Engineering and Error Analysis with UIMA

Name: Qichen Pan   AndrewID: pqichen

Introduction

In this homework, we are supposed to build a UIMA pipeline to apply Vector Space Retrieval algorithm on Q&A process. Give a question and some candidate answers, our pipeline will rank the answers based on the similarities between them and the question. I implemented 3 different methods to compute the similarities, Cosine Similarity, Dice Coefficient and Jaccard Coefficient. I also did some error analysis on the output of the baseline system, and performed some optimization to improve the performance of the whole system.

Task 1 Build Vector Space Retrieval Model using UIMA

1.1 Task Introduction

In this task, I added codes to the original framework given by the TAs to realize all the basic functions of the baseline system. The baseline system can rank the answers based on the Cosine Similarities between them and a certain question. And I used MRR to measure the performance of the system.

I also implemented other methods to compute the similarities, such as Dice Coefficient and Jaccard Coefficient.

1.2 File Description

Filename: DocumentReader.java

In this file, each document will be read to generate appropriate doc cas into UIMA pipeline. In this specific case, documents consist of only one sentence. Qid represents query Id, Rel represents relevance. Our cas also stores doc's text.

Filename: DocumentVectorAnnotator.java

In this file, Tokenlist of each doc will be generated. Also, each Token in the TokenList will have its exact appearance as its frequency.

Filename: RetrievalEvaluator.java

In this file, our UIMA pipeline will calculate the similarities between questions and all the answers. All the answers will be ranked based on their scores. UIMA pipeline will also compute MRR to measure the performance of the whole process. I applied 3 methods for computing similarities: CosineSimilarity, DiceCoefficiency and JaccardCoefficiency. I used stopwords.txt to optimize the pipeline to achieve higher MRR.

1.3 System Output

The output of the pipeline is shown below:

```
-------------------------------------------------------------------
Using Cosine Similarity:
Score: 0.45226701686664544 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.30618621784789724 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.5070925528371099 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.2581988897471611 rank=3 rel=1 qid=4 If you see a friend without a smile, give him one of yours
Score: 0.15811388300841897 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Cosine Similarity::0.8666666666666668

-------------------------------------------------------------------
Using Dice Coefficiency:
Score: 0.4 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.3 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.5 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.25 rank=3 rel=1 qid=4 If you see a friend without a smile, give him one of yours
Score: 0.14285714285714285 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Dice Coefficiency::0.8666666666666668

-------------------------------------------------------------------
Using Jaccard Coefficiency:
Score: 0.25 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.17647058823529413 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.3333333333333333 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.14285714285714285 rank=3 rel=1 qid=4 If you see a friend without a smile, give him one of yours
Score: 0.07692307692307693 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Jaccard Coefficiency::0.8666666666666668
Total time taken: 0.565
```

1.4 Comparison and Observations

From the results we can observe that all the 3 methods can give good predictions on the test dataset. For case 1, 2, 3, 5, all the methods can get exactly the right answer, but for case 4, all the methods failed to figure out the correct rank of the true answer. Some more details around this issue will be given in the next segment.

The reason why these different methods share similar performance may lies on that the metric they use to calculate similarity are very identical to each other, especially on text analysis areas. The fundamental element taken into consideration is Token, and the way to represent similarity is to count overlaps.

Task 2 Error Analysis and Improvement

2.1 Error Analysis

From the system output we can find that all 3 methods failed in the 4th case on predicting the right answer. Let us take a close look at the special case. The question is "The shortest distance between new friends is a smile". The first difficulty to handle this case is that there are a lot of trivial words in this sentence that may be meaningless in our similarity computing methods such as "a", "the", "is" and another "a". These words have the same weight as other important words such as "friends", "smile" and "distance". So if we can eliminate these noises in the questions and answers, or at least reduce their weights, we can expect some improvement on the overall performance.

Also, there are some other ways to make potential boost of performance such as detecting negative relations in the sentence, transfer all words to their root forms and get semantic meanings by parsing the sentence.

2.2 Eliminating Stopwords

I apply the first thought of potential ways to improve our system from the previous segment, which is to delete all the meaningless words in the question and answers to take only keywords into consideration.

There is a file named "stopwords" given by TAs containing all the ordinary words that have few concrete meanings. Our system can generate a StopWordsList by reading through this file. And when we are going to calculate the similarities between question and answers, we firstly iterate through all the words in the sentence to see if they are in the StopWordsList. In this way, all redundant words can be eliminated from our metric.

The output of the improved system is shown below:

```
-------------------------------------------------------------------
Using Cosine Similarity:
Score: 0.6123724356957945 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.4082482904638631 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.5 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.13608276348795434 rank=2 rel=1 qid=4 If you see a friend without a smile, give him one of yours
```

```
Score: 0.2182178902359924 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Cosine Similarity::0.9

-------------------------------------------------------------------
Using Dice Coefficiency:
Score: 0.5454545454545454 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.4 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.5 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.13333333333333333 rank=2 rel=1 qid=4 If you see a friend without a smile, give him one of yours
Score: 0.2 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Dice Coefficiency::0.9

-------------------------------------------------------------------
Using Jaccard Coefficiency:
Score: 0.375 rank=1 rel=1 qid=1 Classical music may never be the most popular music
Score: 0.25 rank=1 rel=1 qid=2 Climate change and energy use are two sides of the same coin.
Score: 0.3333333333333333 rank=1 rel=1 qid=3 The best mirror is an old friend
Score: 0.07142857142857142 rank=2 rel=1 qid=4 If you see a friend without a smile, give him one of yours
Score: 0.1111111111111111 rank=1 rel=1 qid=5 Old friends are best
(MRR) Mean Reciprocal Rank with Jaccard Coefficiency::0.9
Total time taken: 0.596
```

From the output we can see that there is a slightly improvement on case 4, that the ranking of the correct answer changes from 3rd to 2nd while all other 4 cases remain 100% correctness. I think we can expect a larger improvement on other dataset which may have more data.

2.3 Reduce Time Consumption

From the previous output we can see that the execution time of improved system is slightly longer than the baseline system. This is because whenever we are going to calculate the similarity, our system has to iterate through all the words in the StopWordsList(which is comparatively very large) k times(k represents the number of words in a sentence).

In case both of the StopWordsList and TokenList are sorted list, we can terminate this process when the word in the StopWordsList is larger than the one in our original TokenList. This mechanism can reduce the total time consumption of the whole pipeline. In our experiment, the time period reduced from 0.596s to 0.572s.