Universiteit Leiden

Master Computer Science

Investigating the Explainability Potentials of the Deep Relevance Matching Model

Name:	Ioannis Chios
Student ID:	s2149133
Date:	07/07/2020
Specialisation:	Advanced Data Analytics
1st supervisor:	Dr. Suzan Verberne
2nd supervisor:	Prof.dr. Stephan Raaijmakers

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

In this thesis we address the explainability of web search engines and in particular of neural ranking models for ad-hoc retrieval. We propose two explainable elements on the search engine result page: a visualization of query term importance and a visualization of passage relevance. The idea is that search engines that indicate to the user why results are retrieved improve user satisfaction and gain user trust. We deduce the query term weights from the term gating network in the Deep Relevance Matching Model (DRMM) and visualize them as a doughnut chart. In addition, we train a passage-level version of DRMM that selects the most relevant passage from each document and shows it as snippet on the result page. Next to the snippet we show a document thumbnail with this passage highlighted. We evaluate the proposed interface in an online user study, asking participants to judge the explainability and assessability of the interface. We find that users judge our proposed interface significantly more explainable and easier to assess than a regular search engine result page. However, they are not significantly better in selecting the relevant documents from the top-5. This indicates that the explainability of the search engine result page leads to a better user experience. Thus, we conclude that the proposed explainable elements are promising as visualization for search engine users.

Acknowledgements

First and foremost, I would like to thank my supervisor Dr. Suzan Verberne for her trust and her useful guidance during this project. I would also like to express my gratitude to all the participants of the user study that is presented in this work. I would also like to thank all my friends in Leiden for helping and supporting me throughout the past two years. A special thanks to Fili for her support, motivation and patience. Last but not least, I would like to thank my parents for always being there for me and supporting every single decision of mine.

Contents

Chapter 1

Introduction

Search engines are extremely commonly used for real-world applications like digital libraries and Web search. Today machine learning is utilized to make search engines more powerful. Retrieval models make use of features like click logs and relevance judgments to improve their performance as well as the satisfaction of the users. The more powerful and sophisticated these search engines become, the harder it is for users to understand why the retrieved documents or Web pages are relevant to their search query. This has motivated a line of research to make search engines and recommendation systems explainable to their users [1]. The idea is that if the system indicates to the users why results are retrieved or recommended it gains more trust from the users

There has been more attention to explainable recommendation than to explainable search, with a focus on explaining the user profiling aspect of recommendation systems [2]. Providing explanations which clarify why an item is recommended improves the transparency, persuasiveness, effectiveness, trustworthiness and user satisfaction of the recommendation systems [3]. It also helps system designers to diagnose and improve the algorithm. As a first step towards explainable search engines, we address the explainability of non-personalized web search engines. The explanation of personalization in search engines is a very interesting topic for future work. In this work, we focus on the explanation of query–document relevance on the search engine result page. By helping users understand what the search engine does, we help them become more efficient and effective searchers [4].

Some search engines include explainable features like the contribution of terms to ranking formulas[1] or spelling correction[2]. The most common type of explanation of relevance on the result page is the search snippet, giving a preview of each retrieved document and thereby an indication of its relevance. Query keywords are typically marked in boldface in the snippet to show the user the query relevance of the document. The interpretability of search result snippets has been investigated by Mi and Jiang [5]. They found that snippets play an important role in explaining why documents are retrieved and how useful those documents are.

In this thesis we propose a visualization of the explainable aspects of a neural ranking model on the search engine result page. We use the Deep Relevance Matching Model (DRMM) [6] which is a neural ranking model specifically designed for the ad-hoc retrieval task. Ad-hoc retrieval is the information retrieval task behind search engines. Specifically, given a query, the system's objective is to retrieve the most relevant documents from a corpus. By adding explainable elements to the DRMM we attempt to design a search engine that is more interpretable to the users.

Mi and Jiang [5], in their work, evaluate the search result summaries of an existing web search engine[3]. They conduct an extensive user study to investigate how much users are informed by regular search result summaries. On our part, we evaluate our own explainable search engine interface with a user study. For the evaluation of our explainable interface we follow-up on the work of Mi and Jiang by further investigating the explainability and assessability potentials of search engine result pages. The explainability refers to the ability of the user to interpret why the system

[1] `https://www.elastic.co/guide/en/elasticsearch/reference/current/search-explain.html`
[2] `https://swiftype.com/search-concepts/spelling-correction`
[3] `https://www.bing.com/`

returned the specific results. The assessability refers to the ability of the user to understand which of the results are useful based solely on the search engine result page.

The contributions of this thesis can be summarized as follows. We make the importance of each query term explicit using features of the model's architecture and training process. Additionally, we split the document in smaller passages to investigate the different matching scores between the query and individual passages. We use the passage with the highest matching score as the search result snippet of the document where it derives from. We propose an explainable user interface that shows the query term importance as a doughnut chart, and the passage relevance as a document thumbnail with passage highlights. We evaluate our explainable interface in a small-scale user study in which participants judge the search engine result page on explainability and assessability.

1.1 Research Questions

The main goal of this thesis is to investigate and evaluate the explainability potentials of neural ranking models and in particular those of the Deep Relevance Matching Model (DRMM). We attack the explainability task on two levels: on the query level and on the document level. Furthermore, we design and implement an explainable search engine result page interface and conduct a user study to evaluate it.

The research questions that we address in this thesis are the following:

1. What is the ranking effectiveness of DRMM when selecting the most relevant passage of each document?
2. How do users judge the explainability and assessability of our explainable search engine result page compared to a regular result page?
3. How well can users select the relevant documents based on only the snippets on the result page, in the explainable interface compared to the regular interface?

1.2 Thesis Organization

Chapter 2 introduces related work on ad-hoc retrieval, neural information retrieval and explainable search. Chapter 3 introduces the toolkits and the methods that we used for the ad-hoc retrieval. We will also describe the explainable features that we developed to enhance the search engine result page. In Chapter 4 we describe the experimental setup and the results of the retrieval experiments. In Chapter 5 we describe the design and the results of the user study for evaluating our explainable interface. We conclude this thesis in Chapter 6 with answers to the research questions and suggestions for future work.

Chapter 2

Background

2.1 Ad-hoc retrieval

Ad-hoc retrieval is a classic information retrieval task where the user gives a query as input and the system returns a ranked list of documents that are relevant to the query. The retrieved documents that are ranked higher in the list are more likely to be relevant to the query. By the term relevant we characterize documents that contain sufficient information to answer the specific query, whereas all other documents are called non-relevant. A key characteristic of the ad-hoc retrieval task is that the document collection remains relatively static, while new queries are given to the system periodically. Additionally, the retrieval and hence the ranking of the documents is based solely on the documents' text and not on additional information like user feedback or outgoing links, which are commonly used by web search engines [7, 8].

Given a large document collection, usually containing millions of documents, and a user query, which is usually very short, we use a ranking function to retrieve the most relevant documents. The ranking function calculates a relevance score for each query document pair and ranks the documents in descending order based on that score. The top K documents are then presented to the user as an output to the desired query. One of the most widely used ranking functions in ad-hoc retrieval is Okapi BM25 [9]. It is based on the probabilistic relevance model as well as the Term Frequency (TF) and the Inverse Document Frequency (IDF) to rank the retrieved documents. In more detail, given a query q with n terms $q_1, q_2, ..., q_n$ and a document d BM25 calculates a relevance score:

$$score(q,d) = \sum_{n}^{i=1} IDF(q_i) \cdot \frac{TF(q_i,d) \cdot (k_1+1)}{TF(q_i,d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgDocLen})} \tag{2.1}$$

where $TF(q_i, d)$ is the Term Frequency of the ith query term in document d and $IDF(q_i)$ its Inverse Document Frequency, $|d|$ is the length of document d in tokens and $avgDocLen$ is the average document length in the corpus. Finally, k_1 and b are two parameters that represent the term frequency saturation and the field length normalization respectively. These two parameters are tunable but typically they are set as $k_1 \in [1.2, 2.0]$ and $b = 0.75$.

$IDF(q_i)$ is usually computed as:

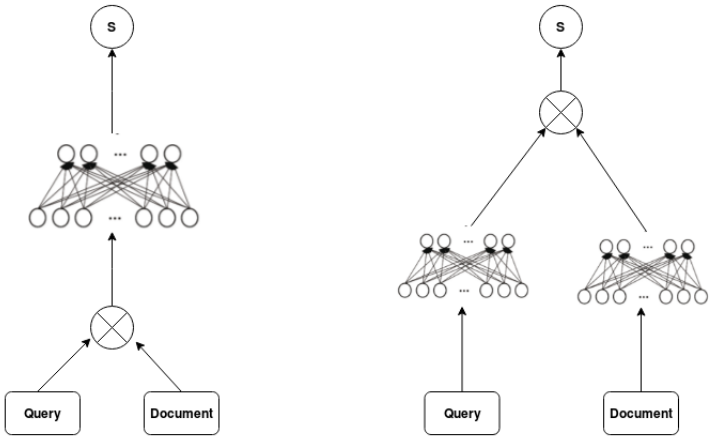
$$IDF(q_i) = log\frac{N}{df(q_i)} \tag{2.2}$$

where N is the total number of documents in the corpus, $df(q_i)$ is the document frequency of q_i. Despite being introduced several years ago, BM25 is still highly effective and is considered as a very strong information retrieval baseline for large document collections.

Over the last years, various information retrieval toolkits have been built by the scientific community to facilitate the research of existing and the development of novel information retrieval methods. Anserini [10] is an open source information retrieval toolkit built on top of Lucene[1].

[1] `https://lucene.apache.org/`

Figure 2.1: The two types of architectures for neural ranking models for information retrieval: Interaction-focused models (left) and Representation-focused models (right)



The intuition behind its development is Lucene's lack of systematic support for evaluation over standard test collections. The advantage of Anserini over other research oriented IR toolkits such as Indri[2] and Galago[3] is its efficiency in indexing and retrieval of large document collections. BM25 is included in the Anserini framework along with other traditional information retrieval algorithms such as QL [11] and RM3 [12].

2.2 Neural information retrieval

Since the 2000s machine learning models have been applied to information retrieval. Learning to rank (LTR) [13] models started outperforming traditional probabilistic methods for ad-hoc retrieval tasks mostly based on hand-crafted features. With the rise of deep learning, neural networks emerged in the information retrieval field as well since they have a lot more potential for learning than traditional models. In this thesis we focus on neural ranking models for ad-hoc retrieval, which is a central task in information retrieval, but not the only one where neural ranking models can be used.

Guo et al. [6] divided existing neural ranking models for information retrieval in two categories, namely representation-focused and interaction-focused. Representation-focused models build separate complex representations for the query and the document and then calculate a relevance score using some simple evaluation function. Different deep neural networks have been applied as representation functions, including convolutional and recurrent networks [14]. This type of architecture is more suitable for tasks concerned with semantic matching, such as automatic conversation and paraphrase identification.

Interaction-focused models on the other hand build joint representations of query-document pairs. This approach is based on the underlying assumption that relevance relies on the relation between the two inputs. The representation functions of interaction-focused models reflect the similarity or distance over each pair of input word vectors. In contrast to the representation-focused architectures their representation functions are usually very simple, such as cosine similarity function and dot-product function, while their evaluation functions are complex (i.e., deep neural networks). This architecture fits tasks with heterogeneous inputs concerned with relevance matching, like ad-hoc retrieval and question answering.

In this thesis we use one of the most used neural ranking models, the Deep Relevance Matching Model [6], which is an interaction-focused neural ranking model specifically designed for ad-hoc retrieval. We choose this model because it integrates interpretable components. This components

[2] `http://www.lemurproject.org/indri/`
[3] `http://www.lemurproject.org/galago.php`

can be deployed to attack the explainability task. In more detail, DRMM builds matching signals between query terms and documents using word embeddings and gives them as input to a feed forward matching network. Moreover, it employs a term gating network to capture the importance of each query term and output a matching score for each query-document pair. The later component is of high importance for our task since the query term importance will be visualized to improve the interpretability of the retrieval results.

2.3 Explainable Search

Recent work on explainable search has been focused on three directions: explainable product search [15], interpretability of neural retrieval models [16, 17] and their axiomatic analysis [18]. Explainable product search is closely related to explainable recommendation since both aim at enhancing the user experience in online shopping and hence in increasing the profits of e-commerce companies. Axiomatic analyses of neural retrieval models investigate to what extent the formal constraints [19] of retrieval models are satisfied from neural ranking models.

On the other hand, research on the interpretability of neural retrieval models attempts to give more insights to the user regarding the search process. Although current search engines have made steps towards the transparency of their results, many users still have little understanding on how they work. Singh and Anand [16] developed an explainable search engine designed to aid the users in the search task. They used a modified version of LIME [20], adapted for rank learning, to interpret the results of neural ranking algorithms. Moreover, they designed an interface for their search engine in which they visualized the explanations.

The work by Singh and Anand does not involve user feedback for evaluating how helpful the search engine is according to the users. Mi and Jiang [5] conducted an extensive user study to address explainability of result pages. They evaluated the search result summaries of a commercial search engine in which the document snippets are the explainable elements on the result page. In the current thesis, we build on this work by implementing an explainable neural search engine and evaluating its retrieval effectiveness as well as its explainable elements with user feedback.

Chapter 3

Methods

3.1 Ad-hoc retrieval architecture

A crucial part in the ad-hoc retrieval task, when neural ranking methods are used, is the combination of a retrieval step with a re-ranking step. The purpose of this combination is the reduction of the computational demands of the neural ranking models. In the first step, K candidate documents for each query are selected using a traditional IR baseline. Then the retrieved documents are reranked using the neural model in order to achieve better performance.

Specifically, for the integration between the initial ranking and the neural re-ranking we implement a rather simple technique. We use and process the query-document relevance file that is very common in hand-labeled ad-hoc retrieval datasets. In more detail, after retrieving the highest ranked (*top K*) documents for every query using BM25, we reconstruct the relevance file using only the query-document pairs that were retrieved. The neural ranking is then performed on the filtered query-document relevance file.

Although this technique may ignore some relevant documents from the re-ranking process, it actually does not affect the results of the neural models while their training time is considerably reduced. It is an information retrieval pipeline widely adopted by many previous works in neural re-ranking for ad-hoc retrieval [6, 21]. The pipeline of the ranking and re-ranking process is shown in Figure 3.1.

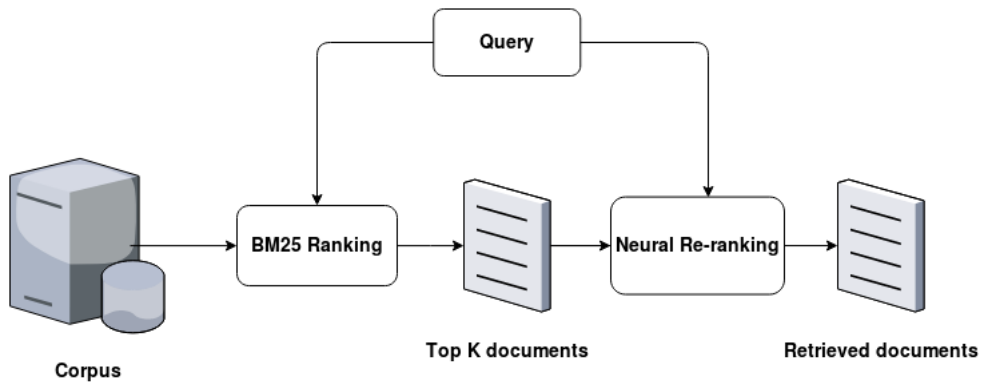


Figure 3.1: Architecture of the ad-hoc retrieval with neural re-ranking pipeline

3.2 Ranking Documents

In our implementation we use Anserini[1] to retrieve the top K documents for each query as described in Section 3.1. The starting point is to build an inverted index for our document collection. After that we use the index to retrieve the top K relevant documents for each query with BM25 [9]. Anserini includes an end-to-end information retrieval pipeline that supports Lucene's multi-threaded indexing and provides preprocessing for standard document formats used in IR – TREC-style XML files, web pages in WARC format, etc. Additionaly, it has implementations for various information retrieval algorithms. We interact with Anserini, which is written in Java, using its command line interface. Hence, it offers an out of the box solution that can build an inverted index and perform document ranking for a large document collection in a few minutes using its own interface and limited computational resources.

3.3 Re-Ranking Documents

The second step of the ad-hoc retrieval pipeline is the re-ranking process. We perform this step using MatchZoo [22], a text-matching toolkit that facilitates the design, the comparison and the sharing of deep text matching models. We use Matchzoo v2.2[2] which is written in Python and offers implementations for data processing, neural matching models as well as their training and evaluation. Its data processing module contains standard text preprocessing methods such as word tokenization, stemming etc. using NLTK[3] and various dataframe operation using Pandas[4]. All neural network models are implemented using Keras[5] and Tensorflow[6].

We use MatchZoo's implementation for DRMM to perform the neural re-ranking step. The Deep Relevance Matching Model [6] is an interaction-focused neural ranking model specifically designed for ad-hoc retrieval. It builds matching signals between query terms and documents using word embeddings and gives them as input to a feed forward matching network that outputs a matching score for each query-document pair.

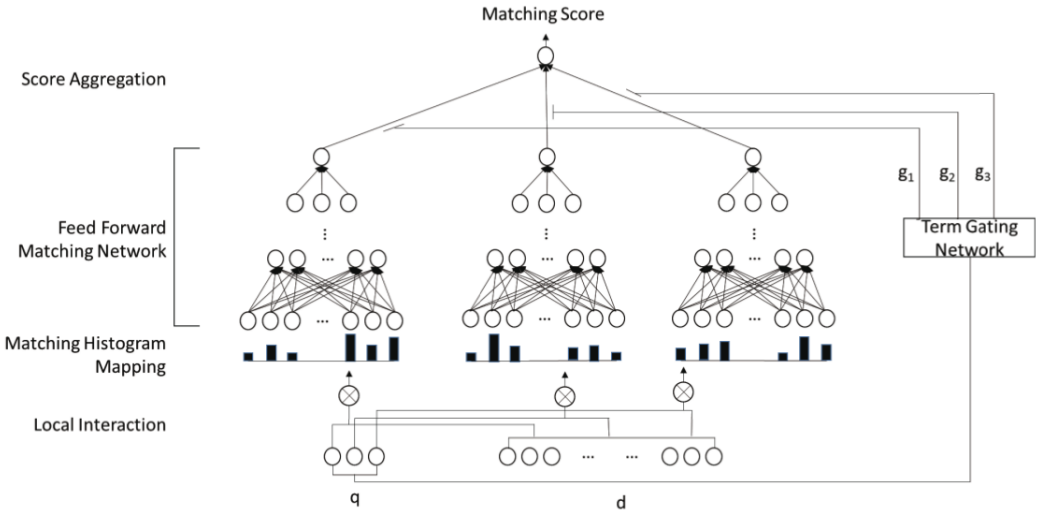


Figure 3.2: Architecture of the Deep Relevance Matching Model [6]

[1] http://anserini.io
[2] https://github.com/NTMC-Community/MatchZoo
[3] https://www.nltk.org/
[4] https://pandas.pydata.org/
[5] https://keras.io/
[6] https://www.tensorflow.org/

3.3.1 DRMM Model Architecture

In more detail, DRMM takes as input the term vectors of a query $q = \{w_1^{(q)}, w_2^{(q)}, \ldots, w_M^{(q)}\}$ and a document $d = \{w_1^{(d)}, w_2^{(d)}, \ldots, w_N^{(d)}\}$ where $w_i^{(q)}$ denotes the word embedding vector of the i-th query term and $w_j^{(d)}$ the word embedding vector of the j-th document term. These term vectors are used to build local interactions between each pair of terms. For each query term the local interactions are transformed into a fixed-length matching histogram. Consequently, the matching histograms are the input of a feed forward neural network which learns hierarchical matching patterns and produces a matching score for each query term. To generate the overall matching score of a query the scores for each term are aggregated with a term gating network. The matching score for each query-document pair is the dot product of the query term importance weights and the output vector of the feed-forward network.

Matching Histograms

The first step in capturing local interactions between query terms and documents in DRMM is to measure the cosine similarity between the word embedding vectors of every query term and every document term. The major drawback of this approach is the variance in the sizes of the local interactions due to the different lengths of queries and documents. In order to overcome this obstacle, DRMM introduced the matching histogram mapping which is a key element in its implementation. The basic idea behind matching histograms is the grouping of local interactions according to different levels of signal strength. Since cosine similarity between the term vectors is within the interval $[-1, 1]$, this interval is discretized into equal sized bins in an ascending order and each similarity score is assigned to the corresponding bin. The exact matching between query and document term (*cosine similarity=1*) is treated as a separate bin. There are three types of matching histograms:

- **Count-based Histogram:** Each bin contains the count of local interactions as the histogram value.
- **Normalized Histogram:** The count value in each bin is normalized by the total count of terms in the document.
- **LogCount-based Histogram:** The count value in each bin is logarithmized to reduce the range of histogram values. In our research, we only experimented with this type of matching histograms since previous work [6] shows they outperform the other ways of matching histogram mapping. An example of LogCount based histograms is shown in Figure 3.3.

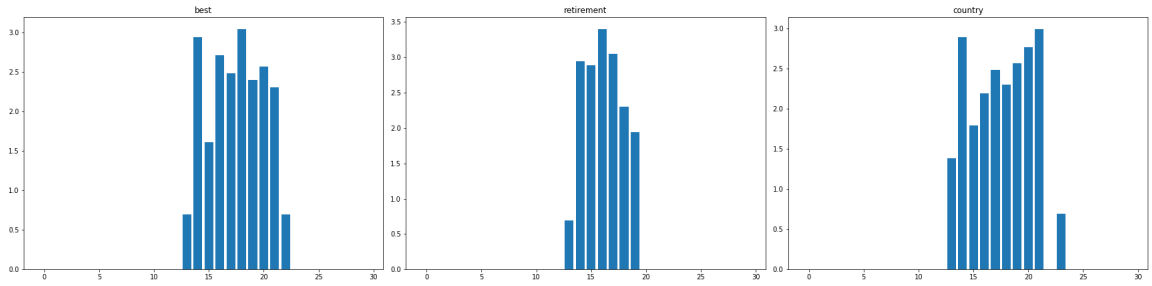
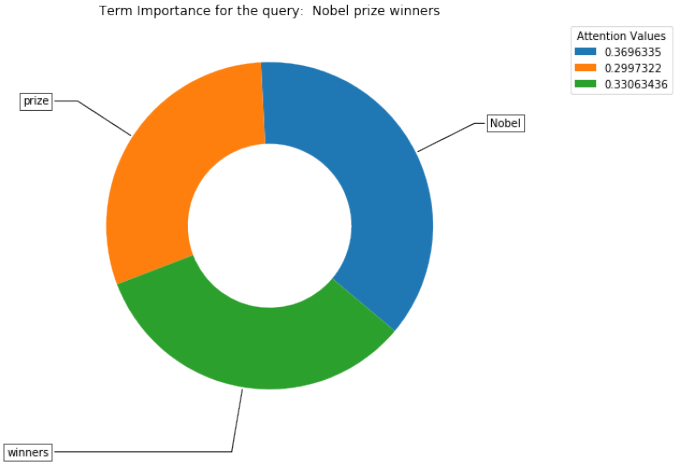


Figure 3.3: The LogCount-based matching histograms for the query "Best retirement country" and a relevant document from Robust04. Each histogram shows the interaction between a term from the query and the specific document. Here we discretize the cosine similarity interval [-1,1] in 30 bins.

Feed forward Matching Network

Using the matching histograms as input, DRMM employs a feed forward matching network to learn the hierarchical matching patterns between documents and query terms. The choice of a feed forward network in the architecture of the model ignores the position of the terms and focuses on the strength of the signals. The output of the network is a matching score for each query term.

Figure 3.4: A doughnut chart that depicts the importance of each term in the query "Nobel prize winners" based on their respective attention values from DRMM's term gating network



Term Gating Network

Another very important feature in the architecture of DRMM is the deployment of the term gating network. The purpose of this addition is the modeling of query term importance. Instead of just summing up the respective matching scores of each query term, the term gating network controls the contribution of each matching score to the final relevance score. To calculate this query term importance it produces an aggregation weight for each term using linear self-attention. Specifically, for each query q_i, its importance weight g_i is returned by a softmax function.

$$g_i = \frac{exp(w_g x_i^{(q)})}{\sum_{j=1}^{M} exp(w_g x_j^{(q)})} \ , \quad i = 1, ..., M \tag{3.1}$$

where $x_i^{(q)}$ is the i-th query term input. This input can be either the embedding vector of the corresponding term or its inverted document frequency. Consequently, w_g which denotes the weight parameter of the gating network is either a weight vector with the same dimensionality as the embedding vector or a scalar weight when the IDF value is chosen as the input.

3.3.2 DRMM Model Training

To train the deep relevance matching model we employ a pairwise ranking loss function, which is widely used in neural ranking and in ad-hoc information retrieval in general, called hinge loss. Like in all pairwise loss functions, we compute hinge loss by taking the permutations of all document pairs. Given two matching scores $s(q, d^+)$ and $s(q, d^-)$ where d^+ is ranked higher than d^- with respect to query q, hinge loss is defined as:

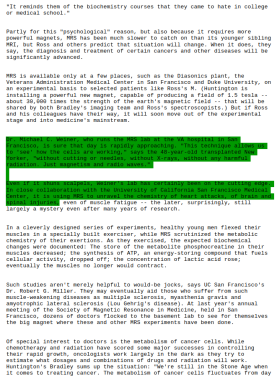
$$L(q, d^+, d^-; \Theta) = max(0, 1 - s(q, d^+) + s(q, d^-)) \tag{3.2}$$

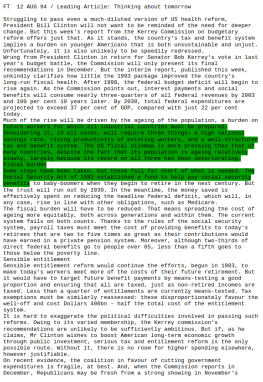
with Θ as the weights used both in the feed forward matching network and the term gating network. The model parameters are updated via standard backprogation using the Adadelta [23] optimizer with mini-batches.

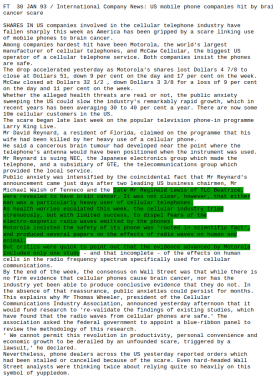
3.4 Explainability

In this thesis we attack the explainability task on two levels: on the query level and on the document level. Based on the model's architecture and training process we try to explain the importance of each query term in a query. Additionally, we split the document in smaller passages to investigate the different matching scores between queries and document passages.

Figure 3.5: Example thumbnails of retrieved documents for the query "Radio waves and brain cancer" with the most relevance passage highlighted.







3.4.1 Query Term Importance

Queries are usually based on keywords and do not contain complex grammatical structures. Moreover, most ad-hoc retrieval algorithms are based on the assumption that all query terms should be of equal importance. Contrary to this hypothesis DRMM handles queries as separate terms but, takes into account the importance of each term by employing the term gating network *(see Subsection 3.3.1)*.

On our part, we consider query term importance as a very important aspect of search results interpretation. It is an actual real-time feedback to the user regarding the query he/she chose in order to satisfy his/her information needs. These term weights are able to teach the users how to search for what they are looking for, given a certain information retrieval system.

To obtain these weights we use the term gating network's softmax function (see Equation 3.1). This gating network can be alternatively described as the simplest form of attention function [24], a linear self-attention. Subsequently, the outputs g_i of the gating network can be described as the outputs of an attention layer, also known as attention values. Since they are the output of a softmax function these probabilities sum to 1, which makes their visualization trivial.

A simple example would be the term importance for the query "Nobel prize winners" that is shown in Figure 3.4. Given this query a relevant document is expected to be about winners of the Nobel prize. Intuitively the term "Nobel" is more important than the other two terms, in the sense that retrieved documents that do not refer to it are probably irrelevant and that documents describing other aspects of "Nobel" would be more relevant than documents about another "prize" for instance. As expected, the term "Nobel" has the greatest attention probability among the other two terms while the term "winners", which is also very important, comes second.

3.4.2 Passage Ranking

For the explainability of the document text matching we decided to adopt a passage-level approach. We split each document in passages to be able to deduce the matching scores between the query and each passage, thereby allowing the selection of the best matching passage. Passages are non-overlapping and have a length of 100 tokens each. Many documents in Robust04 are shorter than 200 tokens and could not be split into longer passages. Moreover smaller passages can be used as snippets for the search engine interface.

Following DRMM's pipeline we build a matching histogram for each query–passage term pair and then give it as input to the model. Since passage-level labels are not available, in order to train the neural ranking model we have to assign a ground-truth relevance label to each passage-query pair. To that end, we transfer the document relevance label to each passage in the document as

ground truth for learning the passage relevance.[7]

After we have trained the model and assigned a matching score to each passage using DRMM, we use several methods to evaluate our model on the document level. Following Dai et. al [26] we take the score of the passage with the highest matching score ('maxP'), the sum of the matching scores ('sumP') or the score of the first passage ('firstP') and use it to rank the documents. Furthermore, we experiment with the mean of the matching scores ('meanP') of each document as well as their median ('medianP'). The results of our experiments are presented in Section 4.3.

Visualization of passage relevance. The passage with the highest matching score from each query–document pair is considered to be the most relevant passage of the document and is shown to the user as document snippet in the search results interface. We also employ another type of visualization to enhance the explainability of the results. We show a thumbnail of the whole document to the user in which the most relevant passage is highlighted, as illustrated in Figure 3.5. This allows the user to better judge the relevance of the document based on the position of the passage in the text and also gives information of where to search in the document for the most relevant passage. This can be especially helpful in long documents where the information needed by the user could be at the bottom of the document.

[7] This is a simple approach that might lead to some passages being mislabeled [25], but for our current goal this suffices.

Chapter 4

Retrieval Evaluation

4.1 Data

For our experiments we used the Robust04 dataset [27] which is widely used to evaluate methods for ad-hoc information retrieval. The Robust04 corpus was used in the TREC Robust track in 2004 and contains 528,155 documents from the Financial Times, the Federal Register 94, the LA Times, and FBIS. Additionally the dataset has 250 topics which contain both a short title and a description. We only used the topic titles as queries in our experiments which are short (a few keywords) and therefore fit our the query term importance approach. The number of relevant documents per query varies a lot in Robust04. The most relevant documents for a single query are 448 while there is one query that does not have any relevant documents and was subsequently excluded from the experiments. The median number of relevant documents per query is 41.

In Subsection 3.4.2 we describe the methods we used to split the Robust04 on small passages. We split each document in passages with a length of 100 tokens. Since the documents in the corpus vary a lot in length, the number of passages from each document varies as well. However most of the documents are short with the 28.3% having only one passage after the split. The average document length is 4.5 passages and the longest document contains 2,284 passages. In Figure 4.1 we can see the frequency of the count of passages in the document collection.

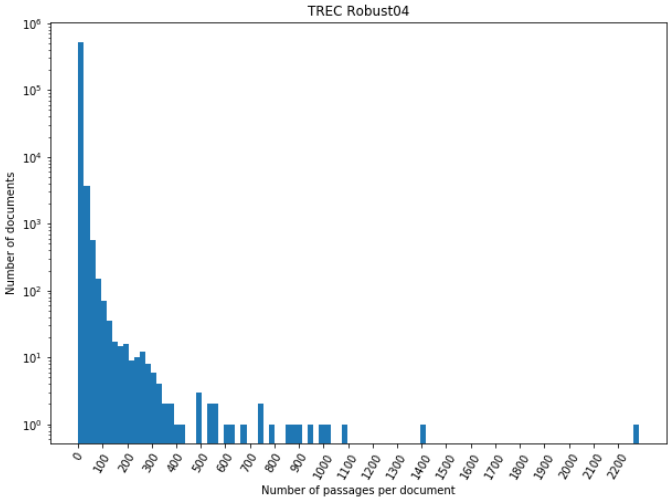
4.2 Experimental Setup

Initial ranking. In the initial retrieval step, we use the Anserini toolkit and more specifically its implementation of BM25, as we describe in Section 3.2. We choose $K = 1000$ documents, following Yang [28], when retrieving complete documents and $K = 100$ when splitting the documents into passages. We chose to reduce the number of initially retrieved documents in the passage approach because of computational limitations. The caveat of this approach is a drop in the performance of the initial retrieval model.

Neural re-ranking. For the re-ranking step we use MatchZoo's implementation of the Deep Relevance Matching Model (see Section 3.3). A drawback of this implementation is that it truncates the first N words and takes them as input. In our experiments this had an impact when retrieving complete documents where we used $N = 500$, following Yang's work [28]. Furthermore, we split the dataset in 5 folds with 50 queries each, following Huston and Croft [29]. We then conduct 5-fold cross-validation where we use 3 folds for training, 1 for validation and 1 for testing [6, 30]. For the DRMM implementation we use LogCount-based histograms with the bin size set to 30, following Guo et al. [6]. In the feed-forward matching networks we used two hidden layers with 5 nodes each and *tanh* as activation function. This specific architecture was chosen based on development evaluation. For the embedding layer we used Glove embeddings [31] pretrained on 6B tokens with 300 dimensions.[1] The query term embeddings were used as the input of the term gating network.

[1] `https://nlp.stanford.edu/projects/glove/`

Figure 4.1: Frequency of passage count per document on the Robust04 corpus.



Implementation Details. All the aforementioned experiments were performed using the servers of the Data Science Lab (DSLAB) of the Leiden Institute of Advanced Computer Science (LIACS). For the training of the Deep Relevance Matching Model, which was the most computationally expensive, `duranium` server was used. `Duranium` is a GPU server with six NVIDIA GeForce GTX 980 Ti and two NVIDIA GeForce GTX TITAN X Graphic Processing Units (GPU). Additionally, the server has two Intel Xeon Processor E5-2650 v3 @ 2.30GHz with 20 cores (40 threads in total) and 128GB of RAM. The preprocessing and the passage splitting of the Robust04 collection as well as the initial document ranking were performed on `adamant` server. `Adamant` is a CPU server with two Intel Xeon CPU E5-2630 v3 @ 2.40GHz with 16 cores (32 threads in total) and 512Gb of RAM. Both servers run on CentOS Linux 7 (Core) OS.

4.3 Results

Table 4.1 shows the results of the retrieval experiments on the test set. The first row in Table 4.1 are the results of Anserini's BM25 on the document level with $K = 1000$ and acts as a strong baseline. The second row are the results of our DRMM implementation on the document level.[2]

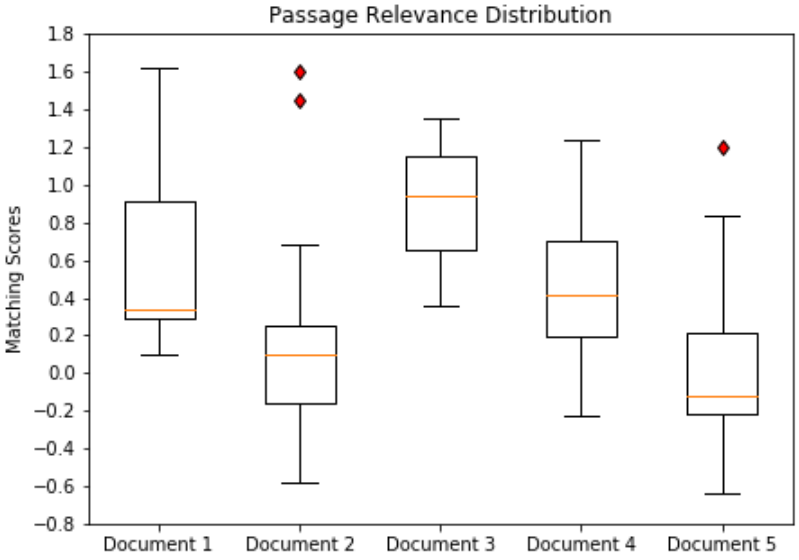
	MAP	P@20	nDCG@20
BM25	0.2531	**0.3631**	**0.4240**
DRMM	0.2662	0.2974	0.3706
DRMM-maxP	**0.3172**	0.2650	0.3177
DRMM-sumP	0.2800	0.2309	0.2690
DRMM-firstP	0.3003	0.2700	0.3025
DRMM-meanP	0.3043	0.2760	0.3069
DRMM-medianP	0.2960	0.2720	0.2947

Table 4.1: Retrieval results on the Robust04 test data

The next rows show the results of our own methods for passage retrieval with evaluation on the document level, as explained in Subsection 3.4.2. The results indicate that in terms of Precision@20 and nDCG@20 our models are less effective than BM25 and document-level DRMM, but they are a bit better in terms of MAP. Even though we optimize for MAP the significant drop in terms of Precision@20 and nDCG@20 of our proposed models shows that they cannot beat the strong

[2]The results in terms of P@20 and nDCG@20 are lower than in the original paper [6], but our implementation is not identical to the original. Other papers have reported different results as well [30]

Figure 4.2: Passage matching score dispersion for the top 5 most relevant documents for the query: "Radio waves and brain cancer".



baselines of BM25 and the standard DRMM. In more detail, the DRMM-maxP model performs the best in terms of MAP and nDCG@20 among our proposed models, while DRMM-meanP performs better in terms of P@20. For the evaluation of the explainable features that we developed we use the DRMM-maxP model whose performance is considered sufficient.

In Figure 4.2 we can see the distribution of matching scores across passages of the same document. The five documents are the highest ranked documents for the query "radio waves and brain cancer". For the ranking of the documents the DRMM-maxP model was used. This means that the documents are ranked based on the highest matching score among their passages. We observe that in most documents there is a significant difference between the highest and the lowest scoring passages of a documents.

The boxplot for Document 2 shows that the lowest ranked passage has a matching score of -0.6 while the highest has a score of 1.6 which is an outlier for the specific document. The fact that highest ranked passages are outliers in many cases could possibly lead to many documents ranking higher than they should. However, other approaches like DRMM-meanP or DRMM-medianP where scores that depict the overall matching of the documents seem to perform worse in terms of most evaluation metrics as shown in Table 4.1. The explanation is that in many relevant documents the information that are related to the query occur in small parts of the document, that can even fit in one passage. Thus, a high matching score of a single passage can lead to a good ranking of the whole document.

Chapter 5

User Evaluation

5.1 Study Design

To evaluate the explainability of our methods we conducted a user study. For the study purposes we created two interfaces for our search engine result pages (SERP), namely one explainable and one regular. The regular interface resembles most web search engines showing the result's title and abstract. The explainable interface builds up to that and includes the query term importance doughnut and the document thumbnails as described in Section 3.4. The two interfaces were compared in an online user study. The study used a 2x2 within-subject design to evaluate the two different interfaces. The participants were split in two groups and each group evaluated different interfaces of the same result pages in alternating order.

Query selection. We hand-selected six queries from the Robust04 dataset[1] in order to show their result pages to the participants. The selected queries are shown in Table 5.1. The search engine result pages contained the query and the top-5 documents that were retrieved based on the query. We selected the queries so as to satisfy three basic criteria:

1. At least one relevant document should occur in the five highest ranked documents of the query.
2. All five retrieved documents should have a title field[2] to be shown in their SERP.
3. The query should consist of at least three terms, to have an informative query term importance doughnut chart.

For each query we retrieved the most relevant documents based on our ranking and neural re-ranking methods as described in chapter 3. Specifically, the DRMM-maxP method was used for the retrieval of the documents that were used in the user study. In this method the relevant documents are ranked based on their highest ranked passage. The highest ranked passages were also used as the snippets of the documents in the SERP.

our query id	Robust04 query id	query	relevant documents in top-5
1	310	radio waves and brain cancer	A
2	318	best retirement country	A,E
3	384	space station moon	A
4	400	amazon rain forest	C,D,E
5	613	berlin wall disposal	B
6	615	timber exports asia	B

Table 5.1: Selected queries for the user study. The relevant documents in the top 5 are indicated with letters, where A denotes the first-ranked document and E denotes the fifth-ranked document

[1] Only the titles of the topics were used as queries

[2] Documents from Federal Register 94 do not contain a title field and are difficult to be visualized in a SERP.

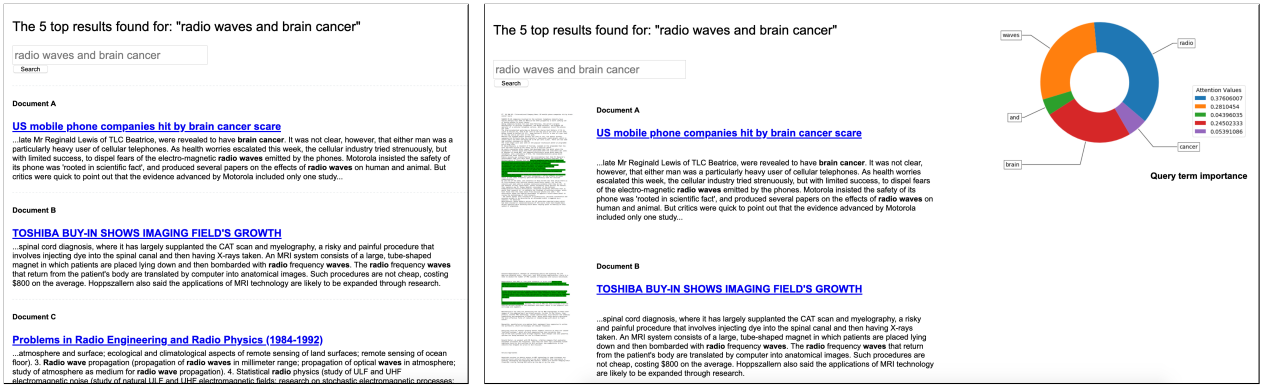


Figure 5.1: Left: Interface R (regular). Right: Interface E (explainable)

User interfaces. We created two static interfaces for our search engine result pages to compare them. Two custom templates were designed, one for each interface, in order to look as close to most search engine result pages to that day as possible. The web pages were hosted on the web server of LIACS (Leiden Institute of Advanced Computer Science) in order to be accessible to all participants remotely. Both interfaces are shown in 5.1:

- Interface R *(regular)* is a classic search engine result page, showing the title and a snippet for each retrieved document. The snippets shown are the most relevant passages from the document and the query terms in the snippet are marked with boldface.
- Interface E *(explainable)* is the experimental interface that we built by adding explainable characteristics to the classic search engine result page. The title and the most relevant passage as a snippet are shown together with the query terms marked with boldface. Additionally, a thumbnail on the left of each document indicates the position of the most relevant passage in the document. On the upper right of the page the doughnut chart shows the importance weight (attention value) of each query term.

Participants. The user study was performed online and 22 volunteers participated. The participants were randomly split into two groups (Group A and B). The two groups judged the same set of queries but based on different interfaces. In more detail, participants in group A saw queries 1,3 and 5 in interface R and queries 2,4 and 6 in interface E. Participants in group B saw queries 1,3 and 5 in interface E and queries 2,4 and 6 in interface R. The queries were shown to the groups in different order but in a way that the interfaces were alternating, starting from the interface R.

5.2 Result Page Judgments

We asked from the participants to read the result pages and answer to a few questions regarding the results. To create our evaluation form we were based on the paper by Mi and Jiang [5], but as opposed to their work the users had to evaluate the result pages as a whole instead of the individual snippets. The evaluation criteria of our user study were explainability and assessability. The two following questions were used to evaluate the aforementioned criteria. Participants were asked to respond to them using a five point Likert-type scale from 1 (Strongly Disagree) to 5 (Strongly Agree).

- ***(Explainability)*** "By looking at the result page, I can understand why the search engine returned these results for the shown query."
- ***(Assessability)*** "By looking at the result page, I can tell which results are useful without opening the links."

Additionally, the participants were asked to respond to one more question: "By looking at the result page, can you tell which of the returned documents are relevant to the query?" (checkboxes for documents A-E, or 'None of the above'). The users were not able to read the full documents, they had to answer the question by solely looking at the search engine result page.

5.3 Results

We examine the explainability and assessability scores of the two interfaces over all queries and all participants. For each interface we have 66 scores for explainability and 66 for assessability. The mean explainability score for the regular interface was 3.4 and for the explainable interface was 4.2. The mean assessability score for the regular interface is 3.6 while for the explainable interface was 4.4. We observe that the explainable interface scored higher than the regular in terms of both evaluation criteria. The dispersion of assessability and explainability scores for the two interfaces is shown in Figure 5.2.

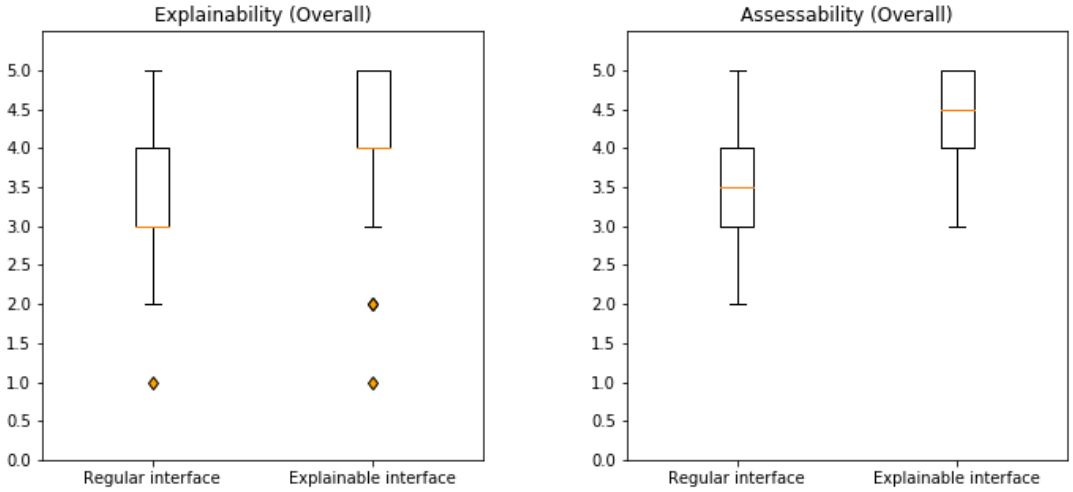


Figure 5.2: Dispersion of assessability (left) and explainability (right) scores for both interfaces.

5.3.1 Significance testing

Moreover, we used three-way ANOVA to test the significance of the differences of the explainability and assessability scores between the two interfaces. The three factors (independent variables) of the analysis are "interface","participant" and "query" and the dependent variables are explainability and assessability (we performed a three-way ANOVA for each of them separately). In Table 5.2 and Table 5.3 we can see the summaries for the three-way anova analyses for explainability and assessability respectively. It is clear from the analyses' results ($F(1,1) = 30.7$, $p < 0.001$ for explainability ; $F(1,1) = 39.6$, $p < 0.001$ for assessability) that the differences between the two interfaces are significant for both evaluation metrics. Moreover, the very large F-Values of the interface variable show that there is a large variation between the two interfaces and hence we can conclude that the interface is the most significant factor for both assessability and explainability.

	F-Value	P-Value
interface	30.7	0.000
participant	3.38	0.000
query	3.29	0.008

Table 5.2: Three-way anova summary for explainability (dependent variable)

	F-Value	P-Value
interface	39.6	0.000
participant	3.09	0.000
query	0.99	0.424

Table 5.3: Three-way anova summary for assessability (dependent variable)

Per-participant analysis. Additionaly, the three-way anova analyses show that the there are significant differences between the answers of participants as well ($F(1,21) = 3.38$, $p < 0.001$ for explainability ; $F(1,21) = 3.09$, $p < 0.001$ for assessability). To further analyse the significance of the difference between the participants we use the Wilcoxon signed-rank test. The scores of the test (dependent samples with $n = 22$) indicate that the two interfaces are significantly different both in terms of explainability ($W = 22.5$, $p = 0.001$) and assessability ($W = 18.5$, $p = 0.002$). The vast majority of participants (17 and 18 respectively) judge both the explainability and the assessability of the explainable interface higher than those of the regular interface.

Per-query analysis Finally, the anova results for the query factor show that the differences in the explainability scores between the queries are significant ($F(1,5) = 3.29$, $p = 0.008$) while they are not significant for the assessability scores ($F(1,5) = 0.99$, $p = 0.424$). These results are further explained by Figure 5.3 which shows a breakout per query. For each of the six queries, the average explainability and assessability scores are plotted for the two interfaces. The left plot (for explainability) shows quite some variation between the queries. The query represented by the cyan line is the only query for which the explainable interface scores slightly lower (4.1) than the regular interface (4.2). This is query 4, "amazon rain forest". One characteristic of this query is that it has three relevant documents in the top-5, where the other queries had one and in one case two. This could have led to a relatively high explainability score in the regular interface (highest of all five queries in the regular interface). On the other hand, assessability is judged higher for all queries in the explainable interface than in the regular interface. This supports the anova analysis results which show that the query variable does not significantly affect the assessability score and also reflects the preference of the users towards the explainable interface.

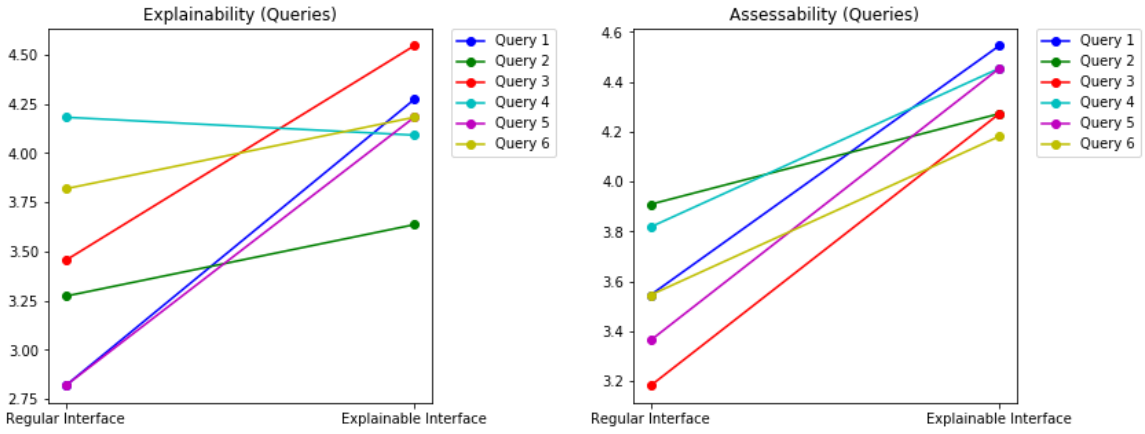


Figure 5.3: Average explainability (left) and assessability (right) scores per query, in both interfaces. Each colored line denotes one query, with on the left side its mean score in the regular interface and on the right side its mean score in the explainable interface.

Relevance assessments. We compared the participants' relevance assessments (based on the result page only) to the ground truth relevance assessments in the Robust04 data, and measured precision and recall per query, participant and interface type. The mean precision and recall scores per interface are in Table 5.4. Although the scores seem higher for the explainable interface than for the regular interface, the deviation is large. Analysis on the participant level with the Wilcoxon Signed-ranks Test (dependent samples with $n = 22$) indicates that the differences between the two interfaces in terms of precision and recall of the relevance assessments are not significant ($p = 0.82$ and $p = 0.50$, respectively).

Table 5.4: Mean precision and recall scores (standard deviation between brackets) that the participants obtain by selecting snippets compared to the ground truth document relevance assessments in Robust04.

	Precision	Recall
Regular Interface	53% (0.42)	62% (0.45)
Explainable Interface	59% (0.40)	75% (0.41)

From another perspective the relevance assessments of the participants can be seen as an evaluation form for the documents' snippets. In order for a user to judge if a document is relevant to the query by solely looking at the result page one has to read each document's snippet. The results in Table 5.4 show that using the most relevant passages of the documents as snippets helped the users to identify the relevant documents in both interfaces.

Chapter 6

Conclusion

6.1 Conclusions

In this thesis, we work on the explainability potentials of neural ranking models for ad-hoc retrieval and propose an explainable interface for the search engine result page. The explainable interface integrated the visualization of the query term importance and the passage relevance into the result page. Furthermore, we evaluate our interface in a small-scale user study. The conclusions to the research questions that we address are the following:

RQ1. *What is the ranking effectiveness of DRMM when selecting the most relevant passage of each document?*

We found that in terms of Precision@20 and nDCG@20 our passage-level ranking model is less effective than BM25 and document-level DRMM, but it is a bit better in terms of MAP. Overall the ranking effectiveness of DRMM when selecting the most relevant passage of each document is lower than the baselines. However, the nDCG@20 score of DRMM-maxP (0.3177) shows that we can use the model for the evaluation of the explainable features developed.

RQ2. *How do users judge the explainability and assessability of our explainable search engine result page compared to a regular result page?*

We found that users judge our proposed interface significantly more explainable and easier to assess than a regular search engine result page. This indicates that the explainability of the search engine result page leads to a better user experience.

RQ3. *How well can users select the relevant documents based on only the snippets on the result page, in the explainable interface compared to the regular interface?*

We cannot prove that the users are better in selecting the relevant documents from the top-5; there is a large deviation in the precision and recall scores that the users obtain.

Overall, we conclude that the proposed explainable elements are promising as visualization for search engine users, based on their subjective experience.

6.2 Future Work

We are interested in investigating the possibility of adding explainable elements to state-of-the-art neural ranking models for information retrieval. Neural ranking models like CEDR-DRMM [21] and POSIT-DRMM [30] are built on top of DRMM's architecture but improve its performance on ad-hoc retrieval tasks significantly. We believe that our methods can be modified accordingly in order to build explainable search engines that achieve performance comparable to the latest neural models.

Furthermore, we would like to address explainable search in two research directions: professional search contexts, and personalized search contexts. We think that explainable search is particularly relevant for professional search contexts, where users are critical towards search results and have the need to be in control [32, 33]. In these contexts, trust is even more important than in generic web search. In the case of personalized search, it is essential that the search engine is sufficiently transparent for two reasons: (1) to gain trust from the user that the personalization does not lead to loss in quality, even though other users get different results; (2) to be able to show the user which personal information is used in the result ranking, and make it explicit when documents are considered relevant because of matches to the users personal preferences.

Bibliography

[1] Yongfeng Zhang, Jiaxin Mao, and Qingyao Ai. SIGIR 2019 tutorial on explainable recommendation and search. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1417–1418, 2019.

[2] Yongfeng Zhang, Yi Zhang, Min Zhang, and Chirag Shah. Ears 2019: The 2nd international workshop on explainable recommendation and search. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1438–1440, 2019.

[3] Yongfeng Zhang and Xu Chen. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*, 2018.

[4] Paul Thomas, Bodo Billerbeck, Nick Craswell, and Ryen W White. Investigating searchers' mental models to inform search explanations. *ACM Transactions on Information Systems (TOIS)*, 38(1):1–25, 2019.

[5] Siyu Mi and Jiepu Jiang. Understanding the interpretability of search result summaries. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 989–992, 2019.

[6] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 55–64, 2016.

[7] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[8] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002.

[9] Stephen E Robertson and Steve Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*, pages 232–241. Springer, 1994.

[10] Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256, 2017.

[11] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *ACM SIGIR Forum*, volume 51, pages 268–276. ACM New York, NY, USA, 2017.

[12] Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. Umass at trec 2004: Novelty and hard. *Computer Science Department Faculty Publication Series*, page 189, 2004.

[13] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[14] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Zamani, Chen Wu, W Bruce Croft, and Xueqi Cheng. A deep look into neural ranking models for information retrieval. *Information Processing & Management*, page 102067, 2019.

[15] Qingyao Ai, Yongfeng Zhang, Keping Bi, and W Bruce Croft. Explainable product search with a dynamic relation embedding model. *ACM Transactions on Information Systems (TOIS)*, 38(1):1–29, 2019.

[16] Jaspreet Singh and Avishek Anand. Exs: Explainable search using local model agnostic interpretability. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 770–773, 2019.

[17] Zeon Trevor Fernando, Jaspreet Singh, and Avishek Anand. A study on the interpretability of neural retrieval models using deepshap. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1005–1008, 2019.

[18] Daniël Rennings, Felipe Moraes, and Claudia Hauff. An axiomatic approach to diagnosing neural ir models. In *European Conference on Information Retrieval*, pages 489–503. Springer, 2019.

[19] Hui Fang, Tao Tao, and ChengXiang Zhai. A formal study of information retrieval heuristics. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56, 2004.

[20] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

[21] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1101–1104, 2019.

[22] Yixing Fan, Liang Pang, JianPeng Hou, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. Matchzoo: A toolkit for deep text matching. *arXiv preprint arXiv:1707.07270*, 2017.

[23] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[25] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. Cross-domain modeling of sentence-level evidence for document retrieval. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3490–3496, Hong Kong, China, November 2019. Association for Computational Linguistics.

[26] Zhuyun Dai and Jamie Callan. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 985–988, 2019.

[27] Ellen M Voorhees. Overview of trec 2004. In *Trec*, 2004.

[28] Wei Yang. End-to-end neural information retrieval. 2019.

[29] Samuel Huston and W Bruce Croft. Parameters learned in the comparison of retrieval models using term dependencies. Technical report, 2014.

[30] Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. Deep relevance ranking using enhanced document-query interactions. *arXiv preprint arXiv:1809.01682*, 2018.

[31] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[32] Suzan Verberne, Jiyin He, Udo Kruschwitz, Gineke Wiggers, Birger Larsen, Tony Russell-Rose, and Arjen P de Vries. First international workshop on professional search. In *ACM SIGIR Forum*, volume 52, pages 153–162. ACM New York, NY, USA, 2019.

[33] Tony Russell-Rose, Jon Chamberlain, and Leif Azzopardi. Information retrieval in the workplace: A comparison of professional search practices. *Information Processing & Management*, 54(6):1042–1057, 2018.