Extractive Explanations for Interpretable Text Ranking

Leonhardt, Jurek; Rudra, Koustav; Anand, Avishek

DOI
10.1145/3576924

Publication date
2023

Document Version
Final published version

Published in
ACM Transactions on Information Systems

Extractive Explanations for Interpretable Text Ranking

JUREK LEONHARDT, L3S Research Center
KOUSTAV RUDRA, Indian Institute of Technology (ISM) Dhanbad
AVISHEK ANAND, Delft University of Technology

Neural document ranking models perform impressively well due to superior language understanding gained from pre-training tasks. However, due to their complexity and large number of parameters these (typically transformer-based) models are often non-interpretable in that ranking decisions can not be clearly attributed to specific parts of the input documents.

In this article, we propose ranking models that are inherently interpretable by generating explanations as a by-product of the prediction decision. We introduce the Select-And-Rank paradigm for document ranking, where we first output an explanation as a selected subset of sentences in a document. Thereafter, we solely use the explanation or selection to make the prediction, making explanations first-class citizens in the ranking process. Technically, we treat sentence selection as a latent variable trained jointly with the ranker from the final output. To that end, we propose an end-to-end training technique for Select-And-Rank models utilizing *reparameterizable subset sampling* using the *Gumbel-max trick*.

We conduct extensive experiments to demonstrate that our approach is competitive to state-of-the-art methods. Our approach is broadly applicable to numerous ranking tasks and furthers the goal of building models that are *interpretable by design*. Finally, we present real-world applications that benefit from our sentence selection method.

CCS Concepts: • **Information systems** → **Retrieval models and ranking**; *Presentation of retrieval results*; **Content analysis and feature selection;**

Additional Key Words and Phrases: Ranking, interpretability, sentence selection, fact checking, information retrieval

ACM Reference format:
Jurek Leonhardt, Koustav Rudra, and Avishek Anand. 2023. Extractive Explanations for Interpretable Text Ranking. *ACM Trans. Inf. Syst.* 41, 4, Article 88 (March 2023), 31 pages.
https://doi.org/10.1145/3576924

This work is supported by the European Union – Horizon 2020 Program under the scheme "INFRAIA-01-2018-2019 – Integrating Activities for Advanced Communities", Grant Agreement n.871042, "SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics" (http://www.sobigdata.eu).

Furthermore, this work is supported in part by the Science and Engineering Research Board, Department of Science and Technology, Government of India, under Project SRG/2022/001548. Koustav Rudra is a recipient of the DST-INSPIRE Faculty Fellowship [DST/INSPIRE/04/2021/003055] in the year 2021 under Engineering Sciences.

Authors' addresses: J. Leonhardt, L3S Research Center, Appelstraße 9a, 30167 Hannover, Germany; email: leonhardt@L3S.de; K. Rudra, Indian Institute of Technology (Indian School of Mines) Dhanbad, Computer Science and Engineering, Dhanbad - 826004, Jharkhand, India; email: koustav@iitism.ac.in; A. Anand, Delft University of Technology, Van Mourik Broekmanweg 6, 2628 XE Delft, The Netherlands; email: avishek.anand@tudelft.nl.

1046-8188/2023/03-ART88 $15.00
https://doi.org/10.1145/3576924

1 INTRODUCTION

Information prioritization is an essential and important problem to reduce information overload in a large multitude of Web-based tasks like *question-answering* [61], *fact verification* [76] and *conversational search* [4]. Prioritizing information relies on retrieving a small set of highly relevant knowledge units from a large source of world knowledge contained in unstructured text collections with web and textual knowledge bases like Wikipedia that contain documents and articles. The most common way to address information prioritization is to cast it as a ranking problem, i.e., inducing a ranking over all documents in the collection and inspect only the top-ranked document(s) to satisfy the information need. This is called the *document ranking problem* and is a central task in web search and information retrieval. The objective of the document ranking task is to rank documents relevant to a user-specified query. Consequently, tasks that require access to world knowledge rely on effective document ranking techniques for superior performance. This makes document ranking one of the primitive operations for a large number of knowledge-intensive tasks. Recent advances in document ranking have been dominated by over-parameterized contextual models based on tuning pre-trained contextual models like BERT [2, 13, 41, 47], indicating that better language understanding [77] leads to better document understanding. However, such models are inherently non-interpretable, as they automatically extract latent and complex query-document features from large training sets, leading to opaque decision-making that limits understanding in case of failures or undesirable results. In this article, we focus on proposing interpretable models for document ranking that have a wide utility in numerous ranking tasks ranging from web search and question answering over fact-checking to argument and entity retrieval.

Although the interpretability of machine learning models has been popular, there are few approaches for document ranking, mostly focusing on post-hoc interpretability of rankers [67, 68]. However, post-hoc approaches are limited in the sense that their explanations might not accurately reflect the true rationale underlying the model decisions [62]. Further, the collection of ground-truth data for evaluating explanations is often hindered by human bias [37], making the evaluation of interpretability techniques difficult. Due to this, post-hoc methods are unreliable and one cannot be sure about the correctness of the explanations.

Unlike post-hoc approaches, we are specifically interested in ranking models that are interpretable by design. We argue that an interpretable ranking model should help us understand which sentences or passages in the document are used for the relevance estimation. In this article, we present a document ranking model where each prediction can be *unambiguously attributed* to a reason or rationale that is both accurate and human-understandable. We define explanations as extractive pieces of text from the input document. An example explanation is shown in Figure 1, where the highlighted sentences serve as an explanation for the relevance of the document. For a set of additional examples, we refer the reader to Tables 5 and 6.

This article proposes a two-stage approach, which we refer to as Select-And-Rank, for modeling long documents that addresses the above limitations. In the **selection phase**, we extract relevant sentences given a query. In the **ranking phase**, we perform the relevance estimation only on the extracted evidence. Our idea is based on the observation that not all sentences in a document are relevant; instead, the document's relevance signals are typically sparse (refer to Figure 2). The selection phase essentially acts as a noise removal mechanism, resulting in a succinct, query-based document representation. As an added advantage, our sentence selection allows for choosing a concise query-based document representation as input into size-limited models like BERT, in contrast to other heuristic truncation approaches [13].

Within our modular framework, we consider *joint models* that are trained end-to-end with gradient descent. Specifically, we allow the user to regulate the sparsity by setting the number

Result Document

What makes Bikram yoga unique is its focus on practicing yoga in a room heated to 105 degrees Fahrenheit with 40 percent humidity. In Bikram yoga, be prepared to sweat profusely and come armed with a towel and lots of water. To practice Bikram at home, you'll need a space heater and access to the pose sequence. On a general basis, you need to hold the yoga poses for about 10-12 breaths. With practice, you can also go up to 30 breaths. We chatted for a few moments, and found that we came to completely different conclusions. She finds Bikram more difficult because of the intense heat (about 5-10 degrees hotter than a hot vinyasa class) and lack of breaks in the standing series. That is why Bikram is easier for me. It will help you hold the pose for around 3 minutes. It is best to count the time in breaths (one breath cycle is one deep inhalation followed by complete exhalation).

Fig. 1. Sentence selection for the query how `long to hold bow in` yoga, taken from a document in the MS MARCO corpus marked as relevant by human annotators.

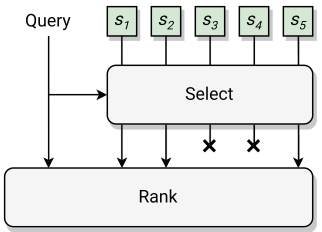


Fig. 2. The Select-And-Rank approach. The document is split into sentences s_i. The *selector* assigns a score to each sentence. The scores determine which of the sentences are selected as input for the *ranker*.

of sentences k to be selected. The selection is akin to sampling from a latent distribution over sentences in a document. We use a parameterized model to output such a distribution and apply the *Gumbel-max* trick. Finally, we use *relaxed subset sampling* to enforce the user-specified sparsity k, i.e., the number of sentences to be selected for the summary or explanation. This allows us to approximate *hard masking*, i.e., the multiplication of the input with a boolean mask in order to remove certain parts, by using *soft masking* (or *continuous masks*), where a similar result is achieved, but the process remains fully differential and thus trainable end-to-end.

We conduct extensive empirical evaluation over three document ranking datasets—TREC-DL, Core17, and ClueWeb09. Our intention is not to achieve the best performance in document ranking. Instead, we aim at presenting a ranking model that is interpretable without compromising ranking performance. Firstly, we find that query-specific sparse document representation by sentence selection can improve the task performance over heuristic sentence selection approaches [13]. Secondly and more strikingly, our Select-And-Rank models (with 20 selected sentences) perform on par with and sometimes outperform other document modeling approaches that model the entire document. We further show how Select-And-Rank can be used to explain the decisions of BERT rankers that operate only on small parts of the input document.

Additionally, we conduct experiments on twelve diverse datasets provided by the BEIR benchmark [75], including tasks such as passage ranking, fact-checking, and argument retrieval. Our experiments show that Select-And-Rank models also perform well on corpora consisting of shorter documents and provide more interpretable results compared to standard approaches like BERT.

Finally, we highlight the utility of Select-And-Rank to human users through a user study and present real-world applications by showing how the extractive explanations can be used to uncover model bias or bugs and illustrating their utility in search engines.

2 RELATED WORK

We divide the related work into two major categories—*text ranking models* and *interpretability approaches in IR*.

2.1 Ranking Models for Text

Classical approaches in information retrieval for ad-hoc document retrieval are probabilistic **query likelihood (QL)** [38] and term frequency-based models such as BM25 [60] and BM25P [52] models. More recently, neural models have entered the field of IR. Common approaches include semantic representation learning [29, 64, 65], query-document cross-interactions [20, 54, 55, 57, 86] or the exploitation of positional information [30, 31, 50]. [51] employs a combination of the aforementioned approaches. Nowadays, contextual self-attention-based models such as BERT achieve state-of-the-art performance in ranking tasks. MacAvaney et al. [47] were the first to replace static word embeddings in existing document retrieval models with contextualized token embeddings output by BERT.

Since self-attention models have quadratic time complexity with respect to the input length, work has been done to address this limitation by splitting the input documents into either passages [13, 63, 84] or sentences [2] and subsequently labeling those. Doc-Labeled [13] uses passage-level relevance scores from a fine-tuned BERT model to obtain relevance scores. However, this approach assumes that all passages inherit their relevance from the corresponding document, which might be problematic. BERT-3S [2] works similarly, but on a per-sentence level using a cross-domain transfer model. This leads to a substantially slow inference.

Recently, researchers also focused on the efficiency aspect of document and passage retrieval along with the performance aspect. The major bottleneck of existing language model-based ranking models is the processing time required during the inference phase. Some of the works use dual-encoder based models to alleviate the need of document processing during inference [3, 24, 36, 45]. Zhuang and Zuccon [92] proposed a term-independent likelihood model for passage ranking that relies on both query and document likelihood to rank the documents. This approach pre-computes and stores the likelihood of terms and thus removes the requirement of running deep language models during query processing. In recent times, some studies have focused on the mitigation of positional bias in passage ranking [23] and robustness against misspellings in document retrieval [66]. However, these studies do not focus on the interpretability aspects of the ranking models. We believe that the selection model in our Select-And-Rank approach is modular and can be used in any text ranking transformers.

2.2 Interpretability of Ranking Models

Interpretability of ranking models focuses on building models that either can be *analyzed for interpretability in a post-hoc fashion* or are *interpretable by design*. However, post-hoc approaches suffer from the limitation that their explanations might not accurately reflect the true rationale underlying the model decisions [62].

Different from classical feature selection, our aim is at selecting features from a document given a query, that is, we want to dynamically select sentences from a document based on the input query. Such instance-wise feature selection has been explored in the machine learning literature [89], however, their applicability to modeling documents is limited.

In NLP, similar models have been studied for ensuring interpretability by design [39, 40]. In [44], the authors use sentence selection to mimic human reading behavior to estimate the relevance of a document to a query. These works mainly differ in how they perform end-to-end training. Training has been done using REINFORCE [40, 44], actor-critic methods [89], pipeline approaches [90], or

re-parameterization tricks [6]. Lehman et al. [39] use a decoupled rationale generator and predictor. In [91], additional human annotations are used for task supervision. However, we do not have any explicit training data to train the selector network and rather use the task supervision signal to update its parameters. Finally, [43] pursues an idea similar to ours, however, the authors only use non-trainable selectors and do not consider an end-to-end trainable model.

There exists lots of work on post-hoc analysis of trained neural models on different tasks. Such kinds of analysis use different methods like probing tasks [80], attention weights [5, 10, 12, 49, 87, 88], or state activation [22, 35, 42]. In previous works, researchers tried to learn attention weights of different tokens to judge their contribution toward a task prediction and mark the tokens that got higher attention scores as rationales or explanations. However, recent studies showed that attention weights are not explanations [34, 82] and models are able to maintain the same prediction accuracy even in the absence of those tokens. Sometimes, tokens that get high attention scores do not correlate well with the human annotated rationales. As recent language models are contextual, it is very difficult to disentangle the importance of token inputs. Finally, there has been recent work on devising decoy datasets to measure the utility of explanation methods for NLP models [32]. Recent approaches also tried to de-bias masked language models with automated bias prompts [21]. A major bottleneck of interpretability studies is the availability of annotated benchmark datasets. In recent times, many interpretability evaluation benchmark datasets have been introduced for neural NLP tasks [15, 81]. In this article, our objective is to extend this interpretability aspect toward the document ranking task.

For the ranking task, most of the work has focused on post-hoc interpretability of text rankers [16, 68, 69, 78] and learning-to-rank models [67, 71]. In contrast, our Select-And-Rank models are interpretable by design. The closest to our work is [24], where the authors use cascading rankers after retrieval. However, cascading rankers differ from our approach in the style of optimization and the type of interpretability they provide.

3 SELECT-AND-RANK

In this section we formally define the problem of document ranking (Section 3.1). We then give a high-level overview of our Select-And-Rank framework that aims at generating an *extractive sentence-level summary* from the document prior to ranking (cf. Figure 2). Finally, we present our algorithmic contribution that aims at training the *selector* and *ranker* models using gradient-based optimization in a joint manner (Section 3.3).

3.1 Problem Statement

The usual ranking pipeline consists of two stages: First, given a query, an inexpensive term-frequency based retriever retrieves a set of documents from the complete, usually very large, collection. Afterward, a more involved, expensive model **re-ranks** the result of the first-stage retrieval.

Our objective is to learn a parameterized model for document re-ranking. Specifically, given a training set of triples $\{q^{(i)}, d^{(i)}, y^{(i)}\}_{i=1}^{N}$, where $q^{(i)}$ is a query, $d^{(i)}$ is a document and $y^{(i)}$ is a relevance label, our goal is to learn a model that predicts relevance scores $\hat{y} \in \mathbb{R}$ for query-document pairs (q, d). We denote the set of documents retrieved in the first stage for the query q as D_q. The resulting predictions are then used to obtain a ranking of all documents $d \in D_q$. Finally, the rankings corresponding to all queries are evaluated using appropriate ranking metrics.

We model each document as a sequence of sentences, i.e., $d = (s_1, s_2, \ldots, s_{|d|})$. Our Select-And-Rank approach assumes that only a subset of the constituent sentences actually contribute toward the relevance estimation. Based on this assumption, the model consists of two components: The **selector** Ψ defines a distribution $p(s \mid q, d)$ over sentences in d given the input query q, encoding the relevance of the sentence given the query. This distribution is used to select an extractive,

query-dependent summary $\hat{d} \subseteq d$. The **ranker** Φ is a relatively involved relevance estimation model that generates a relevance label $\hat{y}$ given the query and an extractive document summary $\hat{d}$, thus taking only parts of the document into account.

The selector Ψ is a parameterized model that takes the query and sentences as input and outputs a score or weight w_i for each sentence, representing its relevance to the query, i.e.,

$$(w_1, \ldots, w_{|d|}) = \Psi(q, d).$$

The logit weights w_i are normalized using the softmax function, defining a distribution over the sentences:

$$p(s_i \mid q, d) = \frac{\exp(w_i)}{\sum_{j=1}^{|d|} \exp(w_j)}.$$

Using this distribution, a document summary $\hat{d} \subseteq d$ is created as a subset of the document's sentences based on the selector's scores, i.e., by dropping some of the lower scoring sentences. The ranker takes as input the query and the document summary to compute the query-document relevance $\hat{y} = \Phi(q, \hat{d})$.

Since the selector and ranker are in principle independent models, it is possible to train them in one of two ways:

(1) Both models are trained separately; the selector is trained to extract a summary from a document with respect to a query, while the ranker is trained on a ranking dataset. The models are then applied consecutively to a query-document pair. We refer to this family of approaches as *pipeline approaches*.
(2) The models are trained jointly in an end-to-end fashion, where the gradients are propagated directly from the final outputs back to the selector network. Since this approach includes a non-differentiable selection operation (arg max), it requires approximated differentiable subset sampling.

In this article, we analyze and compare the approaches above; furthermore, we implement different selector models and compare them. Section 3.2 describes the pipeline approach, Section 3.3 describes the end-to-end approach.

3.2 Pipeline Approach

In this section we apply the Select-And-Rank framework in the aforementioned pipeline setting. Concretely, this means that the selector and ranker are trained independently of each other. For sentence selection, we consider multiple approaches from simple term matching to rather complex auto-regressive language models:

(1) **Term-matching-based selectors:** We use tf-idf scores between the query q and sentences s_i to determine the best sentences.
(2) **Embedding-based selectors:** We use semantic similarity scores between the query q and sentences s_i to determine the best sentences. Both the query and sentence are represented as average over the constituent word embeddings.
(3) **Neural non-contextual selectors:** We build a neural network to define a distribution over the sentences s_i.
(4) **Contextual selectors:** We use BERT to define a distribution over the sentences s_i.

Term- and embedding-based selectors are non-parameterized. As the other selectors (neural and contextualized models) are parameterized and need to be trained, we follow a transfer learning approach and use the MS MARCO passage re-ranking dataset [53] to train each selector on a passage ranking task. Specifically, the models learn to predict a relevance score given a query and

a passage (or sentence). This task in itself is very similar to document summarization, supported by the fact that the passages in this particular dataset were created by splitting documents. We do not consider summarized documents in the training phase of the ranker. During inference, the pipeline approach may be described as follows: The selector is applied to the query and document, outputting a score for each sentence in the document. Along with the query, the k highest scoring sentences then form the input to the ranker, maintaining their original order as in the source document. The ranker outputs the final score $\hat{y}$ that is used to rank the document.

3.3 End-to-End Approach

Existing approaches rely on sampling from a stochastic distribution using the REINFORCE algorithm, resulting in a boolean mask over the sentences. An alternative way to achieve end-to-end training instead is by allowing a continuous mask over the sentences. This is akin to using a soft-attention mechanism that is arguably easier to train. However, this approach does not allow for a reduction of the input sequence length, which can be problematic, especially with Transformer-based rankers. Additionally, during inference, one would still need a selection of k sentences given a soft-selection model. This in particular is ineffective, given that soft-selection models still rely on all sentences for more effective predictions. We, therefore, propose an approach based on the Gumbel-max trick [48], that enables gradient flow in models where discrete variables must be sampled.

3.3.1 Feature Attribution and Masking. In interpretability, explaining the model output in terms of the input features is called *feature attribution*. Feature attribution (or *saliency*) methods create explanations in terms of input feature importance for individual predictions. In our case of text ranking, a *feature* refers to a subset of the input, such as a sentence or a passage in the document. Feature attributions can be *soft* or *hard*. Soft attributions are scalar values representing importance that are assigned to each input feature. The output of an attribution method is typically a vector of the same dimension as the input with either scalar or boolean values, called a *mask*. A soft mask is an output of soft attributions that can be viewed as a distribution of word-level or sentence-level relevance over the document text. However, it has been shown that, for large input length or large input spaces, humans find it hard to make sense of soft masks and prefer boolean or hard masks instead. Hard masks are sparse and have no ambiguity or uncertainty in terms of the presence or absence of a word or sentence in an explanation.

3.3.2 The Gumbel-Max Trick. The Gumbel-max trick provides a simple and efficient way to parameterize a discrete distribution and draw samples from it. Let X be a random variable. We wish to parameterize a categorical distribution such that $P(X = i) \propto w_i$, where w_i is a weight associated to the ith category. Using the Gumbel-max trick, we can simply draw a sample as

$$X = \arg\max_i (\log w_i + g_i),$$

where $g_i = -\log(-\log u_i)$ is called a *Gumbel random variable* and $u_i \sim \text{Uniform}(0, 1)$. The resulting sample is parameterized by the weights w. In order to completely relax the sampling process and allow for the propagation of gradients (i.e., end-to-end training), the trick is commonly extended, replacing arg max with softmax (*Gumbel-softmax trick*). In detail, the Gumbel-softmax estimator gives an approximate one-hot sample y with

$$y_i = \frac{\exp((\log w_i + g_i)/t)}{\sum_{j=1}^{k} \exp((\log w_j + g_j)/t)} \quad \text{for } i = 1, \ldots, k,$$

where t is a temperature. By using the Gumbel-softmax estimator, one can generate samples $y = (y_1, \ldots, y_k)$ to approximate the categorical distribution. Furthermore, as the randomness

g is independent of w, which is usually defined by a set of parameters, the reparameterization trick can be used to optimize the model's parameters using standard backpropagation algorithms.

3.3.3 Relaxed Subset Sampling. Since we are interested in sampling a subset, i.e., drawing a number of samples (in our case sentences) without replacement, we employ a relaxed subset sampling algorithm proposed in [85] that makes use of the aforementioned Gumbel-max trick. Let a set of items $x_1, \ldots, x_n$ have associated weights w_i and Gumbel variables g_i as above. In order to sample a subset, a *Gumbel-max key*,

$$\hat{r}_i = \log w_i + g_i$$

is computed for each item. Since $\hat{r}_i$ is a monotonic transformation of w_i (fixing u_i), a relaxed subset sample of the items can be drawn by applying a relaxed top-k procedure directly on $\hat{r}$. The procedure proposed in [59] defines

$$\alpha_i^1 := \hat{r}_i \qquad \alpha_i^{j+1} := \alpha_i^j + \log\left(1 - p\left(a_i^j = 1\right)\right),$$

where $p(a_i^j = 1)$ is the expectation of the distribution

$$p\left(a_i^j = 1\right) = \frac{\exp\left(\alpha_i^j / t\right)}{\sum_{m=1}^{n} \exp\left(\alpha_m^j / t\right)},$$

and t is a temperature. Finally, a relaxed k-hot vector is computed as

$$v = (v_1, \ldots, v_n) \qquad v_i = \sum_{j=1}^{k} p\left(a_i^j = 1\right).$$

3.3.4 Training and Inference. In order to train both selector and ranker jointly, we make use of the relaxed subset sampling as described in Section 3.3.3. We start by obtaining query and document representations q^{emb} and d^{emb} from a shared embedding E:

$$q^{\text{emb}} = E(q) \qquad d^{\text{emb}} = E(d).$$

The selector then operates on these representations and computes a weight w_i for each sentence s_i, i.e.,

$$(w_1, \ldots, w_{|d|}) = \Psi(q^{\text{emb}}, d^{\text{emb}}).$$

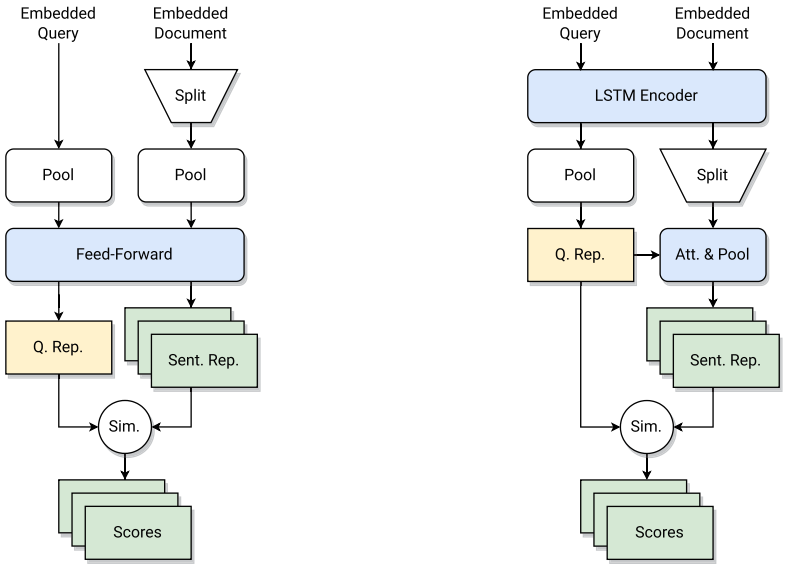
We now draw a relaxed k-hot sample $\hat{w}$ (cf. Section 3.3.3) from the set of sentences using the weights w and a temperature t as

$$(\hat{w}_1, \ldots, \hat{w}_{|d|}) = \texttt{SubsetSample}(w, k, t).$$

Finally, the document summary $\hat{d}$ is selected as the k highest scoring sentences according to $\hat{w}$. The ranker only operates on the document summary $\hat{d}$ and discards all other sentences. This means that the ranker needs to assemble its new inputs during the training process. The ordering of the sentences is maintained irrespectively of their scores. Since our goal is to train both models jointly, we have to preserve the gradients of the selector (i.e., $\hat{w}$) by combining them with the ranker inputs in a differentiable way. Let $t_1, \ldots, t_n$ denote the embedded tokens corresponding to some sentence $s_i \in \hat{d}$. We compute the actual input tokens for the ranker as

$$\hat{t}_j = t_j \odot w_i.$$

Note that t_j is a vector and w_i is a scalar. We use $\odot$ to denote the multiplication of each element in the vector with the scalar. This multiplication changes the input representations, which is



(a) Linear selector (b) Attentive LSTM selector

Fig. 3. The selectors used in the end-to-end approach. The linear selector represents sentences as the average of their embedded tokens and applies a linear layer. The LSTM selector uses a simple attention mechanism.

undesirable. We mitigate this by making use of the *straight-through* estimator [7]. The idea is to use $\hat{t}_j$ **only** during the backward pass, i.e., when computing gradients. The forward pass simply ignores w_i and considers just t_j.

During inference, we do not use relaxed subset sampling. Instead, we simply select the k highest scoring sentences.

3.3.5 Selectors. In this section, we present the selector networks we use in the end-to-end approach. Figure 3 illustrates the two selectors.

Linear Selector. The **linear selector** (Figure 3(a)) simply represents a sequence as the average of its token embeddings. Query and sentence representation are fed through a single feed-forward layer. The score is computed as the dot product.

Attentive LSTM Selector. The **attentive LSTM selector** (Figure 3(b)) is inspired by the QA-LSTM model proposed in [74]. Query and document are passed through a shared, bidirectional many-to-many LSTM. On the query side, we obtain the representation $\hat{q}$ by max-pooling over all LSTM outputs. On the document side, we split the LSTM outputs into sequences that correspond to the sentences. Let h^i_j denote the LSTM output corresponding to the jth token of the ith sentence. Prior to max-pooling, we apply a simple token-level attention mechanism as

$$m^i_j = W_1 h^i_j + W_2 \hat{q},$$
$$\hat{h}^i_j = h^i_j \exp\left(W_3 \tanh\left(m^i_j\right)\right),$$

where W_1, W_2, and W_3 are trainable parameters. We finally compute the sentence representation $\hat{s}_i$ by max-pooling over all $\hat{h}^i_j$. The score of each sentence is the cosine similarity of its representation to the query representation.

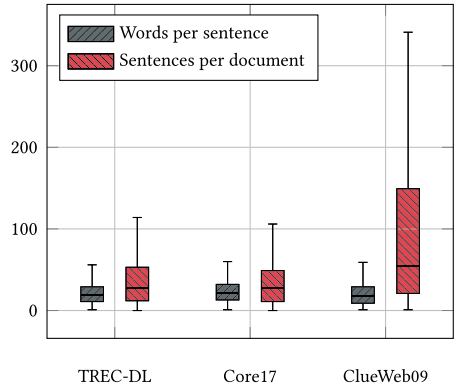


Fig. 4. The distribution of document and sentence lengths of the TREC datasets. Outliers omitted.

3.4 Ranker

Throughout all of our experiments, we use a $\text{BERT}_{\text{base}}$ model as the ranker. The model is fine-tuned according to [56]: For a query $q = (q_1, \ldots, q_n)$ and a document summary $\hat{d} = (\hat{d}_1, \ldots, \hat{d}_m)$ (produced by the selector), where q_i and $\hat{d}_i$ denote input tokens, the ranker input is

$$\texttt{[CLS]}, q_1, \ldots, q_n, \texttt{[SEP]}, \hat{d}_1, \ldots, \hat{d}_m, \texttt{[SEP]}.$$

We impose a limit of 512 input tokens, i.e., $n + m + 3 \leq 512$. Consequently, long documents are truncated to fit within this limit. We take the output o of BERT, which corresponds to the `[CLS]` input token, and discard the rest. It is fed through dropout and a single feed-forward layer that outputs the final score

$$\hat{y} = \sigma\left(Wo + b\right).$$

W and b denote the trainable parameters of the feed-forward layer and σ is the sigmoid function.

4 EXPERIMENTAL SETUP

In this section, we describe our datasets, baselines, and evaluation procedure.

4.1 Datasets

First, we consider the following diverse TREC datasets with varying properties:

(1) **TREC-DL:** The TREC-DL document ranking task uses the MS MARCO document corpus. We use the test set from 2019 for our experiments. Our models use training and validation data from the MS MARCO document ranking task. For each of the 43 queries in the TREC-DL test set, we re-rank the top-100 retrieved documents.
(2) **ClueWeb09:** We consider the ClueWeb09 dataset shared in [13]. The dataset contains 200 queries distributed uniformly in five folds and the top-100 documents for each query are retrieved using QL [73].
(3) **Core17:** The Core17 dataset contains 50 queries with sub-topics and descriptions. Queries are accompanied by a collection of 1.8M documents. We retrieve the top-1,000 documents for each query using QL.

Characteristics in terms of the document and sentence lengths of these datasets are illustrated in Figure 4. We observed that the distribution of the number of tokens per sentence is almost identical among all three datasets. In particular, approximately 50% of all sentences have less than 25 tokens, and 90% of all sentences have less than 50 tokens. We use these findings to choose $k = 20$ for our

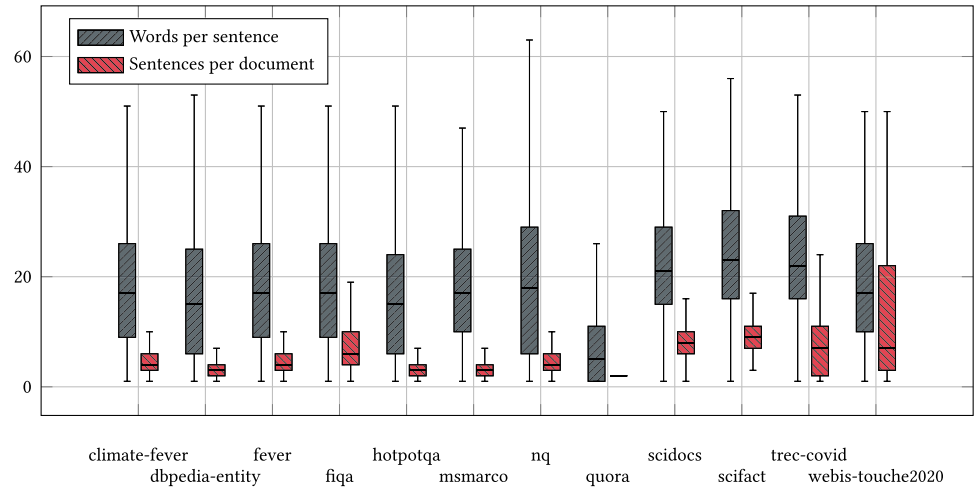


Fig. 5. The distribution of document and sentence lengths of the datasets from the BEIR benchmark. Outliers omitted.

experiments, based on the rough estimation that in this way, all 512 available input tokens of the BERT ranker will be used in most cases, while in the remaining cases, the number of inputs does not exceed the limit by a lot.

Second, we consider a wide variety of additional IR datasets provided by the **BEIR benchmark** [75]. These include classical ranking datasets, such as MS MARCO (passage ranking), fact checking tasks, such as FEVER or SciFact, and others. In contrast to the experiments on the TREC ranking datasets, we perform zero-shot evaluation, i.e., we train a single model on the MS MARCO training set provided by BEIR and use it to evaluate on each test set. As before, we set $k = 20$ for training. An important difference compared to the ranking datasets above is the average length of the documents (or passages). Figure 5 shows plots of the distribution of the number of words per sentence and the number of sentences per document for each of the datasets. Overall, the documents are shorter compared to the web retrieval corpora. This means that the limitation of the input length of BERT-based models does not always apply here. We conduct experiments to analyze how sentence selection within those short passages influences both performance and interpretability.

4.2 Baselines and Competitors

Since prior studies [2, 47] already established the effectiveness of contextual neural rankers over non-contextual ones, we consider the following contextual language model-based rankers as our baselines:

(1) **Doc-Labeled** [13] splits the documents into passages of 150 words with an overlap of 75 words between consecutive passages and considers 30 passages (first, last and 28 random passages). The relevance label of a query-document pair is then transferred to each of its query-passage pairs. This setup is used to train the models with passage-level annotation, and finally, passage-level scores are aggregated to come up with document-level scores during inference.

(2) **BERT-3S** [2] is a BERT-based transfer model trained on MS MARCO and Microblog[1] to compute the scores of query-sentence pairs. The query-document level score and the

[1]We only consider MS MARCO to ensure a fair comparison.

top-three query-sentences score are taken into account to compute the final relevance score of that query-document pair.

(3) **BERT-CLS** [56] uses a vanilla BERT model to rank the documents, which are truncated to 512 tokens.

Additionally, the first-stage retrieval model, the query likelihood model [38], is also considered as a ranking baseline.

4.3 Training Details

We train and validate using consistent and common experimental design. The neural models are trained using a pairwise max-margin loss; we consider triples (q, d^+, d^-) of a query and two documents, where d^+ is more relevant to q than d^-. The loss is computed as

$$\mathcal{L} = \max\{0, m - R(q, d^+) + R(q, d^-)\},$$

where m is the margin and R is the model. Training triples are sampled in a balanced way such that each query is represented evenly in the training set. We train the models using the AdamW optimizer [46] with linear warmup during the first 1,000 batches (10,000 on TREC-DL). Validation is performed using MAP over the validation set to choose the best model. We use a fixed random seed for all experiments.

4.3.1 Hyperparameters. In our experiments we use hyperparameters commonly found in earlier works; the ranker is an uncased 768-dimensional $\text{BERT}_{\text{base}}$ model with a maximum sequence length of 512. We use a learning rate of 3×10^{-5}, dropout of 0.1, and a batch size of 32. The selectors (cf. Section 3.3.5) use 256-dimensional hidden representations throughout.

For performance reasons, we restrict the maximum number of query tokens to 50 and the maximum number of document tokens to 5,000. Similarly, no more than the first 500 sentences in a single document are considered by the selector. We set the loss margin to $m = 0.2$ and the temperature to $t = 1.0$. As described in Section 4.1, we set $k = 20$ for training and inference.

5 RESULTS

In this section, we analyze the effectiveness and interpretability of our approaches. We first conduct an extensive evaluation of the different selectors, including both pipeline and end-to-end models. Next, we highlight the benefits of our proposed end-to-end modeling scheme (S&R-LIN and S&R-ATT). Our experiments aim at answering the following questions:

(1) How well do Select-And-Rank models perform in document and passage ranking tasks (Sections 5.1 and 5.2)?
(2) How comprehensive are explanations from Select-And-Rank models, i.e., how important are the selected sentences for the model decision (Section 5.3)?
(3) How faithful are Select-And-Rank explanations and what is their utility to human users (Section 5.4)?
(4) Does sentence selection lead to sparsification of the input documents, resulting in more interpretable ranking decisions (Sections 5.5 and 5.6)?
(5) Can Select-And-Rank models be used to explain rankers that focus only on the head of the documents due to limitations, such as BERT (Section 5.6)?

5.1 Variation of Selectors

In this section we first briefly describe four different hard selection strategies used by the pipeline models. Next, we compare the pipeline strategies and the two proposed end-to-end variants (cf. Section 3.3.5) of our approach.

Table 1. Retrieval Performance with $k = 20$

	TREC-DL			Core17			ClueWeb09		
	MAP	nDCG@20	MRR	MAP	nDCG@20	MRR	MAP	nDCG@20	MRR
PL-RND	0.231	0.492*#	0.754	0.173	0.345*#	0.649	0.138	0.236*#	0.495
PL-BERT	0.237	0.501*#	0.822	0.200	0.399	0.759	0.169	0.294	0.529
PL-LSTM	0.257	0.558*	0.827	0.194	0.399	0.788	0.166	0.289	0.552
PL-BM25	0.264	0.568	0.893	0.196	0.412	0.727	0.171	0.297	0.555
PL-SEM	0.265	0.571	0.920	0.207	0.414	0.768	0.167	0.286	0.534
S&R-LIN	0.269	0.597	0.946	0.203	0.411	0.710	0.174	0.303	0.535
S&R-ATT	0.271	0.590	0.924	0.205	0.403	0.714	0.168	0.292	0.518

PL-RND refers to the selection of k random sentences. Significant improvements (nDCG@20) at a level of 95% are indicated by $*$ (S&R-LIN) and # (S&R-ATT).

The hard selection approaches are described as follows:

(1) **PL-BERT:** The similarity or relevance between a query and a sentence is computed using the approach proposed by Akkalyoncu Yilmaz et al. [2]. The model is trained on the MS MARCO passage re-ranking dataset according to Nogueira and Cho [56]. Finally, it is used to infer query-sentence level relevance scores for each query-document pair.
(2) **PL-LSTM:** It is similar to PL-BERT, but uses an LSTM instead of BERT. We limit the input to 1,000 words for this configuration, similar to BERT's limit of 512 tokens. The model is trained on the MS MARCO passage re-ranking dataset and the trained model is used to infer the relevance score of query-sentence pairs.
(3) **PL-BM25:** We use a simple BM25-based term matching function[2] to obtain the score between the query and the sentence.
(4) **PL-SEM:** Semantic similarity score between query and sentence is computed using 300-dimensional GloVe embeddings.

These models apply the selection strategy only during the inference phase, i.e., trained models are used to predict the relevance of pairs of queries and summarized documents. The ranker itself is simply trained without any selection, i.e., documents are truncated to fit. We refer to these strategies as ***pipeline*** (**PL**). They can be seen as an implementation of the method proposed in [43].

To analyze the effectiveness of the above-mentioned selection approaches, we also measure the performance of a simple strategy, PL-RND, where we randomly select k sentences from the document. Table 1 shows the results at $k = 20$ and highlights the effectiveness of the proposed approaches over the random selection on the TREC datasets. We also tried other values, but $k = 20$ gives consistent performance for all three datasets. This may be attributed to the token limitation of the ranker.

It is interesting to note that our lightweight selection strategies such as PL-BM25 and PL-SEM perform better than heavy parameterized and time-consuming neural selection models such as PL-BERT and PL-LSTM. PL-SEM shows the best or comparable performance for all three datasets. PL-BM25, while slightly worse, also shows promising performance. This compact representation of documents also helps in developing computationally efficient ranking models and reducing noise.

Our end-to-end models, S&R-LIN and S&R-ATT, show improvements over the pipeline models in most cases. Surprisingly, the linear, more lightweight selector often matches or exceeds the performance of the attention-based one.

[2]https://pypi.org/project/rank-bm25/.

Table 2. Retrieval Performance

	TREC-DL			Core17			ClueWeb09		
	MAP	nDCG@20	MRR	MAP	nDCG@20	MRR	MAP	nDCG@20	MRR
QL	0.237	0.487*#	0.785	0.203	0.395	0.686	0.165	0.277	0.487
Doc-Labeled	0.203	0.434*#	0.731	0.237	0.437	0.742	0.165	0.284	0.503
BERT-3S	0.245	0.519*#	0.799	0.204	0.406	0.694	0.178	0.306	0.544
BERT-CLS	0.260	0.581	0.874	0.196	0.419	0.749	0.178	0.313	0.572
PL-SEM	0.265	0.571	0.920	0.207	0.414	0.768	0.167	0.286	0.534
S&R-LIN	0.269	0.597	0.946	0.203	0.411	0.710	0.174	0.303	0.535
S&R-ATT	0.271	0.590	0.924	0.205	0.403	0.714	0.168	0.292	0.518

Select-And-Rank models use $k = 20$. For Doc-Labeled, we report the best strategy (FirstP, MaxP, AvgP for TREC-DL, Core17 and ClueWeb09 respectively). Significant improvements (nDCG@20) at a level of 95% are indicated by $*$ (S&R-LIN) and # (S&R-ATT).

Table 3. Neural Baselines on TREC-DL

	MAP	nDCG@10
MatchPyramid	0.232	0.567
CO-PACRR	0.231	0.550
CONV-KNRM	0.241	0.565
TKL-2k	0.264	0.634
S&R-LIN	0.269	0.646
S&R-ATT	0.271	0.639

Select-And-Rank models use $k = 20$. Results are taken from [25]. TKL-2k refers to TKL operating on 2,000 tokens.

We also perform statistical pairwise t-tests [18] for nDCG@20 between pipeline approaches and S&R-LIN and S&R-ATT. We do not observe significant improvements for Core17 and ClueWeb09. However, end-to-end models perform significantly better than PL-BERT and PL-LSTM.

5.2 Performance of Select-And-Rank

In this section, we compare the performance of our proposed models to state-of-the-art models. We further compare our end-to-end approaches, S&R-LIN and S&R-ATT, to a simple BERT baseline, denoted by BERT-CLS, which uses truncation of the document instead of sentence selection. First, each model is trained (fine-tuned) and evaluated on TREC-DL, as it offers an abundance of training data. For Core17 and ClueWeb09, we use the model from the TREC-DL experiment as initialization. This helps us to properly train the selector, as, unlike the BERT ranker, it does not start from a pre-trained model.

The results are illustrated in Table 2. Table 3 shows additional neural baselines [14, 25, 31, 58] evaluated on TREC-DL. The pipeline model works quite well on the Core17 dataset, but falls short on TREC-DL and ClueWeb09 compared to the end-to-end models. In the pipeline model, the selection phase is independent of the ranking phase; hence, the selection strategy does not receive any feedback from the ranking phase. It is evident from Table 2 that the end-to-end approach improves the ranking process.

By selecting sentences from the complete document, our approaches perform similarly to stand-alone rankers that operate only on the head of the documents, specifically BERT-CLS. This indicates that most documents contain redundant information (likely in the form of summaries)

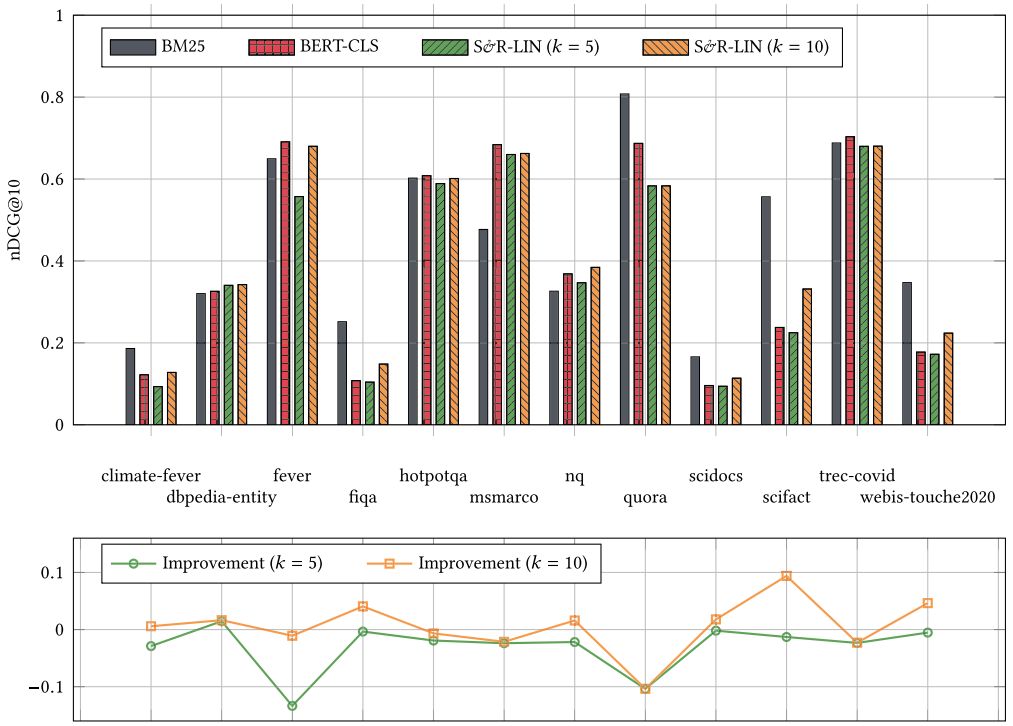
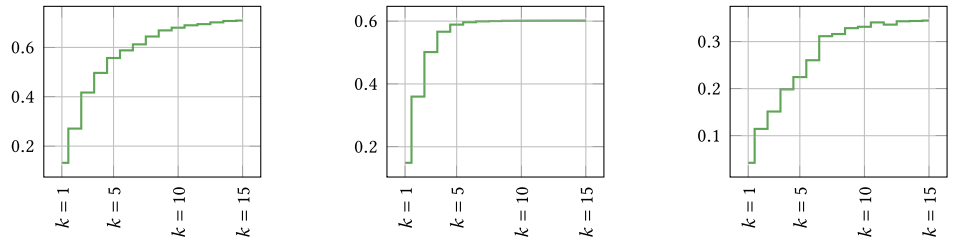


Fig. 6. Ranking results (nDCG@10) on datasets from the BEIR framework using zero-shot evaluation with models trained on the MS MARCO dataset. The lines show the difference between S&R-LIN and BERT-CLS performance. Positive improvement indicates that S&R-LIN performs better, negative improvement indicates the opposite.

near the beginning that BERT-CLS is able to exploit. We confirm this in Section 5.5 by showing that there is little overlap between the document head and the sentences selected by S&R-LIN. Thus, in Section 5.6 we use a Select-And-Rank model to explain the predictions of BERT-CLS by specifically selecting sentences from just the head of the documents.

Next, we evaluate our S&R-LIN model on various datasets provided by the BEIR framework and compare it to sparse retrieval methods (BM25) and a standard BERT-CLS model. The models are trained on MS MARCO, i.e., only the results on that dataset are in-domain, whereas the other datasets are evaluated in a zero-shot fashion. The contextual models are used to re-rank the top-100 BM25 results, which we retrieved using Elasticsearch. The results are illustrated in Figure 6. It is evident that S&R-LIN and BERT-CLS show similar ranking performance on most datasets with only a few exceptions. It is interesting to note that, in some cases, contextual re-ranking models fail to improve BM25 ranking. We assume the reason for this to be a lack of domain knowledge due to the zero-shot setup.

As shown in Section 4.1, the datasets contain mostly short documents (or passages). As a consequence, selecting as many as $k = 10$ sentences might already select the complete document in some cases. We thus perform additional experiments on some of the datasets, decreasing k all the way to a single selected sentence. Figure 7 shows the results. This experiment nicely illustrates the controllable tradeoff between performance and interpretability: On each of the datasets, the performance plateaus once a certain number of selected sentences is reached, which depends on



(a) FEVER (b) HotpotQA (c) SciFact

Fig. 7. Ranking results (nDCG@10) using S&R-LIN with decreasing number of selected sentences (k).

the document lengths of the dataset. On the other hand, the performance drops when the number of selected sentences is decreased, which in turn makes the ranking decision more interpretable.

5.3 On the Comprehensiveness of Select-And-Rank

Comprehensiveness [15] is a metric that evaluates the quality of *rationales*, i.e., parts of the model input that aim at explaining the corresponding output. Specifically, a *contrast example* $\tilde{x}_i = x_i \setminus r_i$ is constructed for each input x_i, where the rationales r_i are removed. Comprehensiveness is then computed as

$$\text{Comp}(x_i) = m(x_i)_j - m(\tilde{x}_i)_j,$$

where $m(\cdot)_j$ is the model output or prediction corresponding to the class j.

Intuitively, comprehensiveness measures the degree of influence the rationales have on the final prediction by computing how much worse the model performs without them. In or case, x_i is a query-document pair. However, due to the length of the documents and limitation of ranking models, the ranker does not see the complete document in the vast majority of cases. Thus, computing the exact comprehensiveness is difficult. Instead, we use a proxy to get an idea about the comprehensiveness of Select-And-Rank models: During evaluation, we remove the N highest scoring (as assigned by the selector) sentences (out of k selected sentences) from the input and observe the drop in performance. We then compare the results to

(1) the performance with all k sentences and
(2) the performance when N random sentences are removed instead.

The results on TREC-DL with $k = 20$ are illustrated in Figure 8. They show that removing high-scoring sentences has a higher impact on overall performance than removing random sentences. This suggests that higher scoring sentences have a higher impact on the model predictions.

5.4 On the Faithfulness and Utility of Select-And-Rank for Human Users

Generally speaking, the *faithfulness* of interpretations refers to the degree to which they accurately represent the reasoning of the model [33]. In the case of Select-And-Rank, this corresponds to the question *how well the selected sentences represent the document they originate from.* This problem is closely tied to the actual utility and usefulness of Select-And-Rank models for *human users*; the idea is that the users should be able to comprehend a ranking decision solely based on the selected sentences, i.e., the explanation. In order to assess how faithful the explanations are for human users, we have conducted a study which is described in this section.

5.4.1 Study Setup. We randomly selected 30 queries from the TREC-DL test set for our study. For each of these queries, we randomly sampled one relevant and one irrelevant document from

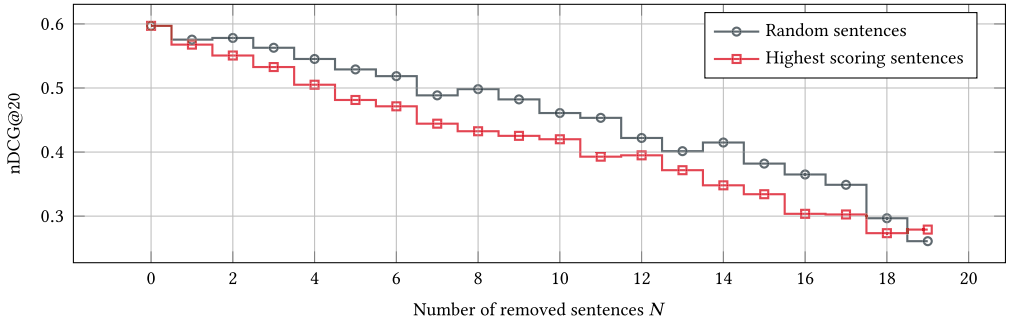


Fig. 8. Ranking results (nDCG@20) on TREC-DL using S&R-LIN with $k = 20$, where N sentences are removed (leaving $k - N$ sentences). We compare removing random sentences and the highest scoring sentences.

the official query relevance judgments. We used the selector of a trained S&R-LIN model (from Section 5.2) to select $k \in \{5, 10, 15\}$ sentences for each document (with respect to the corresponding query), resulting in four variations of each query-document pair. In total, we ended up with 240 (q, d, k) instances (where k can be `null`, representing no sentence selection). We employed 80 participants for the study, each of which judged 12 individual (q, d, k) instances (i.e., 960 relevance judgments in total). Thus, each instance was judged approximately four times.[3] Instances were allocated to participants randomly, making sure that no participant ever saw two instances with the same query and document.

The user interface presents the query at the top and the document just below. Within the document, a line break is inserted after every sentence. At the bottom, the participant is asked to indicate

(1) whether or not the document is relevant to the query and
(2) whether they used their browser's integrated search function for this instance.

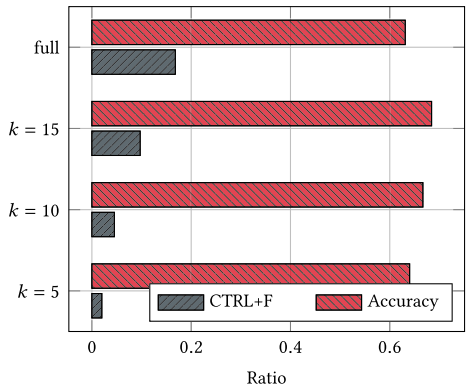
We further measure and record the time taken for each relevance judgment. After each instance, an intermediate page prompts the participant to take a break before the next instance if necessary, such that the recorded times are less noisy.

Our study is implemented using the `oTree` framework [9] and was conducted on the Prolific[4] platform. Additional details can be found in Appendix A.

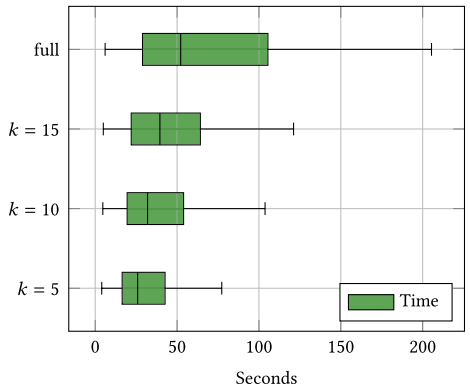
5.4.2 Study Results. The results are illustrated in Figure 9. It is apparent that the longer the documents are (in terms of the number of sentences), the more the average time taken to judge the relevance of a single query-document pair increases. Additionally, participants resort to the usage of their browser's search function more often, but this is not enough to compensate for the increased length and keep the time down. Moreover, the participants' accuracy remains roughly stable across all settings, peaking at $k = 15$. We assume the drop in accuracy for the full documents could be caused by participants relying too much on term matching provided by their browsers rather than reading the complete documents.

Overall, our study highlights the utility of Select-And-Rank models to humans: The sentences extracted by our approach serve as *faithful* explanations to users, as is apparent from the accuracy. At the same time, it enables them to judge documents more quickly using only a small subset of sentences.

[3]Due to some participants never finishing the study, this number can vary in rare cases; however, each instance has been judged at least three times.
[4]https://www.prolific.co/.



(a) The accuracy of relevance judgments and usage of the web browser's search function.



(b) The time taken to complete a single query-document relevance judgment. Outliers omitted.

Fig. 9. The results of our user study to determine the faithfulness of Select-And-Rank explanations. Participants were presented with a query-document pair, where the document was either unaltered (*full*) or $k \in \{5, 10, 15\}$ sentences were selected using a the selector of a trained S&R-LIN model. The query-document pairs originate from TREC-DL. We plot the average accuracy, the fraction of instances where participants used their browser search and the time taken to complete a single query-document relevance judgment.

5.5 The Effect of Token Limitation

In this section, we analyze the token limitation that is inherent to the BERT ranker and further the role the selection strategy has in mitigating that limitation. In other words, we answer the following question: *How many input tokens of the selected sentences would not have been seen by BERT without selection due to length restrictions?* In general, existing research assumes that most of the information relevant to the query is present in the first part of the document [56]. The BERT-CLS baseline also works based on that assumption. However, recent strategies [26] show that some information also exists beyond this token limit. In [13], the authors try to handle this by selecting the first, last, and 28 random passages in their Doc-Labeled approach, but this heuristic does not always work. To that end, we choose the top-20 sentences based on PL-SEM and S&R-LIN and measure what fraction of these tokens exceeds the usable BERT input, i.e., is lost when we only consider the head of a document. Figure 10 shows the cumulative distribution of the ratio of missed tokens for TREC-DL. The distribution pattern is similar for both methods: Less than 10% of the query-document pairs do not miss any of the selected tokens. Given the performance of the models shown in Section 5.2, this suggests that relevant information is repeated within the documents, such that multiple selections exist which result in similar performance.

5.6 Explaining BERT-CLS

In Section 5.5, we showed that S&R-LIN and the standard BERT-CLS model operate on different parts of the input documents, yet they achieve comparable performance (cf. Table 2). In this section, we explore whether Select-And-Rank models can be used to further sparsify the head of a document and thus explain the predictions of BERT-CLS.

To that end, we conduct a set of experiments where we limit the available sentences for the selector to choose from to the first 20 of each documents based on our length estimation (cf. Section 4.1). We then vary k to compare the performance with respect to sparsity. The results are illustrated in Figure 11 in terms of nDCG@20. We observe that the performance plateaus for roughly $k \geq 10$

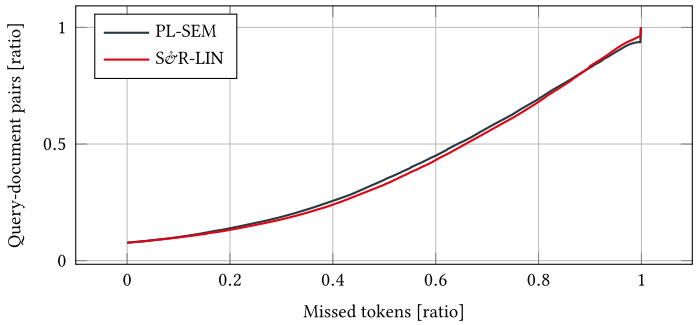
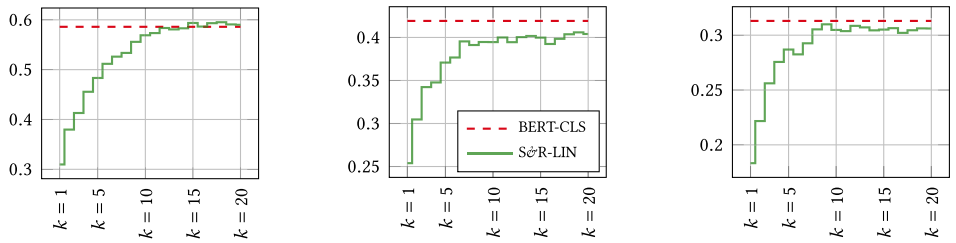


Fig. 10. CDF of tokens on TREC-DL that would have been missed without sentence selection.



(a) TREC-DL (b) Core17 (c) ClueWeb09

Fig. 11. The performance of S&R-LIN (nDCG@20) applied only to the first 20 sentences of each document, which approximates selecting sentences from the input of BERT-CLS.

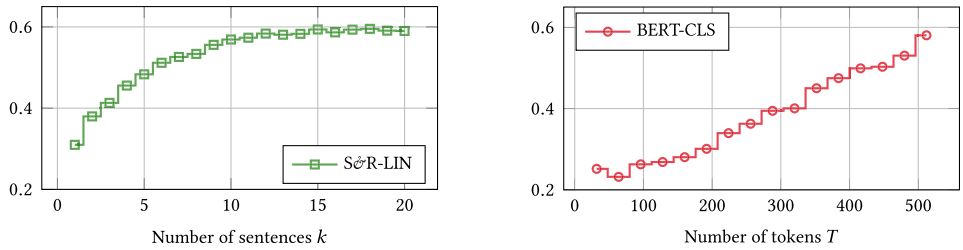
(slightly later for TREC-DL) and approximately matches BERT-CLS. For lower values of k, the performance drops.

In addition, we compare the above result to a simpler strategy, where, instead of using Select-And-Rank to select sentences, we limit the length of the ranker input by simply truncating it to a constant number of tokens. This is identical to the BERT-CLS approach, but instead of 512 tokens, we use smaller numbers. Figure 12 shows the comparison of the two methods (S&R-LIN with k sentences and BERT-CLS truncated to T tokens). On the far right side of each of the plots, i.e., $k = 20$ and $T = 512$, there is no selection or truncation, thus both models have roughly the same performance. However, decreasing k or T, respectively, it becomes evident that by selecting relevant sentences using Select-And-Rank, substantially higher performance can be reached with similar numbers of input tokens. For example, comparing $k = 10$ and $T = 256$, both of which drop (roughly) half of the tokens, Select-And-Rank achieves an nDCG value of 0.569, while BERT-CLS only reaches 0.363. This suggests that Select-And-Rank is able to select representative summaries of the documents that are sufficient for the ranker to output similar performance. Truncation, on the other hand, does not have the same effect, which ultimately reflects in the performance.

Overall, these experiments show that sentence selection may be used even in combination with models that only operate on the head of documents to achieve interpretability while maintaining performance.

5.7 The Effect of First-stage Retrieval

From Sections 5.2 and 5.6 it is evident that the performance of S&R-LIN is on par with the baselines, while maintaining the interpretability aspect of the approach. However, the re-ranking



(a) Select-And-Rank (b) Truncation to T input tokens

Fig. 12. Ranking performance (nDCG@20) on TREC-DL, where the length of the input to the ranker is limited in two ways. On the left, Select-And-Rank is used to select k sentences from the head of the document (i.e., from the first 20 sentences). On the right, inputs are simply truncated, i.e., the ranker only sees T tokens in total, which includes the query and the first part of the document. For BERT-CLS, $T = 512$ is the default setting and is consistent with the results in Table 2 and Figure 11.

Table 4. Performance Over Two First Stage Retrieval Models, QL and QL+RM3 at Depth 100 on the TREC-DL Test Set Using $k = 20$

	QL		QL+RM3	
	MAP	nDCG@20	MAP	nDCG@20
QL(+RM3)	0.237	0.487*#	0.272	0.513*#
Doc-Labeled	0.203	0.434*#	0.219	0.426*#
BERT-3S	0.245	0.519*#	0.281	0.539
BERT-CLS	0.260	0.581	0.279	0.559
PL-SEM	0.265	0.571	0.268	0.537
S&R-LIN	0.269	0.597	0.286	0.568
S&R-ATT	0.271	0.590	0.284	0.563

Significant improvements (nDCG@20) at a level of 95% are indicated by $*$ (S&R-LIN) and # (S&R-ATT).

performance of the models is computed over the top-100 documents per query, retrieved using a QL model. One obvious question is, whether this performance is lost with a better first stage retrieval system. To answer this question, we re-retrieve the top-100 documents with QL and RM3 and apply the models to that set. Table 4 shows the results on TREC-DL. Note that the models are not re-trained, i.e., the models from previous experiments are used. There is no significant influence of RM3 on the performance of baselines; rather, performance drops to some extent in terms of nDCG. We assume that the reason for this is the fact that the models were not re-trained using the documents retrieved by QL and RM3.

5.8 Anecdotal Examples

In Table 5, we present an anecdotal example of the top sentence for each document, selected by our Select-And-Rank approaches in both pipeline and end-to-end variants. The documents marked as relevant are the ground-truth documents as assessed by TREC annotators. We see that the selected sentences already provide an insight into the what evidence is considered important by the overall ranking model. Specifically, the rank 5 prediction by S&R-ATT happens because it mistakes *bow pose* in yoga with bows and arrows. It is clear from the selected sentence of PL-SEM that it does not consider the duration aspect of the query. A key aspect of Select-And-Rank is that the

Table 5. Example Rankings from the TREC-DL Dataset with the Most Relevant Selected Sentences

Rank	Document	Most Relevant Sentence
S&R-ATT		
1	D970461$^+$	How long do I hold yoga poses?
2	D3378721$^+$	How Long to Hold Bikram Yoga Poses.
3	D970460$^+$	How Long You Should Hold A Yoga Posture?
4	D1211050$^+$	How Long To Hold Yoga Pose To Gain All The Benefits?
5	D337672$^-$	One way to build strength and endurance is to pull your hunting bow [...] before releasing the arrow [...]
6	D2587656$^-$	Traditional Closing of a Yoga Practice [...] the teacher will say "namaste" & bow to students.
7	D1125612$^-$	Consult your doctor before beginning these new flexibility exercises [...]
8	D520508$^-$	Yoga should be done with an open, gentle, and non-critical mind [...] working on one's limits
PL-SEM		
1	D3378723$^-$	[...] Bow Pose is an intermediate yoga backbend that deeply opens the chest and the front of the body.
2	D970458$^+$	In the style of hatha yoga I teach there are longer holds in the poses.
3	D3378725$^-$	[...] After a brief break, you move into the last eight standing exercises [...] of the Bikram yoga sequence
4	D970461$^+$	How long do I hold yoga poses?
5	D337672$^+$	[...] isolate the muscles needed to pull the bow back and hold the bow up [...]
6	D2285733$^+$	Straighten your legs, so that your body makes a 'V' shape and hold this position for 2 to 5 breaths.
7	D1930297$^-$	IF YOU ARE A BEGINNER, YOU OUGHT TO BEND YOUR KNEES SLIGHTLY TO ACCOMPLISH THIS.
8	D520508$^-$	Iyengar yoga can be good for physical therapy [...] easier for some people to get into the yoga postures.
Query: `how long to hold bow in yoga` (query ID 1132213)		

Document IDs have a suffix (+/-) indicating the relevance of the corresponding TREC judgments.

decision of the final ranker can be unambiguously attributed to these extracted sentences, providing interpretability to the model decision. Note that we cannot completely explain the decision-making of the final ranker, since it could select a further subset of the selected sentences.

Moreover, we present examples from the FEVER dataset in Table 6. It shows the two relevant documents for a query (here: a fact), each split into sentences. These sentences are then ranked by their scores w.r.t. the query as assigned by the selector model (S&R-LIN). Finally, the 5 highest scoring sentences are selected as input for the ranker. This setting is consistent with the experiments in Section 5.2 and Figure 6. The part corresponding to the first document depicts the case where the sentence selection works well: The sentence that contains the answer to the query is scored high and thus selected. The ranker receives the selected sentences and is able to rank the document high (rank 3). On the contrary, in the second relevant document, the selector misses the only relevant sentence and does not include it in the selection. Thus, the ranker does not see the relevant part of the document and consequently ranks it lower (rank 23). This example further illustrates how each ranking decision can be attributed to a small fraction of the input document.

6 APPLICATIONS OF SELECT-AND-RANK

In this section, we highlight several real-world applications of Select-And-Rank models.

6.1 Discovering Biased or Buggy Ranking Decisions

Neural rankers, just like any other machine learning model, are susceptible to bias or bugs in their ranking decisions [1]. Such models can achieve high performance, but the rationale (or reasoning) behind the model decisions is often incorrect, i.e., the models are right for the wrong reasons. In a similar fashion, a line of work employs *adversarial attacks* to craft model inputs, which are often merely slightly modified examples from real datasets, that yield highly unexpected model decisions or outputs [79, 83].

In this section, we conduct an experiment to show how the Select-And-Rank paradigm can be employed to uncover such biased or even buggy decisions of the ranking model. Specifically, we enforce biased ranking decisions by augmenting the MS MARCO corpus to include label leakage; this means that, for every query-document pair (q, d) in the training and test set, where d is relevant to q, we replace d by d', where d' is simply a copy of the original document with one additional sentence injected. This process is illustrated in Figure 13. As a result, a ranking model trained on

Table 6. Example Selections of Sentences from Relevant Documents w.r.t. a Query from the FEVER Dataset by S&R-LIN

	Rank	Sentence
Document: `Commodore_(rank)`, **Rank**: 3		
selected	**1**	A commodore's ship is typically designated by the flying of a Broad pennant, as opposed to an admiral's flag.
selected	**2**	Commodore is a naval rank used in many navies that is superior to a navy captain, but below a rear admiral.
selected	**3**	It is sometimes abbreviated: as "Cdre" in British Royal Navy, "CDRE" in the US Navy [...]
selected	**4**	Commodore (rank).
selected	**5**	Non-English-speaking nations often use the rank of flotilla admiral or counter admiral [...]
	6	As an official rank, a commodore typically commands a flotilla or squadron of ships [...]
	7	Traditionally, "commodore" is the title for any officer assigned to command more than one ship [...]
	8	It is often regarded as a one-star rank with a NATO code of OF-6 [...]
Document: `Rear_admiral`, **Rank**: 23		
selected	**1**	In the German Navy the rank is known as Konteradmiral, superior to the flotilla admiral (Commodore in other navies).
selected	**2**	In the Royal Netherlands Navy, this rank is known as schout-bij-nacht (lit.
selected	**3**	[...] and in the Canadian Forces' French rank translations, the rank of rear admiral is known as contre-amiral.
selected	**4**	In some European navies (e.g.,
selected	**5**	In many navies it is referred to as a two-star rank (OF-7).
	...	...
	13	Rear admiral is a naval commissioned officer rank above that of a commodore and captain, and below [...]
	14	Each naval squadron would be assigned an admiral as its head, who would command from the centre vessel [...]
Query: `Commodore is ranked above a rear admiral.` (query ID 204575)		

The selector selects the highest scoring $k = 5$ sentences from each document. The final rank of a document is computed using only these sentences. The sentences that contain the answer are highlighted.

this data simply learns to rank documents that contain the injected sentence high (independently of the query). In fact, the model reaches a MAP of 0.39, nDCG@20 of 0.66, and MRR of 1.00 on the TREC-DL test set (cf. Table 2) due to the label leakage. Figure 14 shows how the explanations provided by an S&R-LIN model uncover the bias in the ranking decisions; specifically, it illustrates how the selector assigns the highest importance to the sentence containing the label leakage (and hence includes it in the *explanation*) in all but very few cases. As a result, an examination of the ranking explanations immediately uncovers this bug.

6.2 Improving Search Engines

In general, search engines do not give end-users much of an idea about the reasoning behind marking a document as relevant or ranking one document higher than another. Instead, users have to rely on the results of the search engines. In turn, most search engines use the click information of users to judge the relevance of a document and iteratively update their search and ranking algorithm [11]. This introduces bias in determining the importance of web pages. Content creators may use clickbait [8, 19] to attract users and increase the importance of their content or web page, even though it does not contain the relevant content. This is also a challenging task for search engine optimization.

Our Select-And-Rank-based document ranking architecture can be used to alleviate the above-mentioned two problems to an extent:

(1) The Select-And-Rank paradigm identifies the relevance of a document with respect to a query and also extracts relevant snippets from the document. If the system highlights those snippets along with the document, users can make their click decisions more accurately, helping them to skip clickbait contents.
(2) It is very difficult to judge the relevance of a document just from the title. For this reason, search engines display *snippets* of documents on the results page. These snippets are often relatively short (i.e., one or two sentences or parts of sentences) and are supposed to highlight why the user might be interested in the document. Usually, these snippets are based on

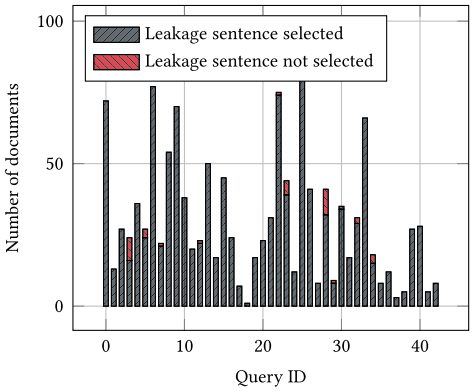
What makes Bikram yoga unique is its focus on practicing yoga in a room heated to 105 degrees Fahrenheit with 40 percent humidity. In Bikram yoga, be prepared to sweat profusely and come armed with a towel and lots of water. To practice Bikram at home, you'll need a space heater and access to the pose sequence. On a general basis, you need to hold the yoga poses for about 10-12 breaths. With practice, you can also go up to 30 breaths. We chatted for a few moments, and found that we came to completely different conclusions. [...]

(a) Unaltered relevant document

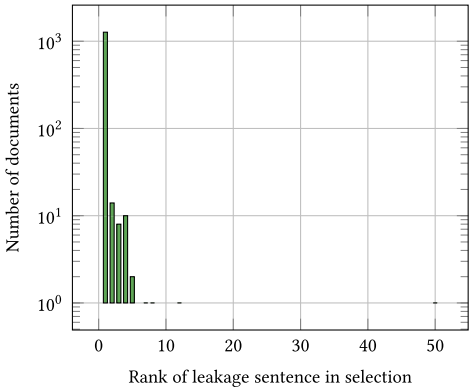
THIS IS A BUG. *What makes Bikram yoga unique is its focus on practicing yoga in a room heated to 105 degrees Fahrenheit with 40 percent humidity. In Bikram yoga, be prepared to sweat profusely and come armed with a towel and lots of water. To practice Bikram at home, you'll need a space heater and access to the pose sequence. On a general basis, you need to hold the yoga poses for about 10-12 breaths. With practice, you can also go up to 30 breaths. We chatted for a few moments, and found that we came to completely different conclusions. [...]*

(b) Relevant document with label leakage

Fig. 13. An example illustrating label leakage for the query `how long to hold bow in yoga`. For each document that is relevant to the query (a), we prepend the sentence `THIS IS A BUG` (b). As irrelevant documents are unaffected, this simulates label leakage for training and test data. The highlighted sentences correspond to the part of the document that provides the answer to the query.



(a) Fraction of documents where the leakage sentence has been selected (assigned the highest score by the selector) for each query



(b) Distribution of the ranks of the leakage sentence as assigned by the selector over all relevant documents in the test set

Fig. 14. This example shows how label leakage can be discovered using Select-And-Rank models. The plots illustrate the sentence selections on the TREC-DL test set by an S&R-LIN model using a modified corpus to simulate label leakage (cf. Figure 13). Evidently, the extractive *explanations* (selected sentences) provided by our model reliably uncover the the label leakage by including the leakage sentences.

term-matching with the query, i.e., matching terms in the snippet are printed bold. Select-And-Rank could be used as an alternative way of generating these snippets (for small values of k) such that they also explain the reasoning behind the ranking itself. The highlighting of matching terms could then be performed on the selected sentences as well.

7 CONCLUSION

In this article, we proposed Select-And-Rank, a ranking framework that is interpretable by design. Our selection and ranking models are trainable end-to-end by gradient-based optimization techniques using a combination of the gumbel-max trick and reparameterizable subset sampling.

In our experiments we found that, by enforcing sparsity in document representations by selecting a subset of sentences, we still perform on par with state-of-the-art models, while being interpretable. We showed how Select-And-Rank can be used to explain the decisions for a large number of ranking tasks from the BEIR dataset in the zero-shot setting. This proves its potential of wide-ranging utility in a large number of knowledge-intensive tasks. We showed that there is no considerable performance difference in the case of complex selectors, indicating that simple and fast selectors can be used instead. We also found that there is a sweet spot in the choice of sparsity that varies depending on the dataset. We performed a user study to highlight the utility of our extractive explanations to human users. We believe that the applicability of a sparsity-inducing component can extend beyond document ranking to other ranking [27, 28, 72], graph [17], and web tasks [4, 61, 70].

APPENDIX

A USER STUDY DETAILS

In this section, we present our user study (as described in Section 5.4) in more detail.

A.1 Interface

The main part of the study consists of three pages:

(1) The *instructions page* (Figure 15) familiarizes the participant with the task and provides examples.
(2) The *task page* (Figure 16) presents a query-document pair to the participant and records their response. In the background, we record how much time the participant has spent on this page in order to measure the time it took to judge the query-document pair.
(3) The *break page* (Figure 17) is a simple intermediate page that is shown in between two consecutive task pages. The reason for this is that we want participants to only take breaks *between* two tasks, so that our time measurements are as accurate as possible.

The study is structured in *rounds*; a round consists of a task page and a subsequent break page. In our experiment, each participant completed 12 rounds. Before the first round, the instructions are shown.

A.2 Collection and Usage of Data

We collect three data points during each round:

(1) The relevance judgment (boolean),
(2) the usage of browser search (boolean) and
(3) the time taken to come up with the relevance judgment (float).

The data is inserted into a database after each page. This allows users to take a break from the study and continue where they left off later, even if they closed their browser in the meantime.

A.2.1 Computation of Metrics. In the results (Section 5.4) we present the following metrics:

(1) **Accuracy**: This is simply the number of correctly judged instances divided by the total number of instances. Correctness is determined using the official TREC query relevances R, which are converted to binary according to the official guidelines, i.e., irrelevant ($R(q, d) = 0$) or relevant ($R(q, d) > 0$).
(2) **Search function usage**: Similarly to accuracy, we divide the number of instances where participants have indicated the usage of their browser's search function by the total number of instances.

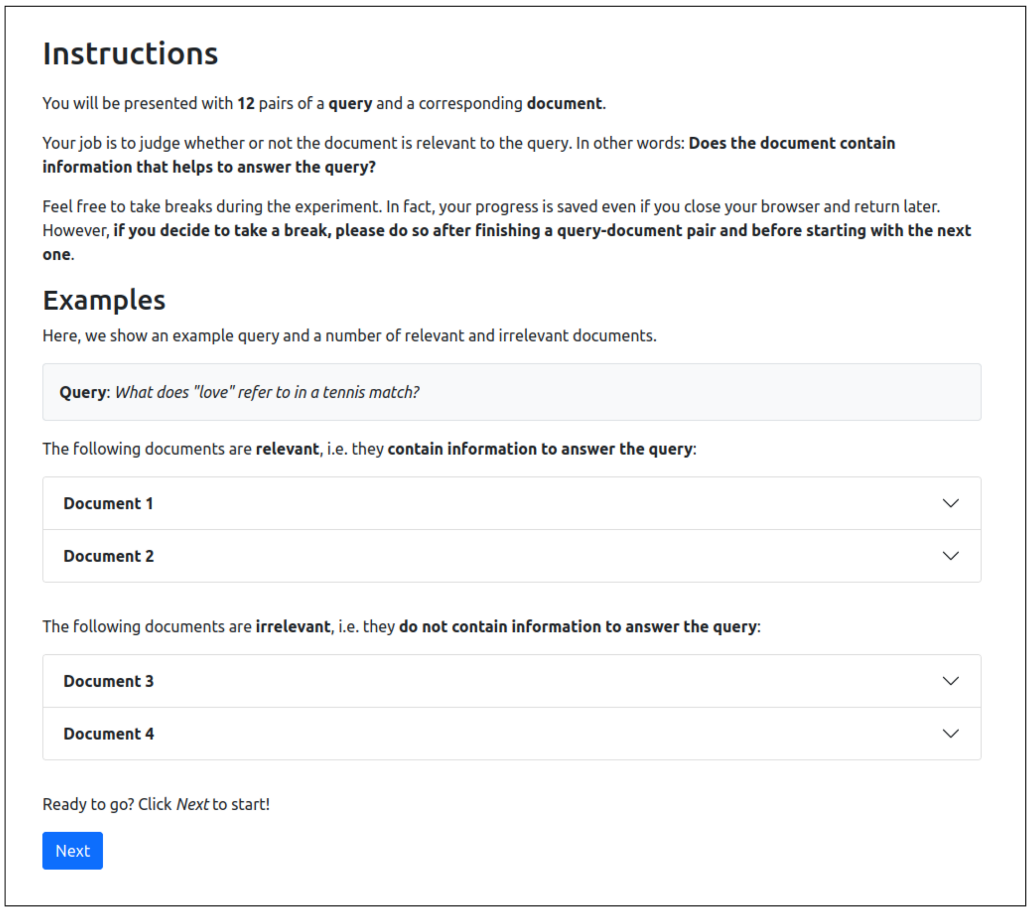


Fig. 15. The instructions page. This page is only displayed once and includes a task description along with some examples.

(3) **Time taken to complete a relevance judgment**: The time is measured as the difference between two-time stamps: The first one is recorded when the participant leaves the break page, and the second one is when the user completes the task page.

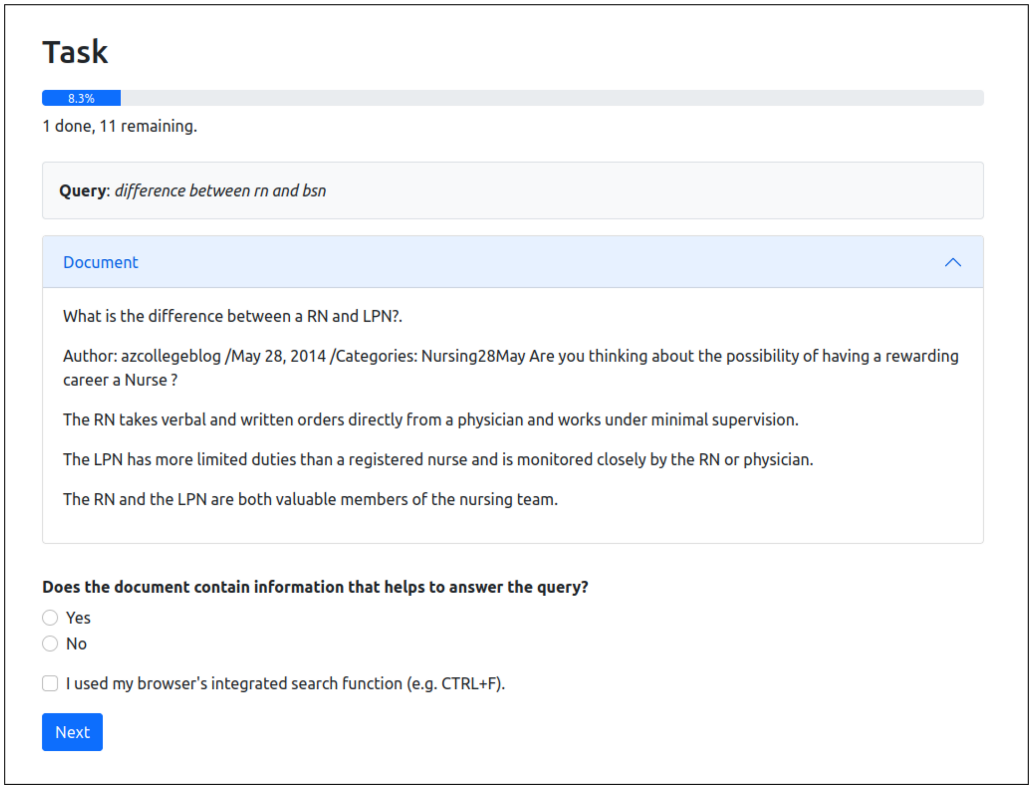


Fig. 16. The task page. It shows a query-document pair and gathers the participant's response (whether the document is relevant to the query) along with whether or not they used the browser search for this instance.

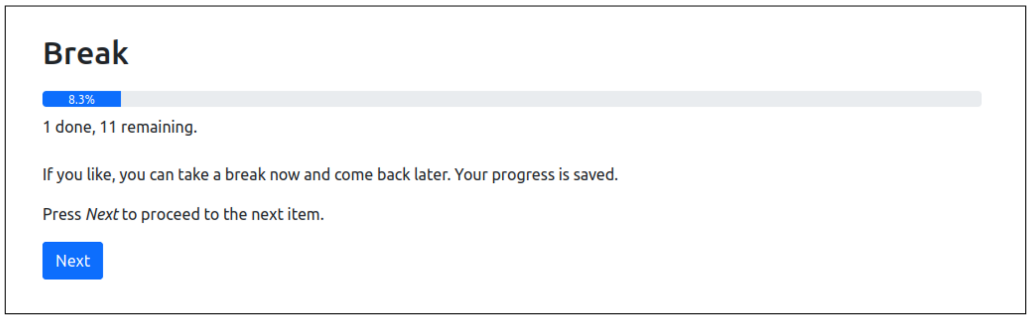


Fig. 17. The break page. It is displayed between two consecutive task pages.

REFERENCES

[1] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. 2020. Debugging tests for model explanations. In *Proceedings of the Advances in Neural Information Processing Systems*. H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33, Curran Associates, Inc., 700–712. Retrieved from https://proceedings.neurips.cc/paper/2020/file/075b051ec3d22dac7b33f788da631fd4-Paper.pdf.

[2] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. 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*. Association for Computational Linguistics. 3490–3496. DOI: https://doi.org/10.18653/v1/D19-1352

[3] Sophia Althammer, Sebastian Hofstätter, Mete Sertkan, Suzan Verberne, and Allan Hanbury. 2022. PARM: A paragraph aggregation retrieval model for dense document-to-document retrieval. In *Proceedings of the Advances in Information Retrieval*. Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty (Eds.), Springer International Publishing, Cham, 19–34.

[4] Avishek Anand, Lawrence Cavedon, Hideo Joho, Mark Sanderson, and Benno Stein. 2020. Conversational search (dagstuhl seminar 19461). In *Proceedings of the Dagstuhl Reports*. . Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473. Retrieved from https://arxiv.org/abs/1409.0473.

[6] Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. 2963–2977. DOI: https://doi.org/10.18653/v1/P19-1284

[7] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. arXiv:1308.3432. Retrieved from https://arxiv.org/abs/1308.3432.

[8] Abhijnan Chakraborty, Bhargavi Paranjape, Sourya Kakarla, and Niloy Ganguly. 2016. Stop clickbait: Detecting and preventing clickbaits in online news media. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* . IEEE, 9–16.

[9] Daniel L. Chen, Martin Schonger, and Chris Wickens. 2016. oTree–An open-source platform for laboratory, online, and field experiments. *Journal of Behavioral and Experimental Finance* 9 (2016), 88–97. https://www.sciencedirect.com/science/article/pii/S2214635016000101.

[10] Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 551–561. DOI: https://doi.org/10.18653/v1/D16-1053

[11] Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. ORCAS: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York, NY, 2983–2989. DOI: https://doi.org/10.1145/3340531.3412779

[12] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Vancouver, Canada, 593–602. DOI: https://doi.org/10.18653/v1/P17-1055

[13] Zhuyun Dai and Jamie Callan. 2019. 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*. Association for Computing Machinery, New York, NY, 985–988. DOI: https://doi.org/10.1145/3331184.3331303

[14] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in Ad-hoc search. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, 126–134. DOI: https://doi.org/10.1145/3159652.3159659

[15] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 4443–4458. DOI: https://doi.org/10.18653/v1/2020.acl-main.408

[16] Zeon Trevor Fernando, Jaspreet Singh, and Avishek Anand. 2019. 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*. Association for Computing Machinery, New York, NY, 1005–1008. DOI: https://doi.org/10.1145/3331184.3331312

[17] Thorben Funke, Megha Khosla, Mandeep Rathee, and Avishek Anand. 2022. ZORRO: Valid, sparse, and stable explanations in graph neural networks. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–12. DOI: https://doi.org/10.1109/TKDE.2022.3201170

[18] Luke Gallagher. 2019. Pairwise t-test on TREC Run Files. Retrieved from https://github.com/lgrz/pairwise-ttest/. Accessed April 2021.

[19] Ayçe Geçkil, Ahmet Anıl Müngen, Esra Gündoğan, and Mehmet Kaya. 2020. A clickbait detection method on news sites. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 932–937.

[20] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for Ad-Hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. . Association for Computing Machinery, New York, NY, 55–64. DOI: https://doi.org/10.1145/2983323.2983769

[21] Yue Guo, Yi Yang, and Ahmed Abbasi. 2022. Auto-debias: Debiasing masked language models with automated biased prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1012–1023. DOI: https://doi.org/10.18653/v1/2022.acl-long.72

[22] Michiel Hermans and Benjamin Schrauwen. 2013. Training and analysing deep recurrent neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*. C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26, Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2013/file/1ff8a7b5dc7a7d1f0ed65aaa29c04b1e-Paper.pdf.

[23] Sebastian Hofstätter, Aldo Lipani, Sophia Althammer, Markus Zlabinger, and Allan Hanbury. 2021. Mitigating the position bias of transformer models in passage re-ranking. In *Proceedings of the Advances in Information Retrieval: 43rd European Conference on IR Research*. Springer-Verlag, Berlin, 238–253. DOI: https://doi.org/10.1007/978-3-030-72113-8_16

[24] Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. 2021. Intra-document cascading: Learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 1349–1358. DOI: https://doi.org/10.1145/3404835.3462889

[25] Sebastian Hofstätter, Hamed Zamani, Bhaskar Mitra, Nick Craswell, and Allan Hanbury. 2020. Local self-attention over long text for efficient document retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 2021–2024. DOI: https://doi.org/10.1145/3397271.3401224

[26] Sebastian Hofstätter, Markus Zlabinger, and Allan Hanbury. 2020. Interpretable & Time-Budget-Constrained Contextualization for Re-Ranking. arXiv:2002.01854. Retrieved from https://arxiv.org/abs/2002.01854.

[27] Helge Holzmann and Avishek Anand. 2016. Tempas: Temporal archive search based on tags. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 207–210. DOI: https://doi.org/10.1145/2872518.2890555

[28] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. 2017. Exploring web archives through temporal anchor texts. In *Proceedings of the 2017 ACM on Web Science Conference*. Association for Computing Machinery, New York, NY, 289–298. DOI: https://doi.org/10.1145/3091478.3091500

[29] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York, NY, 2333–2338. DOI: https://doi.org/10.1145/2505515.2505665

[30] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1049–1058. DOI: https://doi.org/10.18653/v1/D17-1110

[31] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2018. Co-PACRR: A context-aware neural IR model for Ad-Hoc retrieval. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, 279–287. DOI: https://doi.org/10.1145/3159652.3159689

[32] Maximilian Idahl, Lijun Lyu, Ujwal Gadiraju, and Avishek Anand. 2021. Towards benchmarking the utility of explanations for model debugging. In *Proceedings of the 1st Workshop on Trustworthy Natural Language Processing*. Association for Computational Linguistics, Online, 68–73. DOI: https://doi.org/10.18653/v1/2021.trustnlp-1.8

[33] Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness?. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 4198–4205. DOI: https://doi.org/10.18653/v1/2020.acl-main.386

[34] Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 3543–3556. DOI: https://doi.org/10.18653/v1/N19-1357

[35] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2015. Visualizing and Understanding Recurrent Networks. arXiv:1506.02078. Retrieved from https://arxiv.org/abs/1506.02078.

[36] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 39–48. DOI: https://doi.org/10.1145/3397271.3401075

[37] Isaac Lage, Emily Chen, Jeffrey He, Menaka Narayanan, Been Kim, Sam Gershman, and Finale Doshi-Velez. 2019. An Evaluation of the Human-Interpretability of Explanation. arXiv:1902.00006. Retrieved from https://arxiv.org/abs/1902.00006.

[38] Victor Lavrenko and W. Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 120–127. DOI: https://doi.org/10.1145/383952.383972

[39] Eric Lehman, Jay DeYoung, Regina Barzilay, and Byron C. Wallace. 2019. Inferring which medical treatments work from reports of clinical trials. In *Proceedings of the 2019 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, 3705–3717. DOI: https://doi.org/10.18653/v1/N19-1371

[40] Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 107–117. DOI: https://doi.org/10.18653/v1/D16-1011

[41] Jurek Leonhardt, Koustav Rudra, Megha Khosla, Abhijit Anand, and Avishek Anand. 2022. Efficient neural ranking using forward indexes. In *Proceedings of the ACM Web Conference 2022*. Association for Computing Machinery, New York, NY, 266–276. DOI: https://doi.org/10.1145/3485447.3511955

[42] Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding Neural Networks through Representation Erasure. arXiv:1612.08220. Retrieved from https://arxiv.org/abs/1612.08220.

[43] Minghan Li and Eric Gaussier. 2021. KeyBLD: Selecting key blocks with local pre-ranking for long document information retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 2207–2211. DOI: https://doi.org/10.1145/3404835.3463083

[44] Xiangsheng Li, Jiaxin Mao, Chao Wang, Yiqun Liu, Min Zhang, and Shaoping Ma. 2019. Teach machine how to read: Reading behavior inspired relevance estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 795–804. DOI: https://doi.org/10.1145/3331184.3331205

[45] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP*. Association for Computational Linguistics, Online, 163–173. DOI: https://doi.org/10.18653/v1/2021.repl4nlp-1.17

[46] Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=Bkg6RiCqY7.

[47] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 1101–1104. DOI: https://doi.org/10.1145/3331184.3331317

[48] Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* Sampling. In *Proceedings of the Advances in Neural Information Processing Systems*. Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Eds.), Vol. 27, Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2014/file/309fee4e541e51de2e41f21bebb342aa-Paper.pdf.

[49] Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of the 33rd International Conference on Machine Learning*. Maria Florina Balcan and Kilian Q. Weinberger (Eds.), PMLR, New York, New York, 1614–1623. Retrieved from https://proceedings.mlr.press/v48/martins16.html.

[50] Ryan McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1849–1860. DOI: https://doi.org/10.18653/v1/D18-1211

[51] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1291–1299. DOI: https://doi.org/10.1145/3038912.3052579

[52] Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, and Ophir Frieder. 2020. Weighting passages enhances accuracy. *ACM Transactions on Information Systems* 39, 2 (2020), 11 pages. DOI: https://doi.org/10.1145/3428687

[53] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated MAchine reading COmprehension dataset. In *Proceedings of the CoCo@NIPS*. Retrieved from http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf.

[54] Yifan Nie, Yanling Li, and Jian-Yun Nie. 2018. Empirical study of multi-level convolution models for IR based on representations and interactions. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*. Association for Computing Machinery, New York, NY, 59–66. DOI: https://doi.org/10.1145/3234944.3234954

[55] Yifan Nie, Alessandro Sordoni, and Jian-Yun Nie. 2018. Multi-level abstraction convolutional model with weak supervision for information retrieval. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 985–988. DOI: https://doi.org/10.1145/3209978.3210123

[56] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. arXiv:1901.04085. Retrieved from https://arxiv.org/abs/1901.04085.

[57] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2016. A Study of MatchPyramid Models on Ad-hoc Retrieval. arXiv:2002.12478. Retrieved from https://arxiv.org/abs/1606.04648.

[58] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition. *Proceedings of the AAAI Conference on Artificial Intelligence* 30, 1 (2016). DOI: https://doi.org/10.1609/aaai.v30i1.10341

[59] Tobias Plötz and Stefan Roth. 2018. Neural nearest neighbors networks. In *Proceedings of the Advances in Neural Information Processing Systems*. S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31, Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2018/file/f0e52b27a7a5d6a1a87373dffa53dbe5-Paper.pdf.

[60] Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389. https://ieeexplore.ieee.org/document/8187284.

[61] Rishiraj Saha Roy and Avishek Anand. 2021. Question answering for the curated web: Tasks and methods in QA over knowledge bases and text collections. *Synthesis Lectures onSynthesis Lectures on Information Concepts, Retrieval, and Services* 13, 4 (2021), 1–194.

[62] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.

[63] Koustav Rudra and Avishek Anand. 2020. Distant supervision in BERT-based Adhoc document retrieval. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York, NY, 2197–2200. DOI: https://doi.org/10.1145/3340531.3412124

[64] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. Association for Computing Machinery, New York, NY, 101–110. DOI: https://doi.org/10.1145/2661829.2661935

[65] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*. . Association for Computing Machinery, New York, NY, 373–374. DOI: https://doi.org/10.1145/2567948.2577348

[66] Georgios Sidiropoulos and Evangelos Kanoulas. 2022. Analysing the robustness of dual encoders for dense retrieval against misspellings. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 2132–2136. DOI: https://doi.org/10.1145/3477495.3531818

[67] Jaspreet Singh and Avishek Anand. 2018. Posthoc Interpretability of Learning to Rank Models using Secondary Training Data. arXiv:1806.11330. Retrieved from https://arxiv.org/abs/1806.11330.

[68] Jaspreet Singh and Avishek Anand. 2019. EXS: Explainable search using local model agnostic interpretability. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, 770–773. DOI: https://doi.org/10.1145/3289600.3290620

[69] Jaspreet Singh and Avishek Anand. 2020. Model agnostic interpretability of rankers via intent modelling. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. Association for Computing Machinery, New York, NY, 618–628. DOI: https://doi.org/10.1145/3351095.3375234

[70] Jaspreet Singh, Johannes Hoffart, and Avishek Anand. 2016. Discovering entities with just a little help from you. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. Association for Computing Machinery, New York, NY, 1331–1340. DOI: https://doi.org/10.1145/2983323.2983798

[71] Jaspreet Singh, Megha Khosla, Wang Zhenye, and Avishek Anand. 2021. Extracting per query valid explanations for blackbox learning-to-rank models. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. Association for Computing Machinery, New York, NY, 203–210. DOI: https://doi.org/10.1145/3471158.3472241

[72] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. 2016. Expedition: A time-aware exploratory search system designed for scholars. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 1105–1108. DOI: https://doi.org/10.1145/2911451.2911465

[73] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. 2005. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*. 2–6.

[74] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 464–473. DOI: https://doi.org/10.18653/v1/P16-1044

[75] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the 35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*. Retrieved from https://openreview.net/forum?id=wCu6T5xFjeJ.

[76] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: A large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 809–819. DOI: https://doi.org/10.18653/v1/N18-1074

[77] Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does BERT answer questions? A layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery, New York, NY, 1823–1832. DOI: https://doi.org/10.1145/3357384.3358028

[78] Michael Völske, Alexander Bondarenko, Maik Fröbe, Benno Stein, Jaspreet Singh, Matthias Hagen, and Avishek Anand. 2021. Towards axiomatic explanations for neural ranking models. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. Association for Computing Machinery, New York, NY, 13–22. DOI: https://doi.org/10.1145/3471158.3472256

[79] Eric Wallace, Tony Zhao, Shi Feng, and Sameer Singh. 2021. Concealed data poisoning attacks on NLP models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 139–150. DOI: https://doi.org/10.18653/v1/2021.naacl-main.13

[80] Jonas Wallat, Jaspreet Singh, and Avishek Anand. 2020. BERTnesia: Investigating the capture and forgetting of knowledge in BERT. In *Proceedings of the 3rd BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Online, 174–183. DOI: https://doi.org/10.18653/v1/2020.blackboxnlp-1.17

[81] Lijie Wang, Yaozong Shen, Shuyuan Peng, Shuai Zhang, Xinyan Xiao, Hao Liu, Hongxuan Tang, Ying Chen, Hua Wu, and Haifeng Wang. 2022. A fine-grained interpretability evaluation benchmark for neural NLP. (2022). arXiv:10.48550. Retrieved from https://arxiv.org/abs/10.48550.

[82] Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 11–20. DOI: https://doi.org/10.18653/v1/D19-1002

[83] Chen Wu, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2022. PRADA: Practical Black-Box Adversarial Attacks against Neural Ranking Models. arXiv:2204.01321. Retrieved from https://arxiv.org/abs/2204.01321.

[84] Zhijing Wu, Jiaxin Mao, Yiqun Liu, Jingtao Zhan, Yukun Zheng, Min Zhang, and Shaoping Ma. 2020. Leveraging passage-level cumulative gain for document ranking. In *Proceedings of the Web Conference 2020*. Association for Computing Machinery, New York, NY, 2421–2431. DOI: https://doi.org/10.1145/3366423.3380305

[85] Sang Michael Xie and Stefano Ermon. 2019. Reparameterizable subset sampling via continuous relaxations. In *Proceedings of the IJCAI*. 3919–3925. DOI: https://doi.org/10.24963/ijcai.2019/544.

[86] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural Ad-Hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 55–64. DOI: https://doi.org/10.1145/3077136.3080809

[87] Huijuan Xu, Subhashini Venugopalan, Vasili Ramanishka, Marcus Rohrbach, and Kate Saenko. 2015. A Multi-scale Multiple Instance Video Description Network. arXiv:1505.05914. Retrieved from https://arxiv.org/abs/1505.05914.

[88] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 1480–1489. DOI: https://doi.org/10.18653/v1/N16-1174

[89] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2019. INVASE: Instance-wise variable selection using neural networks. In *Proceedings of the International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=BJg_roAcK7.

[90] Zijian Zhang, Koustav Rudra, and Avishek Anand. 2021. Explain and predict, and then predict again. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, 418–426. DOI: https://doi.org/10.1145/3437963.3441758

[91] Ruiqi Zhong, Steven Shao, and Kathleen McKeown. 2019. Fine-grained Sentiment Analysis with Faithful Attention. arXiv:1908.06870. Retrieved from https://arxiv.org/abs/1908.06870.

[92] Shengyao Zhuang and Guido Zuccon. 2021. TILDE: Term independent likelihood MoDEl for passage re-ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, 1483–1492. DOI: https://doi.org/10.1145/3404835.3462922

Received 31 May 2022; revised 19 October 2022; accepted 21 November 2022