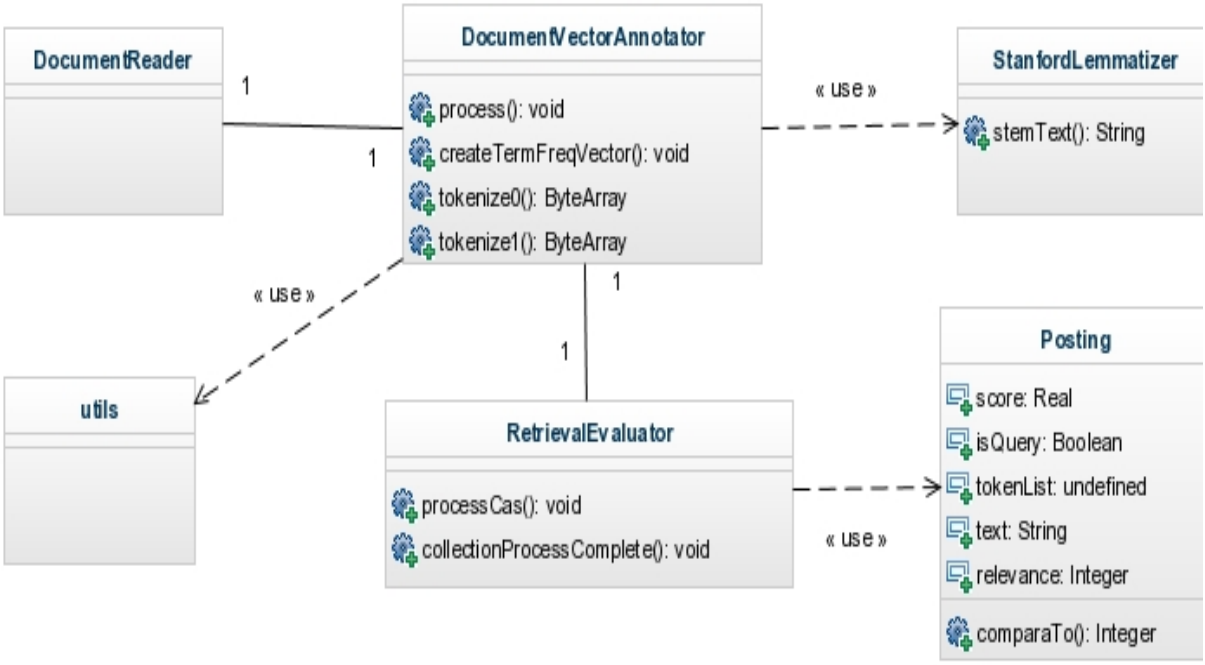
Homework 3 Report

Rui Wang (ruiw1)

# 1. System Design

## 1.1 Class Model




## 1.2 Type System

### 1.2.1 Document

Document is used to save query and document information, like query id, relevance. The most importance of Document is to save term frequency vector.

### 1.2.2 Token

Token is basic unit of term frequency vector, containing feature like term and term frequency. It is the basic part of token list in vector.

### 1.2.3 Query (not used)

This type was designed to save query, in order to distinguish with document themselves. However, during implementation, I found it is not necessary to use two kinds of types because it increases complexity of system. However, it is still a good practice to save query independently.

## 1.3 Annotator

There is only one annotator this time, named DocumentVectorAnnotator. This annotator is used to build term frequency vector.

# 2. Error Analysis

## 2.1 Error Type

2.1.1 Vocabulary Mismatch

This error mainly caused by lack or inappropriate stemming. For instance, in most context, feelings and feeling have the same meaning. However, without stemming, VSM will not consider this term's weight. Stemming could find the connection between Noun and Adjective, Singular and Plural, etc. in different vectors.

a) Different term forms

```
qid=3   rel=99  In which year did a purchase of Alaska happen?
qid=3   rel=1   Alaska was purchased from Russia in year 1867.
qid=3   rel=0   1867 - U.S. President Andrew Jackson proclaims treaty for purchase of Alaska from Russia.
qid=3   rel=0   William Seward negotiated a purchase of Alaska for $7.2 million.
```

In this query, "purchase" is a noun, but in relevant document, we get "purchased". So Stemming can eliminate this mismatch error.

```
qid=4   rel=99  What year did Wilt Chamberlain score 100 points?
qid=4   rel=1   On March 2, 1962, Wilt Chamberlain scored a record 100 points in a   game against the New York Knicks.
qid=4   rel=0   A 100 point game was a highlight in a career of Wilt Chamberlain
qid=4   rel=0   Wilt Chamberlain most famous record is the 100 points he scored in the   Philadelphia Warriors' 169-147
```

Query: What year did Wilt Chamberlain score 100 points?

Document: On March 2, 1962, Wilt Chamberlain scored a record 100 points in a game against the New York Knicks.

Without stemming, "score" and "scored" will mismatch.

b) Punctuation

Another possible reason may result in mismatch problem is about Punctuation. Simple tokenization on white spaces cannot handle other symbols, e.g. comma, question mark. Stemming will fail on cases like "China's ". Stemming considers this is a Plural and return "China' ". Similar problem happens like "happen? ", "Pompeii; ". So more rules should be applied to tokenize.

2.1.2 Semantic misundstanding

Due to variety of natural language, one concept can be expressed by different ways and different phrases and words. It result in two main errors.

a) Term replacement

```
qid=3   rel=99  In which year did a purchase of Alaska happen?
qid=3   rel=1   Alaska was purchased from Russia in year 1867.
qid=3   rel=0   1867 - U.S. President Andrew Jackson proclaims treaty for purchase of Alaska from Russia.
qid=3   rel=0   William Seward negotiated a purchase of Alaska for $7.2 million.
```

In this query, question is about a year purchase happened. However, relevant document answer gives exactly the year itself, and term vector cannot catch this kind of match.

b) Term expression

```
3 qid=6   rel=99  Who was the first person to run the mile in less than four minutes
4 qid=6   rel=1   Roger Bannister was the first to break the four-minute mile barrier.
5 qid=6   rel=0   The claim that a 4-minute mile was once thought to be impossible byinformed obs
6 qid=6   rel=0   It is hard for humans to run the mile faster than in four minutes
```

For example, in query “Who was the first person to run the mile in less than four minutes” uses “four minutes” to describe speed of this person. However, in relevant document, “four-minute” is used as an effective expression, but result in a query mismatch.

2.1.3 Term overweighting

Term overweighting is a problem that some unimportant terms are assigned too much weight. The most common reason is stopwords.

```
qid=20  rel=99  What is the Keystone State?
qid=20  rel=1   They call it the Keystone State, and in this unpredictable election year, Pennsylvania is living up to its name.
qid=20  rel=0   Keystone Resort is the largest ski resort in Summit County located in Keystone Colorado.
qid=20  rel=0   The Keytstone Resort is owned and operated by Vail Resort.
```

For this set of document, the third one ranked highest because “is” and “the” could give higher similairty in vector space model. The relevant document does not have “the”, so for term vector similarity between query and real relevant document is weaker.

2.2 Solutions

2.2.1 Punctuation

I write a new rule based tokenization algorithm, in order to remove unnecessary symbols. If a token is end with “ ? “, “ ; “ , “ ’ “, regular expression I define will remove these symbols.

2.2.2 Stemming

To achieve stemming, I use Stanford Lemmatizer. It effectively transforms different forms of a word into its origin. The stemming package of Lucene is another choice.

2.2.3 Synonym

An approach to find synonym could be used to reduce semantic error. However, I didn’t implement synonym algorithm because it is too complicated, and it is context dependent, which increases problem complexity.

2.2.4 Stopwords

A empirical rule is that the most frequent terms in vocabulary are terms like

and, is, this, etc. If considering stopwords' importance in our retrieval model, maybe it would cost computational waste, because if every document contains more or less stopwords, it does not matter if I remove these words, and result will not be changed.

### 2.3 Statistics

| Error Types | Cases | Frequency |
|---|---|---|
| Term forms | Q4, Q14 | 2 |
| Punctuation | Q1, Q5, Q3, Q11, Q18 | 5 |
| Term replacement | Q20, Q7 | 2 |
| Term expression | Q1, Q6 | 2 |
| Term overweight | Q3, Q12 | 2 |

### 2.4 Performance Improvement

| Solution | MRR Before utilization | MRR After utilization |
|---|---|---|
| Punctuation | 0.4375 | 0.4708 |
| Stemming | 0.4375 | 0.5500 |
| Punctuation&&Stemming | 0.4375 | 0.5833 |
| Stopwords | 0.4375 | 0.4750 |
| Punctuation&&Stopwords | 0.4375 | 0.4750 |
| Punctuation &&Stopwords &&Stemming | 0.4375 | 0.5417 |

Combine different methods have different effect, but they do have positive effect on accuracy. Stemming greatly contributes to performance improvement. Punctuation processing and removing stopwords helps a little.

## 3. Similarity Measures

There are different kinds of similarity computation, so I implement two more measures, Dice coefficient and Jaccard coefficient.

### 3.1 Jaccard Coefficient

This coefficient is defined as the size of the intersection divided by the size of the union of sets. It makes sense that this approach can measure similairty of two vectors. For example, if two vector are the same, then their intersection is equal to their union, the value of Jaccard coefficient is 1. If two vectors have no intersection, then coefficient is 0.

This is implemented in **RetrievalEvaluator.java**

3.2 Dice Coefficient

Being similar to Jaccard Coefficient, this similarity measure is also used to measure data presence and absence. The range of this measure usually is from 0 to 1, where 0 means total irrelevance and 1 means two vectors are the same.

This is implemented in **RetrievalEvaluator.java**

3.3 Measurement

| | Cosine Similairty | Jaccard Coefficient | Dice Coefficient |
|---|---|---|---|
| Naive | 0.4375 | 0.4083 | 0.4625 |
| Processing Punctuation | 0.4708 | 0.4250 | 0.4250 |
| Stemming | 0.5500 | 0.4542 | 0.4542 |
| Punctuation&&Stemming | 0.5833 | 0.4667 | 0.4667 |
| Stopwords | 0.4750 | 0.4083 | 0.4083 |
| Punctuation&&Stopwords | 0.4750 | 0.4083 | 0.4083 |
| Punctuation &&Stopwords &&Stemming | 0.5417 | 0.4792 | 0.4792 |

3.4 Experiment

After implementing two more measures, I started to do performance test. At beginning, experiment plan just includes naïve test case. More specifically, I tested these three measures under very limited tokenization processing.

However, I realized that different solutions for handling error types might have different effect on measures. I thought they should be same. Experiment shows that different processing did have different effect. For example, removing stopwords decrease MRR of Cosine Similarity while it increase Dice Coefficient. If I process punctuation, it improves Cosine similarity and Jaccard Coefficient, and if I combine all solutions I implement, it improves every measures' MMR.