TILDE: Term Independent Likelihood moDEl for Passage Re-ranking

Shengyao Zhuang
The University of Queensland
Brisbane, QLD, Australia
s.zhuang@uq.edu.au

Guido Zuccon
The University of Queensland
Brisbane, QLD, Australia
g.zuccon@uq.edu.au

ABSTRACT

Deep language models (deep LMs) are increasingly being used for full text retrieval or within cascade retrieval pipelines as later-stage re-rankers. A problem with using deep LMs is that, at query time, a slow inference step needs to be performed – this hinders the practical adoption of these powerful retrieval models, or limits sensibly how many documents can be considered for re-ranking.

We propose the novel, BERT-based, Term Independent Likelihood moDEl (TILDE), which ranks documents by both query and document likelihood. At query time, our model does not require the inference step of deep language models based retrieval approaches, thus providing consistent time-savings, as the prediction of query terms' likelihood can be pre-computed and stored during index creation. This is achieved by relaxing the term dependence assumption made by the deep LMs. In addition, we have devised a novel bi-directional training loss which allows TILDE to maximise both query and document likelihood at the same time during training. At query time, TILDE can rely on its query likelihood component (TILDE-QL) solely, or the combination of TILDE-QL and its document likelihood component (TILDE-DL), thus providing a flexible trade-off between efficiency and effectiveness. Exploiting both components provide the highest effectiveness at a higher computational cost while relying only on TILDE-QL trades off effectiveness for faster response time due to no inference being required.

TILDE is evaluated on the MS MARCO and TREC Deep Learning 2019 and 2020 passage ranking datasets. Empirical results show that, compared to other approaches that aim to make deep language models viable operationally, TILDE achieves competitive effectiveness coupled with low query latency.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**; **Information retrieval query processing**.

KEYWORDS

Query likelihood model, Neural IR, BERT

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462922

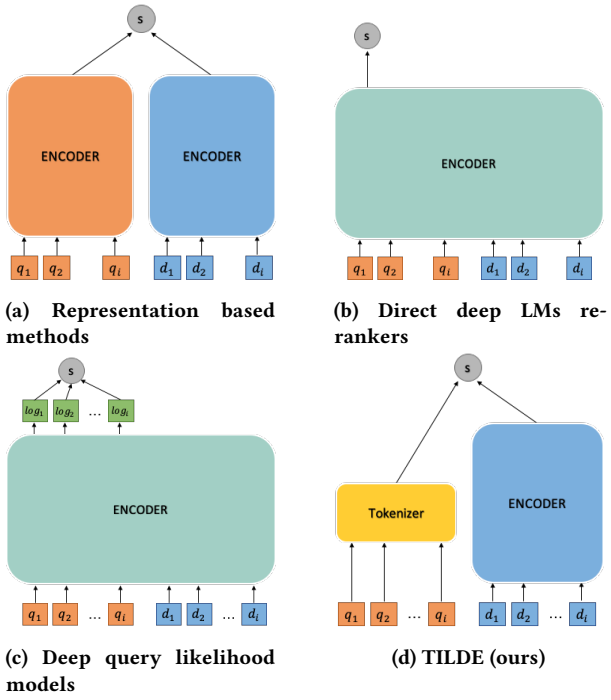
ACM Reference Format:
Shengyao Zhuang and Guido Zuccon. 2021. TILDE: Term Independent Likelihood moDEl for Passage Re-ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462922

1 INTRODUCTION

Deep language models like BERT [7] and T5 [21] are increasingly being used for full index retrieval [6, 12, 16, 20] or for late-stage re-ranking within cascade retrieval pipelines [8, 18, 19, 33] (i.e., where highly ranked documents as ranked by an inexpensive ranker, e.g., BM25, are re-ranked by a more expensive ranker). These deep language models have been empirically shown to outperform traditional retrieval baselines represent the current state-of-the-art in terms of ranking effectiveness [14, 17].

There are four main patterns of use for deep language models for retrieval: (i) as representation based methods [10, 12, 16, 27, 31], (ii) as modified document text methods [6, 20], (iii) as direct deep language model re-rankers [18, 19], or (iv) as deep query likelihood models [8, 33]. Each pattern has different effectiveness and efficiency trade-offs (see Section 2 and Table 1). Figure 1 illustrates three of these four main patterns, along with the approach taken by our proposed method (TILDE). Representation-based methods (Figure 1, (a)) use two encoders to get query and document vector representations separately. As query and document do not directly interact with each other, the document representation can be pre-computed and stored at indexing. The query representation however needs to be computed at query time. On the other hand, direct deep language model re-rankers and deep query likelihood re-rankers (Figure 1, (a) and (b)) require query-document pairs as input to the model, hence the inference step can only be performed at query time. While deep language model and deep query likelihood model re-rankers are often highly effective, they are also very inefficient. On the other hand, representation-based methods and those based on modified document text demonstrate lower effectiveness but a higher efficiency.

In this paper, we focus on deep query likelihood models [8, 33] that typically rely on a transformer-based decoder as the model architecture [14]. In this architecture, the likelihood of a query term is estimated with respect to the previous terms in the query (term dependence) through an inference mechanism, and the whole collection shares a single model. Unlike other deterministic neural re-rankers such as those that employ BERT [18, 19], a document's relevance score provided by a deep query likelihood model is more



(a) Representation based methods

(b) Direct deep LMs re-rankers

(c) Deep query likelihood models

(d) TILDE (ours)

Figure 1: High level architectures of three of the four main patterns of use of deep language models in search, along with the architecture of the proposed TILDE method. The architecture of modified document text methods is not shown for brevity.

explainable. This is because the likelihoods produced by these models can be considered to directly represent the importance of query terms.

However, deep query likelihood models are very inefficient, even when used as the final stage in a cascade pipeline. This inefficiency is because the transformer model's inference step is computationally expensive and query likelihood cannot be pre-computed offline[1] (e.g., during indexing). This is because the query needs to be observed to compute query term likelihood due to how deep query likelihood models compute query term dependencies. Performing the inference step at runtime is infeasible as it would introduce latency in the search system that is not acceptable by most users in typical applications such as web, email, and desktop search. Hence, often current deep query likelihood models cannot be used in realistic or practical search scenarios.

To overcome these inefficiencies, we propose TILDE, a novel BERT-based Term Independent document-query Likelihood moDEl. Unlike current deep query likelihood models, TILDE assumes that query terms are independent. Importantly, each term's query likelihood can be pre-computed offline and stored in the index. By doing so, TILDE does not require the expensive inference step at runtime, typical of deep language models and deep query likelihood models.

We further propose a novel loss function that can be used to train TILDE: the bidirectional query-document likelihood loss (BiQDL).

[1]Unless for a subset of pre-determined, perhaps popular, queries.

Table 1: High-level comparison of trade-off between effectiveness and efficiency made by the proposed TILDE and the four typical patterns of use of deep LMs for search.

Method	Effectiveness	Efficiency
Representation Based [12, 16, 31]	++	+
Modified Document [6, 20]	+	+++
Deep LMs Re-ranker [18, 19]	+++	- -
Deep QLMs Re-ranker [8, 33]	++	- - -
TILDE (ours)	++	+++

Unlike traditional query likelihood models [29, 30] and deep query likelihood models [8, 33] which only optimise query likelihood during training, BiQDL is used to train TILDE to maximise both query and document likelihood at the same time. After training, the learned model can be used to estimate query or document likelihoods. This provides further flexibility in the trade-off between efficiency and effectiveness. When TILDE is set to rely only on query likelihood (TILDE-QL), at query time the deep language model's tokenisation step (see Figure 1 (d)) – a very fast operation – is the only additional step required to conventional inverted index retrieval. When TILDE is set to rely on both query and document likelihoods (TILDE-QDL), then at query time both the deep language model's tokenisation and one typical deep language model's inference step are both required: while reducing effectiveness, the inference step does provide additional relevance signals to the re-ranker, thus further increasing effectiveness.

We investigate the effectiveness and efficiency of TILDE on the MS MARCO passage ranking dataset and the TREC 2019 and 2020 Deep Learning datasets. Compared to representative deep LMs based methods, TILDE exhibits a better trade-off between effectiveness and efficiency than comparative deep language models for search.

2 RELATED WORK

Recent advances in natural language processing have seen the introduction of deep language models, typically trained on large amount of text [7, 21]. These deep LMs have been exploited to improve search along four main patterns, which are characterised by a trade-off between effectiveness and efficiency, exposed in the high-level comparison of Table 1:

- **representation based methods**, like EPIC [16], ColBERT [12], RepBERT [31], CLEAR [10] and ANCE [27]. These methods compute a vector-based representation for each document at indexing (akin to signatures). These vectors are stored in memory and used at retrieval time by doing a single inference to obtain the query vector representation and multiple vector comparisons between documents' and query's (dense) vectors: a generally fast operation.
- **modified document text methods**, like docTTTTTquery [20] and DeepCT[2] [6]. These methods use deep LMs to generate terms to append to the original documents, which

[2]DeepCT modifies term frequency statistics in the inverted index. This could be implemented by modifying each document by repeating the terms in the text as required to achieve the required term frequency. This is the approach followed by Dai et al. in their original implementation [6].

are then indexed once expanded. By doing this, new terms are added to the document and existing terms are also re-weighted. No deep LM is used at query time.
- **direct deep language model re-rankers**, like the BERT re-ranker [18] and Mono-T5 [19]. These methods fine-tune deep (pre-trained) language models on text matching tasks. They require both query and document text as model input, and use a classification layer on top of the model's output embeddings (e.g., the [CLS] embedding for BERT) to produce a score indicating the relevance of a document to a query.
- **deep query likelihood models**, like ranking by generation [8] and QLM-T5 [33]. These methods embrace the query likelihood ranking paradigm where the probability of a document being relevant to a query is estimated by the likelihood of generating the query text given the document text. As for direct deep language model re-rankers, the common approach is to feed query–document pairs through the model and sum the log probabilities of the query tokens' predictions provided as output by the model.

Direct deep language model re-rankers have shown to provide significant effectiveness improvement over other neural rerankers [14]. However, because at query time these methods require to infer scores from the deep LMs, a process that is computationally demanding and incurs high latency, they are typically ran as late-stage re-rankers within re-ranking pipelines because of the latency added by the inference process. Even so, though, some methods are infeasible to be used in production-ready systems. For example the additional latency imposed by the BERT re-ranker [18] is in the order of 3,500 ms on high-performance GPUs. Recent works have attempted to dynamically identify appropriate points for early exciting for accelerating the BERT inference step, at the expense of small degradation of model quality [11, 26, 32]; yet, the up to 40% savings in runtime [26] still render these methods impractical for deployed search engines. Deep query likelihood models follow along the same lines, though being generally less effective and less efficient than direct deep language model re-rankers [8, 33].

The representation based methods and the modified document text methods address the efficiency drawback of the previous methods. In the first type of methods, this is achieved by storing the vectors computed offline as signatures, which are then used at retrieval by performing vector based similarity with the query vector. Nevertheless, in these approaches the query vector does still need to be computed online, thus requiring the query to be ran through the inference process of the deep LMs: this adds a latency factor in the order of tens or hundreds (tens) of milliseconds, when computation is performed on CPU (GPU) [16]. In the second type of methods, at indexing time each document is expanded (adding new and existing terms) using the deep LMs: these expanded documents are the ones stored in the index. Then, traditional inverted file retrieval methods (e.g., BM25) are applied and no deep LMs inference is required at query time, thus exhibiting the fastest runtimes among methods exploiting deep LMs. However, the low latency obtained by these two types of methods does not come free as speed up in retrieval are traded off for lower effectiveness compared to direct deep language model re-rankers.

The method proposed in this paper, TILDE shares similarities with representation based methods [12, 16, 31], with two fundamental differences. First, these previous representation based methods fine-tune their deep LMs to maximize the similarities between the representation of documents and the queries for which they are relevant. These learned representations are dense vectors in a high dimensional space, rendering them of little interpretability (i.e., not explainable). On the other hand, our fine-tuning objective is rooted on query and document likelihood, and the prediction made by TILDE can be considered as an indicator of term importance, hence more interpretable. Second, although previous methods are able to avoid performing inference for documents at query time, they still need to perform inference for the query representation. Since commonly the size of these models is large, the inference time required when answering a query is considerable. In contrast, TILDE-QL, which only relies on query likelihood, requires just the deep LM's tokenizer to extract the token IDs from the query, hence avoiding the expensive inference step at query time.

3 METHOD

TILDE is influenced by deep query likelihood and deep document likelihood ranking paradigms [8, 33]. In this section, we provide an explanation of deep query likelihood and document likelihood models (3.1), a theoretical framework describing how deep query likelihood and document likelihood models are used in TILDE (3.2) as well as our new bidirectional query-document likelihood loss (BiQDL) that is used to fine tune TILDE (3.3).

3.1 Ranking by Query & Document Likelihoods

Deep language models are able to predict the conditional probability of the next token x_i given the previous tokens in a sequence [2]:

$$P(x_i) = P_\theta(x_i|x_{<i}) \tag{1}$$

where θ are the weights of the deep language model. Based on Eq.1, typical deep query likelihood models [8, 33] compute the log query likelihood (QL) as:

$$\text{QL}(q|d^k) = \sum_i^{|q|} \log(P_\theta(q_i|q_{<i}, d^k)) \tag{2}$$

where q is the query text and d^k is the text of the k-th document in the candidate set D ($d^k \in D$). That is, the log probability of the query token q_i is conditioned on all the document tokens and all the previous query tokens: hence this type of query likelihood model are *term dependent models*. The candidate documents are then ranked in descending order of the log query likelihood estimated by the query likelihood model.

Similarly, candidate documents also can be ranked according to the log document likelihood (DL) function:

$$\text{DL}(d^k|q) = \frac{1}{|d^k|} \sum_i^{|d^k|} \log(P_\theta(d_i^k|d_{<i}^k, q)) \tag{3}$$

where the log probability of document tokens is estimated by a deep document language model. When ranking documents with document likelihood, the final document log probability typically needs to be normalized by the length of the document (the first

term in the equation). This is because longer documents tend to have lower log probabilities and thus without normalization are more likely to be incorrectly ranked lower than short documents. Deep document language models have a similar model architectural as the deep query likelihood models with one key difference: the objective function (Eq. 2 vs. Eq. 3) that is used to fine tune the model.

Deep query likelihood and document likelihood models have been shown to be much more effective than traditional statistical query likelihood models [8, 33]. This is because deep query likelihood and document likelihood models can more effectively leverage patterns in human language embedded in the pre-trained deep language models, providing more accurate likelihood estimations. However, this improvement in effectiveness comes with two trade-offs. First, deep language models employ large neural networks with millions or even billions of parameters [3]. Because of this, inferring likelihoods in these models is expensive[3]. Second, as the likelihood of the next token is dependent on all previous context tokens (Eq. 1), both a query and a document need to be provided to the model at the same time. As query likelihoods typically cannot be predicted ahead of time, inferences must be performed online (at runtime). Also, the number of candidate documents to be ranked is usually large (e.g. top 1,000) and, thus, multiple inference steps are required. As result, deep query likelihood and document likelihood methods, e.g., QLM-T5 [33], exhibit poor runtime efficiency (high query latency), preventing the adoption of them in production-level search engines.

3.2 TILDE: Term Independent Likelihood moDEl

The key factor influencing the efficiency of existing deep query likelihood and document likelihood models is that terms in a document and a query are modelled as dependent on each other. While this is realistic, it is often relaxed in typical information retrieval scenarios. For example, traditional bag of words [22] and language modelling [29, 30] approaches assume terms are independent. In devising TILDE, we follow this common assumption of term independence. Thus the likelihood of a query (TILDE-QL) or a document (TILDE-DL) can be computed as:

$$\text{TILDE-QL}(q|d^k) = \sum_{i}^{|q|} \log(P_\theta(q_i|d^k)) \tag{4}$$

$$\text{TILDE-DL}(d^k|q) = \frac{1}{|d^k|} \sum_{i}^{|d^k|} \log(P_\theta(d_i^k|q)) \tag{5}$$

TILDE is based on the BERT deep language model [7]. To compute the term independent query likelihood (TILDE-QL), only the text of a document is required as input for BERT; the output is the log probability for each query token. We compute this using a language modelling head on top of the [CLS] token provided as output by BERT. Then, the query text is tokenised into query token IDs by the BERT tokeniser. The IDs are used to look up a corresponding log probability from the likelihood distribution predicted by TILDE-QL. Finally, all the corresponding log probabilities are summed to form the query likelihood. The term independent document likelihood (TILDE-DL) is computed in a similar fashion, but with the query taking the place of the document.

Previous studies have shown that ranking documents by query likelihood is much more effective than ranking by document likelihood [8]. This is because documents are usually much longer than queries. When a deep query likelihood model estimates query likelihood, the model has been provided with richer information, and hence tends to be more accurate. However, note that document likelihood models are similar to generative models as they have been trained to predict the next tokens in a document with only a few query tokens as input. It is well known that deep language models have the ability to generate in-topic text [**?**], hence document likelihood models should also provide useful topic-level information about the document. Based on these intuitions, we combine the deep query and document likelihood components of TILDE into a unique query-document likelihood function:

$$\begin{aligned}&\text{TILDE-QDL}(q, d^k) = \\ &= \alpha \cdot \text{TILDE-QL}(q|d^k) + (1-\alpha) \cdot \text{TILDE-DL}(d^k|q)\end{aligned} \tag{6}$$

where α is a weight factor used to control the influence of query likelihood over document likelihood. For example, when $\alpha = 1$, the document is ranked solely by TILDE-QL, and when $\alpha = 1$ the document is ranked solely by TILDE-DL.

Compared to standard deep query likelihood and document likelihood models [8, 33] based on Eq. 2 and 3, TILDE only requires the query text or the document text as the input. The output is the log probabilities for all the tokens in the deep language model's vocabulary. In other words, TILDE maps the input query or document text to a log probability distribution over the entire vocabulary, and this distribution can be considered as the query or document representation. As for representation-based methods [10, 12, 16, 27], the document representation (log probability distribution) that TILDE produces can be pre-computed and stored offline, during indexing time. If TILDE-QL is used, no expensive deep language model inference step is required, as documents have been pre-processed at indexing time and the query only requires tokenisation. If TILDE-DL is used, only one deep language model inference is required to obtain a query representation, while the documents' tokenisation can also be performed at indexing. These attributes enable TILDE to deliver a significantly lower query latency compared to other deep query likelihood and document likelihood methods.

3.3 Bi-directional query-document likelihood loss (BiQDL)

Standard deep query likelihood and document likelihood models are typically trained with negative log likelihood loss over relevant query-document pairs provided by the training dataset, i.e., $D = \{(q^1, d^1), (q^2, d^2), ..., (q^{|D|}, d^{|D|})\}$ [8, 33]. However, this loss function is not suitable for TILDE as we treat predicted tokens independently and aim to predict all possible relevant query or document tokens given a document or query. This problem is similar to the *multi-label classification* task, where in our case the number of classes is the size of the vocabulary and token IDs in the relevant query or document are the target labels. Thus, a better choice to train our model is *negative log binary cross entropy loss*:

[3] And usually GPUs are required to provide acceptable inference time.

$$\mathcal{L}_{QL}(D) = -\sum_{(q,d)\in D} \frac{1}{|V|} \sum_{i}^{|V|} y \log(P_\theta(t_i|d)) + (1-y)\log(1-P_\theta(t_i|d)) \quad (7)$$

where V is the vocabulary, t_i is i-th token in the vocabulary, and $y = 1$ if t_i is in the query text, otherwise $y = 0$. Note, Eq. 7 is used to train the TILDE-QL model. To train the TILDE-DL model, we use the $\mathcal{L}_{DL}$ loss where the only two differences are that the document d is replaced with query q, and y becomes the indicator of whether t_i is present in the document text or not.

As we discussed in Section 3.1, query and document likelihoods are both useful in providing information about a query and a document, thus it is desirable to train TILDE-QL and TILDE-DL with $\mathcal{L}_{QL}$ and $\mathcal{L}_{DL}$ separately so that TILDE-QDL (Eq. 6) can be used to rank documents. However, inspired by *Symmetric Cross Entropy* [24], an alternative way of training the full TILDE-QDL model is to leverage both $\mathcal{L}_{QL}$ and $\mathcal{L}_{DL}$ at the same time. To achieve this, we propose the *bi-directional query-document likelihood loss* (BiQDL):

$$\mathcal{L}_{BiQDL}(D) = \frac{\mathcal{L}_{QL}(D) + \mathcal{L}_{DL}(D)}{2} \quad (8)$$

By using $\mathcal{L}_{BiQDL}$, TILDE maximises both QL and DL simultaneously during training. As the result, TILDE trained with $\mathcal{L}_{BiQDL}$ can be used to serve as both QLM and DLM at the same time.

4 EXPERIMENTAL SETTINGS

In order to guide the empirical experiments for studying TILDE, we propose the following research questions:

RQ1: What is the effect of assuming query term independence in deep query likelihood models for retrieval?
RQ2: How does TILDE trade-off effectiveness for efficiency using the TILDE-QL and TILDE-DL components?
RQ3: What effect has the bi-direction query-document likelihood loss have on effectiveness for TILDE?
RQ4: What effect does the weight factor α have on TILDE?

The code that implements TILDE and the other methods considered in this paper is available at https://github.com/ielab/TILDE, along with the experimental results.

4.1 Datasets and Evaluation Measures

We conduct our experiments on the MS MARCO dataset [4] and the two recent TREC Deep Learning tracks (DL2019, DL2020) [4, 5].

The MS MARCO dataset consists of approximately 8.8 million passages (average length: 73.1 terms) extracted from Web pages, and approximately 1 million natural-language questions (average length: 7.5) gathered from the Bing search engine's query log. Queries in the dataset are split into train, dev, and eval sets[5]. Each query is associated with shallowly-annotated judgments where on average only one passage is marked as relevant and no irrelevant passages are identified. We consider unjudged documents as irrelevant when computing evaluation metrics. Because of the large number of queries, the MS MARCO dataset has been widely used for training deep language models based retrieval methods. Following standard practice from the literature [12, 14, 16, 33], in our empirical evaluation we use the dev set (≈7k) and the official evaluation measure MRR@10[6].

The TREC DL2019 and DL2020 collections share the passage corpus with MS MARCO. Unlike MS MARCO, these collections provide small query sets of four-point scaled dense judgments (43 for DL 2019, 54 for DL2020). Following standard practice in TREC DL [4, 5], we use nDCG@10 and MAP to evaluate our experiments on these collections to easily compare our methods to past and future work.

For all evaluation measures, differences between methods are tested for statistical significance using a paired two-tailed t-test, with Bonferroni correction.

In addition to effectiveness, we also measure the efficiency of the approaches. We do so by measuring the average query latency on the MS MARCO dataset. We do not measure latency on the TREC DL collections because of the relatively fewer queries. As the MS MARCO dataset contains more queries, average latencies measured on this dataset are more reliable. Efficiency experiments were conducted on a 80 CPUs Linux CentOS server with Intel(R) Xeon(R) CPU E7-4830 v2 @ 2.20GHz, and 1TB of DDR 3 RAM; no GPU computation was used for the efficiency experiments.

4.2 Baselines

We compare the effectiveness and efficiency of TILDE to the following baselines:

BM25 [22]: A widely used statistical bag-of-words approach that is commonly used as the first-stage retrieval method by most deep language model based re-rankers. We also use BM25 as the first-stage ranker, on top of which we apply TILDE (and other deep LM re-rankers). We use the prebuilt Anserini [28] MS MARCO index and *pyserini* [1] with default settings for retrieval.

docTquery-T5 [20]: A T5-based modified document text method. Instead of directly modifying document term frequency values stored in the index as done by DeepCT, this method performs term expansion to documents ahead of indexing. This expansion is done by generating possible queries and appending them at the end of the original documents. The modified documents are then indexed and the standard BM25 ranker can be used at retrieval – as such, docTquery-T5 can be used as first-stage retrieval method. Compared to DeepCT, docTquery-T5 has been empirically found to provide higher effectiveness gains over BM25 [14]. Thus, in our experiments we also use docTquery-T5 as first-stage retrieval model. We use *pyserini* to implement docTquery-T5.

EPIC [16]: A recent representation based method that uses BERT. Document and query representations in EPIC are fine tuned using similarity matching, and thus at query time BERT inference has to be applied on the query. We use the original implementation of EPIC provided in the *OpenNIR* toolkit [15].

BERT-base / BERT-large re-ranker [18]: A direct deep language model re-ranker approach, based on BERT. This approach requires that both the query and the document are jointly provided as model inputs at query time, as for deep query likelihood models.

[4] https://microsoft.github.io/msmarco/
[5] The eval set is not made publicly available as it is used for the MS MARCO leaderboard

[6] Despite recent, somewhat differing opinions expressed in the community [9, 23], and recent empirical evidence [34], MRR@10 is still the official (and only) measure used for reporting on MS MARCO, and thus we adapt to this convention to allow our results to be easily compared to past and future work.

Instead of using query likelihood to directly estimate relevance, this model treats ranking as a classification task and predicts a score for each query-document pair. The BERT-large re-ranker differs from the BERT-base from the (larger) number of parameters.

QLM-T5 / QLM-BERT [33] A recent deep query likelihood model that assume term dependencies with respect to terms in queries and documents. In addition, we provide a further extension of this model by considering a version based on the BERT deep language model. This is done to be able compare to TILDE to determine the impact of taking the term independence assumption, as the language models would have the same architecture and number of parameters. However, we cannot directly use the original BERT implementation because BERT uses a transformer-encoder, which applies bidirectional self-attention: the input tokens can pay attention to the next tokens in the sequence and thus the model can easily predict any target token – these next tokens then need to be masked out. To make the BERT-based term-dependent model work, we apply attention masks on each layer of BERT so as to allow the current token to pay attention only to itself and the previous tokens. By doing this, we change BERT into a left-to-right architecture and can thus apply the same next token prediction training objective used by the T5-based model. For the QLM-T5 model, we follow the training settings described by Zhuang et al. [33].

4.3 Training Details

We use the BERT-base-uncased deep language model [7] as the main architecture of TILDE. This language model has been widely adopted in other relevant approaches [6, 12, 16, 18].

We fine tune TILDE using the BiQDL loss function (Section 3.3) and the official MS MARCO train set queries and judgements (≈533k relevance judgements). We then evaluate this fine tuned model the MS MARCO, TREC DL2019, and DL2020 collections. Query and passage pairs are tokenised using the BERT tokeniser provided in the Huggingface implementation [25]. The tokeniser maps each term in the target text into one or more token IDs into BERT's word-piece vocabulary. The original vocabulary size of BERT is 30,522; however, we find that removing stopwords and symbols in the target tokens improves effectiveness (Note, the model input texts are unchanged). [7] This is reasonable because stopwords such as "is" appear almost in every passage, thus, in the term-independent model, these terms would be likely incorrectly considered as extremely important. After stopwords removal, the BERT word-piece vocabulary comprises of 28,403 tokens.

To investigate the impact of the proposed BiQDL loss function, we separately train three TILDE models each with the $\mathcal{L}_{DL}$ or $\mathcal{L}_{QL}$ or $\mathcal{L}_{BiQDL}$.

For all BERT-based models, we use the Adam optimizer [13] with learning rate of 2e-5. For the T5-based models, we use a learning rate of 1e-3. We train all models on 10 epochs with batch size of 128 across the entire training set.

5 RESULTS

The main results of our empirical experiments aimed at comparing TILDE with previous representative methods and investigating the research questions of Section 4 are reported in Table 2. Next, we examine these results in details by answering each of the research questions.

5.1 RQ1: Impact of Term Independence on Deep Query Likelihood Models

RQ1 investigates the impact of relaxing term dependence modelling in deep query likelihood models. This aspect in fact represents the main difference between the proposed TILDE approach, which assumes term independence, and the previous deep QLMs [8, 33], which instead model term dependencies. To answer RQ1, we compare the results obtained by TILDE with those obtained by Zhuang et al.'s QLM-T5 and the modification of that method that we introduced in Section 4.2, QLM-BERT. For both methods we only focus on the versions with query likelihood only for RQ1 (i.e. TILDE-QL and QLM-BERT-QL). For all approaches, we perform a re-ranking of the top 1,000 results from BM25. Throughout the analysis we refer to the results reported in Table 2.

We first start by examining TILDE-QL and QLM-BERT-QL: these methods rely on the same pretrained deep language model, BERT. This comparison allows us to ascertain that differences in performance are due to the term independence assumption (and the consequent changes in the ranking method) rather than the underlying deep language model (as T5, used in QLM-T5, has a larger number of parameters than BERT). On MS MARCO, both TILDE-QL ($p = 3.0 \times 10^{-46}$) and QLM-BERT-QL ($p = 7.7 \times 10^{-43}$) significantly improve BM25 results; and QLM-BERT-QL obtains a higher MRR@10 than TILDE-QL, but the difference is not highly statistically significant (p=0.045). Similar trends are also observed for DL2019 and DL2020 in terms of nDCG@10 and MAP, except that for on DL2020, TILDE-QL surprisingly achieves a higher MAP score (0.406 vs 0.391), though the difference is not statistically significant (p=0.77).

We also consider the effectiveness of QLM-T5-QL. This model records higher effectiveness than the BERT-based counterpart: this is due to a more powerful pre-trained language model, T5, which compared to BERT features a higher number of model parameters. Differences between QLM-T5-QL and QLM-BERT-QL are not statistically significant in DL2019 and DL2020, but are for MRR@10 on MS MARCO (p=0.03). Differences between QLM-T5-QL and TILDE-QL are statistically significant for nDCG@10 (p=0.009) but not for MAP (p=0.07) on DL2019 and for both measures on DL2020 (nDCG@10: p=0.45; MAP: p=0.69).

The results indicate that relaxing term dependence in deep QLMs decreases effectiveness, although not significantly. This result is to be expected (and thus the results for MAP on DL2020 are surprising), because QLM-BERT-QL takes query-document pairs as the input to the model and thus the query text can directly interact with the document text, hence providing more accurate likelihood predictions. Moreover, predicting the next query token based on the document text and the previous query tokens should provide smoother and more grammatically correct predictions as these predictions are dynamically changed based on the previous query tokens. On the other hand, TILDE-QL predicts the query likelihoods only based on the document text hence it loses the benefits of the query context. Nevertheless, the modelling of term dependencies in QLMs comes

[7] We use the stopwords provide by the Python `nltk` library, but we keep `where`, `how`, `what`, `when`, `which`, `why`, `who`.

Table 2: Evaluation results. Latency is measured in milliseconds per query (query inference + re-ranking, lower the better). Note: the latency for re-rankers does not include the latency of their first stage retriever.
***: BM25+BERT-base/large results are obtained from the literature (and thus not available for all measures/datasets); latency values for these models were obtained using a GPU server.**
****: All deep query likelihood efficiency results have been obtained by us also using a GPU server, rather than the CPU server used for the remaining experiments.**

	MS MARCO		DL2019		DL2020	
Method	**MRR@10**	**Latency**	**nDCG@10**	**MAP**	**nDCG@10**	**MAP**
BM25	0.187	130	0.506	0.377	0.480	0.286
(i) Representation based						
BM25 + EPIC	0.270	356 + 108	0.609	0.411	0.576	0.349
docTquery-T5 + EPIC	0.302	279 + 20	0.686	0.473	0.624	0.405
(ii) Modified document text						
docTquery-T5	0.277	143	0.641	0.462	0.619	0.407
(iii) Direct deep language model						
BM25 + BERT-base*	0.347	2,970	0.703	—	0.668	0.431
BM25 + BERT-large*	0.365	3,500	0.738	0.506	—	—
(iv) Deep query likelihood						
BM25 + QLM-BERT**						
QL	0.281	4,500	0.641	0.482	0.625	0.391
DQL	0.290	9,000	0.662	0.484	0.635	0.401
BM25 + QLM-T5**						
QL	0.294	5,000	0.653	0.497	0.652	0.426
DQL	0.301	10,000	0.672	0.505	0.665	0.435
TILDE (ours)						
BM25 + TILDE						
TILDE-QL	0.269	0.5 + 29	0.579	0.406	0.620	0.406
TILDE-QDL with BiQDL	0.280	290 + 64	0.609	0.420	0.621	0.412
docTquery-T5 + TILDE						
TILDE-QL	0.285	0.5 + 0.9	0.650	0.467	0.624	0.417
TILDE-QDL with BiQDL	0.295	290 + 3.1	0.654	0.468	0.622	0.413

at a cost: QLM-BERT-QL and QLM-T5-QL have a latency of several orders of magnitude higher than that of TILDE-QL. In fact, TILDE-QL only adds 29.5 ms to the retrieval pipeline (as measured on the CPU server) and thus can easily be considered for deployment in practice. Conversely, QLM-BERT/T5 require approximately 5,000 ms (on a faster, GPU server), rendering them of impractical use.

5.2 RQ2: Trade-off between Effectiveness and Efficiency in TILDE

RQ2 investigates the trade-offs that TILDE enables between effectiveness and efficiency by allowing to use the deep query likelihood component alone, or this in conjunction with the deep document likelihood component. In fact, when TILDE is set to only rely on the deep query likelihood component (TILDE-QL, i.e., $\alpha = 1$ in Eq. 6), queries are processed by the BERT tokeniser, no model inference is required at query time, and only the summation of the log probability associated to the target query tokens for each document needs to be computed. When TILDE instead is set to rely on both deep query and document likelihood components (TILDE-QDL), along with the use of the BERT tokenizer to process queries, one inference step is needed to obtain the likelihood distribution for the document tokens from the query. In addition, document log likelihoods also need to be summed . As the TILDE-DL component can provide extra topic-level information about documents, as discussed in Section 3, we expect TILDE-QDL to achieve higher effectiveness than TILDE-QL.

To answer RQ2 we compare the results reported in Table 2 for TILDE-QL and TILDE-QDL; note that the results for TILDE-QDL were obtained setting $\alpha = 0.5$ (a study of different settings for α is performed for RQ4). We observe that in the majority of the cases TILDE-QDL is better than TILDE-QL, across datasets, measures and initial stage of retrieval (BM25 and docTquery-T5), with the only exception of when re-ranking docTquery-T5 results on DL2020, where both nDCG@10 and MAP of TILDE-QL are higher than those of TILDE-QDL. All differences between TILDE-QL and TILDE-QDL are however not statistically significant.

We further note that the same general trend is observed also for the QLM-BERT/T5 methods, for which variants that employ both deep query and document likelihoods (DQL) are better than the corresponding ones using only query likelihood (QL).

Table 3: Ablation study of loss and scoring functions on MS MARCO.

Scoring	Loss	MRR@10
TILDE-QL	$\mathcal{L}_{QL}$ (w/o clean_vocab)	0.257
TILDE-QL	$\mathcal{L}_{QL}$	0.264
TILDE-DL	$\mathcal{L}_{DL}$	0.056
TILDE-QDL	$\mathcal{L}_{QL}$, $\mathcal{L}_{DL}$	0.276
TILDE-QL	$\mathcal{L}_{BiQDL}$	0.269
TILDE-DL	$\mathcal{L}_{BiQDL}$	0.056
TILDE-QDL	$\mathcal{L}_{BiQDL}$	0.280

The trends identified above suggest that the deep document likelihood component generally improves the effectiveness of both TILDE and term-dependent deep query likelihood approaches, though not significantly.

The analysis so far, however, has only considered the effectiveness of the approaches. Next we consider their efficiency. TILDE-QL is sensibly more efficient than TILDE-QDL. This is because TILDE-QL uses the BERT tokenizer (a look-up table) to process queries; by doing so TILDE-QL only requires 0.5ms to generate the query representation. For document re-ranking, TILDE-QL requires 29ms to re-rank the top 1,000 results for BM25 and 0.9ms to re-rank the top 20 results for docTquery-T5. TILDE-QDL in fact requires the additional inference step for processing the query and the computation of document likelihoods. Thus, the small improvements provided by TILDE-QDL over TILDE-QL in terms of effectiveness come at a cost in terms of efficiency.

When compared to other approaches that make use of deep language models, we observe that there are no statistically significant differences among TILDE-QL and EPIC effectiveness results (across all datasets and measures). However, because TILDE-QL requires no inference, it enjoys a speed up in runtime of approximately 15 times compared to EPIC when re-ranking BM25, and it is even faster when re-ranking docTquery-T5 results (as only 20 passages are re-ranked – as for EPIC). On the other hand, TILDE-QDL (i.e. when using also the deep document likelihood) features a runtime comparable to that of EPIC, as both methods required a inference step at query time – and no statistically significant differences in terms of effectiveness are detected between the two methods. Compared to term-dependent QLMs and BERT-based re-rankers, any version of TILDE generally provides lower effectiveness; however TILDE is order of magnitude more efficient.

5.3 RQ3: Impact of BiQDL on Effectiveness

RQ3 investigates the impact of the proposed bi-direction query-document likelihood loss (BiQDL) on TILDE. To perform this we report the results of an ablation study performed on MS MARCO when re-ranking results from BM25. This study considers fine tuning TILDE with the different loss functions described in Section 3.3. We also consider the effect the stopword removal performed on the BERT vocabulary has on effectiveness.

The results of the ablation study are reported in Table 3. We first note that removing stopwords and symbols from the original BERT vocabulary is helpful as they are not important for term-independent models. Thus for the reminder of the ablation study we use the reduced vocabulary.

When using TILDE-QL, we find that the version trained with $\mathcal{L}_{BiQDL}$ is more effective than that trained with $\mathcal{L}_{QL}$ (0.269 vs 0.264, differences not statistically significant). This is interesting as $\mathcal{L}_{QL}$ aims to optimize deep query likelihood only while $\mathcal{L}_{BiQDL}$ aims to optimize both deep query and document likelihood. Thus, the addition of the document likelihood loss component in $\mathcal{L}_{BiQDL}$ also helps optimizing the query likelihood. We then compare the results obtained for TILDE-QDL: not surprisingly, when fine tuned with $\mathcal{L}_{BiQDL}$, this version of TILDE achieves better effectiveness than when scoring with models fine tuned with $\mathcal{L}_{QL}$ and $\mathcal{L}_{DL}$ separately (0.276 vs 0.280). Thus, in answer to RQ4, the bi-direction query-document likelihood loss improves the fine tuning of TILDE.

Further note that all models that use TILDE-DL for ranking perform badly, even when $\mathcal{L}_{DL}$ is used as loss function. This finding is in line with previous work [8] that reported that deep document likelihood alone is unable to provide a reliable relevance signal. However, we do find that combining the deep query likelihood and document likelihood provide a higher effectiveness than the query likelihood alone. For more details, we refer the reader to the next section where we study the effect of the weight factor α.

5.4 RQ4: Impact α on Effectiveness

RQ4 investigates the impact on effectiveness of the weight factor α, which controls the mix of deep query and document likelihood in TILDE-QDL. When examining the previous research questions, we reported the results obtained by fixing $\alpha = 0.5$. Here instead we use Eq. 6 as the scoring function and vary α between $\alpha = 0$ (DL only) to $\alpha = 1$ (QL only), with point-wise increments. Results are evaluated in terms of nDCG@10 and MAP on both DL2019 and DL2020 (results on MS MARCO show less stable trends because MRR@10 is not a stable metric).

Empirical results are reported in Figure 2. When setting $\alpha = 0$, passage relevance is estimated using deep document likelihood only, and effectiveness is low across datasets and evaluation measures. Surprisingly, when increasing α to 0.1, the effectiveness already increases significantly in both datasets. This suggests that the deep query likelihood component is very important as it helps to improve effectiveness even just by only adding a small portion of it. We observe that the effectiveness further continues to increase up to $\alpha = 0.4$; after this point, the effectiveness starts to drop down. This finding is important, as it means that the deep document likelihood is also helpful to deliver high effectiveness, confirming our hypothesis that document likelihood provides a useful relevance signal.

6 CONCLUSION

In this paper, we propose a novel BERT-based deep query likelihood model called TILDE that computes likelihoods without token dependencies. This unique feature of TILDE means that the tokens' likelihood can be pre-computed and stored at indexing time. Our model can further provide a flexible trade-off between effectiveness and efficiency: this is achieved by balancing two scoring components, namely the deep query likelihood (QL) and the deep

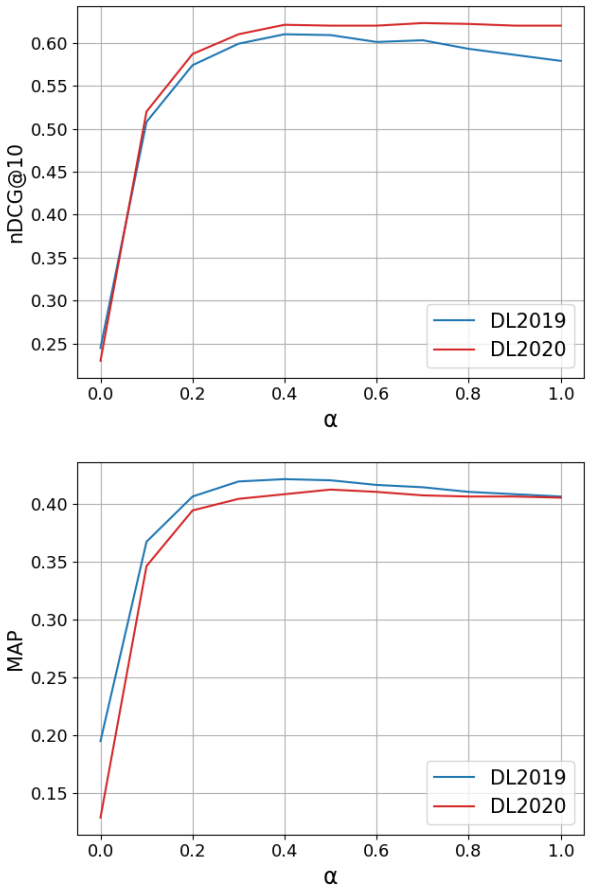


Figure 2: nDCG@10 (first row) and MAP (second row) values for different weight factor α on DL 2019 and 2020 datasets.

document likelihood (DL). When TILDE solely relies on QL, queries are processed by the pre-trained deep language model tokeniser and therefore no expensive inference is required at query time. In this case, TILDE achieves considerable lower query latency than other deep language model based re-rankers. without sacrificing much effectiveness. When the highest effectiveness is preferred, TILDE can trade-off efficiency for effectiveness by leveraging its DL scoring component to obtain a stronger document relevance signal, thus achieving higher effectiveness. To fine tune TILDE to better leverage both deep likelihood components, we further propose the bi-direction query-document likelihood loss (BiQDL). When BiQDL is used, TILDE maximizes both QL and DL at the fine tuning stage. We perform an ablation study that shows TILDE fine tuned with BiQDL is more effective than when TILDE is fine tuned trained with standard loss functions. Future work will focus on investigating more complex model architectures designed specifically for the QL and DL components as the standard BERT pre-trained model simply treats these as the same.

Broader impact

The research presented in this paper may inspire a new direction for improving the efficiency of BERT-based rankers. Our work has shown that a complex query encoder for BERT-based rankers is actually not necessary. The reason for this is intuitive: queries usually are much shorter than documents (sometimes are just a single keyword), thus query encoders can be simpler than document encoders. A simple query encoder is especially beneficial for representation-based rankers for which the query latency heavily depends on the query encoder inference time. Our proposed TILDE model can be thought of as an extreme case of query encoder simplification – no neural network layers are used in TILDE's encoder. Because of this, TILDE's query encoder inference can be performed efficiently in devices with low computational resources, such as mobile phones and other embedded devices.

Acknowledgement

Dr Guido Zuccon is the recipient of an Australian Research Council DECRA Research Fellowship (DE180101579). This research is partially funded by the Grain Research and Development Corporation project AgAsk (UOQ2003-009RTX). The authors are thankful to Dr Ahmed Mourad and Mr Harrisen Scells (ielab, UQ) for suggestions on early drafts of this work, and Dr Miao Xu (UQ) for discussions on loss functions.

REFERENCES

[1] Zeynep Akkalyoncu Yilmaz, Charles LA Clarke, and Jimmy Lin. 2020. A light-weight environment for learning experimental IR research practices. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2113–2116.
[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The journal of machine learning research* 3 (2003), 1137–1155.
[3] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).
[4] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2019. Overview of the trec 2019 deep learning track. In *Proceedings of TREC 2020*.
[5] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2020 deep learning track. In *Proceedings of TREC 2020*.
[6] Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1533–1536.
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4171–4186.
[8] Cicero dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through Ranking by Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1722–1727.
[9] Norbert Fuhr. 2018. Some common mistakes in IR evaluation, and how they can be avoided. In *ACM SIGIR Forum*, Vol. 51. ACM New York, NY, USA, 32–41.
[10] Luyu Gao, Zhuyun Dai, Zhen Fan, and Jamie Callan. 2020. Complementing lexical retrieval with semantic residual embedding. *arXiv preprint arXiv:2004.13969* (2020).
[11] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. DynaBERT: Dynamic BERT with Adaptive Width and Depth. *Advances in Neural Information Processing Systems* 33 (2020).
[12] 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*. 39–48.
[13] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.
[14] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2010.06467* (2020).
[15] Sean MacAvaney. 2020. OpenNIR: A Complete Neural Ad-Hoc Ranking Pipeline. In *WSDM 2020*.

[16] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via prediction of importance with contextualization. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1573–1576.

[17] Bhaskar Mitra and Nick Craswell. 2018. *An introduction to neural information retrieval.* Now Foundations and Trends.

[18] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[19] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document Ranking with a Pretrained Sequence-to-Sequence Model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, 708–718.

[20] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

[21] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[22] Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond.* Now Publishers Inc.

[23] Tetsuya Sakai. 2020. On Fuhr's Guideline for IR Evaluation. In *SIGIR Forum*, Vol. 54. p14.

[24] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 322–330.

[25] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. HuggingFace's Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).

[26] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic Early Exiting for Accelerating BERT Inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2246–2251.

[27] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808* (2020).

[28] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using Lucene. *Journal of Data and Information Quality (JDIQ)* 10, 4 (2018), 1–20.

[29] ChengXiang Zhai et al. 2008. Statistical Language Models for Information Retrieval A Critical Review. *Foundations and Trends® in Information Retrieval* 2, 3 (2008), 137–213.

[30] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)* 22, 2 (2004), 179–214.

[31] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. RepBERT: Contextualized Text Embeddings for First-Stage Retrieval. *arXiv preprint arXiv:2006.15498* (2020).

[32] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. BERT Loses Patience: Fast and Robust Inference with Early Exit. *Advances in Neural Information Processing Systems* 33 (2020).

[33] Shengyao Zhuang, Hang Li, and Guido Zuccon. 2021. Deep Query Likelihood Model for Information Retrieval. In *The 43rd European Conference On Information Retrieval (ECIR)*.

[34] Justin Zobel and Lida Rashidi. 2020. Corpus Bootstrapping for Assessment of the Properties of Effectiveness Measures. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1933–1952.