11-693 Software Method for Biotechnology Homework 1
Name: Tao Guo
id: taog

Report of Logical Data Model Design for a Sample Information Processing Task

1. Requirement analysis
Due to the assignment, the information processing pipeline consists of 5 steps.

1.1 Test Element Annotation
In this step, the system reads in the input file, annotates the question and answer spans and record whether the answer is correct or not. In the model design, the program generates the initial annotation for the whole test called *TestElement*. After that, it create instances of *Question* and *Answer*, and record the correct-or-not information in the instances of *Answer*.

1.2 Token Annotation
In this step, the system annotates each token span. In the model design, the program generates instances of the type *Token*, and links them with their *Question* instances.

1.3 NGram Annotation
In this step, the system annotates 1-, 2- and 3-grams of consecutive tokens. In the model design, the program generates instances of the type *NGram*, recording the number of tokens the type owns and keeping track of them.

1.4 Answer Scoring
In this step, the system assigns answer scores to the questions. The program lets the instances of *Question* record them.

1.5 Evaluation
In this step, the system sorts the answer according to scores and calculates the precision. The program lets the instance of the *TestElement* record that precision.

2. Logical Data Model Design

The logical data model design is illustrated as the following UML type diagram (figure).

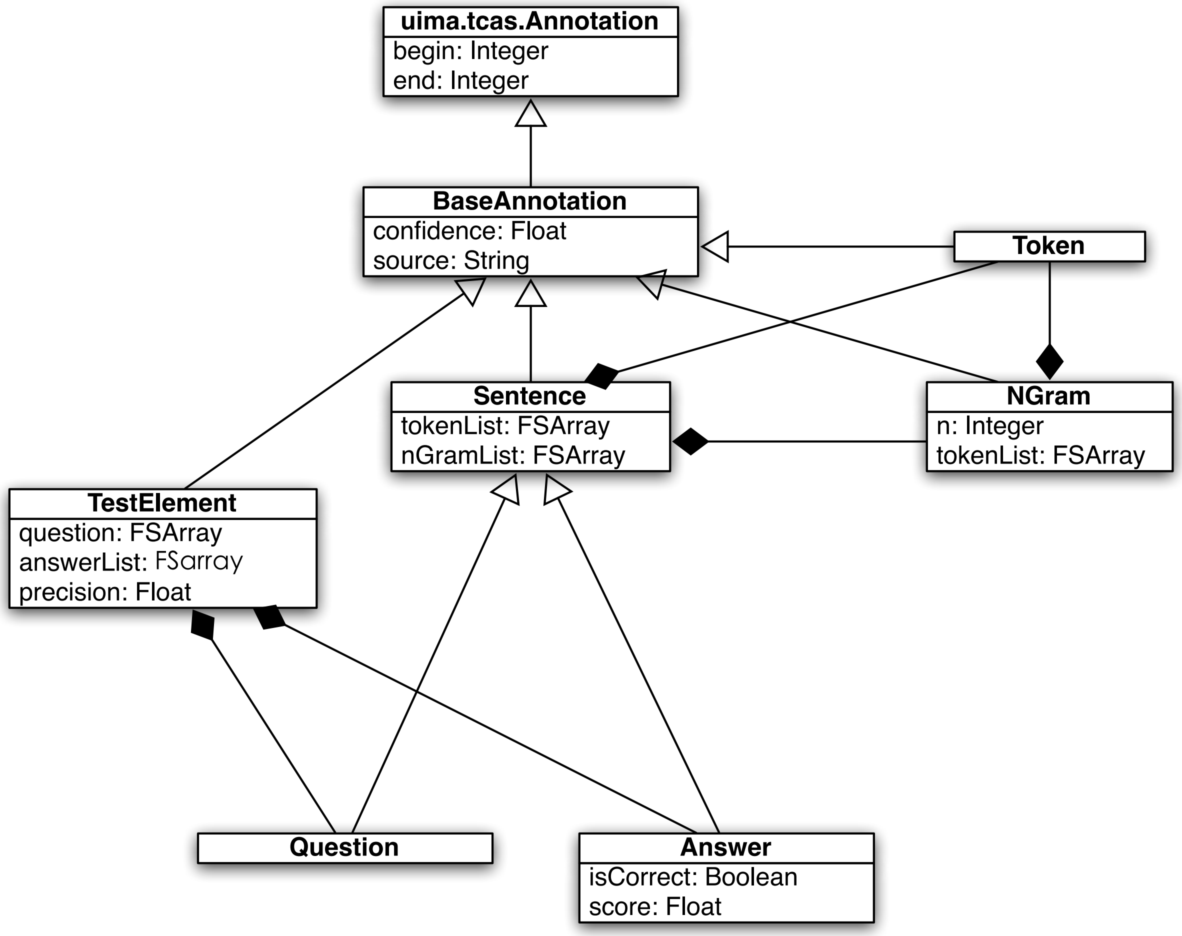

Figure: UML Type Diagram of the Design

2.1 Type *Annotation*
The type *Annotation* is provided by the UIMA framework. It provides the basic characteristics and features such as the begin and end features. All the other types in this design inherit from it.

2.2 Type *BaseAnnotation*
The type *BaseAnnotation* directly inherits from the type *Annotation*. It contains the feature *confidence* to help keep track of how confidence the annotation was. And another feature *source* to record where it was originally made by. These two features are inherited by all the subtypes.

2.3 Type *Sentence*
The type *Sentence* directly inherits from the type *BaseAnnotation*. It is the supertype for *Question* and *Answer* to provide *tokenList* and *nGramList* features to record the list of tokens and n-grams.

2.4 Type *Question*
The type *Question* directly inherits from the type *Sentence*. It denotes to the question of the test element.

2.5 Type *Answer*

The type *Answer* directly inherits from the type *Sentence*. It denotes to the answer of the test element. Specifically, it has the *isCorrect* and *score* feature to record if it is originally correct and the score assigned to it.

2.6 Type *TestElement*

The *TestElement* directly inherits from the type *BaseAnnotation*. An instance was originally created to represent the all test. It has a two FSArray features *question* and *answerList* pointing to the questions and answers involved in the test. It also has a feature precision to record the precision of the test.

2.7 Type *Token*

The type *Token* directly inherits from the type *BaseAnnotation*. It denotes to a token in the sentences and the n-grams.

2.8 Type *NGram*

The type *NGram* directly inherits from the type *BaseAnnotation*. It denotes an n-gram, thus a contiguous sequence of n tokens from the text. It has a feature *n* to record the length of the n-gram. It also has an FSArray feature *tokenList* to refer to the tokens it contains.

All these types work together and formed the logical data model of the design.