DESIGN AND ENGINEERING OF INTELLIGENT INFORMATION SYSTEMS

HW2: Logical Data Model and UIMA Type System Design and Implementation

Fernando Garza - 146368
Februaruy 28, 2014

1 INTRODUCTION

The purpose of this project was to create a logical data model using UIMA framework to solve an unstructured information problem. Given a set of questions and answers, the job was to design the processing pipeline in order to select the best answer, based on a score given to each of the provided answers.

For this homework, we were required to code the actual processors, following each step. The typesystem was already given.

2 PIPELINE

The information processing pipeline had to contain the following steps:

1. Test Element Annotation
2. Token Annotation
3. NGram Annotation
4. Answer Scoring
5. Evaluation

So, the job is to read the text file, identify the elements that correspond to a question or to an answer, annotate them, be able to form n-grams of consecutive tokens and assign an answer score (based on the processing algorithms and classes).

2.1 Test Element Annotation

This descriptor has as output a Question or Answer annotation. The code is located in the package *components* and in the class *TestElementAnnotator.java*. This code splits the input into lines, and then analizes each pattern found on the line, in order to see if its a question or an answer, and annotates it.

2.2 Token Annotation

This descriptor has a as output tokens for the answers, this tokens will be later used on the pipeline. The code is located in the package *components* and in the class *TokensAnnotator.java*. The code selects all of the input text from the answers and extracts whitespace and punctuation, leaving just tokenized words.

2.3 NGram Annotation

This descriptor has a as input the tokens formed on step 2 of the pipeline and the answers. The output is an array of 1, 2 and 3-grams correspondingly for each answer. The code is located in the package *components* and in the class *NGramAnnotation.java*.

2.4 AnswerScoring

Takes as an input the question, answers, and NGrams, and outputs the answer score. Basically it compares each of the answer's grams to the question grams. If there is a match, then a point is added. Also, a weighing of the points was made, based on the theory that the coincidence of trigrams is more valuable than de coincidence of unigrams, so the following ponderation was applied: 20% for unigrams, 30% for bigrams and 50% for trigrams. The code is located in the package *components* and in the class *AnswerScoring.java*.

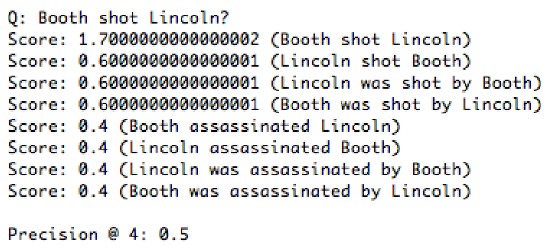
2.5 Evaluation

This part of the pipeline sorts the answers according to their score in descending order. Then calculates the number of correct answers (gold standard) in order to calculate TOP-N precision, thus it checks wether the top-n questions are correct or not and the precision is calculated as the quotient of the top-n correct questions and the total correct ones. This element of the pipeline prints the sorted questions and the precision.

3 RESULTS

3.1 BOOTH SHOT LINCOLN?

On the first set of given q/a, it is strange because although the best answer was the correct one (see figure 3.1), the next one in the rank was the opposite one.



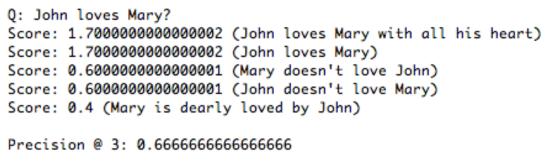
```
Q: Booth shot Lincoln?
Score: 1.7000000000000002 (Booth shot Lincoln)
Score: 0.6000000000000001 (Lincoln shot Booth)
Score: 0.6000000000000001 (Lincoln was shot by Booth)
Score: 0.6000000000000001 (Booth was shot by Lincoln)
Score: 0.4 (Booth assassinated Lincoln)
Score: 0.4 (Lincoln assassinated Booth)
Score: 0.4 (Lincoln was assassinated by Booth)
Score: 0.4 (Booth was assassinated by Lincoln)

Precision @ 4: 0.5
```

Figure 3.1: Score and precision for q/a set 1.

Also it is good to note that the answers had very similar scores, if it not where for the last digit on answers 3,4 and 5, they would be at the same level. And we find again that between answers 4 and 5 (which are opposite) there is a close relation in terms of score.

3.2 JOHN LOVES MARY?

This time, it is easy for the system to segregate correct from bad answers, except on the last one which clearly requires some POS analysis.



```
Q: John loves Mary?
Score: 1.7000000000000002 (John loves Mary with all his heart)
Score: 1.7000000000000002 (John loves Mary)
Score: 0.6000000000000001 (Mary doesn't love John)
Score: 0.6000000000000001 (John doesn't love Mary)
Score: 0.4 (Mary is dearly loved by John)

Precision @ 3: 0.6666666666666666
```

Figure 3.2: Score and precision for q/a set 1.

3.3 FINAL NOTES

Although this system makes a somewhat nice characterization of the questions and answers, for a good practical application we would need to analyze parts of speech from the question and answer, so that we can detect meaning in context.

Also, the code is not optimized, so the system will become slow if a lot of questions and -mainly- possible answers are to be processed.

4 References

- Kano, Y., Dorado, R., McCrohon, L., Ananiadou, S., & Jun'ichi Tsujii. (2010, May). U-Compare: An Integrated Language Resource Evaluation Platform Including a Comprehensive UIMA Resource Library. In LREC.
- Reina, J. R. FJ Martin Mateos. Procesamiento del Lenguaje Natural, Aplicaciones a la Inteligencia Artificial. Recovered from http://www.cs.us.es/.
- *UIMA Tutorial and developers guide*, retrieved from http://uima.apache.org/d/uimaj-2.5.0/tutorials_and_users_guides.pdf