Information Extraction using the Maluuba API

Team Members: David Greis, Khoa Tran

Option 2: Explore the Maluuba API and determine how it works on a collection

Corpus: We found the data while scouring the Internet for labeled entity recognition datasets that we could access free of charge. We found the MUC 3 and MUC 4 datasets available on the Technion's (Israel) website:

http://www.itl.nist.gov/iaui/894.02/related_projects/muc/muc_data/muc_data_index.html

The corpus contains the raw text of many news articles from the early 90s along with keys to each article, which contain entities against which we could test the Maluuba API. The keys suggest that the purpose of the data was to automatically detect violent incidents from foreign countries (most likely as a means for the US government to monitor terrorist or guerrilla activities abroad). This goal complicated matters for the purposes of our project, as we will describe below.

Repository Location: https://github.com/kqdtran/info_extraction_i256

Results

In the end, the Maluuba API allowed us to process nearly all 400 articles from the answer key set (n=396). As we progressed in our project, we discovered that our estimates for precision, and particularly for recall, would be significantly biased due to 1) an idiosyncrasy of the Maluuba API and 2) the quality of the labeling in the key data.

The Maluuba API returns the recognized location entities as a single string, not as a list of separate strings of the separate entities as one might expect. Because of this, two-word entities (e.g. 'San Francisco') are difficult to match against the answer key. We decided to be generous to the Maluuba API and rather than do a direct entity search, we would look for any intersection of tokens between the key and the long Maluuba string. We recognize this might give Maluuba too much credit in some circumstances. For example, the word 'city' is very general could be in the intersection even if Maluuba doesn't correctly recognize which city it is. Thus, this has the potential to upwardly bias both precision and recall.

The much bigger problem comes in the quality of labels in the answer key. As we progressed in our project, we discovered that the labeled data was focused on terrorist incidents, not on recognizing any and all locations in the articles. In other words, for each article, the key's location had the location of where a specific incident occurred, not a list of all locations described in an article.

This property of the key data severely biases our estimate of recall, because the Maluuba API will find entities that the key doesn't consider 'relevant'. Instead, they'll just artificially inflate the number of retrieved (but not relevant) entities, which in turn will bias recall downward. Our estimate of precision will not be as affected by this property. That is because even if the answer key picks out only one of the locations in the article, the Maluuba API should still be able to get it. When it can't, this is a legitimate place to dock Maluuba's precision score.

We see this reflected in our results. We estimated: Precision: 0.5588 & Recall: 0.0927. While we think our estimate of recall cannot be taken seriously, we think our estimate of precision is reasonable, despite the issues we have discussed above.

Please see the file outputDetails.txt for more details about our final result

Challenges

We decided to just go with 'location' instead of using all entities that Maluuba returns in a request. The reasons for this is as followed:

- The labeled data has really unique entity types related to counter terrorism that Maluuba would never have. Beside date and location, there were not much else that can be used to compare with the given entities in the corpus.
- Dates recognition seem to be impossible because Maluuba infers every date to happen in this year (2013) instead of the year that the event actually took place. Furthermore, the dates are returned as date ranges, so only either the start or end date would be correct.

Responsibility of each member

- David worked on making a master dictionary from the test data, where a key of the dict is the article ID, and the corresponding value is a list of sentences to be interpreted by the Malluba API one by one.

He also worked on determining the precision and recall of the final output to see how accurate the Maluuba API is.

- Khoa worked on making the test dictionary given the entities and their values. Each key in the dictionary is the article ID as in the master dictionary, while each value is another dictionary of entities, which follows the format: {entityName: entityValue}, e.g. {'city': Berkeley}.

He also worked on sending request to the Maluuba API and extracting the response based on the sentences curated by David. The result is then compared and contrasted with the test dictionary to determine the accuracy of the API and how well it works on the input data of 400 articles.

- Both team members contribute to the report, and to the general codebase to make sure that we are on the right track.