to enrich a database table with new information and facts, the database should be converted to RDF model. Converting a database to RDF model is not a trivial task. After many experiments, we found that Karma integration tool is the most efficient tool to be used in this project because of its reliable and accurate results. Mapping a database to RDF model requires a structure and an ontology so that the contents of the database will be mapped to RDF model based on the given ontology. In this project, we focus on food, water, and energy data. For the mentioned systems, there are no enough ontologies to be used. As a consequence, we developed a new ontology that contains the most common terms and concepts used in FEW data. After that, when mapping FEW databases to RDF model, FEW ontology was used to specify the desirable relationships to be used in the generated RDF triples. The following figure shows few “Production” relationships were generated based on FEW ontology which have the URI of “<http://umkc.edu/ontology/FEW/>”



The screenshot displays the Karma tool's configuration interface for defining a property. At the top, the 'Semantic Types' section is set to 'Property of Class'. Below this, the 'Property' field contains the text 'production'. A section titled 'Properties of Selected Class' shows 'none'. A scrollable list of 'All Properties' includes: 'production end date http://umkc.edu/ontology/FEW/commercialProduction', 'production start date http://umkc.edu/ontology/FEW/FIPProduction', and 'active years end date manager http://umkc.edu/ontology/FEW/OnFarmProd'. At the bottom, there are fields for 'Literal Type' and 'Language', an 'Advanced Options' button, and 'Cancel' and 'Save' buttons.

Figure 18: FEW ontology while using Karma tool

There was no need to create new classes in our FEW ontology since we can import other ontologies to use such as DBpedia to reuse the pre-existing classes and other relationships. Using Karma, a user needs to provide the general URI that will be used as the

common URI for all the subjects in that particular RDF model. The provided URI describes the location and the primary key of a table. For instance, based on the following URI, we can understand that this URI describes the “bread” row in the “example” database that exist in “http://catalog.data.gov/restaurant”.

“<http://catalog.data.gov/restaurant/example/bread>”

4.3 Enriching RDF data triples

Adding extra triples to a dataset based on the existing triples is a complex task especially when using the semantic similarity between the concepts to validate the related information and facts. This phase of the project contains many levels. In this section, we discuss these levels one by one to make it easier and more understandable.

4.3.1 Choosing the target triples

Adding extra information and triples to a dataset cannot be unorganized. Therefore, in the first step a number of triples should be taken as target triples in order to make a comparison between these target triples. Then, based on the comparison between the target triples, extra information will be added and the target triples with the new triples will be stored in a file. To achieve the most accurate results from comparing triples semantically, two triples will be compared at a time. Particularly, not the whole target triples will be compared to each other, but only the extracted subjects.

We have developed a code to extract the concepts from each triple by eliminating the similar part between the rest of the subject. The following figure shows how the data will be after extracting the concepts.

```
<bread> <Expiration> “20/8/2017” .  
<bread> <Quantity> “100” .  
<bread> <Price> “250” .  
<bread> <Best_By> “13/8/2017” .
```

<roll> <Expiration> “20/8/2017” .
<roll> <Quantity> “100” .
<roll> <Price> “302” .
<roll> <Best_By> “13/8/2017” .

At this point, the data is ready for the next step of processing.

4.3.2 Running entities on ConceptNet

In this part of the paper, we discuss our approach and how to calculate an accurate semantic similarity score between any two given concepts. As mentioned before, the target triples represent the first two triples in a dataset and two entities were extracted from these triples. Even though both of the extracted entities are subjects, but we will call the first entity a subject and the second entity an object in order to distinguish between them.

First, our program starts by running the subject (the first entity) on ConceptNet. Running the subject on ConceptNet will return hundreds of related concepts, synonyms, and their relationships weights. The essential part of our program is that it starts analyzing the returned concepts by searching for the object. Once the object occurs in the returned entities, the program stops searching and starts calculating the semantic similarity between the subject and the object based on the average of the searching tree. The level of searching tree is determined by the user. In case the searching tree reached the maximum level without finding the object, the searching operation will stop and the average semantic similarity starting from the subject and ending with the object is 0.

The next step in our algorithm is running the object on ConceptNet and searching for the subject within the return entities. If the subject was found before reaching the maximum level of the searching tree, then the program will stop searching and starts calculating the semantic similarity score based on the number of levels in the searching tree. In contrast, if the

searching tree reached the maximum number of levels without finding the subject, then no relationship was determined between the object and the subject. Hence, there is no relationship at all between the subject and the object. Our algorithm:

```

sub ← subject1
obj ← subject2
subResult ← ConceptNet(sub)
objResult ← ConceptNet(obj)
for each entity in subResult tree do:
    if obj equals entity:
        get (weight, relationship)
        calculateObjSimilarity(weight)
        return(sub, rel, obj, similarity)
for each entity in objResult tree:
    if sub equals entity:
        get (weight, relationship)
        calculateSubSimilarity(weight)
        return(obj, rel, sub, similarity)
else
    return "No relationship"

```

4.3.3 Levels of searching tree

Whenever running an entity on ConceptNet, it returns many related concepts. These related concepts are considered to be in the first level of the searching tree. For instance, running “Corn” returns many concepts such as (maize, crop, food, etc.) and their weight. If one of these related terms is the object, then the similarity will be taken directly based on the edge weight between these two concepts. The following figure shows the first five related terms to “Corn”. If the object we are looking for is “maize,” then the searching part stops and returns 4.95 as the semantic relationship between “corn” and “maize”.

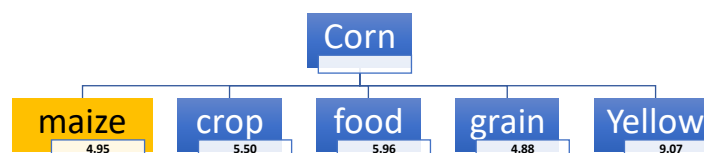


Figure 19: First level of the searching tree

Otherwise, if the program did not find any similar concept to the object, it starts running each of the related words as a new subject since there might be a relationship but not a direct relationship. For example, if the subject is “Flower” and the object is “Green” then we can see that “Green” was not found in the first level. The program will run each related term as the subject again starting from “Rose”. This technique builds a huge tree to search all the related concepts that might be related to the object. In our example, we see that “Green” was found in the third level of the tree. Therefore, the semantic similarity is calculated based on adding all the weight and divide the total by the number of levels. The semantic similarity for this example is 5.17.

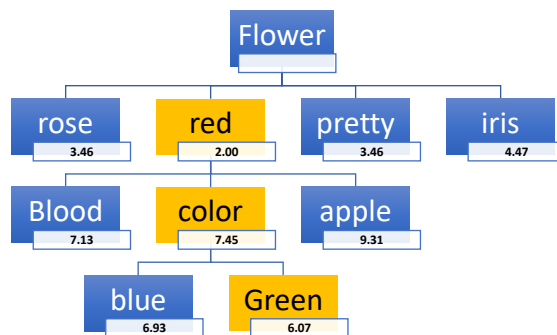


Figure 20: Third level of the searching tree

Lastly, if there is no relation at all between the subject and the object, then the similarity will be 0 and no extra triples will be added. The following example illustrates two entities “Flower” and “fire” having no relationship between them no matter how many levels the tree is.

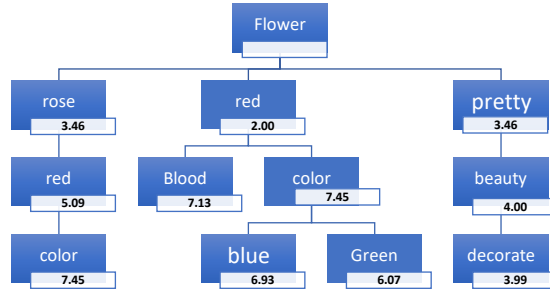


Figure 21: The searching tree for the term "Flower"

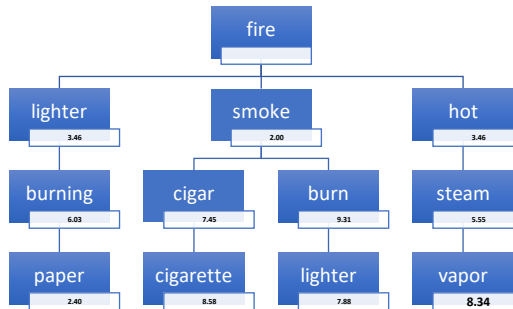


Figure 22: The searching tree for the term "Fire"

4.3.4 Adding the extra triples

In case there is a semantic similarity between these entities after running them on ConceptNet, we add a new triple after the target triples containing the similarity between the subject and the object with the relationship between them. For example, if we have the following triples:

```
<corn> <type_of> "crop"
<maize> <type_of> "plant"
<flower> <exist_in> "park"
<green> <color_of> "tree"
<flower> <exist_in> "park"
<fire> <related_to> "smoke"
```

After our program run, it output the following triples:

```
<corn> <type_of> "crop"
<maize> <type_of> "plant"
<corn> <is_a> "maize" 4.95
<flower> <exist_in> "park"
<green> <color_of> "tree"
```

```

<green> <color_of> "flower" 5.38
<flower> <exist_in> "park"
<fire> <related_to> "smoke"

```

Two triples were added to this dataset based on the relationship between “corn” and “maize,” “flower” and “green” with the semantic similarity between these new relationships. Since each quad contains only four parts, we had to add blank nodes to store the new generated triple with adding the URI of ConceptNet. For example, the triple <corn> <isA> “maize” will be converted using a hash function into eight-digit number, i.e. 12345678, followed by extra information to form the final quads.

```
_:12345678 <similarityScore> "4.95" <Food>.
```

```
_:12345678 <sourceURL> "http://conceptnet.io/c/en/corn" <Food>.
```

4.4 Architecture

In this section, we provide our system architecture that illustrates the path of the data starting from the raw data on USDA then to Karma that converts these databases into RDF model, ConceptNet that provides related entities, passing through our relationship generator, and finally using the hash function and adding quads for the final output.

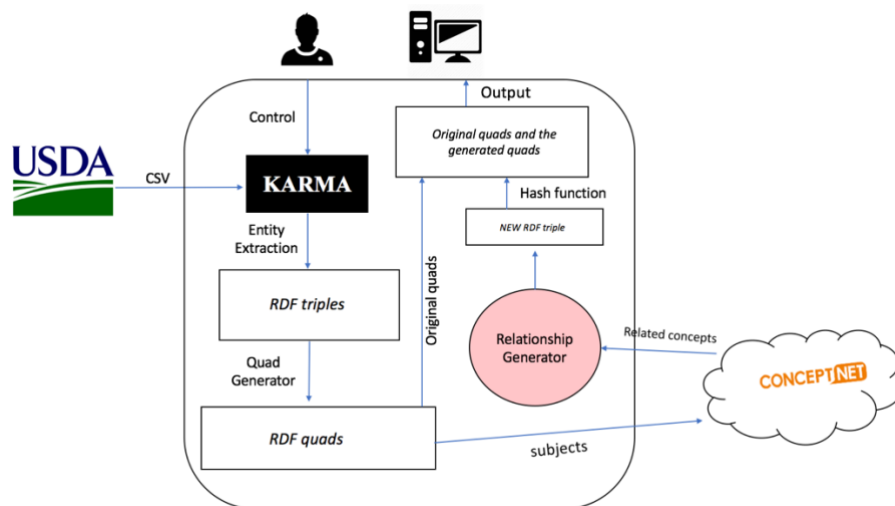


Figure 23: System Architecture

CHAPTER 5

EVALUATION

4.5 Implementation

We implement our program using Python programming language with the help of some libraries such as Requests and hashlib. Requests library was used to send requests to ConceptNet to run the processing terms. Whereas hashlib package was used to create a hash value from the extra triple since each quad contains only 4 parts including the context. The new generated quad contains subject, predicate, object, as a hash value, then the relationship `<semanticSimilarity>` followed by the score, and the context. Using the hash values, we convert the first three parts to a hash number generate based on a formula in hashlib and we create a blank node. The hash value is now the first part of a triple, a relationship is the second value, the similarity score is third value, and the context is the last value. To add the page link of the node with the highest similarity, we added another quad that contains the same hash value as the first part, the relationship `<sourceURL>`, then the URL link, and the context. The following triples illustrate the hash value implementations.

```
<http://umkc.edu/example1><http://umkc.edu/predicate1> "object1" <http://umkc.edu/context>
```

```
<http://umkc.edu/example2><http://umkc.edu/predicate2> "object2" <http://umkc.edu/context>
```

```
<http://umkc.edu/example1><http://umkc.edu/relationship> "example2" <http://umkc.edu/context>
```

```
_:12345678 <semantic_similarity> "4.93450204" <http://umkc.edu/context>
```

```
_:12345678 <source_URL> "http://conceptnet.io/c/en/a_Term" <http://umkc.edu/context>
```

_:12345678 this is the hash value that contains the following triple:

```
<http://umkc.edu/example1><http://umkc.edu/relationship> "example2"
```

4.6 Work load

Our program has been tested by running thousands of lines of RDF triples from different datasets. We provided it with a strong and reliable approach to be used in real life applications. When running a dataset containing 5 lines of RDF triples or 500 triples, the program performance remains the same. The only obvious difference is the running time. The following table shows the average of required time to run different datasets with different sizes.

Table 7: Time needed for different RDF datasets

Experiment number	Number of RDF triples	Required time
Experiment 1	120	1.51737523
Experiment 2	1200	8.73779988
Experiment 3	3000	17.00693297

As we can see that our program accepts any number of RDF triples despite of how big or small the dataset is. The major difference is that the amount of time required which is increasing gradually with the size of a dataset.

4.7 Results

We have developed a new approach to enrich a database table with extra information and facts related to the contents after converting it to an RDF model. This approach helps scientists and users interested in the web of semantics to enhance existing data on the internet in way that makes the searching process faster and provides users with more reliable information about entities.

Our project aims to build a robust and reliable knowledge graph that shed light on FEW systems. We conclude these results with three different examples to illustrate how the actual input and output will be. Given the following table of different datasets:

Table 8: FEW systems input experiments

Experiment number	Dataset type	Triples number	Time required
Experiment 1	Food	185	1.59 sec
Experiment 2	Water	23400	3.23 min
Experiment 3	Energy	900	5.13 sec

The following three tables shows the results for the aforementioned datasets, table 9 shows the Ingredients which is under the Food category.

Table 9: An input example (Ingredients)

Ingredients	Quantity	Price	Starting Date	Ending Date
Bread	100 loaves	\$33	13/8/2017	20/8/2017
roll	30 packs	\$150	13/8/2017	20/8/2017
sugar	20 lb	\$40	13/8/2017	2018
salt	5 lb	\$10	13/8/2017	2018
cake	10 cakes	\$200	13/8/2017	20/8/2017
cupcake	150 cupcakes	\$300	13/8/2017	20/8/2017
cookie	200 cookies	\$100	13/8/2017	20/8/2017
almond	10 lb	#300	13/8/2017	20/8/2017

As we mentioned before, our project contains two main parts. The first part is converting the previous table to RDF model as shown below:

```

<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/bread> <http://umkc.edu/ontology/FEW/price> "533" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/bread> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/bread> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/bread> <http://umkc.edu/ontology/FEW/Best_By> "13/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/roll> <http://umkc.edu/ontology/FEW/Quantity> "100 loaves" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/roll> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/roll> <http://umkc.edu/ontology/FEW/Quantity> "30 packs" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/roll> <http://umkc.edu/ontology/FEW/price> "$150" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/roll> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/sugar> <http://umkc.edu/ontology/FEW/Expiration> "2018" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/sugar> <http://umkc.edu/ontology/FEW/Quantity> "20 lb" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/sugar> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/sugar> <http://umkc.edu/ontology/FEW/Best_By> "13/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/sugar> <http://umkc.edu/ontology/FEW/price> "$40" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/salt> <http://umkc.edu/ontology/FEW/price> "$10" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/salt> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/salt> <http://umkc.edu/ontology/FEW/Expiration> "2018" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/salt> <http://umkc.edu/ontology/FEW/Best_By> "13/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/salt> <http://umkc.edu/ontology/FEW/Quantity> "5 lb" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cake> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cake> <http://umkc.edu/ontology/FEW/price> "$200" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cake> <http://umkc.edu/ontology/FEW/Quantity> "10 cakes" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cupcake> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cupcake> <http://umkc.edu/ontology/FEW/Quantity> "150 cupcakes" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cupcake> <http://umkc.edu/ontology/FEW/price> "$300" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cupcake> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cupcake> <http://umkc.edu/ontology/FEW/Best_By> "13/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cookie> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cookie> <http://umkc.edu/ontology/FEW/Quantity> "200 cookies" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cookie> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/cookie> <http://umkc.edu/ontology/FEW/price> "$100" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/almond> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://dbpedia.org/ontology/Food> <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/almond> <http://umkc.edu/ontology/FEW/Expiration> "20/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/almond> <http://umkc.edu/ontology/FEW/price> "#300" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/almond> <http://umkc.edu/ontology/FEW/Best_By> "13/8/2017" <http://data.gov/FEW_systems/Food>
<http://catalog.data.gov/dataset/food/CheeseCake_restaurant/almond> <http://umkc.edu/ontology/FEW/Quantity> "10 lb" <http://data.gov/FEW_systems/Food>

```

After converting a database to RDF model, the second part of our project starts by processing these triples to generate new triples that enrich the dataset based on the semantic

CHAPTER 6

CONCLUSION AND FUTURE WORK

In our project, we developed a new approach to enrich a dataset of RDF triples by adding extra triples containing similar information and facts based on the existing knowledge. Our project aims to build a reliable knowledge graph that serves FEW systems by using a raw database from the USDA, then converting it to RDF model, and finally enhancing these triples by adding semantic knowledge. In addition, we developed a FEW ontology that can be used while mapping FEW databases to RDF model. FEW ontology contains many vocabularies that can be used to specify the relationships between columns when converting FEW databases.

In the future, we plan to automate the process of converting a database to RDF model. A single table can be mapped to RDF model using Karma, but still there is a lot of work required by a user. Hence, we are planning to work on enhancing the conversion process in a way that multiple databases can be mapped to RDF model at the same time in order to reduce the manual work required by users.

REFERENCES

- [1] Anas Katib, Praveen Rao, and Vasil Slavov. “A Tool for Efficiently Processing SPARQL Queries on RDF Quads,” 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017.
- [2] Anas Katib, Vasil Slavov, and Praveen Rao. “RIQ: Fast Processing of SPARQL Queries on RDF Quadruples,” *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 2016, vol. 37, pp 90-111.
- [3] Clay Shields. “Text-Based Document Similarity Matching Using Sdtext,” 49th Hawaii International Conference on System Sciences, Hawaii, USA, January 5-8, 2016.
- [4] Clifford Lynch. “Big data: How Do Your Data Grow?,” *Nature*, 2008, vol. 455.7209, pp 28-29.
- [5] Dandelion API, <https://dandelion.eu/>. Retrieved August 2017.
- [6] Daniel Abadi, Adam Marcus, Samuel Madden, and Kate Hollenbach. “SW-Store: A Vertically Partitioned DBMS for Semantic Web Data Management,” *The VLDB Journal*, 2009, vol. 18.2, pp 385–406.
- [7] Erik Boiy and Marie-Francine Moens. “A Machine Learning Approach to Sentiment Analysis in Multilingual Web Texts,” *Information Retrieval*, 2009, vol. 12.5, pp 526–558.
- [8] Hugo Liu and Push Singh. “ConceptNet — A Practical Commonsense Reasoning Tool-Kit,” *BT Technology Journal*, 2004, vol. 22.4, pp 211–226.
- [9] Apache Software Foundation, <https://any23.apache.org/index.html>. Retrieved June 2017.
- [10] Wikipedia: DBpedia, <https://en.wikipedia.org/wiki/DBpedia>. Retrieved May 2017.
- [11] Wikipedia, <https://en.wikipedia.org/wiki/Wikipedia>. Retrieved May 2017.
- [12] Wikipedia: WordNet, <https://en.wikipedia.org/wiki/WordNet>. Retrieved May 2017.
- [13] Craig Smith, <https://expandedramblings.com/index.php/youtube-statistics/>. Retrieved July 2017.

- [14] Doug Bernard, <https://learningenglish.voanews.com/a/facebook-has-more-users-than-china-population/2732122.html>. Retrieved June 2017.
- [15] DBpedia, <http://wiki.dbpedia.org/>. Retrieved May 2017.
- [16] WordNet, <https://wordnet.princeton.edu/wordnet/frequently-asked-questions/for-application-developer/>. Retrieved May 2017.
- [17] Tibi Puiu, <https://www.zmescience.com/research/technology/smartphone-power-compared-to-apollo-432/>. Retrieved June 2017.
- [18] Hung Chim and Xiaotie Deng. “Efficient Phrase-Based Document Similarity for Clustering,” *IEEE Transactions on Knowledge and Data Engineering*, 2008, vol. 20.9, pp 1217–1229.
- [19] Talis Information Ltd, <http://research.talis.com/2005/rdf-intro/>. Retrieved June 2017.
- [20] Ming-Hung Hsu, Ming-Feng Tsai, and Hsin-Hsi Chen. “Combining WordNet and ConceptNet for Automatic Query Expansion: A Learning Approach,” *Information Retrieval Technology*, 2008, vol. 4993, pp 213–224.
- [21] Minh Pham, Suresh Alse, Craig Knoblock, and Pedro Szekely. “Semantic Labeling: A Domain-Independent Approach,” *Lecture Notes in Computer Science*, 2016, vol. 9981, pp 446–462.
- [22] Mohamed Hazber, Ruixuan Li, Guandong Xu, and Khaled Alalayah. “An Approach for Automatically Generating R2RML-Based Direct Mapping from Relational Databases,” *International Conference of Young Computer Scientists*, Harbin, China, August 20-22, 2016.
- [23] ParallelDots AI API, <https://www.paralleldots.com/text-analysis-apis>. Retrieved August 2017.
- [24] Phalpheaktra Chhaya, Kyung-Hee Lee, Kwang-Soo Shin, Chi-Hwan Choi, Wan-Sup Cho, and Young-Sung Lee. “Using D2RQ and Ontop to Publish Relational Database as Linked Data,” *Eighth International Conference on Ubiquitous and Future Networks*, Vienna, Austria, July 5-8, 2016.
- [25] Pieter Pauwels, Edward Corry, and James O’Donnell. “Making SimModel Information Available as RDF Graphs,” *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM*, 2014, pp 439–445. CRC Press.

- [26] Praveen Rao, Anas Katib, and Daniel Ernesto Lopez Barron. "A Knowledge Ecosystem for the Food, Energy, and Water System," KDD 2016 Workshop on Data Science for Food, Energy and Water, San Francisco, USA, August 14, 2016.
- [27] Robert Speer and Catherine Havasi. "ConceptNet 5: A Large Semantic Network for Relational Knowledge," *The People's Web Meets NLP*, 2013, pp 161-176. Springer-Verlag Berlin Heidelberg.
- [28] Serique Kleberson, José Santos, and Dilvan de Abreu Moreira. "BioDSL: A Domain-Specific Language for Mapping and Dissemination of Biodiversity Data in the LOD," *Brazilian e-Science Workshop*, Porto Alegre, Brazil, July 4-7, 2016.
- [29] Shi-Jinn Horng. "Big data: Challenges and Practical Application," *International Conference on Science in Information Technology*, Yogyakarta, Indonesia, October 27-28, 2015.
- [30] Clark & Persia LLC, <http://clarkpersia.github.io/csv2rdf/>. Retrieved May 2017.
- [31] S.K. Ramnandan, Amol Mittal, Craig Knoblock, and Pedro Szekely. "Assigning Semantic Labels to Data Sources," *12th European Semantic Web Conference*, Portoroz, Slovenia, May 31-June 4, 2015.
- [32] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. "DBpedia: A Nucleus for a Web of Open Data," *The Semantic Web: Lecture Notes in Computer Science*, 2007, vol. 4825, pp 722-735.
- [33] Stadler Claus, Jörg Unbehauen, Patrick Westphal, Mohamed Ahmed Sherif, and Jens Lehmann. "Simplified RDB2RDF Mapping," *WWW 2015: Linked Data on the Web Workshop*, Florence, Italy, April 20, 2015.
- [34] Steve Spagnola and Carl Lagoze. "Edge Dependent Pathway Scoring for Calculating Semantic Similarity in ConceptNet," *Ninth International Conference on Computational Semantics*, Oxford, UK, January 12-14, 2011.
- [35] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. "WordNet::Similarity: Measuring the Relatedness of Concepts," *Demonstration Papers at HLT-NAACL 2004 on XX - HLT-NAACL*, 2004, pp 38-41.
- [36] Thomas Neumann and Gerhard Weikum. "The RDF-3X engine for scalable management of RDF data," *The VLDB Journal*, 2009, vol. 19.1, pp 91–113.
- [37] Vadim Eisenberg and Yaron Kanza. "D2RQ/Update: Updating Relational Data via Virtual RDF," *21st International Conference Companion on World Wide Web*, Lyon, France, April 16-20, 2012.

- [38] Vasil Slavov, Anas Katib, Praveen Rao, Srivenu Paturi, and Dinesh Barenkala. "Fast Processing of SPARQL Queries on RDF Quadruples," 17th International Workshop on the Web and Databases, Snowbird, USA, June 22, 2014.
- [39] Ruben Verborgh and Max De Wilde. "Using OpenRefine," 2013. Packt Publishing Ltd.

VITA

Mohamed Gharibi

Mohamed Gharibi joined UMKC in 2015 to pursue a Master of Science degree in Computer Science. His research interest includes Data Science -Big Data. He is also pursuing a degree in higher-education teaching, Preparing Future Faculty, from the School of Education supported with a Scholar Award from the School of Graduate Studies at UMKC. Mohamed is working under the supervision of Dr. Praveen Rao.