B.Sc. in Computer Science and Engineering Thesis

Building A Knowledge Graph Using Twitter Data

Submitted by

Sukanta Kundu
201205068

Anindya Biswas
201405006

Qazi Fahim Farhan
201405052

Supervised by

Dr. Muhammad Masroor Ali

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

Dhaka, Bangladesh

February 2018

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis, titled, "Building A Knowledge Graph Using Twitter Data", is the outcome of the investigation and research carried out by us under the supervision of Dr. Muhammad Masroor Ali.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Sukanta Kundu
201205068

Anindya Biswas
201405006

Qazi Fahim Farhan
201405052

CERTIFICATION

This thesis titled, **"Building A Knowledge Graph Using Twitter Data"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in February 2018.

Group Members:

Sukanta Kundu

Anindya Biswas

Qazi Fahim Farhan

Supervisor:

Dr. Muhammad Masroor Ali
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

ACKNOWLEDGEMENT

We are indebted to our supervisor Professor Dr. Muhammad Mashroor Ali got introduing us to the Knowledge Graph. We've learned how to think critically like a researcher, carry on a research, read scientific papers, and how to write , speak and present properly. We show our gratitude for his dedication and patience in correcting our mistakes, reviewing our ideas, methodologies, proofs, etc. We couldn't have done it without him.

Dhaka
February 2018

Sukanta Kundu

Anindya Biswas

Qazi Fahim Farhan

Contents

List of Figures

List of Tables

List of Algorithms

ABSTRACT

In this thesis, we wish to work on creating a Knowledge Graph using Twitter micro-data. A Knowledge Graph is a representation of human knowledge in the form of a graph. Just like any graph, it has nodes (entities), attributes and edges (relations). Since tweets give us limited data (maximum length of 300 characters), it becomes very difficult to extract adequate information to create a knowledge graph.

Chapter 1

Introduction

The present world is a place with immeasurable date or informations due to world wide web. Each and every day this amount is increasing. It is getting harder day by day for people to extract exact information he/she needs and to analysis them to have a desired goal. However this big data has become a challenge for computer science to organize, store and most importantly efficient searching. (As far as the word efficient searching means searching a particular word with all its variationns.)

A single word can represent a person, place, name of organization, restaurant . Again a word slightly different spelling can represent totally different thing. It is very hard for a single person or a particular softwere to do this. So, we need to make machine learn way of efficient searching.

It may be helpful for us to write grammers for machine to learn than writing all the rules of searching by ourselves. Now for machine to learn to work or search in order to search efficiently we need data or information to be organized in a way that machine can analyze those data. Here knowledge graph comes into play.

Chapter 2

Related Works

Previously there have been many researches conducted on knowledge graphs. Let us discuss them one by one.

In the paper 'Knowledge Graph Identification' [1], the authors have extracted uncertain entities and their relations to create an 'extraction graph'. Next they removed noise, added missing information and determined candidate facts that resulted into a Knowledge Graph Identification. To disambiguate and relate a concept with similar concept, the researchers have used Probabilistic Soft Logic (PSL) [2] ontological constraints, identify coreferences etc.

One common problem in building a Knowledge Graph is to find appropriate links between various concepts. Mihalcea annd Csomai et al [3] have worked on link detection. They used machine learning on a wikipedia dump to link an article with its corresponding mention. There they have also calculated link probability of various phrases or mentions. Next they disambiguated links by taking help of the surrounding words (the words themselves and their parts of speech), comparing the common meaning with the relatedness to the corresponding context and check with training data. For example, the word 'tree' has a common meaning 'plant' and so 97% of the time, it will link to a page on plants. But if we find 'binary tree' in a document, the relatedness of 'tree' to 'binary' suggests that this phrase is related to data structure. That is how almost everyone disambiguates links.

Next we have an important work of Larry Heck and Hongzhao Huang [4]. First they developed a robust way to represent concepts. Many NLP tools use string concatenation to represent ideas. But most of the time it fails to represent appropriate concepts. For example 'Madrid' is a place, 'Real' means something authentic or true, but 'Real Madrid' refers to a football club. This is a counter example where string concatenation fails. A possible alternative is to use word hashing with n-gram (tri-gram in this case) word representation. This way turns words into concept-vectors. This representation is robust because it can be used even for unseen words. After that they did neural embedding of knowledge graph. To do so, they tracked a concept and its corresponding subgraph, encode the knowledge as featured vector. then they trained Deep Neural Network to get semantic relationships among various concepts. After that, they took tweets and used their deep neural network on those tweets.

References

[1] L. G. W. C. Jay Pujara, Hui Miao, “Knowledge graph identification,” *International Semantic Web*, pp. 0–1, 2013.

[2] M. B. B. H. L. G. Angelika Kimming, Stephen H. Batch, “A short introduction to probabilistic soft logic,” pp. 0–1, 2012.

[3] I. H. W. David Milne, “Learning to link with wikipedia,” *CIKM*, pp. 0–1, 2014.

[4] H. H. Larry Heck, “Deep learning of knowledge graph embeddings for semantic parsing of twitter dialogs,” *Signal and Information Processing*, pp. 0–1, 2014.

Appendix A

Algorithms

A.1 Sample Algorithm

In Algorithm 1 we show how to calcute $y = x^n$.

Algorithm 1 Calculate $y = x^n$

```
Require: $n \geq 0 \vee x \neq 0$
Ensure: $y = x^n$
  $y \leftarrow 1$
  if $n < 0$ then
    $X \leftarrow 1/x$
    $N \leftarrow -n$
  else
    $X \leftarrow x$
    $N \leftarrow n$
  end if
  while $N \neq 0$ do
    if $N$ is even then
      $X \leftarrow X \times X$
      $N \leftarrow N/2$
    else $\{N$ is odd$\}$
      $y \leftarrow y \times X$
      $N \leftarrow N - 1$
    end if
  end while
```

Appendix B

Codes

B.1 Sample Code

We use this code to find out…

```c
#include <stdio.h>
int Fibonacci(int);

main()
{
  int n, i = 0, c;

  printf("Enter␣the␣value␣of␣n:␣");
  scanf("%d",&n);

  printf("\nFibonacci␣series\n");

  for (c = 1 ; c <= n ; c++)
    {
      printf("%d\n", Fibonacci(i));
      i++;
    }

  return 0;
}

int Fibonacci(int n)
{
  if (n == 0)
    return 0;
  else if (n == 1)
    return 1;
  else
    return (Fibonacci(n-1) + Fibonacci(n-2));
}
```

B.2 Another Sample Code

```
SELECT associations2.object_id, associations2.term_id,
       associations2.cat_ID, associations2.term_taxonomy_id
FROM (SELECT objects_tags.object_id, objects_tags.term_id,
     wp_cb_tags2cats.cat_ID, categories.term_taxonomy_id
FROM (SELECT wp_term_relationships.object_id,
     wp_term_taxonomy.term_id, wp_term_taxonomy.term_taxonomy_id
FROM wp_term_relationships
LEFT JOIN wp_term_taxonomy ON
     wp_term_relationships.term_taxonomy_id =
     wp_term_taxonomy.term_taxonomy_id
ORDER BY object_id ASC, term_id ASC)
AS objects_tags
LEFT JOIN wp_cb_tags2cats ON objects_tags.term_id =
     wp_cb_tags2cats.tag_ID
LEFT JOIN (SELECT wp_term_relationships.object_id,
     wp_term_taxonomy.term_id as cat_ID,
     wp_term_taxonomy.term_taxonomy_id
FROM wp_term_relationships
LEFT JOIN wp_term_taxonomy ON
     wp_term_relationships.term_taxonomy_id =
     wp_term_taxonomy.term_taxonomy_id
WHERE wp_term_taxonomy.taxonomy = 'category'
GROUP BY object_id, cat_ID, term_taxonomy_id
ORDER BY object_id, cat_ID, term_taxonomy_id)
AS categories on wp_cb_tags2cats.cat_ID = categories.term_id
WHERE objects_tags.term_id = wp_cb_tags2cats.tag_ID
GROUP BY object_id, term_id, cat_ID, term_taxonomy_id
ORDER BY object_id ASC, term_id ASC, cat_ID ASC)
AS associations2
LEFT JOIN categories ON associations2.object_id =
    categories.object_id
WHERE associations2.cat_ID <> categories.cat_ID
GROUP BY object_id, term_id, cat_ID, term_taxonomy_id
ORDER BY object_id, term_id, cat_ID, term_taxonomy_id
```