Extracting Relations and Information from KB

Vedsar Kushwaha
Indian Institute of Science, Bangalore
vedsarkushwaha@ssl.serc.iisc.in

Partha Pratim Talukdar
Indian Institute of Science, Bangalore
ppt@serc.iisc.in

1 Introduction

Harvesting knowledge from the graph is still a hard problem to solve. Different techniques have been proposed by researchers. But getting the correct and exact information from the graph based on the user query still needs some more research to solve. In this work, we are also addressing a similar kind of problem. The problem is to extract information from the knowledge graph based on the user query. Also extract relations that contains those edges. There are different ways to solve these kinds of problem. In this work, we are using string matching techniques to solve the problem. We are also using natural language toolkit to parse the sentence and knowledge graph. NELL SVO triples are used to connect user query with the knowledge graph. Rest of the sections are as follows:(2) Related Work, this section describes the current work for the problem (3) Data set Used, this section describes the different dataset useds(4) Tools Used, this section describes the different tools used(5) Method, this section describe our methodolgy (6) Experiments, this section describes the experiments we performed and the results that we obtained (7) Further improvements, this section explains the further which can be improved to get better results.

2 Related Work

Some work has already been done in this field. Semantic Parsing via Paraphrasing (Berant and Liang, 2014), explains a technique where for an given input utterance, they first convert it to some set of candidate logical forms. Then, they heuristically generate canonical utterances for each logical form. Finally, they output the canonical utterance that best paraphrases the input utterance, and thereby the logical form that generated it.
Work on Open Information Extraction (IE) (Anthony, Stephen and Oren, 2011) does the task of extracting assertions from massive corpora without requiring a pre-specified vocabulary. This solves the problem by identifying relation phrases that denote relations in English sentences. OLLIE is the extension to this work (Mausam, 2012). OLLIE initially achieves high yield by extracting relations from nouns, adjectives, etc and then using a context-analysis step, it increases the precision by including contextual information from the sentence in the extractions.

3 Data Set Used

We are using NELL as a knowledge graph(KB) (Carlson, 10) to extract information for user query. We are also using NELL SVO triples (Talukdar, 2012) to connect parsed user query with the knowledge graph.

4 Tools Used

Whole project is build using python as programming language. We used python Natural Language toolkit(nltk) for text processing. Description of these tools are as follows:

- Word Tokenizer: This tool is used to tokenize user query into list of words.
- Stop Word: This tool gives the list of stop-words, so that programmer can easily remove

stop words from any word list.

- Parts of speech tagger: This tool is used to tag set of words with respective parts of speech.
- Snowball stemmer: This tool is used to remove -s, -es, -ing etc from word set to make it plain set of words.

5 Method

5.1 Problem Statement

Given user query and knowledge graph, extract relations from the user query based on the knowledge graph and further extract answer from the graph for that user query.

5.2 Approach

We are breaking the approach into three steps. In first step, we are parsing the user query and extracting usefull information from that. In second step, we are loading graph into memory. This loading of graph is done in such a way that extracting relations and answer to the user query will become easier and faster. In final step, we are merging the above two steps. In this step, we are getting relations and answers to user query from knowledge graph by putting information extracted from user query in step into step two. Details of each step are as follows:

5.3 Phase 1

Initially, we are loading knowledge graph into memory. Here the knowledge graph contains relations and each relation has a list of subject-object pair. We are loading knowledge graph into memory because we need to search for relations for a particular subject or object. So, we need to load the knowledge graph in such a way so that searching time for relations will be as less as possible. We are using two hash map for this, one for subject and other for object. Hash map for subject has key as subject string and value as relation set. Similarly hash map for object has key as object string and value as relation set. Note that subject-object pair has one relation connected with it but only subject or only object can have more than one relation. Now to search for a relation based on subject-object string, we simply find the set of relations for subject and set of relations for object using corresponding hash map. Then we do a intersection between these two relations to get common relations. These common relations are the required relations present in the graph for the subject-object string.
Steps for loading graph:

1. Create two dictionary data structure, one for subject and other for object.
2. Subject dictionary will have subject as key and respective relation set as value. Same for Object dictionary data structure. (Note that the graph that we have is in the form of list of relation files, each relation files contains list of node pairs).

We are also building verb distribution for each subject-object pair of each relation to capture semantics of user query. Here, for each subject-object pair, we are extracting SVO triples from SVO NELL database. The set of verbs obtained in each step are marked as similar.
Formation of verb distribution:

1. For each relation of knowledge graph, do the following steps:
2. For each node pair(which are in form of subject-object) of each relation, do the following steps:
3. Search for SVO in SVO NELL database with respect to node pair to get list of SVO triplets. Mark all the verbs obtained using one node pair as similar verbs and put them into one set.

5.4 Phase 2

In second phase, we are taking input from user in the form of question sentence. After getting this, we are using python natural language toolkit(nltk), parts of speech tagger(postag) to tag verbs from the input user query. After this we are getting some collection of verbs, call it set A. Now, we are matching this verb collection(set A) with the verb set collections build in phase one(call it set B). If there is any common verbs between both set A and set B, then we will expand the size of set A and add all verbs of that partcular collection of set B into set A. This phase will helps us to capture semantics of user query. After getting the collection of verbs, we are searching for subject and object for each verb in NELL SVO triples. These Subject-Verb-Object (SVO) triples are generated as part of the NELL project. For this project, we are using 604 million triples of NELL SVO. Each line in the file consists of the Subject, Verb(+Preposition), Object, Frequency. So, for each verb extracted from user query, we extract its corresponding subject and object from NELL SVO triples. Since, this list extracted SVO triples will be huge because one verb will corresponde to very big set of subject and object and we may have more than one verb per user query. So, after getting SVO triple list, we filter it out using user query. Here the idea is, user query either contain verb and subject and asking for object or contain verb and object and asking for subject. So, we match subject and object of SVO triples. We remove all those triples whoes subject or object did not matched to user query. Now this filtered SVO triples will have verb relavent to user query and subject and object again relevant to user query.

1. Take query from user.
2. Do a word tokenize on the input query and extract verbs using parts of speech tagger.
3. Now, expand this verb collection by getting similar verbs using the verb distibution build in phase 1.
4. Using this verb collection, get SVO triples from NELL SVO.
5. Filter out the SVO, if neither subject nor object is present in the user query.

5.5 Final Phase

After phase one, we have knowledge graph loaded in memory in the form of hash map. After phase two, we have filtered SVO triples based on the user query. Now, in final phase we extract relations based on the filtered SVO triples. Our first output will be these relations. For each relation check whether subject of relation is present in the query or not. If subject is present in the user query then append object in your $answertoquery$ list. Similarly, if object of the relation is present in the user query then append subject in the $answertoquery$ list. After scanning through each relations obtained by using SVO and hashmap, the second and final output will be $answertoquery$ list.

1. Get a empty list of relation R and $answertoquery$
2. Get the relations based on SVO triples obtained in phase 2 and put it into relation R. Note that relations can be extracted in constant time using dictionary data structure loaded in memeory in phase 1.
3. For each relation R obtained, if subject is present in user query then append the object into $answertoquery$, or if object is present in user query then append the subject into $answertoquery$.
4. This set of relations R and $answertoquery$ will be the final output, where R contains the relation which holds the edge name and $answertoquery$ contains the answer to user query.

6 Experiment

We evaluate our system based on 30 different queries provided by different users. For all those queries whose words matches with the relations of NELL KB are giving exact results. For example queries like ”which athlete also known as crazy legs?” or ”Which university offers journalism as academic program?” or ”What are the academic programs offered at ohio state university?” are giving good results. The reason behind this is that there are relations with name ”university offers academic program” and ”athlete also known as”. For queries where question asked are similar to the previous example but words are changed, for example ”What are the degrees provided by ohio state university?”, doesn’t gives exact answer. The query matches with the required relation as well as with some other relations too, because of verb distribution set. Note that in previous example of academic program also this is happening, but we are giving high score if there is an exact match between user query and graph relations. We also tested for queries whose answers are not there in the KB. For example, ”who is the leader for political party BJP?”. For these types of queries, the system didn’t gives any results and simply outputs ”No information found”.

7 Future Work

Our system works well for the user queries where words in the queries matches with the words in the relations of the knowledge graph and node pairs of the knowledge graph. It also gives good results where alternate verbs are used instead of the verbs used in KB relations. But if someone did spelling mistakes in words then the exact answer will not be extracted since at the lower level we are doing string matching. So some kind of spell checker should be there to correct the spelling mistakes. One more improvement will be in the side of semantics. We are using verb distribution created in phase 1 to capture semantics. But due to verb distribution, other results are also coming which is creating noise in original results. So either improvement in semantics extraction or improvement in result scoring will further improve the results.

8 Acknowledgement

I would like to express my special thanks of gratitude to my course teacher Prof. Partha Pratim Talukdar who gave me the golden opportunity to do this wonderful project, which also helped me in doing a lot of Research and I came to know about so many new things. I am really thankful to him. I would also like to thank my colleague manjunath hegde for providing user queries which helps me in evaluating the goodness of the project.

I am making this project not only for marks but to also increase my knowledge . THANKS AGAIN TO ALL WHO HELPED ME.

References

Jonathan Berant and Percy Liang. 2014. *Semantic Parsing via Paraphrasing.* Association for Computational Linguistics (ACL).

Anthony Fader, Stephen Soderland and Oren Etzioni. 2011. *Identifying Relations for Open Information Extraction.* Conference on Empirical Methods in Natural Language Processing.

Mausam, M. Schmitz, R. Bart, S. Soderland and O. Etzioni. 2012. *Open Language Learning for Information Extraction.* Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CONLL).

Talukdar Partha Pratim, Wijaya Derry, Mitchell Tom. 2012 *Acquiring Temporal Constraints Between Relations*. Proceedings of the 21st ACM International Conference on Information and Knowledge Management.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, Tom M. Mitchell. 2010 *Toward an architecture for never-ending language learning*. *In AAAI.*