# Task 1

In this task I have built vector space retrieval model using UIMA. Pipeline consists of only three steps:

1. **DocumentReader** (collection reader) – reads sentences (documents) with queries from input file and save them in CASes
2. **DocumentVectorAnnotator** (annotator) – that splits sentences (documents) into the separate tokens. Very simple tokenizer was used. It splits only on whitespaces and does not split on punctuation. It con-siders a sequence of adjacent white-spaces to be a single separator. It also does not lowercase the input and uses it as is.
3. **RetrievalEvaluator** (cas consumer) – process data annotated by DocumentVectorAnnotator and compute all important metrics.

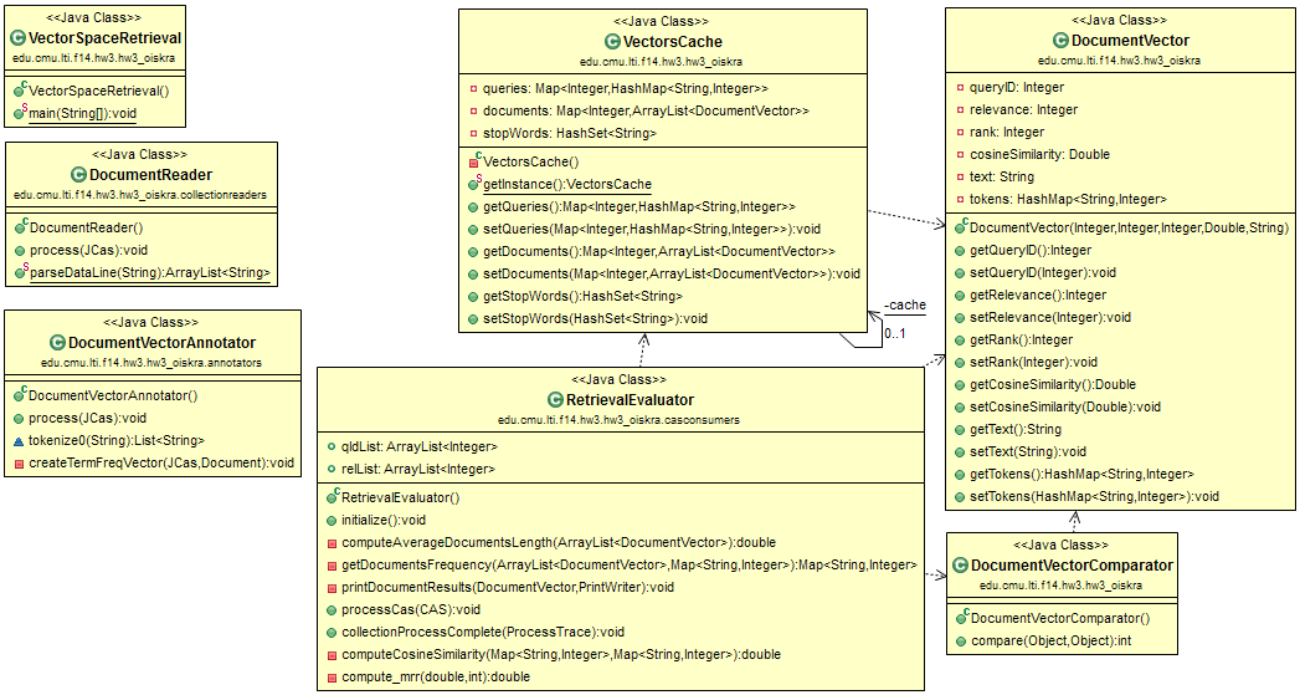


**Figure 1. Main parts of the pipeline. Data flows from upper left to down left corner and then right.**

In the vector space model, both the query and documents are sparse term vectors. Given a query vector and a set of short documents, we compute the cosine similarity values between the query vector and document vectors. Then, all documents are ranked in the decreasing order of the cosine similarity. [[1]] To save **sparse vectors** was built the separate class **DocumentVector** which stores tokens into HashMap<String, Integer>.

**Cosine similarity** is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any other angle. It is thus a judgement of orientation and not magnitude: two vectors with the same orientation have a Cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a

[1] From assignment description

similarity of -1, independent of their magnitude. Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1]. One of the reasons for the popularity of Cosine similarity is that it is very efficient to evaluate, especially for sparse vectors, as only the non-zero dimensions need to be considered.

Note that these bounds apply for any number of dimensions, and Cosine similarity is most commonly used in high-dimensional positive spaces. For example, in Information Retrieval and text mining, each term is notionally assigned a different dimension and a document is characterized by a vector where the value of each dimension corresponds to the number of times that term appears in the document. Cosine similarity then gives a useful measure of how similar two documents are likely to be in terms of their subject matter.

The cosine of two vectors can be derived by using the Euclidean dot product formula:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \cos\theta$$

Given two vectors of attributes, A and B, the cosine similarity, cos(θ), is represented using a dot product and magnitude as

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

The resulting similarity ranges from −1 meaning exactly opposite, to 1 meaning exactly the same, with 0 usually indicating independence, and in-between values indicating intermediate similarity or dissimilarity.

For text matching, the attribute vectors A and B are usually the term frequency vectors of the documents. The cosine similarity can be seen as a method of normalizing document length during comparison. [2]

To evaluate performance of my space retrieval system I used **Mean Reciprocal Rank** (MRR). It is a statistic measure for evaluating any process that produces a list of possible responses to a sample of queries, ordered by probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer. The mean reciprocal rank is the average of the reciprocal ranks of results for a sample of queries Q:

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

The reciprocal value of the mean reciprocal rank corresponds to the harmonic mean of the ranks. [3]

[2] Cosine similarity - https://en.wikipedia.org/wiki/Cosine_similarity
[3] Mean reciprocal rank - https://en.wikipedia.org/wiki/Mean_reciprocal_rank

Both cosine similarity and mean reciprocal rank were computed based completely on preceding formulas.

To evaluate performance I needed to aggregate information supplied with different CASes. Because the pipeline processes one CAS at a time, I created separate singleton class **VectorCache** that serve for the needs of caching resource. To understand the architecture of pipeline I provided its UML diagram (Figure 1).

## Task 2

After the baseline pipeline was done I performed the error analysis and on the examples that were not retrieved appropriately. I also categorized types of errors and counted the amount of error of every type. For better understanding data are arranged into the table.

| Id | Rank | Question & Answer | Query token | Token missed in relevant document | Error type |
|---|---|---|---|---|---|
| 1 | 2 | Q: Give us the name of the volcano that destroyed the ancient city of Pompeii<br>A: In A.D. 79, long-dormant Mount Vesuvius erupted, burying in volcanic ash the Roman city of Pompeii; an estimated 20,000 people died. | a) Pompeii<br>b) volcano | a) Pompeii;<br>b) volcanic | a) Tokenization<br>b) Stemming |
| 2 | 2 | Q: What has been the largest crowd to ever come see Michael Jordan<br>A: When Michael Jordan--one of the greatest basketball player of all time--made what was expected to be his last trip to play in Atlanta last March, an NBA record 62,046 fans turned out to see him and the Bulls. | a) Jordan | a) Jordan--one | a) Tokenization |
| 3 | 3 | Q: In which year did a purchase of Alaska happen?<br>A: Alaska was purchased from Russia in year 1867. | a) purchase | a) purchased | a) Stemming |
| 4 | 2 | Q: What year did Wilt Chamberlain score 100 points?<br>A: On March 2, 1962, Wilt Chamberlain scored a record 100 points in a game against the New York Knicks. | a) points?<br>b) score | a) points<br>b) scored | a) Tokenization<br>b) Stemming |
| 5 | 3 | Q: What river is called China's Sorrow?<br>A: People of China have mixed feelings about River, which they often call "sorrow of China" | a) China's<br><br>b) China's<br>c) river<br>d) Sorrow? | a) China<br><br>b) China"<br>c) River<br>d) "sorrow | a) Stemming & Tokenization<br>b) Tokenization<br>c) Casing<br>d) Tokenization & Casing |
| 6 | 2 | Q: Who was the first person to run the mile in less than four minutes | a) minutes | a) four-minute | a) Tokenization & Stemming |

| | | | | | |
|---|---|---|---|---|---|
| | | A: Roger Bannister was the first to break the four-minute mile barrier. | | | |
| 7 | 3 | Q: What year did Alaska become a state?<br>A: And that's not even to mention the breathtaking beauty of Alaska that became, in 1959, our 49th state. | a) state? | a) state. | a) Tokenization |
| 8 | 2 | Q: When did Mike Tyson bite Holyfield's ear?<br>A: Fighting for Holyfield's WBA heavyweight title on June 28, 1997, Tyson inexplicably bit his opponent's right ear. | a) bite<br>b) ear? | a) bit<br>b) ear. | a) Stemming<br>b) Tokenization |
| 9 | 2 | Q: What was the first spaceship on the moon<br>A: Luna 2 was the first spacecraft to reach the surface of the Moon. | a) moon<br><br>b) spaceship | a) Moon.<br><br>b) spacecraft | a) Tokenization & Casing<br>b) Synonyms |
| 10 | 1 | Q: Who won the Nobel Peace Prize in 1992?<br>A: Menchu won the Nobel peace prize in 1992. | a) 1992?<br>b) Peace<br>c) Prize | a) 1992.<br>b) peace<br>c) prize | a) Tokenization<br>b) Casing<br>c) Casing |

After combining errors by categories we can arrange error types from the most to less common:

- Tokenization – 11 errors
- Stemming – 6 errors
- Casing – 5 errors
- Synonyms – 1 error

The distribution of errors by categories gives us an idea how to improve our pipeline to get more useful results. I decided to use more robust versions of tokenizer, apply stemming, turn all letters to lowercase and remove stop words, provided in the separate input file, from input data. Tokenization was done by Stanford NLP Tokenizer [4], lemmatization – using Stanford CoreNLP lemmatization [5].

Except cosine similarity three other measures were used in addition to more robust tokenization, stemming and casing: Jaccard Index, Sørensen–Dice coefficient, and Ocapi BM 25.

The **Jaccard index**, also known as the Jaccard similarity coefficient (originally coined coefficient de communauté by Paul Jaccard), is a statistic used for comparing the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets:

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

(If A and B are both empty, we define J(A,B) = 1.) Clearly,

[4] Stanford NLP Tokenizer - http://nlp.stanford.edu/software/tokenizer.shtml
[5] Stanford CoreNLP - http://nlp.stanford.edu/software/corenlp.shtml

$$0 \leq J(A,B) \leq 1.$$

The MinHash min-wise independent permutations locality sensitive hashing scheme may be used to efficiently compute an accurate estimate of the Jaccard similarity coefficient of pairs of sets, where each set is represented by a constant-sized signature derived from the minimum values of a hash function. [6]

The **Sørensen–Dice index**, also known by other names (see Names, below), is a statistic used for comparing the similarity of two samples. It was independently developed by the botanists Thorvald Sørensen and Lee Raymond Dice, who published in 1948 and 1945 respectively.

Sørensen's original formula was intended to be applied to presence/absence data, and is

$$QS = \frac{2C}{A+B} = \frac{2|A \cap B|}{|A| + |B|}$$

where A and B are the number of species in samples A and B, respectively, and C is the number of species shared by the two samples; QS is the quotient of similarity and ranges from 0 to 1.[5] which is always in [0, 1] range.

It can be viewed as a similarity measure over sets:

$$s = \frac{2|X \cap Y|}{|X| + |Y|}$$

Similarly to Jaccard, the set operations can be expressed in terms of vector operations over binary vectors A and B:

$$s_v = \frac{2|A \cdot B|}{|A|^2 + |B|^2}$$

which gives the same outcome over binary vectors and also gives a more general similarity metric over vectors in general terms. [7]

In information retrieval, **Okapi BM25** is a ranking function used by search engines to rank matching documents according to their relevance to a given search query. It is based on the probabilistic retrieval framework developed in the 1970s and 1980s by Stephen E. Robertson, Karen Spärck Jones, and others.

The name of the actual ranking function is BM25. To set the right context, however, it usually referred to as "Okapi BM25", since the Okapi information retrieval system, implemented at London's City University in the 1980s and 1990s, was the first system to implement this function.

---

[6] Jaccard index - https://en.wikipedia.org/wiki/Jaccard_index
[7] Sørensen–Dice coefficient - https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient

BM25, and its newer variants, e.g. BM25F (a version of BM25 that can take document structure and anchor text into account), represent state-of-the-art TF-IDF-like retrieval functions used in document retrieval, such as web search.

BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document (e.g., their relative proximity). It is not a single function, but actually a whole family of scoring functions, with slightly different components and parameters. One of the most prominent instantiations of the function is as follows.

Given a query $Q$, containing keywords $q_1, \ldots, q_n$, the BM25 score of a document $D$ is:

$$\text{score}(D,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgdl}})},$$

where $f(q_i, D)$ is $q_i$'s term frequency in the document $D$, $|D|$ is the length of the document $D$ in words, and $avgdl$ is the average document length in the text collection from which documents are drawn. $k_1$ and $b$ are free parameters, usually chosen, in absence of an advanced optimization, as $k_1 \in [1.2, 2.0]$ and $b = 0.75$. $\text{IDF}(q_i)$ is the IDF (inverse document frequency) weight of the query term $q_i$. It is usually computed as:

$$\text{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5},$$

where $N$ is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing $q_i$. [8]

Different combinations of Tokenization, Stemming and Similarity Measures were assessed. Some of the results are combined into the table in form convenient for future analysis:

| Tokenization | Stemming | Stop words | Similarity | MMR |
|---|---|---|---|---|
| Stanford tokenizer | Yes | Yes | Jaccard index | 0.9667 |
| Whitespace tokenizer | Yes | No | Jaccard index | 0.8375 |
| Stanford tokenizer | Yes | No | Jaccard index | 0.8292 |
| Stanford tokenizer | Yes | Yes | Ocapi BM25 | 0.6625 |
| Stanford tokenizer | Yes | No | Cosine Similarity | 0.6625 |
| Stanford tokenizer | Yes | Yes | Sørensen–Dice coefficient | 0.6458 |
| Stanford tokenizer | Yes | No | Sørensen–Dice coefficient | 0.6292 |
| Whitespace tokenizer | Yes | No | Ocapi BM25 | 0.6250 |
| Stanford tokenizer | Yes | No | Ocapi BM25 | 0.6042 |
| Whitespace tokenizer | Yes | No | Cosine Similarity | 0.5500 |
| Whitespace tokenizer | Yes | No | Sørensen–Dice coefficient | 0.5000 |
| Whitespace tokenizer | No | No | Cosine Similarity | 0.4375 |

[8] Okapi BM25 - https://en.wikipedia.org/wiki/Okapi_BM25

At first sight seems that Jaccard Index provides pretty good similarity measure, at least on given input data. To decide whether those measures are statistically significant I performed paired t-test using Apache Commons TTest class [9]. Only the combinations that marked with green color got p-value more than 0.05 in paired t-test with baseline annotator. The combination of Stanford tokenizer, Stemming, Stop words pruning and Jaccard index got p-value equals to 1.3880e-08 despite the fact that MMR for this combination was the highest and equal to 0.9667. That is mean that performance analysis has to be complex in regard to measurements and statistical significance.

The future improvements could be probably achieved using Configuration Space Exploration Framework that we were studying at the class but it is out of the scope of this homework.

[9] Apache Common's TTest class - http://commons.apache.org/proper/commons-math/apidocs/org/apache/commons/math3/stat/inference/TTest.html