3) collectionreaders/DocummentReader.java

This class is dedicated to text collection.

We had to made sure that everything was being done correctly.

This class separates: the text, quid and rel for each document (each line of the .txt provided in the project).

4) annotators/DocummentVectorAnnotator.java

//TODO: construct a vector of tokens and update the tokenList in CAS

Created a vector of tokens. Main emphasis is in method “createTermFreqVector”, which:

i) obtains each text, line by line

ii) separates the line into tokens.

iii) Populates “text”, “frequency” for type “Token” in CAS.

iv) Assigns correct frequency to each token, by counting its appearance in each line.

v) Assign tokens to “tokenList”, in order to populate “Document” in CAS.

5) casconsumers/RetrievalEvaluator.java

//TODO :: 1. construct the global word dictionary.

// 2. keep the word frequency for each sentence.

The first two steps are done by reading the corresponding values from the previously populated CAS.

Term-frequency vectors were created for this purpose.

// 3. Compute Cosine Similarity (between 2 sentences) and rank the retrieved sentences.

The cosine similarity is defined as:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum\limits_{i=1}^{n} A_i \times B_i}{\sqrt{\sum\limits_{i=1}^{n} (A_i)^2} \times \sqrt{\sum\limits_{i=1}^{n} (B_i)^2}}$$ [1]

// 4. Compute the MRR metric.

The similarity scores from the previous step need to be sorted (from highest to lowest) in order to rank the correct sentences (rel=1) and threfore obtain the MRR metric:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}.$$ [2]

[1] http://en.wikipedia.org/wiki/Cosine_similarity

[2] http://en.wikipedia.org/wiki/Mean_reciprocal_rank

3.3. Bonus

Other type of improvements would involve programming other similarity measure and comparing the results, in terms of scores and MRR.

Other two very similar and popular similarity measures are Tanimoto and Jaccard. They are defined as follows.

Tanimoto Similarity:

$$f(A,B) = \frac{A \cdot B}{|A|^2 + |B|^2 - A \cdot B}$$ [3]

Jaccard Similarity:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$ [4]

Since the MRR is based on the ranking, which in turn is done through a similarity measure, therefore changing the similarity index has great impact on the MRR's final result. The results are shown on the next section.

4. Results

The following results were obtained when using cosine similarity:

```
******************** Started VectorSpaceRetrieval ********************

Query: Classical music is dying
Correct Answer: Classical music may never be the most popular music
Score: 0.3849001794597505   rank =1    quid=1

Query: Energy plays an important role in climate change
Correct Answer: Climate change and energy use are two sides of the same
coin.
Score: 0.4629100498862757   rank =1    quid=2

Query: One's best friend is oneself
Correct Answer: The best mirror is an old friend
Score: 0.5  rank =2    quid=3
```

[3] http://en.wikipedia.org/wiki/Jaccard_index

[4] *Idem*

```
Query: Energy plays an important role in climate change
Correct Answer: Climate change and energy use are two sides of the same
coin.
Score: 0.1962615682814125   rank =3    quid=2

Query: One's best friend is oneself
Correct Answer: The best mirror is an old friend
Score: 0.25      rank =1    quid=3

(MRR) Mean Reciprocal Rank ::0.611111111111111


Total time taken: 1.087

***************************************************************
```

From this experiment we can see that, at least for the documents we have for testing, the cosine similarity is far better at calculating similarity scores than the tanimoto similarity.