Yan He (yanhe)

October 22, 2014

Software Methods Homework 3 Report

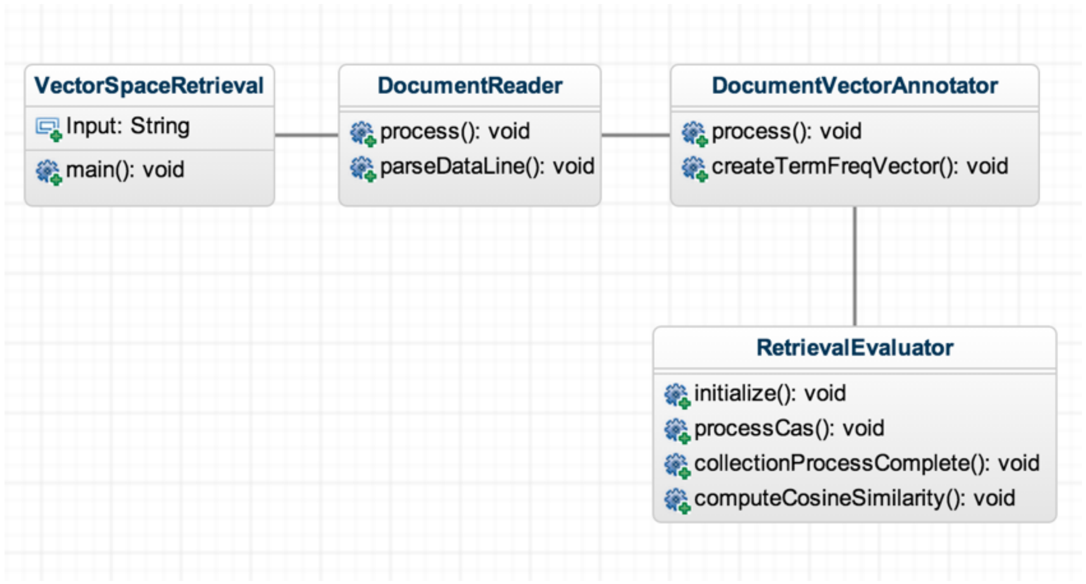
Task 1 Building Vector space Retrieval Model

1. **System Architecture**

   For this assignment we need to build a vector space retrieval model using UIMA. In the vector space retrieval model, both the query and documents are sparse term vectors, then we need build this model to compute the cosine similarity values between query vector and document vectors.

2. **Data flow**

   The basic data flow of this model is as follows. Basically, the data is read by DocumentReader and pass through DocumentVectorAnnotator to split the sentence and finally calculate and evaluate by RetrievalEvaluator



3. **Vector Space Retrieval**

   This is the main function of this retrieval system. The analysis engine is Sentence Analysis as the analysis engine. It contains three analysis components: The DocumentReader, the DocumentVectorAnnotator and the RetrievalEvaluator.

4. **Document Reader**

   Document Reader is provided in the archetype. Its main function is to read the lines in the document and get the document information such as query id, relevance assessment, and document text. Then these information is saved in the CAS.

5. **Document Vector Annotator**

   The main function of Document Vector Annotator is to creates sparse term vectors. It could construct a vector of tokens and update the tokenList in the CAS. For task 1, I only use the tokenize0 function to split the sentence by white-space. Then I create a map entry and set the type value for each token.

6. **Retrieval Evaluator**

   The main function of Retrieval Evaluator is to calculate the value of cosine similarity and MRR. Then it saves the values into the document which name is “report.txt”.

Task 2 Error Analysis

1. **Variation**

   The first error type is variation, such as the tenses of the word. For example, the word “purchase” could not be given correctly because in the right answer it is the word “purchased”. Moreover, same thing could happen to “old” and “Old”.

2. **Answer mismatch**

   The second error is answer mismatch, which means that the answer that given by the system is similar to the actual answer, but in fact it is not right. For example, the right answer should be “Keystone State” while in the output answer it is just “Keystone”.

3. **Word with punctuation**

   The third error type is related to punctuation, which means that the word with punctuation could not be recognized correctly. For example, the word “Mendocino Tree" in the question sentence can not be regarded same as “Mendocino Tree,”.

4. **None meaningful word**

   The system could not recognize the word that doesn’t have any real meaning, such as “a”, “an”, “the” and so on.

5. **Ideas about improvement**

   For dealing with synonyms and different forms of a word, we could build a dictionary (or reference) for nouns and verbs. When we find a noun or a verb, we will search the dictionary and try its other forms or synonyms. Different forms of a word of synonyms can be regarded as the same when we check the comparison.

   For dealing with none meaningful word, we could build a dictionary of unmeaningful word. Before we put a word in token list, we could tranverse the dictionary to check whether it is an unmeaningful word or not. Then we could discard this word if it appears in the dictionary.

Extra point

For the extra points, I’ve implemented three independent functions. The function *jaccardSimilarity* uses the measurement of Jaccard similarity coefficient. Its result output is named as “reportForJaccardSimilarity.txt” under the same directory as the original report. The function *diceSimilarity* uses the measurement of Dice similarity coefficient. Its result output is named as “reportForDiceSimilarity.txt”. And the function *tfidfSimilarity* uses the measurement of TF-IDF. Its result output is named as “reportForTfidfSimilarity.txt” These different kinds of measurement could improve the final value of MRR within a small range.