Hw2 Documentation, LTI 11791

Abhimanu Kumar
Andre Id - ABHIMANK

January 11, 2014

1 Architectutre Design

I describe here the design considerations, UIMA issues and other NLP issues that I took into account while designing the logical architecture and implementing UIMA analysis engine design. We have a total of 5 annotator classes: 1) TokenAnnotator, 2)NGrammAnnotator, 3) QuestionAnnotator, 4)QuestionAnnotator, and finally 5)AnswerScoreAnnotator. Each class has a corresponding Analysis Engine Descriptor XML file that defines its input output set and types used.

1.1 Class TokenAnnotator

This class is the basic token annotator class that annotates every token encountered in the dataset. It stores the basic tokens encountered in the text.

1.2 Class NgramAnnotator

This is the class that encapsulates the group of tokens that are taken as bi-gram tri-gram etc. I create uni-grams, bi-grams as well as tri-grams, and all that creation logic in the same class. It creats a set of overlapping bi-grams and tri-grams.

1.3 Class QuestionAnnotator

This class encapsulates the methd to get the sentence body of the question. This class just extracts the begin and end of the question.

1.4 Class AnswerAnnotator

This class encapsulates the answers provided to a question. Apart from extracting the end and begin of the answer it also stores the correctness of the answer.

2 Special Points

1) I have coded logic to create uni-gram, bi-gram and tri-grams in a single class. This helps in code reusability as well as compactness of the code.

2) The scoring function used is the number of NGram overlap between an answer and its question. It is a simple yet good scoring method. It gives a precision of 0.5 nnd 0.6666 on the two documents given as input.

3) The precision is defined as (number of correct answers in top N)/N where N is the total number of correct answer.

3 Checking The Output

Run the hw2-abhimank-aae.xml, the aggregate analysis engine, and it gives the precision of the documents in the console.