Sentence-level representations for document ranking

Manuel João Sousa Santos

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Bruno Emanuel Da Graça Martins
Prof. Pável Pereira Calado

Examination Committee

Chairperson: Prof. Alberto Manuel Rodrigues da Silva
Supervisor: Prof. Bruno Emanuel Da Graça Martins
Member of the Committee: Dr. Daniel Coelho Gomes

January 2021

Acknowledgments

I would like to thank my supervisors, professors Bruno Martins and Pável Calado, for the constant support throughout this work. I am very thankful for the sharing of knowledge and wisdom.

To my family, specially my parents and grandparents, for the unconditional love and constant care, for always being there when I needed, thank you.

To my friends, thank you all for the patience you have, for all the moments shared, and for the ones to come.

To my academic friends, for all the adventures, all the laughter and tears, for making me grow and be a better person, for all the memories, thank you.

Abstract

Pre-trained contextual language models based on Transformers have been successful in a number of applications in Natural Language Processing (NLP), and more recently also on Information Retrieval (IR) problems. In this thesis, we propose the use of sentence-level representations, built through this type of models, for ad-hoc document ranking problems. We predict relevance scores for long documents by aggregating sentence-level scores from a pool of candidate sentences, determined by a RoBERTa-based model. Experiments on the TREC GOV collection show that the proposed approach produces better results than using simpler ranking function based on sparse representations, like BM25.

Keywords

Natural Language Processing; Information Retrieval; Ad-Hoc Document Ranking; Pre-Trained Language Models

Resumo

Modelos de linguagem contextuais pré-treinados têm sido bem sucedidos em várias aplicações na área de processamento de linguagem natural, e mais recentemente em problemas de recuperação de informação. Neste trabalho, propomos o uso de representações de frases, criadas a partir deste tipo de modelos, para problemas de classificação de documentos. Calculamos a relevância de documentos extensos com base na agregação das pontuações de frases candidatas, determinadas por um modelo RoBERTa. Experiências na coleção GOV do TREC mostram que a abordagem proposta produz melhores resultados do que usar funções de classificação mais simples, baseadas em representações esparsas, como o BM25.

Palavras Chave

Processamento de Linguagem Natural; Recuperação de Informação; Classificação de Documentos; Modelos de Linguagem Pré-Treinados

Contents

List of Figures

List of Tables

Acronyms

IR	Information Retrieval
NLP	Natural Language Processing
MAP	Mean Average Precision
nDCG	Normalized Discounted Cumulative Gain
TF	Term Frequency
IDF	Inverse Document Frequency

1 Introduction

Contents

In this chapter, we introduce the theme of our thesis, formulated with an initial context of the problem we are facing and the correspondent motivation to solve it. Next, we state the objectives we achieved, followed by a brief summary, containing a list of the contributions to this area. Finally, we report the organization of the document.

1.1 Context and Motivation

The use of neural networks in Information Retrieval (IR) and particularly in ad hoc document ranking, has been expanding in the past years. Pre-trained contextual language models (PLMs), like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), are achieving state-of-the-art results on standard ad-hoc retrieval benchmarks and in a number of related natural language processing (NLP) tasks, such as question answering and text summarization. These models are being particularly successful because, unlike traditional context-free language embedding models like word2vec (Mikolov et al., 2013) or unidirectional language models like ELMo (Peters et al., 2018), BERT creates deep bidirectional representations. BERT relies on Transformer encoder (Vaswani et al., 2017) to generate a fixed sized length output representation which has a quadratic computational complexity to the input sequence, so the input sequence length is usually limited to 512 tokens. Therefore, when applying PLMs to the task of document ranking, these models often fall short to encode the entirety of most document contents, since their size usually surpasses the model limit. To avoid this problem, several previous studies predict relevance scores over sentences or passages, to be then aggregated into a document relevance score (Dai and Callan, 2019; Yilmaz et al., 2019).

An issue with passage-level approaches is that the majority of ad-hoc collections only have relevance judgments for the whole document, making it difficult to fine-tune a passage-based ranking model in the same domain. Given this problem, models based on BERT are mostly fine-tuned on MSMARCO, i.e. a passage ranking dataset (Nguyen et al., 2016), and to our knowledge no one has yet tried to modify an ad-hoc collection into a sentence-level weak labeled dataset. Given this problem, one of our motivations is to initially explore an unsupervised approach based on a RoBERTa model, previously trained for the task of semantic similarity between sentences. This way, our model can take advantage of the datasets built with sentence-labeled pairs to be used for document retrieval. Additionally, we consider the findings from Yang et al. (2019), that demonstrated the increased effectiveness of BERT when fined-tuned on the same task, to further fine-tune RoBERTa with a weak signaled dataset.

1.2 Objectives

In this thesis, we analyse how a RoBERTa model can be utilized in the task of document ranking, based on a sentence-level approach. We infer a document's relevance score by aggregating RoBERTa's scores of the document's best sentences. These candidate sentences are chosen based on their position and query term similarity. Furthermore, we adapt document-level relevance judgments into a weak supervised sentence-level dataset, in order to create an environment where RoBERTa can be fine-tuned and tested on the same domain and task. We evaluate the efficiency of our proposal on a TREC ad-hoc collection, concluding that our approach has promising results, outperforming the function BM25.

1.3 Summary of Contributions

In brief, this thesis has the following contributions:

- The proposal of a document ranking method based on a sentence-level approach, where only the best sentences are processed by a RoBERTa model, trained for sentence similarity, and aggregated into a final document-level relevance score.
- The creation of a weak-labeled dataset, where the document labels are adapted into sentence-level weak signals, in order to analyse how RoBERTa benefits from being fine-tuned on the same domain and task.
- An evaluation of our proposal on a standard ad-hoc TREC collection, showing the effectiveness of our sentence-level representation approaches, as well as further investigation about the impact of the following method's hyper-parameters: the variation of the RoBERTa model; the number of sentences considered; the number of documents reranked.

1.4 Document Organization

Our document is organized in the following way: In Chapter 1 we introduce the context and motivation of the proposed theme, followed by the objectives achieved. Chapter 2 presents an explanation of the fundamental concepts that serve as base to our proposed work, succeeded by the state-of-the-art approaches accomplished in the same areas of interest. Chapter 3 addresses the methodology and subsequent architecture of the proposed solution. Chapter 4 describes the experiments made, in order to test our proposed method. This includes an explanation of our experimental setup, followed by a deep analysis of the obtained results. Finally, Chapter 5 exposes the main contributions oh this thesis, as well as a delineation of promising future work that can be developed.

2

Concepts and Related Work

Contents

In this chapter, we present an initial explanation of the fundamental concepts that are the root of our work, followed by a review of the most relevant research done in connection to our theme.

2.1 Fundamental Concepts

In Subsection 2.1.1, we describe the classic ranking models that serve as baseline functions to our reranking methods. Subsection 2.1.2 explains the architecture of the most recent pre-trained contextual language models.

2.1.1 Classic Ranking Models

One type of Information Retrieval (IR) ranking models is based on vector space, where documents and queries are represented as vectors. Usually, each vector dimension is associated to a term, and if that term appears in the document, it is represented by a non-zero value weight. There are several ways to compute these weights. It can be calculated by the Term Frequency (TF), where the more relevant terms should have a higher number of appearances on the document. It can also be estimated by the Inverse Document Frequency (IDF), i.e. the most common terms that occur throughout the corpus have less relevance.

A popular approach, called TF-IDF, calculates the product between the TF and the IDF. This way, the relevance of a term occurring a high number of times in a document is balanced by the number of times that term appears in the remaining documents of the corpus. To estimate the final relevance score, we can compute the cosine similarity between the two vectors.

Another classic type of ranking models are the probabilistic models, which estimate the probability of a document being relevant to a query. This approach assumes that the documents with higher probabilistic scores should be ranked higher in the ranking list. Okapi BM25 is perhaps the most used ranking function for search engines, and also the most used baseline method in reranking neural models. Given a query q and a document D, the BM25 formula is denoted as follows, where $f(q_i, D)$ is the term frequency of term q_i and $|D|$ is the length of document D. The values k_1 and b are free parameters, used for optimization purposes, usually with $k_1 \in [1.2, 2.0]$ and $b = 0.75$.

$$\mathrm{rel}(q, D) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)} \tag{2.1}$$

The IDF is the inverse document frequency of term q_i, and is calculated by the following expression, where N is the total number of documents in the collection and $n(q_i)$ is the number of documents containing the term q_i.

$$\mathrm{IDF}(q_i) = \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right) \tag{2.2}$$

2.1.2 Bidirectional Encoding Representations from Transformers

Prior to the work of Vaswani et al. (2017), the most common encoders for deep neural ranking models were based on recurrent or convolutional neural networks. The new architecture Transformers (Vaswani et al., 2017) is based solely on attention mechanisms. It uses stacked self-attention and point-wise, fully connected layers for both the encoder and decoder. The encoder-decoder is composed by 6 identical layers, where each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second one is a fully connected feed-forward network. The decoder has an additional third sub-layer, which performs full head attention over the output of the encoder.

Vaswani et al. (2017) describes an attention function as mapping a query Q and a set of key-valued pairs (K, V) to an output. The output is then computed as a weighted sum of the values, where each value weight is calculated by a compatibility function of the query and the corresponding key. The attention function is computed by the dot product of the query with all keys, divided by the square root of the keys dimension d_k. This value is given as input to a softmax function, and the correspondent output is multiplied by the value of each word in the sequence. The attention function is described as follows:

$$\mathrm{Attention}(K, V, D) = \mathrm{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{2.3}$$

The multi-head attention function allows the model to gather information from different subspaces at different positions by linearly projecting the queries, keys and values h times with different, learned linear projections to d_q, d_k, and d_v dimensions, respectively. On each projected version is applied the attention function in parallel, resulting d_v dimensional output values. Finally, the values are concatenated and projected, resulting in the final values. The multi-head function is described as follows:

$$\begin{aligned} \mathrm{MultiHead}(K, V, D) &= \mathrm{Concat}(h_1, \ldots, h_n) \cdot W^O \\ h_i &= \mathrm{Attention}(Q \cdot W_i^Q, K \cdot W_i^K, V \cdot W_i^V) \end{aligned} \tag{2.4}$$

Devlin et al. (2019) introduced a new pre-trained contextual language model, named BERT. This model is designed to pre-train deep bidirectional representations from unlabeled text. BERT's architecture has two main phases: the pre-training and the fine-tuning. In the first one, the model is trained on unsupervised data for two different tasks. In the last phase, the BERT model is initialized with the pre-trained parameters, and all of the parameters are fine-tuned with labeled data for a particular downstream task. This has the advantage of being a model with an unified architecture across different tasks.

BERT is designed with a multi-layer bidirectional Transformer encoder, based on the work of Vaswani et al. (2017) previously described. In order to handle a variety of tasks, BERT input representation can hold a single sentence or a pair of sentences in one token sequence. A sequence is referring to the input token sequence to BERT. The first token is a special classification token, [CLS], that is used for classification tasks (e.g., the similarity between the two sentences). Thus, for a given token, the correspondent representation is constructed by the sum of its token, segment and position embeddings.

BERT is pre-trained in two different tasks. The first one is called Masked Language Model, and in order to train a deep bidirectional model, masks a percentage of tokens at random, and then predict those masked tokens. The second pre-training task is called Next Sentence Prediction, which prepares the model for downstream tasks, such as Question Answering or Natural Language Inference. The idea is to choose two sentences A and B, where 50$\%$ of the times B is the actual next sentence that follows sentence A, and the other 50$\%$ a random sentence is chosen from the corpus.

RoBERTa (Liu et al., 2019) is an optimized version of BERT, which fine-tunes a number of hyper-parameters, such as the learning rate, the number of warmup steps, and the size of the training batches. However, the biggest improvement comes from extending the pre-training phase to a larger number of datasets, with varying sizes and domains. Since BERT relies on large quantities of text, the increase of data size resulted in performance improvements over the downstream tasks.

2.2 Related Work

This section contains the state-of-the-art work of two different fields in IR. Subsection 2.2.1 includes recent work developed in the area of passage-level relevance ranking. Subsection 2.2.2 contains recent research about ranking models for the task of document ranking.

2.2.1 Passage-Level Relevance Ranking

In document ranking benchmarks, relevance judgments are almost always associated to the whole document, and hence traditional retrieval models calculate relevance scores based on document-level signals. However, of all the sentences that compose a document, only a select few are perhaps relevant for a given query. Given the increase of document lengths in full-text collections, Callan (1994) first proposed to consider passage relevance for retrieval tasks. He defined passages by splitting a document into three different ways: paragraph passages, bounded-paragraph passages, and window passages. After a document is split into passages, we can obtain passage relevance signals that can be used to calculate a final document-level relevance score (Liu and Croft, 2002).

More recently, Wu et al. (2019) studied the relation between passage-level relevance and document-level relevance judgments. They showed that position and query similarity of passages play a significant role in the determination of document-level relevance. These authors also demonstrated that on the

THUCNews[1] dataset, in average, a relevant document only has 23$\%$ of highly relevant passages. In subsequent work, Wu et al. (2020) proposed a model that uses a passage-level representation based on a cumulative gain, where the last passage cumulative gain represents the document-level cumulative gain. Unlike our work, they deal with a dataset with a passage-level ground truth.

In our work, we take into account the aforementioned findings in order to select a pool of candidate sentences to build a document relevance score. With this approach, we differ from most passage-level representation models, as we only aggregate the relevance scores of the most relevant sentences. This reduction of sentences processed by RoBERTa drastically decreases the computational costs.

2.2.2 Neural Ranking Models for IR

There is a large variety of ranking models, including vector space models (e.g., classic TF-IDF), probabilistic models (e.g., BM25 (Robertson et al., 1996)), feature-based learning to rank models (e.g., LambdaMART (Burges, 2010)) and neural ranking models (e.g., DSSM (Huang et al., 2013) or DRMM (Guo et al., 2016)). However, the contextual capacity of the aforementioned models is much more limited than a BERT-based model, pre-trained on a large-scale corpus. Recent work has shown that PLMs achieve state-of-the-art results in many NLP tasks, and also in IR problems (Lin et al., 2020). Nogueira and Cho (2019) first utilized BERT as a passage reranker, using the MSMARCO passage ranking dataset for fine-tuning the model. The authors use BERT's [CLS] vector as input to a single layer neural network, to obtain a final probability score. In subsequent work, Nogueira et al. (2019) developed a multi-stage document ranking architecture with BERT. In the first stage, the top-k_0 documents retrieved by a standard ranking function are reranked by a first BERT model. After that, the top-k_1 documents are then reranked by duoBERT, a second BERT model trained through a pair-wise classification approach. This has the ability to trade off quality against latency by controlling the number of documents that enter each stage.

Birch (Yilmaz et al., 2019) is another recent approach which started to utilize sentence-level labels, using BERT to create a document reranker. The authors estimate a document relevance score from the combination of the document's original score (e.g., obtained through a model like BM25) with the aggregation of the top-n most relevant sentences according to BERT. BERT-MaxP (Dai and Callan, 2019) is also a document reranker that instead explores passage-level signals. The authors adopt a simple passage-level approach by splitting the document into overlapping passages. BERT is then used to predict the relevance of each passage independently, and the final document score is obtained with the best passage.

CEDR (MacAvaney et al., 2019) corresponds to a joint approach that incorporates BERT's vector representation into existing neural models, such as DRMM. The paper's method is to use BERT's [CLS] vector, benefiting from deep semantic information, as well as individual contextualized token matches.

[1] http://thuctc.thunlp.org

PARADE (Li et al., 2020) is an end-to-end document reranking model that aggregates passage-level representations, overcoming the problem of performing inference over passages independently. The first step of PARADE is to represent a document as passages. To do so, a sliding window of 150 words is applied to the document with a stride of 100 words. In the next step, each passage is represented by BERT's [CLS] token, built from the concatenation between the query and the passage. In the passage aggregation phase, all passage representations are concatenated and the resulting vector is given as input to Transformer (Vaswani et al., 2017) layers, enabling interaction between passages and exploiting the ordering and dependencies between them. Finally, the [CLS] vector of the last Transformer output layer is given as input to a single-layer feed-forward network to generate the final document relevance score. BERT-QE (Zheng et al., 2020) outperforms standard BERT-based models by adding a phase of contextualized query expansion in their three phased approach. In phase one, a BERT model is used to re-rank a list of documents based on an unsupervised ranking model. In phase two, the top-k_d documents from the previous phase are selected to return the most relevant chunks of text, to serve as feedback information. In phase three, the selected chunks are used in combination with the query and the document to compute a final relevance score. For a deeper understanding about the evolution of text ranking, Lin et al. (2020) presented an overview on modern techniques.

2.3 Overview

Recent state-of-the-art research shows the importance of considering passage-level relevance for retrieval tasks. Pre-trained contextual language models such as BERT achieve state-of-the-art results in IR benchmarks by benefiting from their deep bidirectional encoding representations. However, since BERT-based models have a limited number of token capacity, the ranking models started to apply different innovative passage-level approaches to surpass this problem. In the next chapter, we describe a new sentence-level approach that utilizes the model RoBERTa fine-tuned for sentence similarity tasks.

3 Sentence-Level Representations for Document Ranking

Contents

In this chapter, we present the proposed method for document ranking using a sentence-level approach. For a given query q and a document D, we calculate a relevance score $\mathrm{rel}(q, D)$ that determines the importance of document D for the query q. This relevance is performed by aggregating the top-n best sentence-level scores, obtained by a neural model such as RoBERTa trained on sentence similarity tasks, (Reimers and Gurevych, 2019) into a document-level score. Figure 3.1 illustrates the general architecture of our proposal.

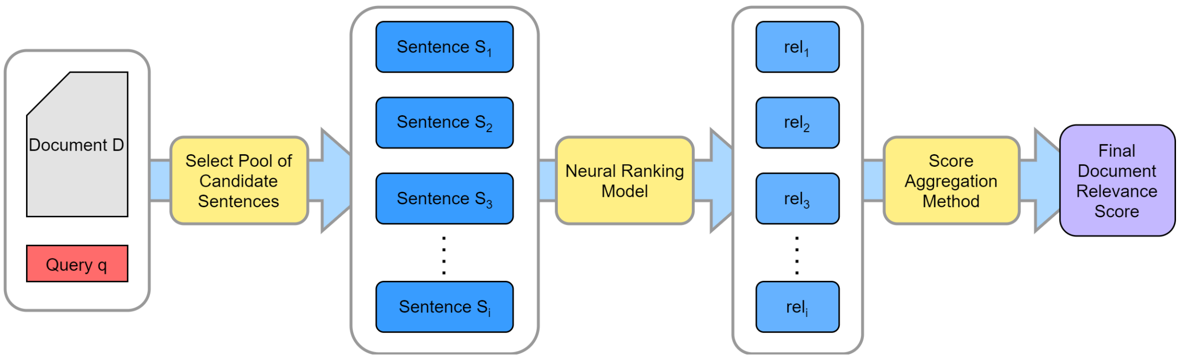

Figure 3.1: Illustration of our general document ranking architecture.

3.1 Choosing Candidate Sentences

We do not aim to use the neural ranking model to encode every single sentence in a document. Instead, we calculate a document relevance score based on a specific pool of candidate sentences, formally expressed as $D_P = \{S_1, \ldots, S_n\}$, where n is the number of sentences. This approach will lead to a significant reduction of computational costs, since most documents have a very large number of sentences.

We tested three different approaches to choose a pool of candidate sentences, based on two criteria: (i) the position of a sentence in the document; or (ii) the number of shared terms between a query and a sentence.

The approach named **FIRST** picks the first sentences of a document, exploring the fact that the most relevant information of a document tends to be near the beginning.

The approach named **TERMF** contains the sentences that have the highest raw term frequency score, denoted as follows:

$$\mathrm{tf}(q, S) = \sum_{t_i \in q} f_{i,S} \tag{3.1}$$

In the previous expression, $f_{i,S}$ is the raw count of query term t_i in sentence S. In the experiments, we ignored all terms that were either stop words or punctuation.

Finally, the approach named **FIRST+TERMF** corresponds to an aggregation of both sets. If the same sentence is in both groups, that sentence is not repeated and another is chosen from TERMF.

3.2 Creating Sentence Scores

For query q and sentence S_i, we use a RoBERTa model to generate a fixed sized vector representation for both query and sentence. This output is computed by calculating the mean of all vectors produced for the individual word pieces generated during tokenization, having q^{avg} and s_i^{avg}, denoted as follows:

$$q^{avg} = \mathrm{RoBERTa}(q) \tag{3.2}$$

$$s_i^{avg} = \mathrm{RoBERTa}(S_i) \tag{3.3}$$

Note that we do not use the traditional inference method of selecting the output token [CLS], given the concatenation of the two strings as input to a RoBERTa cross-encoder. In our experiments, this setup becomes too expensive because we are dealing with too many possible combination pairs. Since our focus is to efficiently find the most similar sentences given a query, it can be more beneficial to build a model properly trained to find semantic similarity between sentences. We follow the work done by Reimers and Gurevych (2019), which adds a mean-pooling operation to the output of RoBERTa (i.e., computing the mean of all output token vectors), in order to derive a fixed sized sentence embedding. With this approach, the authors designed a bi-encoder that maps each input independently, and then determines matching scores with the cosine similarity between the two vectors. In our experiments, we use as base model their version of RoBERTa fine-tuned on the combination of the SNLI (Bowman et al., 2015) and Multi-Genre NLI (Williams et al., 2018) datasets, and then on the Semantic Textual Search benchmark (STS-b) (Cer et al., 2017), since this model achieved state-of-the-art results for sentence similarity tasks.

The relevance score is then obtained by calculating the cosine similarity between the two vectors.

$$\mathrm{rel}(q, S_i) = \cos(\theta) = \frac{q^{avg} \cdot s_i^{avg}}{\|q^{avg}\| \times \|s_i^{avg}\|} \tag{3.4}$$

In the previous equation, $q^{avg} \cdot s_i^{avg}$ corresponds to the dot product between the vectors and $\| * \|$ is the vector norm.

3.3 Aggregating Sentence Relevance Scores

Given the pool of sentence relevance scores $D_{P_{rel}} = \{rel_1, \ldots, rel_n\}$, we can obtain a document relevance score in three different ways.

Max calculates a document relevance score by choosing the sentence with the highest score.

$$\text{rel}(q, D) = \max(rel_1, \ldots, rel_n) \tag{3.5}$$

Sum assumes that all candidate sentences must contribute equally in scoring a document, thus summing all relevance scores.

$$\text{rel}(q, D) = \sum_{i=1}^{n} rel_i \tag{3.6}$$

Weighted Mean considers that sentences with a higher query term frequency must have a higher weight on a document relevance score.

$$\text{rel}(q, D) = \frac{\sum_{i=1}^{n} w_i \times rel_i}{\sum_{i=1}^{n} w_i} \tag{3.7}$$

In the previous equation, w_i is the raw count of query q terms in the correspondent sentence S_i.

3.4 Combining Ranking Systems

Similarly to the work done by Yilmaz et al. (2019), we decided to combine the scores of two ranking systems (i.e., the initial ranking function and RoBERTa), in order to take advantage of both approaches. To do so, we used the fusion algorithm named MAPFuse (Lillis et al., 2010) to create a new ranking system, by combining the document scores given by the baseline ranking function and RoBERTa, with the help of their correspondent Mean Average Precision (MAP) scores over a held-out set of queries. The MAPFuse formula is denoted as follows; where S is the set of input systems that returned document D, MAP_s is the MAP score associated with system s, and $p_s(D)$ is the position of document D ranked by system s.

$$\text{rel}(q, D) = \sum_{s \in S} \frac{MAP_s}{p_s(D)} \tag{3.8}$$

3.5 Overview

Our goal is to find a document-level relevance score, based on a sentence-level representation approach of the most important sentences. The first step of our methodology is to choose those sentences. We used three different strategies, called FIRST, TERMF, and FIRST+TERMF, based on the position of the sentence in the document, and on the number of equal terms between the sentence and the query.

Next, the query and each candidate sentence is converted into a fixed size vector of embeddings by a RoBERTa-based model. This model is built with a bi-encoder and fine-tuned in sentence similarity tasks. The relevance score between each sentence and the query is calculated through the cosine similarity between the two vectors. After obtaining all the sentence relevance scores, the following step is to aggregate them into a final document-level relevance score. To do so, we apply three different aggregation techniques, called Max, Sum, and Weighted Mean.

Finally, after computing the new ranking system, by reranking the top-n documents, we use the fusion algorithm MAPFuse to combine RoBERTa's ranking system with the baseline ranking system, and derive a new one. The evaluation of our sentence-level ranking architecture is described in the next chapter.

4 Experimental Evaluation

Contents

In this chapter, we report the experimental setup developed in order to test our methodology. This chapter contains the dataset used, as well as the steps to evaluate, compare, and further fine-tune our methods. We also show the results obtained, with further analysis on the impact of certain hyper-parameters in our approaches.

4.1 Experimental Setup

To store and index our collection of documents we used Apache Solr[1], which is a well known text search platform. Considering that the majority of the GOV documents are in the HTML format, we created a parser to eliminate all document's unwanted content, like tags and Javascript code.

To split a document into sentences we used an English parser from the Spacy[2] library. When choosing the candidate sentences, we set to 10 the total number of sentences for the approaches named FIRST and TERMF, and 20 for the approach named FIRST+TERMF. These values were tuned based on a trade-off between sentence pool size and performance. In Section 4.2.3, we further investigate the variation of performance, given the different number of sentences processed by RoBERTa.

4.1.1 Dataset

We analysed our method with the ad-hoc retrieval collection named GOV[3]. This is a TREC Web collection crawled from government websites, with approximately 1.25 million documents. In our experiments, we used the TREC 2002 Web Topic Distillation topics as test data, and both TREC 2003 and 2004 Web Track topics as training data. Since we have some queries with only a title and others with title and description, we have chosen to uniformly use the title only for all queries, having a total of 775 queries. In average, each document has a much higher number of tokens than RoBERTa can handle, making GOV a reliable collection to test our hypothesis.

4.1.2 Evaluation Metrics

The evaluation is made with the MAP at cutoff 1000, the Precision at cutoff 10, and the Normalized Discounted Cumulative Gain (nDCG) at cutoff 1000 and 10. The MAP formula is denoted as follows:

$$\mathrm{MAP}(Q) = \frac{\sum_{q=1}^{Q} \mathrm{AP}(q)}{|Q|} \tag{4.1}$$

In the previous equation, $|Q|$ is the total number of queries and $\mathrm{AP}(q)$ is the average precision for query q, which is calculated as follows:

[1] https://lucene.apache.org/solr
[2] https://spacy.io
[3] http://ir.dcs.gla.ac.uk/test_collections/govinfo.html

$$\mathrm{AP}(q) = \frac{\sum_{k=1}^{n} \mathrm{P}(k) \times rel(k)}{\#\mathrm{RelevantDocuments}} \tag{4.2}$$

In the previous equation, $\mathrm{P}(k)$ corresponds to the precision at cutoff k documents and $rel(k)$ is 1 or 0 depending if the document is relevant or non-relevant, respectively.

In turn, the nDCG formula is denoted as follows:

$$\mathrm{nDCG}_p = \frac{\mathrm{DCG}_p}{\mathrm{IDCG}_p} \tag{4.3}$$

In the previous equation, p is a rank position and IDCG_p is the value of DCG_p sorted by relevance. DCG_p can be obtained by the following formula:

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{rel(i)}{\log_2(i+1)} \tag{4.4}$$

The reranking threshold was set to 30 for optimal performance. In Section 4.2.4, we validate this choice by studying how the variation of the number of documents that are reranked affects the overall performance of our method.

4.1.3 Models Under Comparison and Training Strategies

We compare our RoBERTa models against two traditional baselines, both implemented within Solr.

BM25 is an unsupervised ranking function that scores a document based on the term frequency and the inverse document frequency, considering the document length as a normalization factor (Robertson et al., 1996). We set BM25 parameters as default, with $k_1 = 1.2$ and $b = 0.75$.

BM25+Porter combines BM25 with a stemming algorithm that reduces inflected or derived words to their root form (Porter, 1980). This baseline also removes stop words with a Solr predefined list.

To check the performance of our baselines, we validate them against the method implemented by Bennett et al. (2008). These authors reported to have used BM25 tuned with the same parameter values, also having the text pre-processed with Porter's algorithm and a stop words list. As shown in Table 4.1, for all the topic's years considered, we can see a substantial improvement from pre-processing the documents with a stemming algorithm and a list of stop words. There is also a slight improvement from our implementation of BM25+Porter compared with the one made by Bennett et al. (2008) for the year 2002, which validates our reranking baseline. Given these results, we decided to use the top 1000 documents retrieved by the method BM25+Porter in our reranking methods.

As mentioned previously in Section 3.2, we use a publicly available RoBERTa-Base[4] model, already fine-tuned for sentence similarity. In order to further fine-tune RoBERTa for the GOV dataset, we need to

[4]https://github.com/UKPLab/sentence-transformers

	2002 Topics		2003 Topics		2004 Topics	
Model	MAP@1K	P@10	MAP@1K	P@10	MAP@1K	P@10
BM25	0.1617	0.1980	0.0892	0.0680	0.2321	0.0693
BM25+Porter	0.1915	0.2460	0.0858	0.0720	0.2478	0.0707
BM25+Porter (Bennett et al., 2008)	0.1888	0.2420	-	-	-	-

Table 4.1: Results of the different baselines, considering the TREC Web Topics Distillation topics from the years of 2002, 2003, and 2004.

adapt the relevance judgments from documents to sentences. To do so, for each document, we choose the most relevant sentence from the pool of candidate sentences given by FIRST+TERMF and use that sentence as an instance. With this approach, we assume that all instances taken from relevant documents are relevant (i.e., similar to the query title) and all instances taken from non-relevant documents are non-relevant.

Training is performed on a single GPU GeForce GTX 1080, using a triplet loss where the anchor input s_a is compared to a positive input s_p and a negative input s_n (i.e., a query is compared to a relevant and a non-relevant sentence) denoted as:

$$\mathcal{L} = \sum_{i \in b} \max(\|s_{a_i} - s_{p_i}\| - \|s_{a_i} - s_{n_i}\| + \epsilon, 0) \tag{4.5}$$

In the previous equation, b is the batch of training instances, $\|\cdot\|$ is the Euclidean distance metric, and ϵ is a margin. Thus, the model is tuned so that the distance between the query and a relevant sentence is lower than the distance between the query and a non-relevant sentence.

The training data was constructed by pairing a relevant sentence with a non-relevant one from a random document that is picked from the top-50 non-relevant documents retrieved by BM25. We also use data augmentation by repeating each relevant document a total of five times, pairing it with different negative sentences. We fine-tune the model for 2 epochs with batches of 8 training instances, with a 10% random split between training and development data, having a total of 25200 training instances. We use the Adam optimizer with a learning rate of 3e-5 and with 10% of training data for warm-up.

4.2 Experimental Results

In this section, we analyse the results of the different sentence-level representations, as well as the impact of the following questions in our methods:

- How does the number of sentences that is considered affect the performance of our ranking method?
- Can a different version of RoBERTa improve effectiveness without losing efficiency?

	Single System				MAPFuse			
Model	MAP@1K	nDCG@1K	P@10	nDCG@10	MAP@1K	nDCG@1K	P@10	nDCG@10
BM25	0.1617	0.4129	0.1980	0.2440	-	-	-	-
BM25+Porter	0.1915	0.4648	0.2460	0.3049	-	-	-	-
BM25 (Bennett et al., 2008)	0.1888	-	0.2420	-	-	-	-	-
RoBERTa (Full Text)	0.1533	0.4322	0.2400	0.2713	**0.1911**	0.4646	0.2480	0.3074
1. RoBERTa$_{\text{Max}}$	0.1512	0.4286	0.2400	0.2616	0.1887	0.4624	0.2620	0.3124
1. RoBERTa$_{\text{Sum}}$	0.1556	0.4354	0.2540	0.2791	0.1857	0.4603	0.2600	0.3094
1. RoBERTa$_{\text{W.Mean}}$	0.1365	0.4182	0.2120	0.2397	0.1827	0.4559	0.2320	0.2884
2. RoBERTa$_{\text{Max}}$	0.1488	0.4333	0.2100	0.2515	0.1838	0.4613	0.2540	0.3062
2. RoBERTa$_{\text{Sum}}$	0.1592	0.4337	0.2120	0.2504	0.1815	0.4580	0.2340	0.2888
2. RoBERTa$_{\text{W.Mean}}$	0.1417	0.4181	0.2020	0.2256	0.1815	0.4574	0.2340	0.2867
3. RoBERTa$_{\text{Max}}$	0.1527	0.4299	0.2360	0.2578	0.1891	0.4632	0.2620	0.3120
3. RoBERTa$_{\text{Sum}}$	0.1655	0.4418	0.2200	0.2664	0.1898	0.4657	0.2500	0.3117
3. RoBERTa$_{\text{W.Mean}}$	0.1504	0.4285	0.2060	0.2383	0.1840	0.4584	0.2300	0.2872
3. RoBERTa$_{\text{Max}}$ (fine-tuned)	0.1516	0.4394	0.2240	0.2732	0.1884	**0.4663**	**0.2660**	**0.3234**

Table 4.2: Results of different models on the GOV dataset, considering the TREC 2002 Web Topic Distillation topics. The rows labeled with 1. 2. or 3. correspond to the FIRST, TERMF, and FIRST+TERMF approaches, respectively. Best results are in **bold**.

- How does the number of documents that are reranked by RoBERTa affect the performance of our ranking method?

4.2.1 Different Sentence-Level Representations

The ranking performance of our methods is shown in Tables 4.2, 4.3, and 4.4, corresponding to the 2002, 2003, and 2004 topic distillation years, respectively. We can see that when RoBERTa uses the document's full content, cut to the maximum number of word pieces that are allowed, it under-performs over some sentence-level versions for all the different topic distillation years. This confirms that for long-sized documents, RoBERTa benefits from a sentence-level representation approach. We can also conclude that almost all the MAPFuse results, given by the fusion between a RoBERTa reranking system and the BM25+Porter baseline ranking system, perform better than it's systems evaluated separately. In Table 4.3, we can observe that the best approach of the 2003 set, 3. RoBERTa$_{\text{Max}}$, already surpasses the results given by BM25+Porter, suggesting that, in some cases, our RoBERTa approach can outperform the baseline ranking function without the need of a fusion algorithm.

When analysing the different methods for choosing the candidate sentences, FIRST+TERMF yields the best results for the 2002 and 2003 sets, while FIRST is the most efficient approach for the 2004 set. This shows that the first sentences are essential to consider in a sentence-level approach, confirming that relevant information tends to be near the top of a document. On the other hand, the approach TERMF gave the worst results in all three experiments, reinforcing the importance of considering the document's first sentences, even if they have a low number of terms in common with the query.

As for the score aggregation methods, we can see that the most effective relevance score aggrega-

	Single System				MAPFuse			
Model	MAP@1K	nDCG@1K	P@10	nDCG@10	MAP@1K	nDCG@1K	P@10	nDCG@10
BM25	0.0892	0.2897	0.0680	0.1161	-	-	-	-
BM25+Porter	0.0858	0.2935	0.0720	0.1123	-	-	-	-
RoBERTa (Full Text)	0.0889	0.3014	0.0800	0.1194	0.0945	0.3028	0.0880	0.1292
1. $\text{RoBERTa}_{\text{Max}}$	0.0998	0.3056	0.0840	0.1316	0.0866	0.2957	0.0800	0.1158
1. $\text{RoBERTa}_{\text{Sum}}$	0.0855	0.2968	0.0880	0.1242	0.0924	0.3018	**0.0900**	0.1286
1. $\text{RoBERTa}_{\text{W.Mean}}$	0.0854	0.2934	0.0700	0.1105	0.0930	0.2980	0.0820	0.1198
2. $\text{RoBERTa}_{\text{Max}}$	0.0708	0.2801	0.0640	0.0896	0.0754	0.2842	0.0840	0.1075
2. $\text{RoBERTa}_{\text{Sum}}$	0.0910	0.3008	0.0740	0.1205	0.0818	0.2916	0.0760	0.1124
2. $\text{RoBERTa}_{\text{W.Mean}}$	0.0827	0.2931	0.0700	0.1092	0.0846	0.2935	0.0860	0.1201
3. $\text{RoBERTa}_{\text{Max}}$	**0.1092**	**0.3115**	**0.0900**	**0.1374**	0.1019	0.3073	0.0840	0.1300
3. $\text{RoBERTa}_{\text{Sum}}$	0.0975	0.3021	0.0700	0.1151	0.0880	0.2963	0.0760	0.1142
3. $\text{RoBERTa}_{\text{W.Mean}}$	0.0936	0.3056	0.0720	0.1261	0.0918	0.3012	0.0800	0.1247

Table 4.3: Results of different models on the GOV dataset, considering the TREC 2003 Web Topic Distillation topics. The rows labeled with 1. 2. or 3. correspond to the FIRST, TERMF, and FIRST+TERMF approaches, respectively. Best results are in **bold**.

tion method is Sum for the 2002 set and Max for the 2003 and 2004 sets. This suggests that in particular topics it is more advantageous to predict a relevance score based equally on multiple sentences and in others to use the single most relevant sentence. Weighted Mean had always lower results than the other two methods, which is perhaps due to the fact that the first sentences in a document tend to be somewhat equally relevant. Also, BERT based models rely on contextual semantic information to predict it's embeddings, meaning that a high term frequency between query and sentence does not necessarily translate on a high similarity. In terms of the improvements of the nDCG@10 metric, the 2002 set had a maximum improvement of 2.5% over the initial baseline ranker BM25+Porter, while the 2003 set had an increase of 22.4% and the 2004 set improved 11.8%.

The best results regarding the RoBERTa's model fine-tuned on the GOV weak labeled dataset can be seen in the last row of Table 4.2. We can verify that this model achieved the best value of nDCG@10, improving 6.1% over BM25+Porter, while the other metrics are in line with the previous best RoBERTa method. We only reported the most effective approach, which was the FIRST+TERMF sentence pool together with Max aggregation. This was because, when building the dataset, we chose only the best sentence from the document's pool of sentences given by FIRST+TERMF.

4.2.2 Different RoBERTa Models

In this section, we analyse the effectiveness and efficiency of our model compared with RoBERTa-Large, a more robust version of RoBERTa, as well as against a DistilRoBERTa model trained on the MSMARCO dataset. Although approaches based on Transformer models such as RoBERTa have achieved state-of-the-art results, they are computationally expensive. We need to consider that the ranking system will be applied in real time search engines and thus it is necessary to have an efficient architecture.

	Single System				MAPFuse			
Model	MAP@1K	nDCG@1K	P@10	nDCG@10	MAP@1K	nDCG@1K	P@10	nDCG@10
BM25	0.2321	0.3943	0.0693	0.2746	-	-	-	-
BM25+Porter	0.2478	0.4057	0.0707	0.2914	-	-	-	-
RoBERTa (Full Text)	0.2079	0.3741	0.0702	0.2532	0.2498	0.4091	0.0782	0.3035
1. RoBERTa$_{\text{Max}}$	0.2375	0.4011	0.0716	0.2837	**0.2736**	**0.4286**	**0.0800**	**0.3257**
1. RoBERTa$_{\text{Sum}}$	0.2134	0.3785	0.0707	0.2592	0.2662	0.4226	0.0769	0.3162
1. RoBERTa$_{\text{W.Mean}}$	0.2037	0.3705	0.0618	0.2429	0.2640	0.4196	0.0724	0.3059
2. RoBERTa$_{\text{Max}}$	0.1898	0.3637	0.0627	0.2329	0.2591	0.4152	0.0738	0.3026
2. RoBERTa$_{\text{Sum}}$	0.1914	0.3568	0.0551	0.2217	0.2567	0.4123	0.0680	0.2931
2. RoBERTa$_{\text{W.Mean}}$	0.2047	0.3699	0.0591	0.2419	0.2574	0.4129	0.0724	0.2993
3. RoBERTa$_{\text{Max}}$	0.2271	0.3953	0.0764	0.2827	0.2590	0.4173	0.0787	0.3145
3. RoBERTa$_{\text{Sum}}$	0.1765	0.3475	0.0609	0.2179	0.2455	0.4044	0.0724	0.2892
3. RoBERTa$_{\text{W.Mean}}$	0.1669	0.3410	0.0631	0.2131	0.2505	0.4090	0.0724	0.2967

Table 4.4: Results of different models on the GOV dataset, considering the TREC 2004 Web Topic Distillation topics. The rows labeled with 1. 2. or 3. correspond to the FIRST, TERMF, and FIRST+TERMF approaches, respectively. Best results are in **bold**.

In Table 4.5, we have the sizes of the models that were considered, as well as their correspondent inference time over a document. The inference time estimates in seconds the encoding of the query and each sentence, plus the computation of the cosine similarity between them. RoBERTa-Large takes approximately 119% more computational time than RoBERTa-Base, while the distilled version of RoBERTa-Base is 36% more time efficient.

Table 4.6 shows the results achieved by the best methods for the three different models. Regarding the results of RoBERTa-Large, we can see that with no system fusion there is an increase of performance for all it's methods. This fact suggests that when we only consider the scores of RoBERTa, the model with the biggest size tends to perform better. For the MAPFuse results, RoBERTa-Large has better results than RoBERTa-Base when using the Max aggregation method, but performs worse regarding the Sum aggregation method. Overall, we can conclude that for a bigger RoBERTa version, a loss in efficiency does not necessarily translate in a significant improvement of effectiveness.

As for DistilRoBERTa-Base, we wanted to investigate if a distilled version of RoBERTa-Base, fine-tuned on the MSMARCO dataset could bring performance improvements. With this experiment, we are trying to see if there are significant differences when RoBERTa is trained on a similar ranking task instead of a semantic task. We can see in Table 4.6 that DistilRoBERTa has promising results given a single system, beating the other two models. However, it does not achieve considerable improvements over the best methods. Overall, DistilRoBERTa gives a sense that it can be a possible option to explore the model's fine-tuning phase on a passage-level dataset, given the advantage of decreasing the computational costs.

Model	# Layers	# Layer Size	# Parameters	Inference Time (s / doc)
RoBERTa-Large	24	1024	355M	0.10
RoBERTa-Base	12	768	125M	0.05
DistilRoBERTa-Base	6	768	82M	0.03

Table 4.5: Different versions of RoBERTa, compared in terms of size and computational time. Inference time over a document is estimated considering the use of the first 10 sentences.

	Single System				MAPFuse			
Model	MAP@1K	nDCG@1K	P@10	nDCG@10	MAP@1K	nDCG@1K	P@10	nDCG@10
1. RoBERTa-Large$_{\text{Max}}$	0.1550	0.4327	0.2500	0.2723	0.1871	0.4646	0.2660	0.3197
1. RoBERTa-Base$_{\text{Max}}$	0.1512	0.4286	0.2400	0.2616	0.1800	0.4565	0.2660	0.3130
1. DistilRoBERTa-Base$_{\text{Max}}$	0.1546	0.4368	0.2400	0.2808	0.1784	0.4617	0.2580	0.3187
1. RoBERTa-Large$_{\text{Sum}}$	0.1563	0.4355	0.2520	0.2815	0.1844	0.4718	0.2620	0.3308
1. RoBERTa-Base$_{\text{Sum}}$	0.1556	0.4354	0.2540	0.2791	0.1879	**0.4736**	0.2660	**0.3322**
1. DistilRoBERTa-Base$_{\text{Sum}}$	0.1602	0.4421	0.2320	0.2829	0.1870	0.4619	0.2640	0.3181
3. RoBERTa-Large$_{\text{Max}}$	0.1554	0.4322	0.2600	0.2742	0.1738	0.4647	0.2660	0.3202
3. RoBERTa-Base$_{\text{Max}}$	0.1527	0.4299	0.2360	0.2578	0.1803	0.4610	0.2560	0.3086
3. DistilRoBERTa-Base$_{\text{Max}}$	0.1615	0.4512	0.2420	0.2997	0.1927	0.4710	0.2600	0.3317
3. RoBERTa-Large$_{\text{Sum}}$	0.1582	0.4375	0.2320	0.2704	0.1798	0.4671	0.2580	0.3198
3. RoBERTa-Base$_{\text{Sum}}$	0.1655	0.4418	0.2200	0.2664	**0.1938**	0.4702	0.2620	0.3243
3. DistilRoBERTa-Base$_{\text{Sum}}$	0.1733	0.4474	0.2320	0.2826	0.1924	0.4670	**0.2680**	0.3276

Table 4.6: Comparison between the results of different RoBERTa versions. The rows labeled with 1. or 3. correspond to the FIRST and FIRST+TERMF approaches, respectively. Best results are in **bold**.

4.2.3 Number of Considered Sentences

One important hyper-parameter is the number of sentences considered when choosing the document's candidate sentences. In this section, we analyse how the variation of sentences processed by RoBERTa influences the reranking effectiveness of our approaches.

Figure 4.1 shows the results in terms of nDCG@10, with the number of sentences varying from 8 to 64. It would be expected for RoBERTa to increase its performance with a larger amount of document data preserved. However, we can see that for RoBERTa$_{\text{Sum}}$ and RoBERTa$_{\text{W.Mean}}$ the more sentences are considered, the less effective the model becomes. On the other hand, RoBERTa$_{\text{Max}}$, whose score aggregation method only considers the single most relevant sentence, has stable results with the increase in the number of sentences. These results validate our hypothesis that it is beneficial, not only in terms of computational costs but also performance-wise, to select a pool of the document's most relevant sentences to be processed by RoBERTa.

4.2.4 Number of Reranked Documents

The majority of neural ranking methods apply their model to the top-n documents retrieved by an initial ranking function. In our case, RoBERTa reranks the top documents retrieved by the baseline named

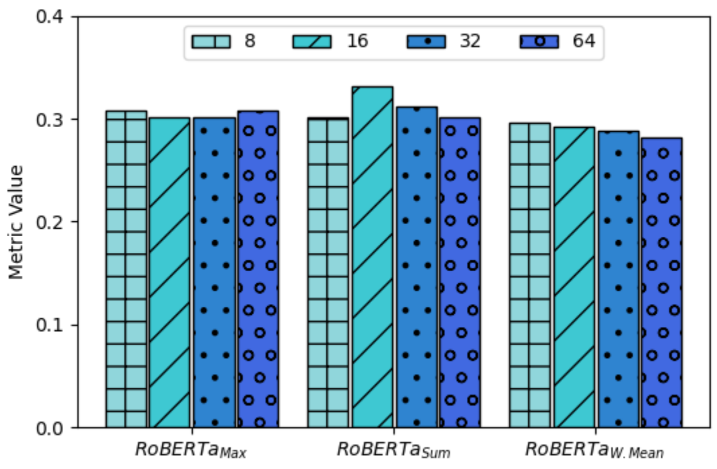

Figure 4.1: Results using 1. RoBERTa for different numbers of candidate sentences. nDCG@10 is reported.

BM25+Porter, which is described in Section 4.1.3. In this section, we investigate the impact of varying the number of documents that are reranked by our method.

Figure 4.2 shows the performance of $\text{RoBERTa}_{\text{Sum}}$ with the FIRST method, with the number of documents reranked by RoBERTa varying from 10 to 100. We can see that nDCG@10 has its highest value at 20 reranked documents, and then slowly decreases with the increase of documents. P@10 reaches its maximum value at 20, 30, and 50 documents and then falls when considering 100 documents. We can conclude that from 50 reranked documents the model is not capable of increasing its performance. Considering these values and the trade-off between effectiveness and computational cost, we decided to fix the reranking threshold to 30 documents in our experiments.

4.3 Overview

We started our experimental setup by choosing the text search platform Apache Solr to store and index the collection of documents we are analysing. This collection is named GOV and it contains approximately 1.25 million documents extracted from government websites. To evaluate our methods, we chose the metrics MAP at cutoff 1000, the Precision at cutoff 10, and the nDCG at cutoff 1000 and 10. Our RoBERTa models reranked the top 1000 documents from the baseline ranking function BM25, with text pre-processed with Porter's algorithm and a list of stop words. These metrics will evaluate three different distillation topics from the years 2002, 2003, and 2004. We also describe the fine-tuning process of the RoBERTa model, in order to test if we could have performance improvements by further fine-tuning the model in a weak-labeled dataset from the same domain.

In the experimental results we concluded that, for all the different experiments, our methods beat the original baseline BM25+Porter, and also beat the method in which all the document's content is

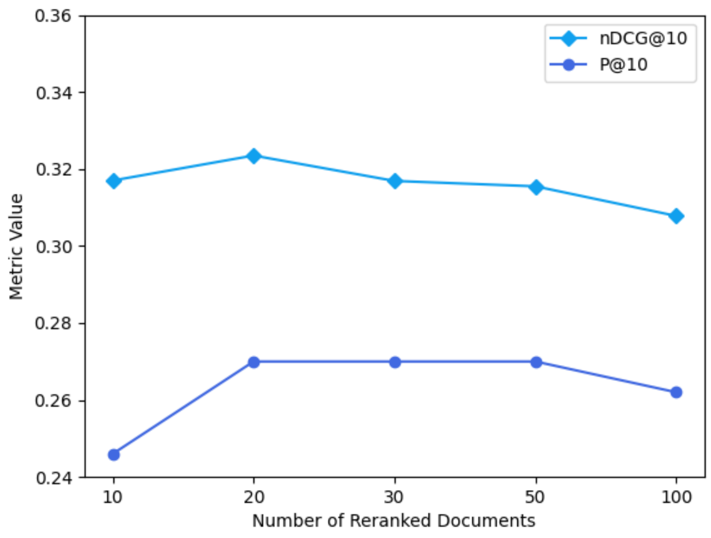

Figure 4.2: Results using 1. RoBERTa$_{\text{Sum}}$ for different numbers of reranked documents. nDCG@10 and P@10 are reported.

used (i.e., using the full token capacity of RoBERTa). Furthermore, we concluded that in some cases, the reranking system RoBERTa achieved better results than the baseline, without the use of the fusion algorithm MAPFuse. In further investigation, we analysed the behaviour of our best methods when we changed some of their hyper-parameters: the type of RoBERTa model used, the number of considered sentences, and the number of documents reranked. The next chapter will conclude this thesis by reporting the main contributions achieved by this work, follows by a description of possible ways to improve the results that were previously mentioned.

5 Conclusion

Contents

This chapter closes our thesis, with a description of the contributions made with this thesis and a discussion of the possible paths to further develop our work.

5.1 Main Contributions

We proposed a sentence-level approach based on RoBERTa for the task of document ranking, analysing its performance on the TREC ad-hoc collection named GOV. First, we pick a pool of candidate sentences to be processed by RoBERTa so as to generate relevance scores, and then aggregate the scores a final document-level relevance score. We studied different ways of choosing the best candidate sentences, as well as different aggregation methods. Additionally, we investigated the importance of creating an environment where the model is fine-tuned on the same domain and task, by converting the document-level labels into weak sentence-based signals. Experimental results show that our approach beats the baseline ranking function BM25 and has better results than using a document-level model architecture.

5.2 Future Work

For further work, we believe it would be interesting to test this method with different collections, that have already been used with recent state-of-the-art models, so that we have a more clear comparison between methods. Since our method is fully based on a sentence-level approach, from the training phase to the inference phase, we could compare it with a similar method, based on a passage-level approach. Also, we only tested a select few relevance score aggregation methods. More advanced functions can be implemented to further improve the results, including the use of rank aggregation methods such as MAPFuse to combine the scores from the candidate sentences. As for the model training phase, more elaborate strategies to choose the representative sentences for each document is also a promising path for future work.

Bibliography

Bennett, G., Scholer, F., and Uitdenbogerd, A. (2008). A comparative study of probabilistic and language models for information retrieval. In *Australasian Database Conference*, pages 65–74. Australian Computer Society.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 632–642. Association for Computational Linguistics.

Burges, C. J. C. (2010). From RankNet to LambdaRank to LambdaMART: An Overview. *Machine Learning*, 11(23-581).

Callan, J. P. (1994). Passage-level evidence in document retrieval. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310. Springer-Verlag.

Cer, D. M., Diab, M. T., Agirre, E., Lopez-Gazpio, I., and Specia, L. (2017). SemEval-2017 task 1: Semantic textual similarity - Multilingual and cross-lingual focused evaluation. *arXiv:1708.00055*.

Dai, Z. and Callan, J. (2019). Deeper text understanding for IR with contextual neural language modeling. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 985–988. Association for Computing Machinery.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805v2*.

Guo, J., Fan, Y., Ai, Q., and Croft, W. B. (2016). A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 55–64. Association for Computing Machinery.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., and Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the ACM International*

Conference on Information and Knowledge Management, pages 2333–2338. Association for Computing Machinery.

Li, C., Yates, A., MacAvaney, S., He, B., and Sun, Y. (2020). PARADE: Passage representation aggregation for document reranking. *arXiv:2008.09093*.

Lillis, D., Zhang, L., Toolan, F., Collier, R. W., Leonard, D., and Dunnion, J. (2010). Estimating probabilities for effective data fusion. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 347–354. Association for Computing Machinery.

Lin, J., Nogueira, R., and Yates, A. (2020). Pretrained transformers for text ranking: BERT and beyond. *arXiv:2010.06467*.

Liu, X. and Croft, W. B. (2002). Passage retrieval based on language models. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 375–382. Association for Computing Machinery.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*.

MacAvaney, S., Yates, A., Cohan, A., and Goharian, N. (2019). CEDR: Contextualized embeddings for document ranking. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1101–1104. Association for Computing Machinery.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). MS MARCO: A human generated machine reading comprehension dataset. *arXiv:1611.09268*.

Nogueira, R. and Cho, K. (2019). Passage re-ranking with BERT. *arXiv:1901.04085*.

Nogueira, R., Yang, W., Cho, K., and Lin, J. (2019). Multi-stage document ranking with BERT. *arXiv:1910.14424*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv:1802.05365*.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992. Association for Computational Linguistics.

Robertson, S., Walker, S., Hancock-Beaulieu, M., Gatford, M., and Payne, A. (1996). Okapi at TREC-4. In *The Text REtrieval Conference*, pages 73–96. Gaithersburg, MD: NIST.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv:1706.03762v5*.

Williams, A., Nangia, N., and Bowman, S. (2018). A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1112–1122. Association for Computational Linguistics.

Wu, Z., Mao, J., Liu, Y., Zhan, J., Zheng, Y., Zhang, M., and Ma, S. (2020). Leveraging passage-level cumulative gain for document ranking. In *Proceedings of The Web Conference*, pages 2421–2431. Association for Computing Machinery.

Wu, Z., Mao, J., Liu, Y., Zhang, M., and Ma, S. (2019). Investigating passage-level relevance and its role in document-level relevance judgment. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 605–614. Association for Computing Machinery.

Yang, W., Zhang, H., and Lin, J. (2019). Simple applications of BERT for ad hoc document retrieval. *arXiv:1903.10972*.

Yilmaz, Z. A., Wang, S., Yang, W., Zhang, H., and Lin, J. (2019). Applying BERT to document retrieval with birch. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 19–24. Association for Computational Linguistics.

Zheng, Z., Hui, K., He, B., Han, X., Sun, L., and Yates, A. (2020). BERT-QE: Contextualized query expansion for document re-ranking. In *Findings of the Association for Computational Linguistics*, pages 4718–4728. Association for Computational Linguistics.