11791 Assignment 2 Report

Mengda Yang

1. Type System

I made some minor modifications on the given type system.

a) Added 2 features in NGram.
   i. Integer N, indicates the 'N' of the N-Gram.
   ii. Annotation Origin, indicates to which Annotation (Question, Answer) this NGram is belong to.

b) Added feature Annotation Origin to type Token.

c) Added feature FSArray Tokens to Question, Answer, which indicates the Token's in this Annotation. Token's stored in the order of appearance.

2. Annotators

a) QAAnnotator

Annotates Question's and Answer's.

I used a regular expression that looks like "A (\\d) .***(\\r|\\n|\\r\\n)**". I think java sees the whole sofa input as a single line, so that I can't use symbols like ^ or $.

b) TokenAnnotator

Annotates Token's in the specified Annotation's.

To increase code reusability, I added a parameter into the UimaContext that can be read out. This parameter indicates among which Annotation types should we process to get the Token's. The Annotation types here must have feature FSArray Tokens.

I could modify the inheritance relationships of these classes, i.e. add an intermediate class between deiis.Annotation and deiis.Question/Answer, but I thought I am not supposed to make such major modifications so I gave up the idea (although it is pretty simple to realize and can simplify my error checking).

I used the Stanford NLP toolkit for tokenization.

The Annotations that is tokenized is updated. Their feature FSArray Tokens is given the tokens of its covered document.

c) NGramAnnotator

Annotates NGram's in the specified Annotation's.

The same as TokenAnnotator, I added a parameter for user to set the Annotations to be processed. These Annotation types still must have feature FSArray Tokens.

I added an Integer array to indicate N. N can have multiple values and these values don't have to be consecutive.

d) AnswerScoreAnnotator

Calculates the score of each annotation Answer.

Score is calculated in this way:

$$score_i = \frac{\sum_{nGram \in NGram_i} nGram * WordCount(nGram) * (nGram \in QuestionNGram)}{\sum_{nGram \in NGram_i} nGram * WordCount(nGram)}$$

However, this scoring function doesn't provide a better result than the one shown in class.