SNLP Mini-Project

NotReallyAGroup

Nino Schnitker – noni4@mail.upb.de

Approach:

The first script *reduceNames.py* extracts all entities from *train.tsv* and *test.tsv* and saves them in *reducedNames.csv*. To extract these entities, we used the Natural Language Processing package *spacy* and the model *en_core_web_sm*. This model was able to recognize most of the entities in both files.

This basic idea, why we extracted these entities was, that it was infeasible to process all articles from Wikipedia. Thus, we needed a rule how to decide which articles we use and which not. We had two ideas: The one with the entity recognition and the usage of all vital articles from Wikipedia (*https://en.wikipedia.org/wiki/Wikipedia:Vital_articles*). In the end, we decided on the first alternative because many people from *train.tsv* and *test.tsv* were not featured in the vital articles level 4 and the viral articles level 5 were infeasible to process.

The second script *buildFactChecker.py* takes all entities from *reducedNames.csv* and queries their Wikipedia articles. Then all nouns and proper nouns are extracted from these articles and saved in *knowledge_matrix.csv*. This way we save all nouns and proper nouns that an entity has a relationship with. This strategy seems to be a simple one but had to be chosen, because of the large number of data that had to be processed. First, we had implemented an algorithm that calculated for every noun and proper noun how strong their relationship is, depending on how many times these words appeared within a certain number of sentences within these articles, but this approach was infeasible. To make it feasible we tried to not use the entire articles from Wikipedia, but a summary. Unfortunately, these summaries missed a lot of facts from *train.tsv* and *test.tsv*. Thus, we decided to use the simpler approach.

The third script *factchecker.py* reads the facts from *train.tsv*, *test.tsv* and *impossible.tsv* (The file that contains ten facts the algorithm is currently not able to handle) and extracts the entities from every fact. For this extraction we again use *spacy* and *en_core_web_sm*. Then the algorithm searches for the entities of every fact in *knowledge_matrix.csv*. And checks how many of the nouns and proper nouns in the fact, every entity of that fact is related to. If the ration between relations that were found and relations that were not found extends a certain boundary, the fact gets classified as true (1.0). Otherwise if gets classified as false (0.0). If there are no entities found in the fact, which happens in a few cases, because

sometimes the entity recognition makes mistakes, the fact gets classified as false. The same happens if the fact only contains entities, were *buildFactChecker.py* was unable to find a Wikipedia article for.