PROJECT REPORT

INFORMATION EXTRACTION

NATURAL LANGUAGE PROCESSING

UNIVERSITY OF TEXAS AT DALLAS

Prof. Mithun Balakrishna

TEAM RAVANA

Student	NetID
Bharat Simha Reddy R S	BXR170015
Ranjith Reddy Kommidi	RXK180004

Project Description:

To Design and Implement Information extraction system to derive templates from given articles that includes

- 10 articles related to Organizations
- 10 articles related to Persons
- 10 articles related to Locations

Extracting templates:

- Set of information templates
- Template #1: *BUY (Buyer, Item, Price, Quantity, Source)*
- Template #2: *WORK (Person, Organization, Position, Location)*
- Template #3: *PART (Location, Location)*

Given the data the following are the tasks need to be implemented:

Task 1: Implement a deep NLP pipeline to extract the following NLP based features from the text articles/documents:

- Split the document into sentences
- Tokenize the sentences into words
- Lemmatize the words to extract lemmas as features
- Part-of-speech (POS) tag the words to extract POS tag features
- Perform dependency parsing or full-syntactic parsing to get parse-tree based patterns as features
- Using WordNet, extract hypernyms, hyponyms, meronyms, AND holonyms as features
- Some additional features that you can think of, which may make your representation better

Task 2: Implement a machine-learning, statistical, or heuristic (or a combination) based approach to extract filled information templates from the corpus of text articles:

- Run the above described deeper NLP on the corpus of text articles and extract NLP features
- Implement a machine-learning, statistical, or heuristic (or a combination) based approach to extract filled information templates from the corpus of text articles

Task 3: Implement a program that will accept an input text document and:

- Run the above described deep NLP on the input text document
- Extract information templates from the input text document using your information extraction approach implemented in Task 2

- Output a JSON file with extracted/filled information templates from the input text document

Sample Extracted Templates:

Example #1:
Sentence(s): In 2017, Amazon acquired Whole Foods Market for US$13.4 billion, which vastly increased Amazon's presence as a brick- and-mortar retailer.
Extracted Template: BUY (“Amazon”, “Whole Foods Market”, “US$13.7 billion”, “”, “”)

Example #2:
Sentence(s): Steven Paul Jobs (; February 24, 1955 – October 5, 2011) was an American business magnate and investor. He was the chairman, chief executive officer (CEO), and co-founder of Apple Inc.
Extracted Template: WORK (“Steven Paul Jobs”, “Apple Inc.”, “chairman; chief executive officer (CEO); co-founder”, “”)
Example #3:

Sentence(s): Richardson is a principal city in Dallas and Collin counties in the U.S. state of Texas.
Extracted Template: PART (“Richardson”, “Dallas
Extracted Template: PART (“Richardson”, “U.S. state of Texas | Texas”)
Extracted Template: PART (“Texas”, “U.S.”)

Tools:

We are using

- Python 3.7.3
- NLTK library
- Wordnet
- Stopwords
- WordNetLemmatizer
- PorterStemmer
- CoreNLP package for Python by Stanford NLP (SNLP 3.9.2)
- DependencyParser
- Spacy
- NeuroCoref

Proposed Solution:

For the templates BUY and WORK we implemented NLP pipeline with spacy loaded with NER and dependency parser. The tags and dependencies are extracted from the spacy and dependency parse tree to extract the relations between tokens.
We built the heuristics required to determine the type of question being asked. With heuristic approach, we can have rules for each of the template and sentence is checked for the possibility of template belongingness. This is because each sentence can belong to more than one template.

Example #1:
Sentence(s): In 2017, Amazon acquired Whole Foods Market for US$13.4 billion, which vastly increased Amazon's presence as a brick- and-mortar retailer.
Extracted Template: BUY ("Amazon", "Whole Foods Market", "US$13.7 billion", "", "")

Example #2:
Sentence(s): Steven Paul Jobs (; February 24, 1955 – October 5, 2011) was an American business magnate and investor. He was the chairman, chief executive officer (CEO), and co-founder of Apple Inc.
Extracted Template: WORK ("Steven Paul Jobs", "Apple Inc.", "chairman; chief executive officer (CEO); co-founder", "")

For the template PART we implemented NLP pipeline with spacy loaded with NER and dependency parser. We implemented Knowledge based graph with nodes as GPE tags and edges as location dependencies. It similar to the dependency tree so that retrieval of the entities became easy compared to the other templates extraction.

Sentence(s): Richardson is a principal city in Dallas and Collin counties in the U.S. state of Texas.
Extracted Template: PART ("Richardson", "Dallas
Extracted Template: PART ("Richardson", "U.S. state of Texas | Texas")
Extracted Template: PART ("Texas", "U.S.")

For each sentence we implemented each template and derived for all the articles. For the sentences with partial arguments it will return empty fields and sentences with no template entities, template can't be extracted as it doesn't belong to any template.

Result and Error Analysis:

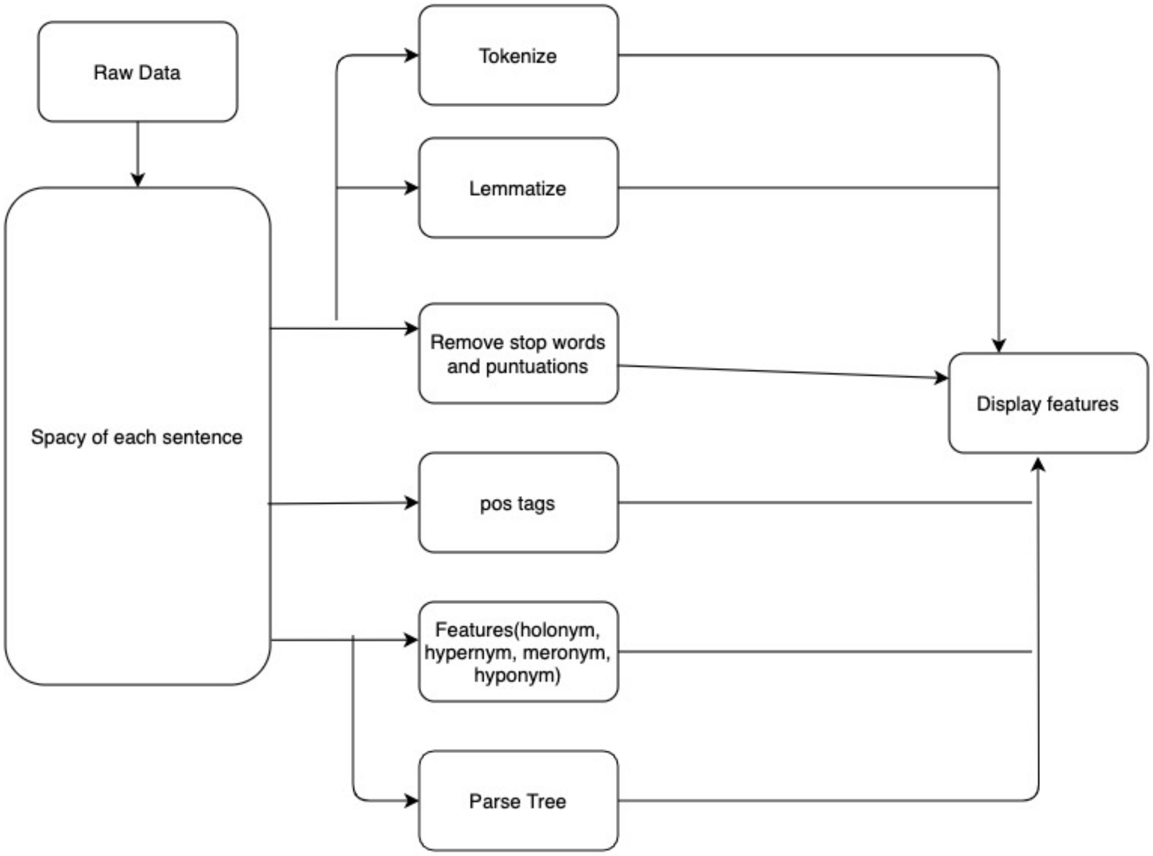
Challenges Faced:

Initial challenge faced was to install Stanford NLP Parser. There were several approaches on how to install. Finally solved the problem by having port access and extracted features for all the documents.

Next challenge for BUY and WORK is to identify the dependencies on the parse tree and roots for the sentences and entities, tags etc. Establishing relations for each template was not easy and writing rules for extraction.

Implementing knowledge graph for PART is the approach which should be implemented for all the templates as it could be a better approach to extract templates with spacy and dependency parser.

The most critical issue is the accuracy of the extraction of templates. We cannot just force update some parsing or entity tagging of spacy. This will cause overfitting of our model. We should improve our model with machine learning method. This might improve the result. Another Approach in our mind is implementing knowledge-based graph.



Pending Issues:

- We can use other dependency parsers to perfectly extract the head word of a given sentence
- Parse and Dependency trees can also be added as a feature to the index to get better results
- For the features – Hypernyms, Meronyms, Synonyms etc. only the top synset has been used. Others can also be explored

Potential Improvements:

- A Term frequency, frequency matrix can be used to retrieve he best results
- Co reference resolution
- Build knowledge based Graph