11791 Homework 1 report - Type System Design

Zhengzhong Liu
Language Technology Institute
Carnegie Mellon University
Pittsburgh, PA 15213

1 Type system design

1.1 Identify types

In this homework, we are required to design a type system for one sample Question Answering task, where the input and proposing processing pipeline are outlined. Hence a type system should be designed to be support the basic requirements in the pipeline and can be easily extended to incorporate possible future development. One important source that affect the type system design is the processing pipeline design. This system design starts with identifying the nouns in the processing pipeline description. From the guideline, there will at least the following steps in this task:

1. Test Element Annotation
   This annotation actually contains two annotations : the question annotation and the answer annotation. *Question* and *Answer* are two clear noun that worth annotating. Thus the type system design should contain these two types. The noun *Test Element Annotation*, can also be considered as a type, which can serve as the super type of Question and Answer. However, the current design exclude this type because there are no significantly important features that should be shared by *Question* and *Answer*.
2. Token Annotation
   Tokens are important language units in understanding the text, which should be considered as annotations.
3. NGram Annotation
   The pipeline relies on NGram as an information source, which should also be one type.
4. Answer Scoring
   The noun score could be considered as an annotation, however, it could also be considered as an attribute that is associated with each answer. The design choice will be given in section 1.2.
5. Evaluation
   Evaluation is a noun, but it could simply be modeled as an action, performed by the annotator (the evaluator). However, to assist evaluation and development, this system design add one type to store the evaluation results (at least the numeric ones) for future references.

To summarize, the type system should contain at least the following basic types: *Question*, *Answer*, *Token*, *NGram*, *Score*, *EvaluationResult*. One further look to the input text file raise another type, *Sentence*, which is another important language unit that could often be useful in language processing (considering that one question or one answer is not always composed by a single sentence).

The input file also contains some portion that are not annotated by the current defined type, such as the letter "Q" and "A", which are markers for test element type, and the number "0", "1" which are used to indicate the correctness of the answer. The current system design consider not to annotate these, but instead will assume that these markers will only be used by annotators such as "TestElementAnnotator" and "GoldStandardAnnotator".

1.2 Type system design and implementation

Given the types to be annotated, the type system design is quite straightforward. One additional step adopted by this design is a base type system, which defines to types: *ComponentAnnotation* and *ComponentTOP*, which extends the basic UIMA type *Annotation* and *TOP* respectively. These two types are all added with the features "componentId" and "confidence". "componentId" is used to indicate where the source of this annotation, "confidence" is used to indicate the confidence score that the annotator assign. These two features are quite universal and can be used by most of the types defined, thus will be served as the main base types. The type system design also notes the differences between *Annotation* and *TOP*, where Annotation contains the features "begin", "end", which *TOP* does not have these. One basic design principle that is likely to be followed in the future is that *Annotation* will be used to annotation text elements, while *TOP* will be used to indicate relations between types.

Most type design are straightforward, and they are named using the reverse URL convention, the package used in this implementation is "edu.cmu.cs.lti.zhengzhl". There are several feature design for each type.

1. Answer
   An answer should have a score associated with it, in order to know what annotator produce this score (e.g. gold standard annotator or system annotator), we need to have a score type, which is able to store both the numeric values of the score and the source that produce the score.
2. Score
   Given the reason stated above, one more score annotation is given, this type should extend *ComponentTOP* because it is not naturally associated with some text span.
3. Question
   A question is not merely a text span, but it should also be associated with its candidate answers, thus there is a feature named "candidateAnswers", which stores a list of answers.
4. Token
   Token are the basic element type in the system, in language processing, we are often interested in several other forms of it, including part of speech, lemma and stem. Thus these features are annotated as features of this type. In addition, "tf" feature will be used to store the term frequency of this token in the file. In the future, there are possibilities that we store document frequency and collection term frequency, but those are more like collection-wise values. The type system doesn't try to model these right now.
5. NGram
   The type NGram should clearly be annotated with the value of "N", this is put as a feature in the system. The design here also have two alternatives, one is to annotate each types of "gram" separately, such as including "Unigram", "Bigram", "Trigram" as types, this method has the benefit also easy to retrieve when we need to have only one particular size of NGram, but it is clearly not easily extensible, thus the current design will use an universal NGram type with a feature "n" indicating the number.