Technische Universitat Dresden

International MSc Program in Computational Logic (MCL)

Institute for Artificial Intelligence

Extracting Vector Representation Of DBPedia Properties from Word2Vec

Student:
Happy Rani Das

Supervisor:
Prof. Dr. Sebastian Rudolph

February 13, 2018

Abstract

Vector representation of words has been learned by word2vec and useful in many natural language processing and information retrieval. Though, surprising fact is that word2vec have not been applied for Dbpedia data. In this paper, I am focusing on how word2vec can be applied for Dbpedia properties in order to get missing information. The main goals of Extracting Vector Representation of Dbpedia Properties are to find corresponding resources from the Dbpedia and to apply Word2Vec technique to find missing information.

1 Introduction

1.1 Background

Word2vec is a neural network, which takes text corpus as input and produces a set of vectors as a result. In the beginning word2vec build a vocabulary from a large text corpus and then produces group of vectors. It first constructs a vocabulary from the training text data and then learns vector representation of words. There are two ways to represent word2vec model architecture 1. continuous bag-of-words (predicts a missing word given a window of context words or word sequence) and 2. skip-gram (predict the neighboring window of target context by using a word) These word2vec model architecture used in machine learning areas, natural language processing and advance research areas. [1].

Continuous bag-of-words (CBOW) and continuous skip-gram model architecture are very popular nowadays in the machine learning areas and further research. Another depiction of words is dense vector came to know by word2vec. Dense vector have exceptionally been displayed to demonstrate same sense and it is beneficent in the immense range from data analytics to natural language processing.

As an example, words that have equivalent explanation will have analogous vectors because of cosine similarity and the words whose doesn't have equivalent explanation will have unalike vectors. It is quite surprising that, word vectors follow the likeness rule. For instance, presume the likeness "Berlin is to Germany as Paris is to France". It gives us the result like following

$$v_{Germany} - v_{Berlin} + v_{Paris} = v_{France}$$

where $v_{Germany}; v_{Berlin}; v_{Paris} and v_{France}$ are the word vectors for Germany, Berlin, Paris, and France respectively. [2].

1.2 Project Description

This project is focused on the Extracting Vector Representation of DBPedia Properties from word2vec. Word2Vec has been applied in several areas with the purpose of detect similarity and to find out nearest word. However, Word2Vec has not been applied for DBPedia properties. The main goal of this project is to introduce techniques that can be used for learning DBPedia data and to find out corresponding missing information from DBPedia. DBpedia (”DB” stand for ”database”) is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web [defined by http://wiki.dbpedia.org/about].

If the user has two sentences like-

1. Berlin is the capital
2. Paris is the capital.

The result of the Word2vec similarity will be the words ending up near to one another. Suppose, if we train a model with (input:Berlin,output:Capital) and (input:Paris,output:Capital) this will eventually give insight the model to understand that, Berlin and Paris both as connected to capital, thus Berlin and Paris closely in the Word2Vec similarity.

The Extracting Vector Representation of DBPedia Properties from word2vec is developed in python which takes DBPedia properties as input and returns nearest or similar missing information as output.

2 Related Work

3 System Architecture

Figure 1 shows the flowchart diagram of our project. Dbpedia data is given as input which is then passed into pre-trained model GoogleNews-vectors-negative300.bin. then we get the vector representation of the given data. Thereafter we try to find out the nearest word based on the given data. After that we try to figured out the missing information like if one word is related to another word, the same type of word would be related to other word which we don't about it. For instance, if Donald Trump is to republican as Barack Obama is to what? And the output would be Democratic.

3.1 DBPedia Properties

This is most important and tricky part in this project. Properties is a connections between objects in DBPedia. To get all of the objects for specified properties we use SPARQL query language. For our project we think about five SPARQL query against DBPedia.

Query 1:

The following code shows how to execute SPARQL query against DBpedia for properties 'country' and 'capital' in order to get all of the country and capital list :

```
SELECT DISTINCT ?country ?capital WHERE

{

?city rdf:type dbo:City ;

rdfs:label ?label ;

dbo:country ?country .

?country dbo:capital ?capital .

} order by ?country
```

References

[1] https://code.google.com/archive/p/word2vec/

[2] http://www.1-4-5.net/ dmm/ml/how_does_word2vec_work.pdf

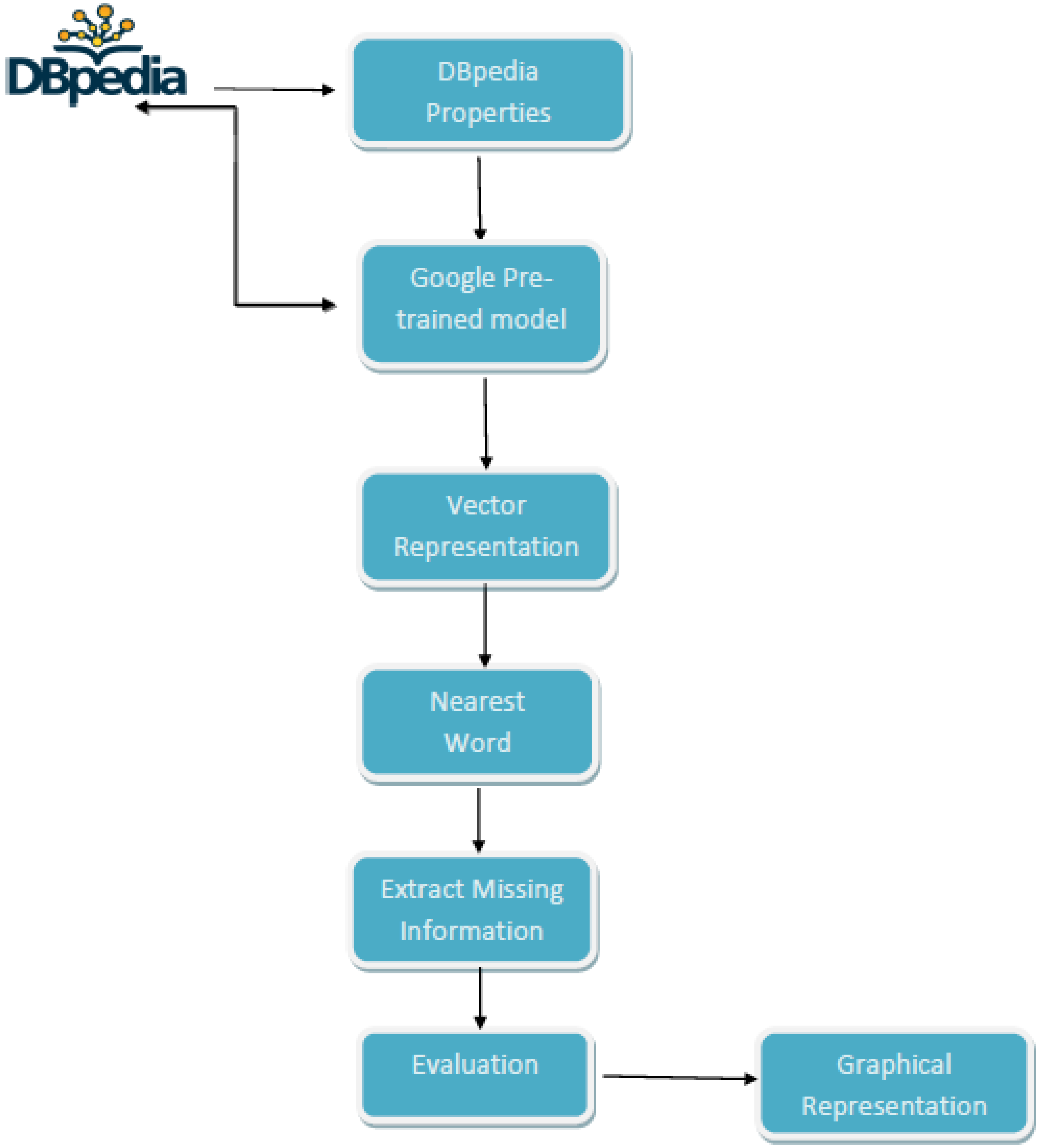
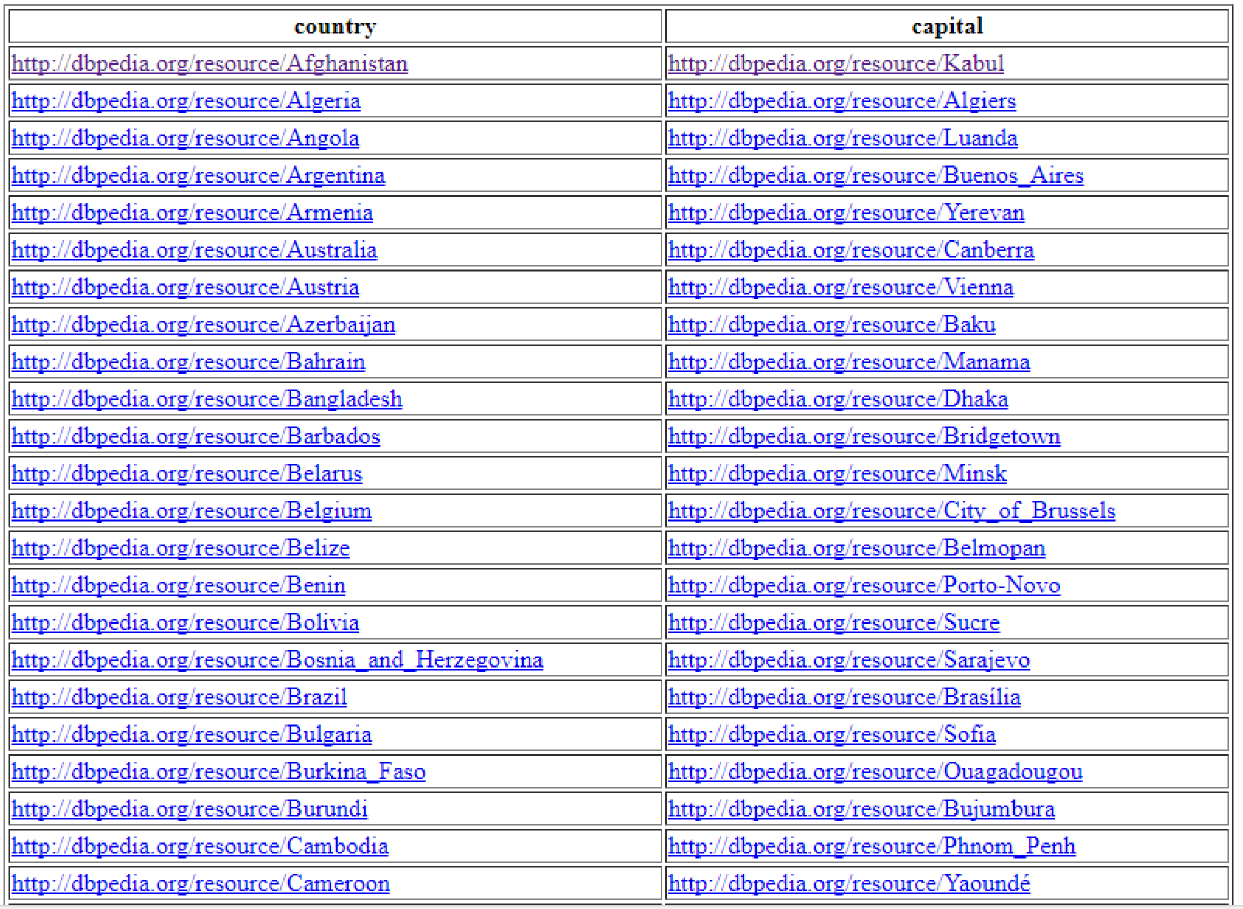


Figure 1: Figure 1: The General Scheme of Extracting Vector Representation of DBpedia Properties from Word2Vec



country	capital
http://dbpedia.org/resource/Afghanistan	http://dbpedia.org/resource/Kabul
http://dbpedia.org/resource/Algeria	http://dbpedia.org/resource/Algiers
http://dbpedia.org/resource/Angola	http://dbpedia.org/resource/Luanda
http://dbpedia.org/resource/Argentina	http://dbpedia.org/resource/Buenos_Aires
http://dbpedia.org/resource/Armenia	http://dbpedia.org/resource/Yerevan
http://dbpedia.org/resource/Australia	http://dbpedia.org/resource/Canberra
http://dbpedia.org/resource/Austria	http://dbpedia.org/resource/Vienna
http://dbpedia.org/resource/Azerbaijan	http://dbpedia.org/resource/Baku
http://dbpedia.org/resource/Bahrain	http://dbpedia.org/resource/Manama
http://dbpedia.org/resource/Bangladesh	http://dbpedia.org/resource/Dhaka
http://dbpedia.org/resource/Barbados	http://dbpedia.org/resource/Bridgetown
http://dbpedia.org/resource/Belarus	http://dbpedia.org/resource/Minsk
http://dbpedia.org/resource/Belgium	http://dbpedia.org/resource/City_of_Brussels
http://dbpedia.org/resource/Belize	http://dbpedia.org/resource/Belmopan
http://dbpedia.org/resource/Benin	http://dbpedia.org/resource/Porto-Novo
http://dbpedia.org/resource/Bolivia	http://dbpedia.org/resource/Sucre
http://dbpedia.org/resource/Bosnia_and_Herzegovina	http://dbpedia.org/resource/Sarajevo
http://dbpedia.org/resource/Brazil	http://dbpedia.org/resource/Brasília
http://dbpedia.org/resource/Bulgaria	http://dbpedia.org/resource/Sofia
http://dbpedia.org/resource/Burkina_Faso	http://dbpedia.org/resource/Ouagadougou
http://dbpedia.org/resource/Burundi	http://dbpedia.org/resource/Bujumbura
http://dbpedia.org/resource/Cambodia	http://dbpedia.org/resource/Phnom_Penh
http://dbpedia.org/resource/Cameroon	http://dbpedia.org/resource/Yaoundé

Table 1: Output for the input properties country and capital.

[3] viewexportask othersask others Philipp Heim, Sebastian Hellmann, Jens Lehmann, Steffen Lohmann, Timo Stegemann: RelFinder: Revealing Relationships in RDF Knowledge Bases. SAMT 2009: 182-187