11-791: Homework 1

Instructed by *Dr. Eric Nyberg*

Due on Sep 11, 2013

Troy, Zhenhao Hua

1 Introduction

There are 10 new types defined in my type system. Two of them are based types and directly extend UIMA cas types. One of them extends Annotation type, is the base type for all other types for different annotations. The other base type extends TOP type, is the base type for all other non-annotation types. Both of the base types have two extra features: source(String) and confidence(Float). So that every type in this system will have these two features.

Upon the base annotation type, Token type direct extends this and Answer, Question, NGram types all have an array of Token as a feature. Question and an array of Answers will form a Block type, which is a computation unit for a Score Scheme. Score Scheme as an element. A Score Scheme should work with different scoring compoments to assign scores for each answer. And a Score Scheme will be sent to a Evaluation type to evaluation metric like Precision at N.

This forms a complete type system from the base annotation Token type to the final Evaluation type.

Please refer to Figure 1 for the entire design diagram. The solid line indicates class extention and the dash line indicates class dependence (using another class in the features).

2 Detailed Documents for Each Type

2.1 BaseAnnotationType

This is the base class for all annotation types in the system. It has two additional features, source and confidence. Source is filled by the annotator to know where the annotation is from. Confidence is a measurement how confident the annotator is when generating the annotation. All annotation types in the system should be a subclass of this one.

Full Name: model.troy.hw.BaseAnnotationType
Super Type: uima.tcas.Annotation
Features:

- source
  - **Range Type:** uima.cas.String
  - **Description:** A string feature for the annotator to provide information so that we can directly get where the annotation is generated.
- confidence
  - **Range Type:** uima.cas.Double
  - **Description:** Annotators should provide a confidence score in this feature for every annotation type.

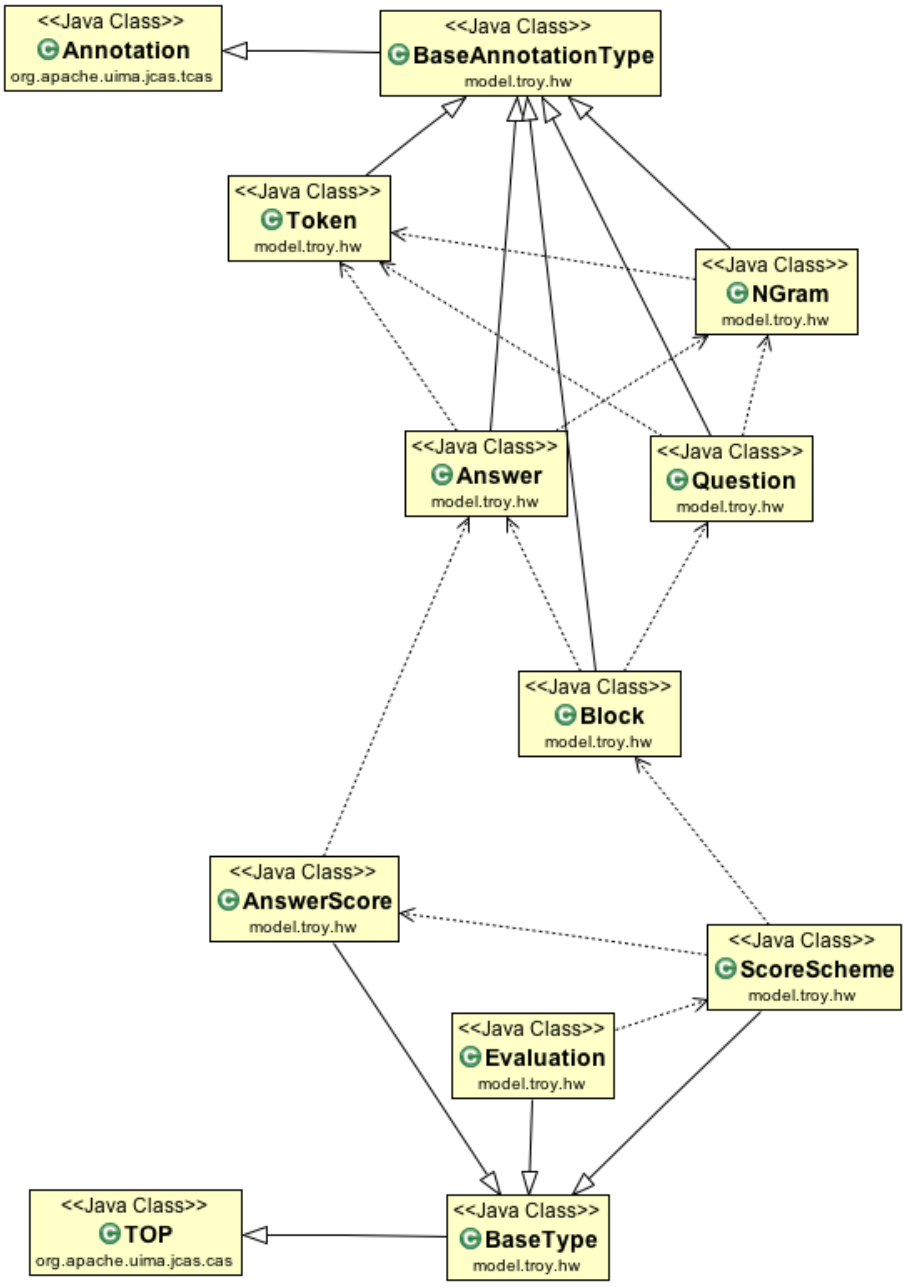

Figure 1: Type Extention and Dependence Diagram

2.2 Token

Token is the smallest annotation unit of a document splitted by white space and punctuations.

Full Name: model.troy.hw.Token
Super Type: model.troy.hw.BaseAnnotationType
Features:
Token type does not have additional features.

2.3 NGram

NGram is a short sequence of tokens, so it contains an array of tokens.

Full Name: model.troy.hw.NGram
Super Type: model.troy.hw.BaseAnnotationType
Features:

- tokenArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.Token
  - **Description:** The tokens contained in this NGram are stored in this array. We can get the exact value of N from the length of the array, so we do not use another feature to store this information.

2.4 Question

Question is the type for annotating questions in the document. It also stores the token information and NGram information.

Full Name: model.troy.hw.Question
Super Type: model.troy.hw.BaseAnnotationType
Features:

- tokenArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.Token
  - **Description:** This stores all the tokens in this question.
- nGramArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.NGram
  - **Description:** This stores all the NGrams in this question. Because we do not need to output NGram with different length separatedly, there are not separated lists for different length of NGram. And a scoring component should consider all the NGrams and have different weights inside the component, so we do not provide separated lists here.

2.5 Answer

Answer is the type for annotating answers in the document. Each instance represents one answer. It also stores the token and NGram information as well as if the answer is correct or not. Although we cannot directly get the corresponding question from this type, the scoring component should look into the Block type which contains answers and corresponding question.

Full Name: model.troy.hw.Answer
Super Type: model.troy.hw.BaseAnnotationType
Features:

- isCorrect
  - **Range Type:** uima.cas.Boolean
  - **Description:** Whether the answer is correct for the corresponding question.
- tokenArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.Token
  - **Description:** This stores all the tokens in this answer.
- nGramArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.NGram
  - **Description:** This stores all the NGrams in this answer.

2.6 Block

Block contains one question and an array of answers corresponding to this question. It is a unit of a whole instance of raw data of the task.

Full Name: model.troy.hw.Block
Super Type: model.troy.hw.BaseAnnotationType
Features:

- question
  - **Range Type:** model.troy.hw.Question
  - **Description:** The question in this text block.
- answerArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.Answer
  - **Description:** An array of all answers in this text block.

2.7 BaseType

The base type of all non-annotation types. Fields are similar to BaseAnnotationType.

Full Name: model.troy.hw.BaseType
Super Type: uima.cas.TOP
Features:

- source
  - **Range Type:** uima.cas.String
  - **Description:** A string feature for the annotator to provide information so that we can directly get where the annotation is generated.
- confidence
  - **Range Type:** uima.cas.Double
  - **Description:** Annotators should provide a confidence score in this feature for every type.

2.8 AnswerScore

It is an answer-score pair.

Full Name: model.troy.hw.AnswerScore
Super Type: uima.cas.TOP
Features:

- answer
  - **Range Type:** model.troy.hw.Answer
  - **Description:** The answer for the score.
- score
  - **Range Type:** uima.cas.Double
  - **Description:** The score to the answer.

2.9 ScoreScheme

ScoreScheme should be the output of a scoring component. It contains a set of whole raw data and an array of scores for all answers.

Full Name: model.troy.hw.ScoreScheme
Super Type: model.troy.hw.BaseType
Features:

- rawData
  - **Range Type:** model.troy.hw.Block
  - **Description:** The whole set of data.
- scoreArray
  - **Range Type:** uima.cas.FSArray
  - **Element Type:** model.troy.hw.AnswerScore
  - **Description:** The array of scores for all the answers in the data.

2.10 Evaluation

An evaluation component should take a ScoreScheme as input and output an Evaluation instance. This class contains the evaluation metric score and other parameters regarding different measurements. For other evaluation metrics, Evaluation class can be extended to add more metrics.

Full Name: model.troy.hw.Evaluation
Super Type: model.troy.hw.BaseType
Features:

- N
  - **Range Type:** uima.cas.Integer
  - **Description:** The N in Precision at N.
- precision
  - **Range Type:** uima.cas.Double
  - **Description:** The precision score of precision at N.
- rawScore
  - **Range Type:** model.troy.hw.ScoreScheme
  - **Description:** The original score from a scoring component.

3 Possible AE and pipeline design to show this type system is applicable

3.1 Pipeline design

1. **Block generation, Answers and Question Extraction.** The first step is to build a **Block** annotation type by a simple parsing on the quesition and answers. In this step, we also need to generate **Question** and an array of **Answer**s. However, we do not need to fill in the token and NGram information at this step.
2. **Tokenization.** The second step is to fill in the token array in every question and answer.
3. **NGram Generation.** The third step is to use the Token generated from previous step to fill in the NGram array in every question and answer.
4. **Scoring.** A scoing component should take the block with full Token information and NGram information as input and generate a score for each answer by creating **AnswerScore** array.
5. **Ranking and Evaluation.** The final step is to evluate the score from the previous step using Precision at N. As raw data are still there in **ScoreScheme**, we can get the ground true and calculate the precision easily.

From this five-step pipeline, we can generate basic Question and Answer text elemetn annotaion, Token Annotation, NGram Annotation, Answer Scoring and final Evaluation.

Acknowledgement: Thank Zi and Di for the discussion about requirement of this assignment.