Project Individual 3: UIMA Analysis Engines

Leah Nicolich-Henkin: lnicolic

September 21, 2015

Overview

This report outlines a UIMA architecture to

- read an input file containing a question and a set of answers
- annotate the file using ngrams
- calculate scores to rank each sentence
- choose the best sentences
- calculate the precision of the system

Type System

The type system in my project was nearly identical to the system provided. The base annotation is the ComponentAnnotation, which provides a componentId and a score for use my all other annotations in the system. I extended this to also include an FSArray of Ngrams, so that both Questions and Answers can be associated with their own ngrams.

Question and Answer both extend the ComponentAnnotation, and each have a sentence feature, annotating their covered text. In addition, Answer has a score. The Token annotation is used for individual words, and can be combined into NGrams. Finally, Questions and Answers can be associated through an InputDocument.

Aside from adding the FSArray of Ngrams, the only change I made to the type system was to change the FSLists to FSArrays, which I found to be much easier to work with.

CPE

The CPE is the organizing descriptor for the annotation process. It directs the system first to the collection reader, then to the annotators, and finally to the consumer. This process is mirrored in `Main.java`.

Collection Reader

The collection reader has the simple job of going through all the files in the input directory and setting their JCas features, so that each can be annotated. It takes the input file directory as a parameter.

Annotators

The annotators work in sequence to annotate the document as different, but sometimes associated, regions of text.

QuestionAnnotator identifies the question sentence in the input document, and annotates it, as well as filling in its ID and sentence features.

The AnswerAnnotator identifies the answer sentences in the input document, and annotates them, as well as filling in its ID, sentence, and label features.

The TokenAnnotator identifies each token in the questions and answers in the input document.

The NGramAnnotator iterates over the tokens, and associated their boundaries with different Questions and Answers. It then uses the `N` parameter to decide how to combine the Tokens into NGrams, which can be added to the Questions and Answers as features.

The InputDocumentAnnotator combines Questions and Answers into a single annotation which can easily be processed by the Consumer.

Consumer

The consumer goes through an InputDocument, assigning scores to each Answer by comparing it with its associated Question. It then ranks and orders the Answers, before choosing the top N in order to calculate precision. It also handles writing to the output fit.

Difficulties

The biggest challenge in this project was simply wrapping my head around all the interactions of the system. UIMA has lots of different components that all have to work together to process data and create output. If a name, configuration, or dependency changes in one place, an Eclipse refactor won't deal with it; instead you have to manually change the related files, which can be challenging and frustrating. Writing the substance of the annotators and calculations for ngrams, precision, etc wasn't particularly challenging, but with the difficulties of figuring out different components and connections, and the dense documentation, the whole project took me over 20 hours.