Abstract

Most modern search engines use Learning to Rank methods for search result ranking. These methods function on the basis of features, extracted from the data. In most cases, these features are based on the text content of the web page. However, recently it was proposed, that the exploitation of a documents visual information can benefit Learning To Rank (LTR). These visual features were used in combination with the text features to improve the feedback for the user. One problem in using the text features is that they rely on handcrafted formulas. A more intuitive solution would be to reproduce the features in a visual manner, and use the spatial information extracted from a webpage for learning and prediction. Such a approach would allow to create a model which depends only on visual information, without the use of handcrafted formulas and heuristics.
Contents

1 Introduction 2

2 Related Work 4
   2.1 Standard text analysis features 4
   2.2 New features in text analysis 5
   2.3 Visual features 5

3 Background information 7
   3.1 Convolution 7
   3.2 Activation 8
   3.3 Pooling 8
   3.4 Fully connected 9

4 Method 10
   4.1 Visual representation 10
      4.1.1 Query-independent representation 10
      4.1.2 Query-dependent representation 11
   4.2 Feature Construction 13
   4.3 Model Architecture 14

5 Experimental Setup 18
   5.1 Dataset 18
   5.2 Metrics 19
   5.3 Parameter Tuning 19

6 Experiments 21
   6.1 Query-independent 21
   6.2 Query-dependent 22
   6.3 Combination 24

7 Conclusion 26
Chapter 1

Introduction

A web search engine is a software system whose main purpose is to search for relevant information on the World Wide Web. Most modern search engines rely on machine learning algorithms [10, 34] in order to find documents, which are relevant to the user query. LTR [29, 43] is an application of machine learning, supervised or reinforcement learning, in the construction of ranking models for information retrieval systems. Using training data, these models are capable of ranking new and unseen data according to their degrees of relevance. [26]. Most web search involve ranking, and many web search technologies can be potentially enhanced by using LTR techniques [21]. In LTR training data consists of lists of items with some partial order specified between items in each list. Query-document pairs are represented by vectors with numerical values, and are called feature vectors. Components of feature vectors are called features and can be divided into three groups [29]: (i) query-independent features - features, that depend only on the document, but not on the query (i.e. document length), (ii) query-dependent - features, that depend both on the contents of the document and the query (i.e. TF-IDF [37]), (iii) query level features - features, that depend only on the query (i.e. query length). Designing features which are capable of characterizing the relevance between a document and a query is a fundamental step of LTR. The majority these features are computed based on the extracted from a document textual elements. However, web pages consist not only of text elements, but they also possess a structured layout, which organizes these elements on the web page by appointing to each of them a position on the web page. This layout can provide useful visual information, which would lead to a better understanding how users identify web page quality for themselves. Although search users perceive this visual information from the web page layout for relevance judgment, this type of information has not been effectively modeled [3, 22].

Visual features have been recently introduced as a new development of LTR algorithms [14]. A web page is rendered into a snapshot as it is shown in web browsers to users. Two types of snapshots for a web page are created: query-independent and query-dependent. The query-independent snapshot captures the raw web page layout information. The query-dependent snapshot aims to capture the web page layout information as well as matching terms of a query, specified by the user. This allows to understand how users see a web page given their current information need. These snapshots are generated by highlighting the matched query words on a web page using some predefined color(s). Given the snapshot of a web page, the visual features are learned automatically via convolutional neural network (CNN) [24]. However, how users perceive web pages may not correspond to the general image view [11]. Since users read row by row from top to bottom, applying a F-bias [15] to the model may lead to better performance of the model on the snapshots. This model is refereed to as the a visual perception (ViP) model. The neural network in the ViP model consists of a combination of convolutional, lstm, and fully connected layers. n this model the extracted visual features are combined together with the textual features and passed through a feed forward network in order to obtain a final score. The results have demonstrated the viability of learning the structural information of document for ranking problems. The results have demonstrated the viability of learning the structural information of document for ranking problems. In this research, we make use of the described architecture, to which we refer as the Segmentation model.

Although the combination of text and visual features provides high ranking scores it requires a number of handcrafted algorithms for the extraction of text features. While the visual information is extracted and learned with the use of a CNN, text features rely on handcrafted algorithms (TF-IDF, BM25, etc.) [38, 46]. A more natural approach, however, would be to create such a representation of the document, which would allow to reproduce the text features in a visual manner. This means that the text features, previously defined by handcrafted formulas, would now be learned from the image itself, exploiting the visual information alone. Since the introduction of visual features proved to be effective in LTR, and the analysis of the structural content does not require any text processing, using a pure visual approach can be an effective solution of this problem. In order to train a ranker, based on visual features, it is first necessary to obtain an accurate visual representation of each document for the entire dataset. We define such a representation as a combination of two structures, which together capture most of the necessary information about a web page. The first structure is defined as
query-independent, and this representation visualizes the layout of a web page and corresponds to its appearance to a user. The information provided by the layout does not depend on a query, but remains constant from user to user. The second structure is query-dependent, and visualizes a user’s search for a particular type of information, which is usually given in the query. The visualization of this query dependent information captures the term frequency, as well as the position of the query terms. To create the defined structures we use several image processing techniques, namely filtering \[42\] and highlighting \[17\]. The query-independent structure can be generated with the use of various image filters, however in this research we make use of only the minimal and laplace filters. For the query-dependent representation we define and implement several highlighting methods, namely term highlighting and term-neighborhood highlighting.

In the final stage the created combined model is compared with the implemented Segmentation model, with a text-based model, and with the best performing models from the visual structures. These comparisons allow to make a better judgment about the advantages and disadvantages of using a combined model, as well as visual features for LTR in general. The accomplished work allows us to answer the next research questions:

- How does the performance of a model, which uses only text information compare to the performance of a visually based model, which uses term highlighting?
- Does the utilization of information in a term’s neighborhood in an image increase the performance of ranking?
- How does the performance of the combined model compare to a model, that uses only text features, the Segmentation model, and the best performing model, which relies on a single visual representation?

The contribution of this work is the implementation of a model that would be text independent, and not reliant on handcrafted formulas, allowing to abstract from the content of the language used in the query and the document. Additionally, we research the possibility of using neighborhood information in order to further improve the visual ranking. This would allow to show how the position affects the relevancy of a document from a visual perspective. Finally, we compare our visual model with the Segmentation model, to understand whether the utilization of visual features can outperform a combination of visual and textual features.

The thesis is organized as follows: In chapter two we provide the history of the research area and the basis of this research, in chapter three we describe the method, implemented in this research. Chapter four describes the parameter search process. In chapter five the experimental setup is covered in detail. Chapter six provides the results of the work, followed by the conclusions in chapter seven.
Chapter 2

Related Work

In this chapter we describe the basis, on which this thesis work is built. First, the history of standard text analysis methods, used to measure the informativeness of a document is provided. After, non-standard methods, namely those, which rely on term order and position in a document, are analyzed. These methods played a big role in introducing new insights about how users perceive and understand documents. The next section extends these models, by utilizing the fact that a document is not just a collection of text information, but also possesses a visual structure, which can also have an affect on document relevancy. Exploiting this visual information can further improve the ranking methods used in LTR.

2.1 Standard text analysis features

A widely used model for feature engineering in LTR is the bag-of-words model [23], also known as the vector space model. It transforms a text into a set of its words, disregarding grammar and word order, while keeping multiplicity. After transforming the text into a bag-of-words, it becomes possible to calculate various measures to characterize the text.

One of the first bag-of-word models which has been commonly used is the term frequency model [31]. In this model we calculate the total amount of times each term appears in each distinct document. One problem with term frequency is that it incorrectly emphasizes documents which are filled with uninformative words ("the", "a", "is") more frequently, without giving enough score to the more meaningful and interesting terms. These more frequent words do not allow to distinguish relevant and non-relevant terms in documents, which leads to irrelevant documents possessing high score and relevancy. Therefore a new statistic, knowns as inverse document frequency [20], which lowers the score of terms that occur very frequently in a document and increases the score of terms that occur rarely, is implemented. The two statistics were combined together into the TF-IDF scoring method [25, 46], which is commonly used as a feature in LTR methods [35]. Additionally, there exist a number of term-weighting methods that are derivatives of TF-IDF. One of them is TF-PDF [5]. TF-PDF was initially introduced in the context of finding emerging media topics. The PDF statistic measures the difference of how often a term occurs in various domains. Another derivative is TF-IDuF [1]. In TF-IDuF, IDF is not counted from the recommended document corpus. Instead, IDF is calculated based on users private document collection. The authors have reported that TF-IDuF was just as effective as TF-IDF but could also be applied in such situations as when a user modeling system has no access to a public document corpus.

Another bag-of-words model, which relies on the term frequency statistic is Okapi BM25 [33]. Okapi BM25 (BM means Best Matching) is a ranking function, which is based on the probabilistic retrieval framework, and is used by search engines to rank matching documents according to their relevance to a given search query. Just like the TF-IDF method, BM25 makes use of term frequency and inverse document frequency statistics, but also introduces the document length statistic, which allows to penalize long documents. Also, several extensions of the BM25 allow to improve the performance of the original model, like BM25F [47], which is a modification of BM25 in which the document is treated as a composition of several fields (such as title, body, anchor) with various degrees of importance and length normalization. Another extension is the BM25+ model [33], which addresses one problem of the standard BM25 in which the term frequency normalization by document length is not accurately lower-bounded; as a result of this issue, documents with big length which have query term matches can be scored unfairly by BM25.

Finally, statistical language models have also been utilized as document features. A statistical language model is a distribution of probability over an ordered collection (sequence) of words. Language models are commonly used in IR in the query likelihood model [9]. Here a unique language model is associated with each document in a collection. Documents are ranked according to the probability of the query in the document’s language model. The unigram language model is widely used for this purpose. A unigram model in IR can be seen as a
combination of several one-state finite automata. It splits the probabilities of different terms in a given context. In this model, the probability of each word only depends on its own probability in the document, so that only one-state finite automata units are present. The automaton itself has a probability distribution over the entire vocabulary, which sum to one.

All the demonstrated bag-of-words models: TF-IDF, BM25, language models, are widely used in rankers as features [36]. These features provide the necessary information about the document. Although these features differ from one another, all of them rely on the term frequency statistic. This statistic is the most important characteristic used in the described methods. Other characteristics are mostly used as rewards and penalties for the term frequency, which themselves depend either on the structure of the entire collection (inverse document frequency), or each individual document (document length). In this thesis, we are interested in modeling the term frequency statistic in a visual manner, since this statistic is straightforward and does not require any complex analysis and hand-crafting.

2.2 New features in text analysis

Although effective, standard features are all based on analyzing document terms. However, a document is not simply a collection of terms. Other characteristics and methods, which can be used as features also exist. Some of these methods introduce new characteristics, which can also contain information useful for improving ranking. The relative position and order of the document terms also correlate with the relevance of a document for the user [44]. Although modeling the term frequency allows to evaluate how relevant a document is to the query, it misses the fact that terms in a document have an order and a position. An example of models, which take these factors into account are phrase based models [13]. In phrase based models words are analyzed not as individual terms, but as a collection, which carries a semantical meaning that can differ from that of the individual words. Although the end goal in such models is to capture the semantics of a sentence by breaking it into phrases, instead of separate terms, the main idea behind the analysis is that the relative position of terms is important in understanding and defining the degree of relevance.

Phrase based models mostly rely on such structures such as N-grams, which introduce term ordering. Another way of acquiring information about the position of terms would be to develop a custom cost function, which would give reward for terms being close to each other, and penalize for the increase in distance [32]. In the context of the current research the second option is more relevant, however it suffers from the need to develop a handcrafted function, which would then allow to obtain a query-document score.

Our work, inspired by the effectiveness of the introduced methods, makes use of the positional analysis of the query terms in the document, by visualizing the query elements according to their relative position in the document. This is more intuitive, since most users perceive a document visually, as a whole image, rather than developing a custom cost function, and then assigning a penalty/reward to terms, based on their positions.

2.3 Visual features

Recently visual features have been proposed as a new development for the improvement of LTR algorithms in [14]. This work leverages the layout information of web pages by rendering the source Web page into a snapshot as is shown in Web browsers perceived by search users. Two types of snapshots for a Web page are considered, namely query-independent snapshot and query-dependent snapshot. The query-independent snapshot captures the raw Web page layout information, which can be directly generated using the render tool over the raw Web page source code. The query-dependent snapshot, on the other hand, aims to capture the Web page layout information as well as matching signals given a specific query. This is to simulate how users perceive a web page given the information need. This is achieved by highlighting the matched query words on a Web page using some background color.

Given the snapshot of a Web page, the visual features are learned automatically for relevance ranking. As the snapshot is an image, the work employs an existing neural model, e.g., the CNN, for this purpose. However, users’ viewing patterns on Web pages may not be the same as that on the general image [41]. A model that can better fit users’ viewing patterns on Web pages may lead to better feature learning performance on the snapshots. In fact, there have been extensive studies on how users view Web pages in the field of visual search. It has been widely accepted that users are accustomed to reading row by row from top to bottom, which forms the well-known F-biased viewing pattern. Inspired by these observations, the authors create a deep neural model that can simulate the F-biased viewing pattern of search users on Web pages to extract visual features from snapshots. The implemented ViP model, consists of a combination of convolutional, lstm, and fully connected layers.

The main contribution of the work lies in the combination of text and visual features and using them for capturing the text content of a web-page, as well as obtaining a structural layout of the page. The work proposes
query independent and query dependent layouts, where the query independent is visualized as the normal web page snapshot, and the query dependent representation having the query term highlighted on the page. This allows to capture both the generic structure of the web page, as well as the positions of the query terms. The combined model, which both types of features, showed an improvement in accuracy compared to models which rely solely on visual or textual content. However, this model requires text features, which themselves, need to be defined according to some algorithms. These algorithms are hand-crafted, which means that they rely on formulas, created by humans, based on some heuristics on what is meaningful. A more reasonable approach would be to learn this information from the visual features, extracted from the visual representation. Such a visual representation would contain both term frequency and positional information, which as described above, is of major importance for understanding how users identify relevant documents.
Chapter 3

Background information

Because we plan to use CNN’s as the main model for learning visual features, we provide a brief introduction on their functionality first. In this chapter we describe the key components, which form a CNN, how they manipulate input data, process it, and return a score, which defines the how successful was the training process and how accurate will be the test predictions.

3.1 Convolution

Convolutional Neural Networks are a major part of Deep Learning tools, which are widely used for image processing. CNNs take their name from the operator known as convolution. The main objective of convolution in a CNN is to extract visual features from the input image. Convolution allows to preserve the spatial relationship between image pixels by learning visual features from the image. CNNs have been widely adopted after showing large success in tasks such as identifying faces and objects. This makes CNNs a comfortable choice for modeling spatially arranged data.

The convolutional layer is the main block of a CNN. These layers apply a convolution operation to the input, after which the result is passed to the next layer. An example of the convolution operation is provided in Figure 3.1:

Here the image patch is convolved with a defined filter to obtain an output.

The layer’s parameters consist of a predefined set of filters (kernels), which have a small receptive field and
extend through the depth of the input volume. These kernels are learnable, and the optimal values for them are obtained during the training phase. It is important to note that only the data for its receptive processed by each convolutional neuron. In the forward pass, each filter is convolved across the image’s width and height, obtaining the dot product between the entries of the filter and the input, which produces a 2-D activation map. This is accomplished for all filters of the CNN. The result is that the network learns filters that are sensitive to some specific type of feature at some spatial location in the input, and activate when these features are detected in the input.

To obtain the full output volume of the convolution layer the activation maps are stacked for all filters along the depth dimension. Each entry in the output volume can therefore be also understood as an output of a neuron that analyzes a small region in the input and shares parameters with other neurons in the same activation map. The key properties, which have been briefly mentioned above, are:

- Local connectivity
- Parameter sharing

Although it is possible to learn features with a fully connected feed-forward neural network, applying this architecture to an image is ineffective. A very big amount of neurons is necessary, due to the fact that the input sizes associated with images are large. The convolution operation solves this problem by reducing the number of parameters, allowing the network to be deeper and have less parameters. Important to mention, it also resolves the vanishing or exploding gradients problem of back-propagation when training neural networks with multiple layers.

Parameter sharing is used to control the number of free parameters. It relies on one assumption: That if a patch feature is useful to compute at some spatial location, then it should also be useful to compute at other positions. This allows to use the same weight and biases for all neurons in a single "slice".

### 3.2 Activation

The activation is a function which calculates a non-linear transformation of the input. The non-linearity allows the network to learn and perform more complex tasks. Without the non-linearity it would become impossible to solve such complex tasks as, for example, object detection in images.

One such activation function, which is widely used in neural networks is ReLU. ReLU stands for Rectified Linear Units. ReLU is an element wise operation, where at each image pixel, negative values in the feature map are replaced by zero. Mathematically, it means applying a non-saturating function

$$ f(x) = \max(0, x) $$

The introduction of this function into the neural network allows to increase the nonlinear properties of the decision-making of the network without modifying the receptive fields of the convolution layer. ReLU is an optimal choice for the activation, due to the fact that it allows to train the neural network several times faster without any major penalty to the accuracy.

### 3.3 Pooling

Convolutional networks often include pooling layers, which take the output values of neurons at one layer and transform it into a single value in the next layer. The main purpose of spatial pooling is dimensionality reduction. An example of pooling is provided in Figure 3.2:
Each feature map retains only the most important information. The input image is partitioned into a set of non-overlapping (in most cases) rectangles and, for each such sub-region, provides a single value output. How this value is calculated is defined by the type of pooling. The intuition behind pooling is that the precise location of a feature is not as important as the relative location to other features. The pooling layer allows to reduce the spatial size of the representation, the number of parameters and also works as a regularizer which reduces overfitting. It is a common practice to insert a pooling layer after the activation function and the next convolutional layer in a CNN architecture. The pooling also provides translation invariance. Due to the aggressive reduction in the size of the representation, the trend is towards using smaller filters.

There are different types of pooling: Max, Average, Sum etc. In Max Pooling, the largest element from the rectified feature map within a window is taken. Instead of taking the maximum value it is also possible to take the average or sum of all elements. In practice however, Max Pooling proved to be the most efficient.

3.4 Fully connected

The Fully Connected layer is a Multi Layer Perceptron. "Fully Connected" means that all neurons from the previous layer are connected to every neuron in the next layer. The results from the convolutional and pooling layers are high-level features of the provided image. The main purpose of the Fully Connected layer is to use these features for prediction.

Apart from prediction, adding a fully-connected layer can also an easy way of learning non-linear combinations of visual features. Most of features from the convolutional and pooling layers separately function well for the various tasks, but combinations of these features provides more information to the model.
Chapter 4

Method

This chapter describes the method which has been used, in order to answer the research questions, put in the beginning of the thesis work. We start by introducing two types of visual representations, namely the query-independent and query-dependent representations. The corresponding sections provide a detailed analysis on the exact techniques, which have been used in order to create these representations. After, the next section describes the text features, which have been utilized in the process of creating a text based model. Defining these features is necessary, since one of the thesis objectives is to compare the visual models with a model, which uses only text features. Finally, we research the various neural network architectures implemented in the research.

4.1 Visual representation

All web pages have a defined visual appearance. That appearance is usually independent of a query and remains the same. On the other hand, each user, when being provided with a web page, has a subjective understanding of how the particular page is related to his/her current needs. Therefore a web page can be represented as a combination of query-independent and a query-dependent features.

The query independent representation attempts to visualize the structure of a document, which is constant. The information provided by this type can be identified as global, as it persists from query to query. The query dependent representation allows to see the representation of a document for a particular user query. These categories of representation do not exclude each other, but can be combined together to form a final document representation. This allows to capture both the local query-document and the global information of a document.

4.1.1 Query-independent representation

First of all, we provide an insight in what can be identified as the structure of a web page. This structure must not depend on a query, and provide information about a document. For each document, this type of information is contained in the spatial location and size of each DOM-element on a web page. The location and size, which identify the position of a web page element are independent from the query terms, as well as allow the user to perceive the documents structure. Since only a documents layout is important, the exact content does not represent any interest. Therefore such information, like the semantical meaning of the text inside the element, is irrelevant, and should be treated as noise. This means that in order to create the query-independent representation it would be necessary to process the image in such a way, that only the information which has been defined as web page structure information is left, while the rest of the information should be either removed or blurred.

To achieve this we rely on various Computer Vision (CV) image processing techniques [7]. A popular technique in the CV field is to apply various filters on the image before extracting visual features [6]. There are many types of filters used like gaussian, sobel, median etc. [8]. The main purpose of these filters is to remove information from an image, which in the current task is not perceived as important.

Since this research mostly focuses on visual data and features, it would be consistent to use CV preprocessing on the available data. The main task would be to apply image filters to capture the web page structure: DOM-elements size and edges, while removing any other information. Therefore the filters to be used in the task should mostly provide blurring and edge detection. In order to meet these goals, two filters have been chosen: (i) Minimum filter, (ii) Laplace filter.

The main reason why these two representations have been considered in this thesis, is that while other filters, like sobel or gaussian allow to accomplish edge detection and blurring, they either remove too much necessary information (i.e. text position), or leave too much unnecessary information (i.e. word character shapes). The
best visual appearance (edges and shapes) has been achieved for these two filter types. The Minimum filter is a type of image transformation. This filter replaces the central pixel with the darkest one in the running window. The size of the kernel in which all values are set to the minimum value is 10, as this value provides the best blurring, while also allowing to visually separate the DOM-elements from the background.

The Laplacian is a 2-D isotropic measure of the second gaussian derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection. The usage of the gaussian derivatives allows to reduce sensitivity to noise. This pre-processing step reduces the high frequency noise components prior to the differentiation step. The operator takes a single gray-level image as input and produces another gray-level image as output. For the Laplacian filter the sigma parameter is set equal to 1.5, which provides the necessary edge detection (DOM-elements edges), as well as blurring effect.

Figure 4.1 provides an example of how these representations look.

![Figure 4.1: Types of query-independent representations](image)

As can be seen from the figure both representation filter the information which defines the context of the document, leaving mostly only the shapes, which reflect the layout of the web page.

### 4.1.2 Query-dependent representation

We attempt to create a visual representation of the web page which would reflect on what the user is most interested in when beginning the web search. Since a query represents a collection of terms, it is reasonable to focus on analyzing these terms in the document. One way we can introduce this focusing aspect in a visual manner is term highlighting. Query terms are first highlighted in the document, and then their positions are extracted. This position is then mapped onto a lower resolution blank image. This algorithm allows to both extract all the necessary terms, as well as preserve their positions. Such modeling allows to visually express the term frequency and position characteristics in a document. Although the modeling of the query term frequency and position is well understood, it is not obvious whether showing only the query terms in the visual representation is sufficient for achieving high prediction accuracy. We experiment with various query-dependent representations, in order to define an such a representation, which would allow to achieve higher accuracy rates at test time.

**Term highlighting**

The single term highlighting (TH) represents the most simple model used in this research. This model takes the positions of the DOM-elements which correspond to the highlighted terms and maps them on a blank image. The result of this operation would be a rectangle, since the position of the DOM-element is calculated based on the bounding box around the query term. The model is implemented both in gray-scale and RGB. The RGB model expands the grayscale, since not only the presence or absence of query terms is important, but also which query terms are in the document affects how the user judges the document’s relevancy. Therefore the RGB model provides a unique color for each query term. This allows the model to weight different terms relatively to each other, unlike in the gray-scale model, where all terms are of the same color. An example of these representations is given in Figure 4.2.
Although the above representations accurately describe the query terms in the document, they suffer from the fact that not only the query terms may be important for the user. Other document terms, which are located close to the term can also possess a certain degree of relevancy. The user may not only process a certain term, but can also be interested in the surrounding information [30].

**Term-neighborhood highlighting**

The visual concentration on a certain area of interest around the query term is modeled with a circular highlighting with a defined radius. This representation is known as the term-neighborhood highlighting (TNH). The circular shape is reasonable as its symmetrical, making the users view unbiased in any direction, and also has soft edges, which is more natural than a shape with sharp angles on edges, from the position of human perception. [15]. The center of the circle is defined as the center of rectangle, bounding the query term. Therefore each query term in the document is located in the center of a circle, with the radius of the circle showing the area of interest of a user. The model is implemented both in gray-scale and RGB, where each query term has it’s own color. Just like in the previous situation the RGB model provides a unique color for each query term, which provides the visual weighting of terms. An example is demonstrated in Figure 4.3.

The problem which can arise with this model, is that it provides the same color for the surrounding space. This means that the surrounding terms are set to be equivalent in terms of relevancy to the query term in the center of the circle. We propose, that making such an assumption is not necessarily correct and can result in lower accuracy prediction for the model.

**Term-neighborhood highlighting with decay**

In order to solve the mentioned above problem, we introduce a decay parameter to the modeling process. The implemented model is called the term-neighborhood highlighting with decay (TNH-D). If previously the color of the area of interest remained constant, now the color changes as the distance from the center of the circle grows. This means that the query term possesses the most relevancy, while the terms around become less and less relevant, as they are located further from the query term. The intuition behind this representation is that while the user can pay attention to a particular region of the web page, with the query term located in the said region,
the information which is located close to the query term is related to the query term, and therefore relevant, however it’s relevancy is not of equal degree to the query term. The representations have been implemented in gray-scale and RGB. An example of the representation is provided in Figure 4.4:

![Grayscale and RGB representations](image)

Figure 4.4: Term-neighborhood highlighting with decay

One disadvantage of this visualization is that query terms, which are located close to each other will eventually overlap, thus leading to a situation when the relevancy in the area of interest for one query term will be ignored, because it is covered by the area of interest from another query term area of interest.

**Additive term-neighborhood highlighting with decay**

To address this problem we create an additive representation, and name the model - additive term-neighborhood highlighting with decay (TNH-AD). This representation is similar to the previous one, except that it sums the pixel values, instead of just overwriting them. This means that the area of interest for each query term is shown correctly. Additionally, since the values add, this would lead to the zone, located in the logical conjunction of two separate areas, having a higher intensity than other zones in the areas of interest of each query term. The relevancy in the logical conjunction would be higher than in other zones of the circles. The conjunction zone would grow as the two (or more) query terms come closer to each other, leading to and even bigger zone with high intensity, and thus a higher relevancy signal in that spatial location. Essentially, such a representation models the positional relationship between query terms, providing higher intensity signals for terms located close to each other. We assume, that this would lead to an increase in performance, compared to the previous version. The model has been implemented in gray-scale and RGB. An example is provided in Figure 4.5.

![Grayscale and RGB representations](image)

Figure 4.5: Additive term-neighborhood highlighting with decay

### 4.2 Feature Construction

For most of the work we rely on CNN’s to extract and learn the features of the provided images automatically, however, since we are interested in comparing the performance of the model, which relies on visual features, with a text-based model, it becomes necessary to construct features for the text model. One simple, yet widely used approach in Information Retrieval for feature construction is the bag-of-words representation [23], also known as the vector space model. It transforms a text into a set of its words, disregarding grammar and word order, while
keeping the frequency. After transforming the text into a bag-of-words, it becomes possible to calculate various measures to characterize the text. There are many types of text features which can be used to learn a model, however in this thesis we rely on the features defined in LETOR 4.0 [36]. LETOR is a package of benchmark datasets for research on LTR, which contains text features, relevance judgments, data partitioning, evaluation tools, and multiple baselines. LETOR has been popular in multiple works in the information retrieval research field [2][11][12], which inspired this thesis to use the standard text features, as defined in LETOR. Using Pythons standard libraries allowed us to implement the next bag-of-words text features, present in LETOR:

- Body TF-IDF
- Title TF-IDF
- Body BM-25
- Title BM-25
- Body Size
- Title Size

All features have been calculated separately for the title and body html tags of a document. As a result, each document from the dataset is represented as a feature vector of length six.

4.3 Model Architecture

Various architectures are available, in order to create an efficient neural network [40]. Because part of the research involves comparison with other models, beside implementing the researches model, it becomes necessary to specify the architecture for each model. A more accurate approach would be to tune the parameters of the neural network, however that lies beyond the scope of the current work. Instead we rely on model architectures, which have been widely used in the science community, and proved to be efficient in the Deep Leaning area of research [16].

The architecture for the text-based model [27]:

\[\text{Figure 4.6: Text model}\]

This architecture specifies a simple feed-forward network, also known as Multilayer perceptron [18]. The input layer takes a feature vector of length six, and feeds it through the network, using standard forward-pass and back-propagation for learning the network parameters [39]. This network uses feature vectors in order to modify its weights, until reaching convergence. It exploits the text information of a document. However, as stated earlier, a document is not only a collection of terms, but terms also possess a relative order and position on a web page. Exploiting these features would increase the performance of the model as shown in Fan et al. [14]. Therefore we implement a CNN model, which would use spatial information, learning it from an image.

The architecture of the CNN [4]:

\[\text{Figure 4.6: Text model}\]
The Single-representation (SR) model is a standard CNN with two convolutional layers, and two fully connected layers. This network is used to learn visual features from the query dependent/query independent representations. Although effective, the architecture of this model uses the entire image as an input, which means that a user, given an image, perceives the entire image immediately. However, as mentioned in Faraday [15], users are biased in their image processing. One such bias is the F-bias. A neural network which segments an image into parts and uses them for feature extraction allows to simulate the F-bias and can result in better performance. We implement a Segmentation model, which is inspired by the neural network from the ViP model.

The structure of the Segmentation network [14]:

![Diagram of the Segmentation network]
The Segmentation network uses a CNN, with two convolutional layers, one lstm layer and two fully connected layers. The convolutional layers are used for the extraction and learning of visual features, while the lstm layer introduces the F-bias.

One problem with the Segmentation model is that it requires extracted text features, which are combined with the learned visual features in order to function. These text features are provided via standard handcrafted text analysis methods. However, we would like to create such a model, which would learn all features automatically, without relying on any handcrafted methods, while being just as effective, as models, which use these methods. In order to solve this problem we propose a Dual-representation (DR) model.

The structure of the DR model:

---

Figure 4.8: Segmentation model
The DR uses four convolutional layers, where each pair is used independently of the other. One of the pair extracts the features from the query independent representation, while the other does the same with the query dependent representation. The extracted features are then flattened and concatenated. To obtain a final score three fully connected layers are introduced in the networks architecture. Unlike the Segmentation model, DR uses only visual information, and unlike the SR model, it utilizes both, the query-independent and query dependent representations.

In order to learn the models, an important step is to define the loss function. The loss function itself depends on the learning approach. Following the paper, which implemented the ViP model, we use pairwise learning. Pairwise approaches look at a pair of documents at a time in the loss function. The loss function is defined as:

$$\text{loss}(x, y) = \max(0, -y \times (x_1 - x_2) + M)$$

The defined criterion measures the loss given inputs $x_1, x_2$, which are visual representations of a pair of documents, where $x_1$ is ranked higher than $x_2$ and a label tensor $y$ with value 1. $M$ is the margin, and is set equal to 1.

The intuition behind the approach is that given a pair of documents, we attempt to come up with an optimal ordering for that pair, based on the ground truth. The main goal is to minimize the number of inversions in ranking, which are those cases where the output pair is given with the wrong order. Pairwise approaches are popular in information retrieval because optimizing document order is closer to the nature of ranking, instead of predicting class label/relevance score.
Chapter 5

Experimental Setup

In this chapter we provide the information about the experimental setup. First of all, the dataset, from which we generate the described visual representations is introduced. The analysis of the dataset provides the necessary information for understanding what the data represents and it’s conversion to a format, which is valid for the model’s training process. Finally, a vital part of obtaining a high ranking accuracy is parameter tuning. Since several hyper-parameters have been introduced in the research it makes sense to tune these parameters in order to achieve the best possible performance.

5.1 Dataset

Choosing an appropriate dataset is just as important as defining the algorithm and the model. A good dataset should be content rich, so that it would be possible to engineer as many features as necessary. Also, the dataset should reflect a real users needs as much as possible, rather than be a simulation. Finally, the dataset must contain enough samples, in order to train the model and achieve a high level of generalization.

The current work uses the ClueWeb09 Category B dataset. Because we are using neural networks in the thesis work, it becomes important to gain access to big data, since neural networks are known for being data-hungry \[48\]. The mentioned dataset contains 246.8 GB of compressed data, which is enough for effectively training a neural model. As the dataset contains only web pages, the relevance judgments have been obtained from the Million Query Track 2009. The Million Query Track 2009 contains scores for a set of query-document pairs, where the average amount of documents per query is 49. Important to note, that this set already handles the problem of retrieving relevant information for the query. Therefore, there is no need on creating any retrieval algorithms, the only task is to correctly rank the provided documents.

Since the research involves visual information, it is important to ensure that the html web pages can be rendered into images (jpg, png, etc). This rendering should be computationally efficient, achieving high performance. To achieve this, all the javascript code in every web document has been removed. The rendering is accomplished by using a standard image manipulation library. During these procedure, it has been revealed that some of the data can not be effectively re-formatted. Additionally some of the converted data was not possible to open using image libraries. The statistics on the documents are provided in table 5.1.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>34012</td>
</tr>
<tr>
<td>Non-convertable Image</td>
<td>2984</td>
</tr>
<tr>
<td>Corrupted</td>
<td>20</td>
</tr>
<tr>
<td>Available</td>
<td>31008</td>
</tr>
</tbody>
</table>

Of 34012 total web pages, 31008 are available for further transformations and learning. The textual information in these html documents has been lower-cased, non-alphabetical symbols have been removed, except numerical values. Non-English information has not been removed from the web pages as it affects the position of the English elements. Removing a word from the document will result in the rest of the text shifting left-ward. English stop-words also have not been deleted for the same reason as with non-English words. The Million Query Track contains 667 queries. Each query is represented by a unique ID and a collection of tokens, which contain the information which the user is interested in. The pre-processing for the queries included the same operations as with the documents, except the stop-words in the queries have been removed.
The resulting vocabulary included 1286 unique words. Removing the stop words is justified, as these words contain little information, and can lead to a drop in performance, as well as leading to an information overload in the visual representation.

5.2 Metrics

Evaluation measures are used to calculate how well the search results of an information retrieval model satisfy the user's information need given in a query. There are two types of such metrics: online metrics, which analyze users interactions with the search system, and offline metrics, which measure how likely each result is to meet the information needs of a user, also known as relevance [46]. The current research focuses on offline metrics for evaluating the model effectiveness. Offline metrics are created by manually giving scores to the quality of search results. Both binary (0 or 1) and multi-level (e.g. from 0 to 5) scales can be used to score each document returned in response to a query.

The implemented metrics in this work are:

- Precision@k (k=1,5,10)
- NDCG@k (k=1,5,10)
- Mean Average Precision (MAP)

Inspired by Fan et al. [14], and in order to ensure better comparison with their model, we implement the same metrics as in the paper, which served as a basis for this thesis project.

Precision [46] is defined as the fraction of relevant documents among all retrieved documents. In modern information retrieval, recall is no longer necessary, as many queries have hundreds and thousands of relevant documents, and no casual user is interested in reading all of them. Precision at k (P@k) corresponds to the number of relevant results among the first k documents on the searching results page. This metric is efficient, since only the top k results need to be examined to determine if they are relevant or not.

DCG [19] is a metric that uses a relevance scale to evaluate how useful is a document, based on its position in the output list. The intuition behind the DCG is that documents with a high value of relevance, that are positioned lower in a search result list should be penalized as the graded relevance value decreases logarithmically proportional to the position of the result. Since for different queries, there may be document lists of varying length, to compare performances the normalized version of DCG uses an ideal DCG, so that all nDCG calculations are relative values on the interval 0.0 to 1.0 and are therefore comparable across queries.

The last metric, the mean average precision for a collection of queries is the mean of the average precision values for each query [28]. While precision and recall metrics are single-value, which are based on the entire list of available documents returned by the system, for systems that return a ranked list of documents, it is necessary to consider the order in which the documents are returned. By calculating precision and recall at every position in the list of documents, we can plot a precision-recall curve, plotting precision as a function of recall. Average precision computes the area under the precision-recall curve.

5.3 Parameter Tuning

In the Method chapter, we have described the various visual representations, which were implemented during the research. However these visualizations introduce a number of parameters, namely the radius of the circle and the strength of the intensity decay from the center of the circle. These parameters are not subject of learning in the current work, however they can have a serious effect on the resulting prediction accuracy. In this situation it would be reasonable to tune these parameters. Parameter tuning in this case would mean considering a certain amount of values for each parameter, and performing an exhaustive search, i.e. taking all possible combinations of parameter values, and picking the combination which provides the best accuracy for the final result. Figure 5.1 provides an illustration how the parameter tuning changes the visual representation.
Figure 5.1: Parameter tuning for visual representations

Up to five values for each parameter have been chosen: values 1,2,3,4,5 for the radius parameter, and values 4,8,12,16,20 for the decay strength. Since we use a linear function for the intensity decay, they decay strength parameter defines how much intensity each pixel position looses, as it becomes further from the center. In total this would lead to a total of twenty five models for representations with two parameters (TNH-D, TNH-AD), and five models - with one parameter (TNH).

The tuning process starts with identifying the positions of query terms in a document. These positions are then mapped onto a blank image. The circle of the defined radius and strength is drawn from the center of the query term position with one combination of the mentioned above radius and decay values.
Chapter 6

Experiments

In this chapter we discuss the experiments, which have been conducted during the thesis work. We start by searching for the best model with a single hyper-parameter. These include the query-independent representation, as well as query-dependent TH and TNH models. The first experiment aims to discover the optimal query-independent visual representation, after which we proceed by tuning the TH and TNH models. In the third experiment the models which include two hyper-parameters (TNH-D, TNH-AD) have been implemented. Out of all the models, the one which provides the highest MAP value is taken for the final experiment. In the last experiment we compare the performance of the BM25, Text, Segmentation, SR for the query-dependent and query-independent representations, and DR models. Finally, we perform a t-test for the models to find the statistical significance of the obtained results.

6.1 Query-independent

The first experiment gives an insight in which visual representation will provide the highest performance. For this experiment the grayscale Normal, Laplace, Minimal query-independent visual representations are used. The resolution values are set to 16x16, 32x32, 64x64, 128x128. For the defined representation-resolution values an exhaustive search is performed, which means taking all possible unique combinations of these parameter values. In order to train a ranker the SR architecture is used. This model is named - single representation query-independent (SR-QI). In total, twelve models are generated. This allows to accomplish two objectives, namely discover the visual representation which provides the highest accuracy, and additionally find the optimal resolution which will be used in later experiments. The experiment results are provided in Figure 6.1:
The best performing representation is the Laplace. One explanation to this would be that unlike the Normal and Minimal representations, the Laplace representation is based on an edge detection technique, which removes all the information from the image, which is not perceived as an edge. The information about the edges in an image is most relevant to that of what defines the structure of a document, thus leading to higher MAP scores. As for the resolution, the performance first increases until reaching a peak, and then decreases. The peak value is a resolution of 64x64 for all representations. The observed pattern may be explained as a small resolution does not provide enough information for the network to learn, and therefore leads to a smaller prediction accuracy. On the other hand, a big resolution provides information which can act as noise during the learning process, distracting the network from accurately learning the visual features.

Finally, it is important to note, that the MAP values displayed in the figure are of extraordinary high values. As mentioned in previous chapters the dataset, which is used in the current thesis work includes relevance judgments for various queries. These judgments are provided for a number of documents, of which many are relevant to the given query. The result of this, is that our ranking problem is in fact re-ranking a small amount of documents, which have already been extracted and judged for each query. This leads to MAP scores, which are higher than in Fan et al. [13].

6.2 Query-dependent

We start the experiment by comparing the performance of the TH (term highlighting) and TNH (term-neighborhood highlighting) models. For the TNH model, the range of the radius is set from one to five. Like in the previous experiment the single representation architecture is used for training. We call this model - single representation query-dependent (SR-QD). The results are provided in Figure 6.2 and Table 6.1:
Figure 6.2: TH, TNH models performance in MAP

From the figure it can be observed, that the best performance is achieved for the representation with zero radius, which is the TH representation. This comes from the fact that the simple TNH model provides equivalent relevancy to all information in the query terms neighborhood, however not all neighborhood information is necessarily related to the query term in the center. As a result, this can lead to a drop in prediction accuracy, which is observed in the current experiment.

After, we tune the parameters of the TNH-D (term-neighborhood with decay) and TNH-AD (term-neighborhood with additive decay) models. Since these models have two hyper-parameters, a set of five values has been chosen for each parameter. Then, just like in the query-independent experiment, an exhaustive search is performed, where all possible combinations are taken. This would result in twenty five models for each representation. The results are provided in Figure 6.3.

The image on the left Figure 5.3 gives the results for the term neighborhood highlighting model, while the image on the right Figure 5.3 is the same model with the additive property. First of all, the results for THN-D demonstrate that the MAP values first increase, then reach a peak and finally start dropping. The TNH-AD results do not follow an intuitive trend. This behavior shows, that the model is not effectively learning, but rather reacting to the input information as noise, leading to results, which do not possess any kind of pattern. Secondly, by comparing the two representations, it can be seen that model without the additive property is more effective. Such an outcome takes place as a result of incorrect modeling of the assumption that documents with words, positioned close to each other, should be more relevant. While the assumption may hold true, modeling it by simply adding the neighborhood intensities proved to be ineffective. This problem takes place, since the adding intensities of closely positioned terms quickly sum up to the maximum value in an image array, which is 255. In case of multiple words overlapping each other, this leads to an entire area of the image possessing the highest pixel values in the image, signaling that all the highlighted zone is completely relevant. One way to solve
this issue in future work, would be to avoid operations which could cause an overflow (addition, multiplication) of intensities in identifying positional relations, and instead take operations, for example, such as median values, or set a unique color for overlapping zones.

Comparing the performance of the TNH-D and TNH-AD leads to a conclusion that the term-neighborhood representation is the most accurate among the representations, which attempt to take the information around the query term into account. This proves that relevancy does decrease as the distance from the query term grows.

After obtaining the MAP scores for all the models, the best model is determined. For the implemented representations, the one with the highest performance is the TH representation. This representation proved to be more effective than any other. While why it is better than the simple term neighborhood model, has already been explained, why this representation achieves the best performance than any neighborhood representation must be explained. We propose, that this takes place due to the fact that modeling the effect which the neighborhood has on relevancy, by using a representation, where the intensity decreases linearly from the center of the circle (which corresponds to the query term) is incorrect. While it is true that the increasing distance from the center should lead to lower relevancy, it does not necessarily have to be linear. Also it does not have to possess a monotonic behavior. Contrary, relevance can have a distorted progression as the distance increases, where relevancy first decreases, than increases, and then starts decreasing again. A solution to the mentioned problems, in future researches, might be to introduce non-linear functions for the decay parameter, as well as introduce a distortion parameter, which adds noise to the visual representation.

In the final step of this experiment, an RGB representation of the term highlighting model has been tested. The results are shown in the table:

<table>
<thead>
<tr>
<th>Colorspace</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grayscale</td>
<td>0.7316</td>
</tr>
<tr>
<td>RGB</td>
<td>0.7453</td>
</tr>
</tbody>
</table>

The RGB representation outperforms the grayscale version. This is accurate, since the grayscale version assigns equal weights to all query words, while the RGB version assigns a unique color to each word. This allows to weigh each of the words separately, increasing the relevancy of documents, where more important term appeared more often, and decreasing the relevancy of documents, where less important term appear often.

### 6.3 Combination

In the last experiment, the various architectures for the best found representations are being compared. The Text model uses the six features described earlier in chapter 3. Additionally we add a standard BM25 evaluation (body and title) in order to show, that the results even with the most standard tools provide high prediction results, which comes from the fact the the average number of documents in each query is not high (49), and the documents are mostly relevant, leaving to us the task of ranking only. The Segmentation model is inspired by the ViP model, and uses the same architecture. The Single and Dual models are alike, except the Dual uses both, a query-independent and a query-dependent representations, while the Single model can use only one. In the current setup, the Segmentation model relies on the RGB TH representation, and the SR model has two implementations, one with the RGB TH (SR-QD), and the second with the Laplace representations (SR-QI). The Dual uses the Laplace and the RGB TH simultaneously. The results of the experiment and the t-test are shown below:

<table>
<thead>
<tr>
<th>Model</th>
<th>P@1</th>
<th>P@5</th>
<th>P@10</th>
<th>N@1</th>
<th>N@5</th>
<th>N@10</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM25</td>
<td>0.824</td>
<td>0.567</td>
<td>0.410</td>
<td>0.686</td>
<td>0.640</td>
<td>0.642</td>
<td>0.676</td>
</tr>
<tr>
<td>Text</td>
<td>0.863</td>
<td>0.549</td>
<td>0.403</td>
<td>0.699</td>
<td>0.654</td>
<td>0.663</td>
<td>0.697</td>
</tr>
<tr>
<td>Segmentation</td>
<td><strong>0.966</strong></td>
<td><strong>0.685</strong></td>
<td><strong>0.498</strong></td>
<td><strong>0.808</strong></td>
<td><strong>0.735</strong></td>
<td><strong>0.727</strong></td>
<td><strong>0.768</strong></td>
</tr>
<tr>
<td>SR-QI</td>
<td>0.932</td>
<td>0.630</td>
<td>0.457</td>
<td>0.765</td>
<td>0.686</td>
<td>0.684</td>
<td>0.719</td>
</tr>
<tr>
<td>SR-QD</td>
<td>0.924</td>
<td>0.668</td>
<td>0.480</td>
<td>0.720</td>
<td>0.689</td>
<td>0.685</td>
<td>0.745</td>
</tr>
<tr>
<td>DR</td>
<td><strong>0.966</strong></td>
<td>0.676</td>
<td>0.496</td>
<td>0.797</td>
<td>0.727</td>
<td><strong>0.730</strong></td>
<td><strong>0.768</strong></td>
</tr>
</tbody>
</table>
Table 6.4: Statistical hypothesis test, p-value (* indicates significance, i.e. p-value ≤ 5%)

<table>
<thead>
<tr>
<th>Model</th>
<th>P@1</th>
<th>P@5</th>
<th>P@10</th>
<th>N@1</th>
<th>N@5</th>
<th>N@10</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text-Segmentation</td>
<td>0.0090*</td>
<td>0.0002*</td>
<td>0.0065*</td>
<td>0.0695</td>
<td>0.0512*</td>
<td>0.1101</td>
<td>0.0514*</td>
</tr>
<tr>
<td>Text-SR-QD</td>
<td>0.8318</td>
<td>0.0089*</td>
<td>0.0220*</td>
<td>0.9364</td>
<td>0.8996</td>
<td>0.8572</td>
<td>0.3242</td>
</tr>
<tr>
<td>Text-SR-QI</td>
<td>0.1590</td>
<td>0.0050*</td>
<td>0.0481*</td>
<td>0.5346</td>
<td>0.3607</td>
<td>0.5270</td>
<td>0.3355</td>
</tr>
<tr>
<td>Text-DR</td>
<td>0.0924</td>
<td>0.0004*</td>
<td>0.0069*</td>
<td>0.5754</td>
<td>0.1814</td>
<td>0.1840</td>
<td>0.0670</td>
</tr>
<tr>
<td>SR-QD-DR</td>
<td>0.0947</td>
<td>0.3377</td>
<td>0.8813</td>
<td>0.3770</td>
<td>0.2003</td>
<td>0.2545</td>
<td>0.3994</td>
</tr>
<tr>
<td>SR-QI-DR</td>
<td>0.5834</td>
<td>0.3512</td>
<td>0.4273</td>
<td>0.8083</td>
<td>0.5072</td>
<td>0.3833</td>
<td>0.2731</td>
</tr>
<tr>
<td>SR-QD-Segmentation</td>
<td>0.0234*</td>
<td>0.3689</td>
<td>0.8274</td>
<td>0.0331*</td>
<td>0.0702</td>
<td>0.1753</td>
<td>0.4534</td>
</tr>
<tr>
<td>SR-QI-Segmentation</td>
<td>0.2377</td>
<td>0.3839</td>
<td>0.3968</td>
<td>0.1286</td>
<td>0.2202</td>
<td>0.2721</td>
<td>0.3220</td>
</tr>
<tr>
<td>DR-Segmentation</td>
<td>0.5201</td>
<td>0.9620</td>
<td>0.9406</td>
<td>0.1972</td>
<td>0.5601</td>
<td>0.8060</td>
<td>0.9476</td>
</tr>
</tbody>
</table>

The BM25 model is not present in the test, since we use it only for the demonstration of the datasets properties. As can be seen from the tables the best performance for most of the metrics is achieved by the Segmentation model, however the results are not statistically significant from the Dual model. Both models stand on par with the Text model for a number of metrics, which means that at worst, these models are capable of achieving close performance to the handcrafted features, and for some metrics (precision at rank and MAP) they even outperform the text features. This allows to conclude that the exploitation of visual features is a viable tool in information retrieval ranking.

In order to better understand the models performance, the training process of the networks has been illustrated and can be seen in Figure 6.4:

The figure shows that the models, which take into account the visual information converge much faster, than the Text model. The lowest loss is achieved for the DR model, however it does not differ strongly from the Segmentation model. The SR (only SR-QD) model converges to the equivalent loss with the Segmentation model, however it’s test performance is worse. Most likely this takes place due to overfitting.
In this thesis work we have researched the subject of creating visual features for information retrieval ranking. Previously, ranking was accomplished based mostly on textual analysis of a collection of documents. A document was represented as a group of tokens, which were then used to construct features. These features were, in most cases, based around using the term frequency statistic in methods such as TF-IDF or BM25, or the distance between query terms in methods like Positional language models. One problem with these methods, is that they require handcrafting, i.e. it is necessary to manually develop penalties/rewards in order to account for different structures of documents (varying document length, informativeness of a document).

A better approach would be to learn features from a visual representation, which would contain all the necessary information in itself, rather than having to extract it by using various handcrafted methods. Instead of relying on these methods, inspired by the recent development of visual features for LTR, we have created a number of visual representations, which allow to capture the inner structure of a document (query-independent) and the term frequency, position of the query terms (query-dependent). The visual features from these representations have been learned automatically, by leveraging the power of neural networks.

The results have demonstrated that visual features are more effective than the implemented text features. In order to obtain the visual features, different representations have been tested, of which the most effective one was the RGB term highlighting for the query dependent on one side, and the laplace representation for the query-independent on the other. Both representations were used in the Dual-representation model, which has either outperformed, or demonstrated almost equal results compared with other visual models.

The mentioned results allow to answer the research questions, which have been put in the beginning of the research:

- How does the performance of a model, which uses only text information compare to the performance of a visually based model, which uses term highlighting?

The model which utilizes visual features outperforms a model, based exclusively on text features. This is accurate for all implemented visual models. This increase in prediction accuracy, is a result of acknowledging and modeling the structural information, present in each document.

- Does the utilization of information in a term’s neighborhood in an image increase the performance of ranking?

Positional information in the query terms neighborhood did not lead to any increase in performance. Contrary, the models which used neighborhood information showed lower prediction accuracy, than models, which relied only on the query terms. We propose that such an outcome could be a result of modeling the decay parameter as a linear function, or modeling the neighborhood as a circle. Since the development of positional models in text analysis has proved to increase the test accuracy, we find it reasonable to assume that the utilization of neighborhood information in a visual manner should also lead to better results, however we could not achieve such results in the current research.

- How does the performance of the combined model compare to the Segmentation model, a text-based model, and the best performing model, which relies on a single visual representation?

The combined (DR) model does not demonstrate any statistically significant difference from the Segmentation model. However, the DR model demonstrates better results for all precision metrics, including MAP, compared to the Text model. As for the Single-representation model, the best results have been achieved, as already mentioned, for the RGB term highlighting representation. The only metric, where the DR model significantly outperforms the SR, is for the precision at rank one. Overall, the DR model shows results either better, or at least not worse than other models. The increase in performance mostly takes place in better identification of
relevant documents, which leads to an increase in precision, however the NDCG results do not improve. Although the thesis included tests with various characteristics, features, visualizations the research possess certain limitations. First, we have not ran tests only with six defined text features. Using more features could lead to an increase in performance for this model. Second, we have not issued enough experiments with neighborhood modeling. There are other ways of modeling neighborhood information besides using circular neighborhoods and linear relevancy decay. The research of a more accurate modeling of the neighborhood information, as well as using other text features, such as language models and positional language models, for a more fair comparison is a subject of future work.
Bibliography


