**Project 2**

**Andrii Ihnatov**

**Taivo Pungas**

**Karim Labib**

We implemented two models for this task. One is based on a term-based approach and the other based on a generative language model.

Though it is easier to just use distinct models separately namely one which uses tf-idf and another which uses a generative language model approach, in reality search engines usually use hundred of different features and score and rank query – document pairs based on all these features.

Therefore we decided to go with this approach of having different features augmented together in order to score query-document pairs.

We will explain each model separately and define what features we are using from each model and then show how we used them together.

**Preliminaries:**

Stop words were removed from both the query and the document. They are not used in the term based or language model.

**Term based approach:**

We defined different features for term-based model. The first basic feature is calculating the tf-idf score of a query-document pair.

We however augmented other features mentioned below:

* Calculation of the percentage of terms in common between the query and the title of the document. The intuition behind that is a more relevant document will more likely have words from the query in its title. For this scoring we used Porter’s stemmer on both the query terms and the title of the document since it is cheap to compute due to short lengths of query and document title.
* Another similar feature is the calculation of tf-idf score for the first 20% of the document. Again, a more relevant document will more likely have words from the query in its top part.
* We also have a tf-idf score for 50% of the document with the same intuition as before.
* Normal tf-idf score for the whole document.
* We also added some extra basic scoring such as percentage of query terms found in the document and a term overlap which is equal to the number of occurrences of query terms in the document divided by the multiplication of document Euclidean length (in vector space representation) and the query length.

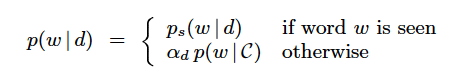
**Language based model:**

Different ways of smoothing for generative language models are present in the literature. We implemented two of them. The first is the *Jelinek-Mercer* methodthat was explained in the lecture. However, when reading a paper by Lafferty[1] he was comparing by experiments the performance of different smoothing methods and there he mentioned that the results showed that “The Dirichlet prior method generally performs

well, but tends to perform much better for concise title queries than for long verbose queries.” This led us to implement the Dirichlet prior method. Which fits our setting since the query terms we have are normally short and concise.

The formulas for two smoothing methods implemented are as follows:





**Classification step:**

By calculating all the previous features we aimed at constructing a feature vector out of them and pass this to a classifier, in our case random forest, and to train the classifier using the *qrel* data we have.

**Metrics calculation:**

When calculating the average precision for each query we divide by min(100, number of relevant documents for topic).

**Results:**

We tested the language model separately. When running the Jelinek-Mercer smoothing computation, the mean average precision was equal to 12%. When running using the Dirichlet smoothing, we found out that the mean average precision was equal to 14%, which affirmed the findings of the paper that Dirichlet smoothing gives better results for short and concise queries.

We also ran the algorithms with Porter’s stemmer. Porter stemmer didn’t affect much the machine learning model with all combined features. However it helped improve the results for the language model separately with an increase of MAP to 17%.

The results for the random forest model using features related to term based approach and with the language model score augmented to it was equal to 24% MAP. Metrics for both models are given below.

------ Language-based model -------

Mean precision: 0.302

Mean recall: 0.126

Mean F1-score: 0.159

Mean average precision: **0.144**

-----------------------------------

-------- Term-based model ---------

Mean precision: 0.274

Mean recall: 0.111

Mean F1-score: 0.140

Mean average precision: **0.245**

-----------------------------------