Information Extraction Project Report

CS 6320: Natural Language Processing

Abdelmounaim Hafid axh170730

Problem Description:

Given 30 text articles with 10 being related to Organizations, 10 related to Persons, and 10 related to Locations, implement an information extraction application using NLP features and various techniques. As part of this assignment there are three tasks that must be fulfilled:

- **Task 1: implement deep NLP Pipeline to extract certain NLP based features from text articles.**
- **Task 2: Implement machine-learning, statistical, or heuristic 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 extract relevant information.**

Proposed Solution:

After some research on the topic and the way other simple information extraction applications were implemented, I decided to go with a rule-based and heuristic-based approach to extracting the information from the articles. This seemed like the best way to extract information, because the English language has reoccurring structures and patterns that could be taken advantage of to extract this information from the articles. Furthermore, this approach can be used to easily control recall, which seems to be more important than precision in this application (at first look). A tightening or loosening of the rules or heuristics can easily change recall and precision, which gives me some control over the extraction. A machine-learning approach seems very complicated and could become computationally intensive to a point that running and testing models would be too time consuming and difficult. Additionally, I did not

feel too confident with using a statistical approach when dealing with only 30 articles worth of data. So, from all the approaches, I decided to use a combination of rules and heuristics to solve the problem.

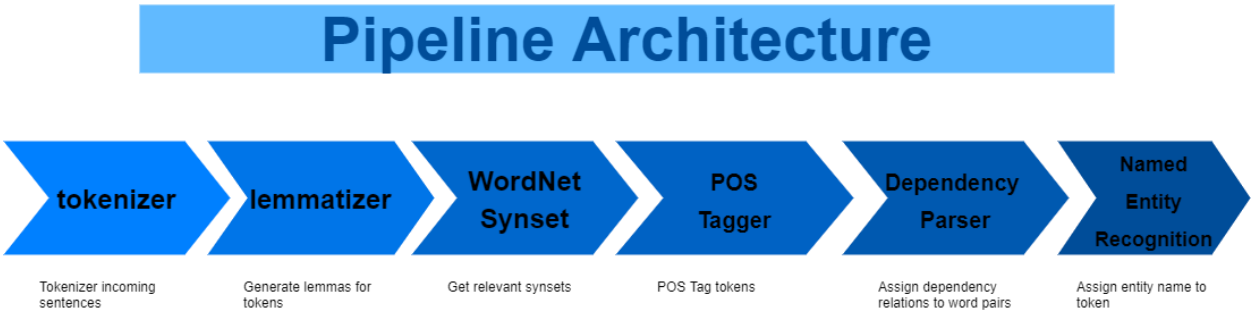
Programming Tools:

This project was implemented using Python 3 and developed in the PyCharm IDE. The tools and libraries used in this project include:

- SpaCy: an industrial level NLP library
- NLTK: natural language processing library

Architectural Diagram:

The diagram below shows the major components that each text article goes through in this data pipeline. One part that is not mentioned is the sentencizer, which parses the documents into individual sentences. Each major component plays a major role in the process of information extraction. The information from this pipeline is used in the implementation phase to create the rule-based and heuristic-based approach to solving the problem at hand.



Full Implementation Details:

The implementation details will be presented by template.

BUY and BORN:

In order to implement the extraction for these two templates, we follow a simple process described below:

1. Identify keyword verbs used to detect whether this is a candidate sentence that needs to be parsed (for example, sentence contains the word acquired for “BUY” relationship).
2. Given one of the verbs, we need a subject of the verb (the “doer”) to identify that is of the correct entity type for that template. We identify said word.
3. Once identified, we also want to identify the object of the verb, (what is being acquired, who or what is born). We traverse the dependency parse tree and find the object, which may also be located in a prepositional phrase, which is accounted for in both template extractions.
4. Finally, the last entity that is required for the templates is the date. We identify DATE entities that are in the dependency subtree for the verb and have a connection to the verb and, therefore, the subject and use it as the date entity for the template. Once again, the date may be located in a prepositional phrase, so we have to traverse the prepositional phrases to make the full traversal and ensure we don’t miss any date entities that may be present.
5. Once all three arguments have been extracted, we can output the results later used in the json creation.

Using the dependency tree along with entity recognition and knowledge of English language sentence structures, we can extract information with relatively good recall. In order to ensure

that we get the full subject phrase or date form, we use the extracted entities and search for them in noun phrases to see if they are part of a larger noun chunk (for example America is part of the phrase United States of America).

PART OF:

The part of template was the most difficult template and required different approaches to try and extract the information that is needed. The process is described below:

1. The first part of the implementation consists of matching phrases like “X is part of Y”, “X is in Y”, “X,Y” (X and Y are locations). These are very common forms and structures that are used to easily and quickly identify this relationship, so we use this to begin our extraction.
2. After trying to look for key phrases, we now use the dependency tree and start traversing subtrees of entities that have the correct entity type. In these subtrees if we have a preposition like “of” or “in” we further traverse to look for the desired entity types and the object of those prepositions to see if it is eligible for the template extraction.
3. Once the arguments have been extracted, we can output to json and display results

This relationship was very difficult to deal with and find the correct relationships, there were several type I and type II errors that had to be dealt with. It seems to be very difficult because of many overlapping structures in the language and no clear and easy way to identify the relationship, unless you’re using specific phrases.

Results and Error Analysis:

Upon analysis of the results found in the json files that were generated from the program, I found that there were a few reoccurring errors. Sometimes, when a date was extracted, only the year or month could have been extracted instead of the whole date, this would occur sometimes, while other times the date would be fully extracted. Another error that was observed were errors that were related to misclassification of entities and incorrect dependency relationships. These errors caused some relationships to be missed.

The most difficult relationship that had some errors was the part of relationship. The issue I had with this one is that it would pick up relationships in phrases like “University of Texas at Dallas” as PART OF(University, Texas). This caused errors in some phrases, but did not always cause errors.

Since I tried to maximize the recall to get more extractions, there seems to be a lot of false positives that were extracted from the articles, which is something that can be worked on in the future.

Summary of Issues:

Most of the problems related to this project are related to the actual extraction of the words, basically, it was a constant struggle to fine tune the extraction method and how to traverse the dependency parse tree. I would usually use a certain method or heuristic and have to fine tune it, because of extraction issues and inaccuracies. One issue that took some time for me to correct was the regexes that I used in the part of relationship. It would match things I didn’t want it to match, so I had to fix it and repeat the process until I fixed it to work as intended. Another issue I had was with extracting the full chunks/phrases of text once a certain word was chosen as part of the extraction. For example, if a word like “French” was tagged as a

person in the name Melinda Ann French, I would have issues extracting the full name with the middle name. Usually this would occur with large organization names. It was very difficult trying to solve this issue, but it was solved in some cases where it happened.

Pending Issues:

The biggest issues that remain are as follows:

- Accuracy: the approach I took attempted to maximize recall over accuracy, so the system suffers a bit in accuracy.
- Complex Sentences: some very complex sentences are hard to deal with. These sentences tend to have several phrases embedded and multiple branches in the dependency tree. It makes extracting some of the location/date/organization/people entities extremely difficult.
- Inaccurate PART-OF Relationships: some of the relationships here were inaccurate primarily because the work was rushed due to certain circumstances that were out of my control, however, with more work and time, they could have been smoothed out.

Possible Improvements:

In the event that I had more time to work on this project I would have made several improvements and changes to increase the quality of extraction and the overall accuracy. The first improvement that I would make would be to increase accuracy and not focus so much on the recall. Basically, create a program that can more accurately detect these relationships, while still maintaining a good level of recall. Additionally, I would like to make a more efficient system that works much faster. The current system took a very long time to process the articles

that were given as samples. These two improvements would make the application several times better, and are probably the biggest improvements that could be made.

Conclusion:

In conclusion, this project has been an interesting and engaging learning experience that has taught me several valuable skills that I can apply in the real world. NLP is still a young and constantly improving field that has several real-world applications. The project here has taught me several valuable things about what the process is like, and, in the future, I hope to increase my knowledge and improve my skills in the area.