Exploiting Sentence-Level Representations for Passage Ranking

Jurek Leonhardt[1], Fabian Beringer[1] and Avishek Anand[1]

[1]L3S Research Center, Appelstraße 9a, 30167 Hannover, Germany

Abstract

Recently, pre-trained contextual models, such as BERT, have shown to perform well in language related tasks. We revisit the design decisions that govern the applicability of these models for the *passage re-ranking* task in open-domain question answering. We find that common approaches in the literature rely on fine-tuning a pre-trained BERT model and using a single, global representation of the input, discarding useful fine-grained relevance signals in token- or sentence-level representations. We argue that these discarded tokens hold useful information that can be leveraged. In this paper, we explicitly model the sentence-level representations by using Dynamic Memory Networks (DMNs) and conduct empirical evaluation to show improvements in passage re-ranking over fine-tuned vanilla BERT models by memory-enhanced explicit sentence modelling on a diverse set of open-domain QA datasets. We further show that freezing the BERT model and only training the DMN layer still comes close to the original performance, while improving training efficiency drastically. This indicates that the usual fine-tuning step mostly helps to aggregate the inherent information in a single output token, as opposed to adapting the whole model to the new task, and only achieves rather small gains.

Keywords

passage ranking, sentence-level, bert, question answering, information retrieval

1. Introduction

Language model pre-training has attracted wide attention and fine-tuning on pre-trained language model has shown to be effective for improving many downstream natural language processing tasks. BERT [1] obtained new state-of-the-art results on a broad spectrum of diverse tasks, offering pre-trained deep bidirectional representations which are conditioned on both left and right context in all layers, which is often followed by discriminative fine-tuning on each specific task, including passage re-ranking for open domain QA.

There are two limitations of using fine-tuned BERT models for re-ranking passages in QA. Firstly, passages are of variable lengths, which affects the quality of BERT-based representations. Specifically, in the fine-tuning regime of BERT for open domain QA and passage re-ranking, a representation is learnt for the entire passage given a question. While this is desirable for small passages or questions that have short and easy answers, it isn't for instances where the passage answers a question using multiple, more complex statements. Secondly, the passage re-ranking task is unlike other QA tasks, like factoid QA and reading comprehension, in that the answers are not limited to a word, phrase or sentence. Potential answers can have varying

LWDA'21: Lernen, Wissen, Daten, Analysen September 01–03, 2021, Munich, Germany

leonhardt@l3s.de (J. Leonhardt); beringer@l3s.de (F. Beringer); anand@l3s.de (A. Anand)

granularity and passages are judged by annotators based on the likelihood of containing the relevant answer. Therefore, the applicability of vanilla BERT models to answering queries that span multiple sentences or might need reasoning across distant sentences in the same passage is limited.

In this paper we deal with the above problems by extending the BERT model to explicitly model sentence representations. This is realized by distilling the sentence representations from the output of the BERT block and aggregating the representations of the tokens that make a sentence. Secondly, once we have the sentence representations, we apply a *Dynamic Memory Network* [2, 3] to model sentence-wise relations for relevance estimation. We are interested in the following research questions:

- By aggregating BERT representations on a sentence level and then reasoning over sentence representations, can we improve re-ranking performance?
- Can we improve training efficiency by light-weight reasoning instead of fine-tuning all parameters of BERT?

We perform experimentation on three diverse open-domain QA datasets and show that the sentence-level representations improve the model's re-ranking performance. We find that explicit sentence modeling using a DMN enables us to reason about the answers that spread across sentences. Additionally, we find that BERT-DMN, although being an extension of BERT, can be used without expensive fine-tuning of the BERT model, resulting in reduced training times. The code will be made publicly available.

2. Related Work

Recent practices in open-domain question answering (QA) can be traced to the Text Retrieval Conferences (TRECs) in the late 1990s. Voorhees [4] defines the task of textual open-domain question answering as using a small text snippet, usually an excerpt from a document as part of a large collection that is being utilized in the process, such as web pages [5]. In the last decade, the focus on open-domain question answering has shifted to the re-ranking stage, where answer identification from candidate documents is performed using learning strategies based on richer and better language understanding models [6, 7, 8, 9, 10]. Our approach also tries to propose models that improve the re-ranking part of the QA pipeline. Specifically, we are different from alternate approaches that perform *end-to-end* question answering that requires some type of term-based retrieval technique to restrict the input text under consideration [11, 12, 13].

Multiple approaches have been proposed to improve re-ranking in open-domain QA. In [6], the authors use LSTMs to encode questions and answers and then perform attention- and CNN-based pooling in order to perform question-answer-matching; [7] follows a similar idea, but produces multiple vector representations for each question and answer, which can then focus on different aspects. Other works like [10] aim to improve the answer selection process by filtering out noisy, irrelevant paragraphs. Afterwards, the answer is selected from the remaining, relevant paragraphs. Some works have used evidence aggregation to re-rank passages based on information from multiple other passages [8] or reinforcement learning to jointly train a *ranking model* to rank the passages and a *reading model* to extract the answer from a passage

[9]. In [14], the authors use weak supervision to train a BERT-based passage ranking model without any ground-truth labels.

The most recent improvement in re-ranking stage of open-domain QA comes from BERT models that have been shown to improve language understanding. Recent works have used BERT-based ranking models, dealing with efficiency [15] and analyzing the attention mechanism [16]. In [17], the authors compare the performance of BERT with and without fine-tuning on various NLP tasks. [18] deals with combining traditional Ranking models with BERT token representations.

Neural architectures for document ranking can be roughly categorized into *representation-based* models for learning semantic representations of the text [19, 20, 21], *interaction-based* models for learning salient interaction patterns from the local interactions between the query and document [22, 23] or a combination of both [24]. Other works [25, 26, 27] try to capture hierarchical matching patterns based on *n-gram* matches from the local interaction matrix of the query-document. More recent approaches [28, 29, 30] have tried to exploit positional information and context of the query terms. Other approaches include query modeling techniques [31, 32] with a query expansion based language model (QLM) that uses word embeddings.

3. Approach

The usual question answering process consists of multiple stages. Given a query, a simple method (like BM25) is used to rank a number of passages with respect to the query. Next, the top-n of these passages are **re-ranked** using a more expensive model. Finally, the top-k ($k < n$) of the re-ranked passages are used to answer the query. This work deals with the passage re-ranking step.

BERT-based models have achieved high performance in passage re-ranking tasks. We find, however, that these models are limited: Firstly, most variants tend to rely solely on BERT's dedicated classification output, operating under the assumption that its internal capabilities of compressing all query and passage representations into a single output are optimal. Secondly, BERT models are very large, which results in slow training.

In this paper we introduce a re-ranking approach that leverages the representations obtained from BERT and aggregates them using a Dynamic Memory Network. We describe DMNs and outline how they can be combined with BERT such that, in addition to the classification output, the query and passage representations are taken into account. Moreover, we investigate how our model can reduce training time by introducing a *lite* version.

3.1. Dynamic Memory Networks

In this section we briefly introduce Dynamic Memory Networks [2, 3], which we use to aggregate BERT outputs. DMNs take as input a sequence of words $w = (w_1, ..., w_T)$, which usually represent multiple sentences such as a document or a passage, and a question $q = (q_1, ..., q_N)$. They are composed of four *modules* (cf. Figure 1):

The **input module** encodes the input words as a sequence of vector representations. The input text is represented by pre-trained word embeddings and fed into a word-level many-to-many GRU. The outputs $h_t^w = \texttt{GRU}(w_t, h_{t-1}^w)$ are then used as inputs in other modules. If the

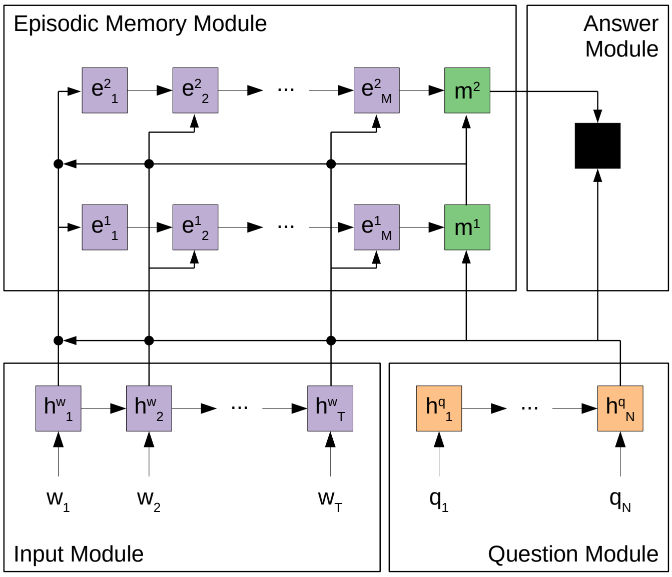

Figure 1: The Dynamic Memory Network architecture with two episodes. The black dots represent the concatenation of vectors. This figure depicts the case where each vector from the input module is used, i.e. $T = M$. In the case of a multi-sentence input, T is greater than M.

input consists of a single sentence, each of the GRU outputs is used; however, if the input consists of multiple sentences, only those GRU outputs h_t are used where t corresponds to an end-of-sentence token (for example periods or question marks), while the rest is discarded. We denote the final sequence of vectors produced by the input module as $s = (s_1, ..., s_M)$.

The **question module** is similar to the input module, as it is used to encode the query (or question) as a fixed-size vector representation. The word embeddings are fed into a many-to-one GRU, which outputs the query representation Q at the end, i.e. $h^q_t = \texttt{GRU}(q_t, h^q_{t-1})$ and $Q = h^q_N$.

The **episodic memory module** maintains a number of *episodes*. An episode e^i produces a *memory* m^i by iterating a GRU over the fact representations from the input module, while taking the previous memory m^{i-1} into account. For this, the GRU's update gate is replaced by a special attention gate at each time step,

$$\texttt{AttGRU}^i(x_t, h_{t-1}) = g^i_t \circ h'_t + (1 - g^i_t) \circ h_{t-1}, \tag{1}$$

where h'_t is the candidate hidden state for the GRU's internal update at time step t. The *attention gate* g^i_t is a function of the input x_t and the memory and question vectors, encoding their similarities (details can be found in [2]). The initial memory is initialized as $m^0 = Q$. The hidden state of an episode e^i is then computed as $e^i_t = \texttt{AttGRU}^i(c_t, e_{t-1})$, where $c_t = [s_t; m^{i-1}]$ is a *candidate fact* and $[\cdot;\cdot]$ denotes concatenation. The new memory value is then simply set to the last hidden state of the episode, i.e. $m^i = e^i_M$. Finally, the output of the episodic memory module is the last output of a GRU that iterates over all memories m^i.

The **answer module** generates the final output of the model and is therefore highly dependent on the task. In our case, it is a simple feed-forward layer that predicts a score to rank passages given the output of the episodic memory module.

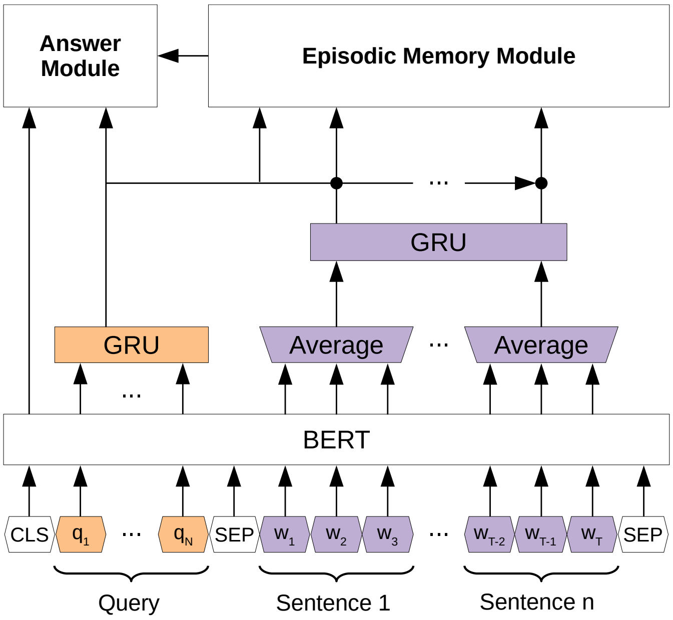

Figure 2: The BERT-DMN model architecture. Note that the padding tokens are omitted here.

3.2. Combining BERT and DMN

Dynamic Memory Networks have proven to be effective in QA tasks such as reading comprehension. In this paper we combine a DMN with contextualized representations, specifically the outputs of BERT, by modifying the input, question and answer module. The resulting model, BERT-DMN, takes all outputs of BERT into account (including the classification token). It processes the token-level outputs by creating query and sentence representations and reasons over them. In the final step, everything is combined to produce the final query-passage score, which is then used to rank the documents. Figure 2 shows the architecture of our approach.

3.2.1. Input and Question Module

Let the query and passage again be denoted by $q = (q_1, ..., q_N)$ and $w = (w_1, ..., w_T)$. We first construct the input for BERT as

$$[\texttt{CLS}], q_1, ..., q_N, [\texttt{SEP}], w_1, ..., w_T, [\texttt{SEP}]. \tag{2}$$

Note that q_i and w_i are not necessarily words, as BERT uses subword tokenization. This input format is identical to the usual way BERT is used, where the first input is a classification token, followed by two text inputs, which are separated by separator tokens.

We split the BERT output $o = (o_1, ..., o_L)$ back into two chunks, where one corresponds to the query and the other one to the passage. The outputs corresponding to the [SEP] tokens are discarded. We then use the token representations output by BERT as a replacement for the word embeddings in the DMN. In practice, instead of simply using the vector corresponding to

	ANTIQUE	InsuranceQA	TREC-DL
No. of train queries	2406	12889	808731
No. of test queries	200	2000	200
Avg. query length	10.55	8.42	6.53
No. of passages	33642	27413	8841823
Avg. passage length	47.83	103.59	64.63
Avg. no. of passages per query	32.95	500	1000
Avg. no. of relevant passages	9.6	1.66	1.69

Table 1
Dataset statistics. InsuranceQA and TREC-DL have dedicated devsets; for ANTIQUE, we use a small fraction of the trainset for validation. Query and passage lengths are measured in words.

the end-of-sentence token to represent the whole sentence, we take the vectors of all tokens in this sentence and average them.

3.2.2. Answer Module

Since the original DMN model was used for reading comprehension tasks, the answer module consisted of a sequence generation network. For the re-ranking task we are only interested in predicting a score, therefore we modify the answer module: Let the final memory be a vector $m \in \mathbb{R}^{d \times 1}$ and the BERT output $c \in \mathbb{R}^{b \times 1}$ correspond to the `[CLS]` token. We concatenate these vectors along with the query representation Q and compute the final score a using a feed-forward layer, i.e.

$$a = \sigma(W_a[c; Q; m] + b_a), \tag{3}$$

where σ is the sigmoid function and $W_a \in \mathbb{R}^{1 \times (b+2d)}$.

4. Experimental Setup

In this section we describe the datasets we use for our experiments and the baselines. We further outline the training process.

4.1. Datasets

We conduct experiments on three diverse passage ranking datasets:

1. **ANTIQUE** [33] is a non-factoid question answering benchmark based on the questions and answers of *Yahoo! Webscope L6*. The questions were filtered to remove ones too short or duplicate. A resulting sample of question-answer pairs was then judged by crowd workers who assigned one of four relevance labels to each pair. All questions have well-formed correct grammar. For the evaluation we follow the authors' recommendation and treat the two higher labels as relevant and the lower two labels as irrelevant.

2. **InsuranceQA** [34] is a dataset from the insurance domain released in 2015. For this work we use the second version, which comes with a predefined train-, dev- and testset. The dev- and testset include for each question the relevant answers as well as a pool of $n \in \{100, 500, 1000, 1500\}$ irrelevant candidate answers. For our experiments, we choose $n = 500$. All queries and passages in this dataset consist of gramatically well-formed sentences.
3. **TREC-DL 2019** is the passage ranking dataset from the TREC deep learning track. It uses MS MARCO [35], a collection of large *Machine Reading Comprehension* datasets released by Microsoft in 2016.[1] This dataset was created using real, anonymized queries from the Bing search engine. The authors automatically identified queries that represented questions and extracted passages from the top-10 search results. These passages then manually received relevance labels from human annotators. The result is a very large dataset with over 8M passages and 1M queries. However, a number of queries have no associated relevant passages. Because of the nature of this dataset, queries and passages are not guaranteed to be grammatically or structurally correct or even made of complete sentences.

Table 1 outlines some dataset statistics. The evaluation (except for the ANTIQUE testset) follows the telescoping setting [36], where a first round of retrieval has already been performed to select candidate passages that are relevant to the queries, followed by a re-ranking step by our models.

5. Baselines

Since we are mainly interested in improving the effectiveness and training efficiency of BERT-based models, the most important baseline is a vanilla **BERT** ranker [37]. The ranking is solely based on the output corresponding to the classification token, which is transformed into a scalar score using a feed-forward classification layer. Additionally, we implement other neural baselines:

1. **QA-LSTM** [6] is based on bidirectional LSTMs and attention. Both query and document are encoded using a shared bidirectional many-to-many LSTM and a pooling operation (maximum or average pooling) to the LSTM outputs. Attention scores are computed using the hidden LSTM states of the document and the pooled query representation. The resulting vectors are then compared using cosine similarity after applying dropout. We set the batch size to 32 and the number of LSTM hidden units to 256. We feed 300-dimensional pre-trained GloVe [38] embeddings to the shared LSTM and use a dropout rate of 0.5.
2. **K-NRM** [39] is a neural ranking model that works via kernel pooling. Starting from pre-trained word embeddings, it builds a *translation matrix*, where each row contains the cosine similarities of a query word to all document words. Each row is then fed into K kernel functions, and the results are pooled by summation. Finally, a single transformation with tanh activation is applied to output a score. The model is trained with a pairwise ranking loss and uses RBF kernels. We use 300-dimensional pre-trained

[1] http://www.msmarco.org/

GloVe embeddings to build the translation matrix. The hyperparameters are adopted from [39]: We set $K = 11$ and use one kernel for exact matches, i.e. $\mu_0 = 1$ and $\sigma_0 = 10^{-3}$. The remaining kernels are spaced evenly in $[-1, 1]$ with $\mu_1 = 0.9$, $\mu_2 = 0.7$, ..., $\mu_{10} = -0.9$ and $\sigma_1 = ... = \sigma_{10} = 0.1$. We use the Adam optimizer with a leaning rate of 0.001 and $\epsilon = 10^{-5}$ and a batch size of 16.

3. **Dynamic Memory Network** [2, 3] serves (in a slightly modified fashion) as the aggregation part of our model, which transforms sentence-level BERT outputs into a relevance score. We also train this model using pre-trained 300-dimensional word vectors in order to analyze if and how much BERT representations improve the performance. For these experiments we use the same DMN hyperparameters as in our experiments with BERT-DMN to make the results more comparable.

5.1. Training Efficiency

As previously mentioned, a drawback of BERT-based models is their training inefficiency, as the time required for even a single training epoch can be substantial, albeit a one-time cost. In order to mitigate this, we propose BERT-DMN$_{\text{lite}}$. While the model architecture remains identical, the BERT layer is excluded from backpropagation, such that its weights remain frozen. This reduces the training time in two ways: The time required to complete the first epoch will be slightly lower, as the majority of the weights are excluded from the backward pass; the second and all subsequent epochs can be sped up significantly, as the BERT outputs can be cached and re-used.

5.2. Training Details

Our models are implemented using PyTorch. We use a pre-trained, uncased BERT$_{\text{Base}}$ model with 12 encoder layers, 12 attention heads and 768-dimensional vector representations. The training is done as follows: We feed all query-passage pairs through the BERT layer to obtain the token representations. We then compute the average of all vectors for each sentence to obtain the inputs for the GRU, which in turn produces representations that serve as the inputs of the episodic memory module. Similarly, we use another GRU to encode the query as a single vector. In the case of BERT-DMN, the fine-tuning of BERT and training of the DMN happens jointly. For BERT-DMN$_{\text{lite}}$, all weights corresponding to BERT are frozen, i.e. they remain unchanged during the optimization. BERT inputs are truncated if they exceed 512 tokens.

The models are trained using the AdamW optimizer with the learning rate set to $3 \cdot 10^{-5}$, following [37], and a pairwise max-margin loss: Let q be a query and p^+ and p^- passages, where p^+ is more relevant to q than p^-. The loss is computed as

$$\mathscr{L} = \max\left\{0, m - R\left(q, p^+\right) + R\left(q, p^-\right)\right\} \tag{4}$$

where m is the margin and R is the model. We use $m = 0.2$ and linear warm-up over the first 1000 steps (10000 on TREC-DL). The DMN hyperparameters are set to 4 episodes, 256-dimensional hidden representations and a dropout rate of 0.1. Dropout is applied at the DMN input, over the attention gates and before the output layer. We use a batch size of 32 throughout our experiments. Validation is performed based on MAP on the devset. We use the same fixed random seed and thus identical training data for all experiments.

	ANTIQUE		InsuranceQA		TREC-DL	
	MAP	MRR	MAP	MRR	MAP	MRR
QA-LSTM	0.488	0.619	0.185	0.231	0.193	0.519
K-NRM	0.511	0.654	0.176	0.215	0.237	0.567
DMN	0.491	0.613	0.092	0.118	0.136	0.274
BERT$_{\text{lite}}$	0.593	0.774	0.259	0.314	0.327	0.739
BERT-DMN$_{\text{lite}}$	0.675	0.851	0.374	0.449	0.418	0.859
BERT	0.697	0.849	0.399	0.476	**0.428**	0.831
BERT-DMN	**0.700**	**0.866**	**0.406**	**0.484**	0.408	**0.889**

Table 2
Passage re-ranking performance. BERT$_{\text{lite}}$ and BERT-DMN$_{\text{lite}}$ are architecturally identical to BERT and BERT-DMN, respectively, but only the classification layer is trained, while all other weights remain frozen. The baselines use pre-trained GloVe embeddings.

5.3. Metrics

The *mean reciprocal rank* (MRR) is defined as

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{5}$$

where Q is the set of all queries and rank_i refers to the highest rank of any relevant document for the i-th query.

Similarly, *mean average precision* (MAP) is defined as

$$\text{AP}(q) = \frac{1}{|R_q|} \sum_{k=1}^{n} \text{P}(k) \times \text{rel}(k) \tag{6}$$

$$\text{MAP} = \frac{1}{|Q|} \sum_{q \in Q} \text{AP}(q) \tag{7}$$

where R_q is the set of all documents relevant to q, n is the total number of retrieved documents, $\text{P}(k)$ is the precision and $\text{rel}(k)$ indicates the relevance of the document at rank k.

6. Results

In this section we present and discuss our results.

6.1. Passage Re-Ranking Performance

Table 2 outlines the passage re-ranking performance of our methods and the baselines on three datasets. It is evident that the BERT-based methods vastly outperform the other baselines on all datasets. BERT$_{\text{lite}}$ performs noticeably worse, but still shows improvements over the non-contextual baselines. Finally, BERT-DMN improves the performance of BERT in all but one case. These results yield the following insights:

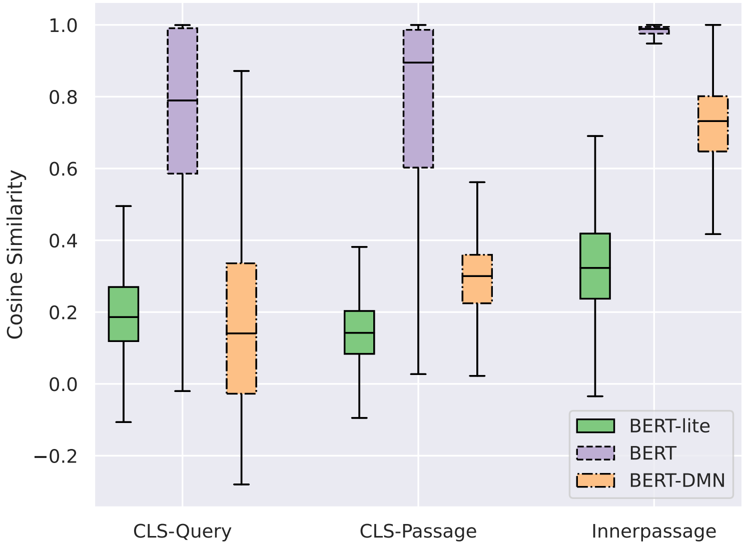

Figure 3: The diffusion of information within BERT representations on TREC-DL, illustrated by cosine similarities between classification token, query tokens and passage tokens.

1. As expected, the contextual token representations obtained from BERT trump non-contextual word embeddings. Even without any fine-tuning, the BERT representations perform well.
2. The contextual sentence representations do in fact hold valuable information. This information is discarded by models which only use the output corresponding to the classification token. End-to-end training further improves the performance.

As a result, the DMN profits vastly from BERT representations (BERT-DMN$_{\text{lite}}$), and the performance improves even more when the model is trained end-to-end (BERT-DMN).

6.2. The Effect of Fine-Tuning

In order to analyze the effect of fine-tuning the parameters of BERT, we conduct additional experiments using *lite* versions of BERT and BERT-DMN. The architectures and hyperparameters of these models are unchanged, however the number of trainable parameters is reduced roughly from 110M to 3M (BERT-DMN$_{\text{lite}}$) or 1k (BERT$_{\text{lite}}$) by freezing the BERT model itself. Table 2 shows slight performance drops of BERT-DMN$_{\text{lite}}$ in all but one case. However, comparing it to the fine-tuned vanilla BERT model shows even smaller differences, and in some cases the performance increases. Conversely, BERT$_{\text{lite}}$ exhibits a much higher loss of performance over BERT. This indicates that most of the information required for the task is already inherent to the pre-trained BERT model, and fine-tuning its parameters is merely required to direct it towards the desired output (usually the classification token). In order to confirm this hypothesis, we adopt a method proposed by Goyal et al. [40] to measure the *diffusion of information* within the contextual representations output by BERT: Given a query-passage pair, we use a BERT model

	ANTIQUE	InsuranceQA
BERT	1.71	1.69
BERT-DMN$_{\text{lite}}$	2.26 → 5.32	2.55 → 5.67

Table 3
The average number of training batches (size 16) per second (higher is better). For BERT-DMN$_{\text{lite}}$, we report one number for the first epoch and one number for all subsequent epochs.

to obtain a representation (in our case a 768-dimensional vector) of each token, corresponding to either query or passage. We then use cosine similarity to compute diffusion of information in three ways:

1. **CLS-Query**: Cosine similarity between the classification token and each query token.
2. **CLS-Passage**: Cosine similarity between the classification token and each passage token.
3. **Innerpassage**: Cosine similarity between each possible pair of two passage tokens.

The results are illustrated in Figure 3 for three BERT models, one without any fine-tuning (BERT$_{\text{lite}}$), one with standard fine-tuning using only the classification output (BERT) and finally one fine-tuned as part of our approach (BERT-DMN). These measurements were performed on roughly 10% of the TREC-DL testset (20k query-passage pairs). We observe that, without any fine-tuning, the outputs are rather dissimilar; with standard fine-tuning, however, the similarity of all representations vastly increases, especially within the passages. The same trend is exhibited by the model fine-tuned with BERT-DMN, but to a much lesser extend. This shows that discarding all but one output during fine-tuning leads to very high diffusion, in that all output vectors become very similar, and taking all outputs into account during fine-tuning alleviates this issue, allowing for a slight performance gain. It further suggests that BERT-DMN$_{\text{lite}}$ is able to combine the classification output and the sentence representations, performing closely to a fine-tuned BERT model.

6.3. Training Efficiency

Since the performances of BERT-DMN$_{\text{lite}}$ and BERT are comparable (cf. Table 2), BERT-DMN$_{\text{lite}}$ can be seen as an alternative to the usual fine-tuning of a BERT model. Since the DMN layer has very few parameters compared to BERT (roughly 3M vs. 100M), the size of the model itself does not change a lot. However, BERT-DMN$_{\text{lite}}$ exhibits noticeable improvements in training efficiency compared to fine-tuning BERT. In order to show this, we measure the number of batches per seconds for both models in Table 3. For BERT-DMN$_{\text{lite}}$, the first epoch is already slightly faster, as the majority of the weights are excluded from the backward pass; the second and all subsequent epochs are sped up significantly, as the BERT outputs can be cached re-used for the remainder of the training. The measurements were performed on a single non-shared NVIDIA GTX 1080Ti GPU.

7. Conclusion and Outlook

The exponential growth in the searchable web [41] has resulted in the proliferation of numerous knowledge-intensive tasks [42, 43], of which question answering tasks are prominent [44, 45]. In this paper we introduced BERT-DMN and BERT-DMN$_{\text{lite}}$, extensions of BERT that utilize dynamic memory networks to perform passage re-ranking. We have shown that our model improves the performance of BERT on three datasets. Moreover, BERT-DMN$_{\text{lite}}$ performs well even without a fine-tuned BERT model, reducing the training time while incurring only a small performance hit. Our findings demonstrate that fine-tuning BERT-based models is not always necessary, as nearly the same result can be achieved using sentence-level representations.

There are many ways to extend BERT-DMN. Firstly, a common problem of over-parameterized models like BERT is that they are less interpretable. There is some initial work in the direction of understanding the rationale behind QA and passage ranking tasks by either sparsification [46], inspecting BERT's parametric memory [47], or in a post-hoc manner [48, 49]. We see the DMN as an interpretable approach to evidence selection for question answering. The dynamic memory module in some sense iteratively computes attention on sentences that reflects their relative importance. We could use this observation to build an interpretable-by-design approach to passage ranking given questions by highlighting evidence sentences from the episodic memory module. Secondly, outside of text datasets, we envision the utility of the DMN in question answering over semi-structured data on the web like anchor text [50], semantic annotations [42], tables [51] and fully structured knowledge graphs. Specifically, the transitive reasoning capability is natural to structured information organized as triples or in a graph.

References

[1] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[2] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, R. Socher, Ask me anything: Dynamic memory networks for natural language processing, in: International conference on machine learning, 2016, pp. 1378–1387.

[3] C. Xiong, S. Merity, R. Socher, Dynamic memory networks for visual and textual question answering, in: International conference on machine learning, 2016, pp. 2397–2406.

[4] E. M. Voorhees, The trec-8 question answering track report, in: In Proceedings of TREC-8, 1999, pp. 77–82.

[5] C. Kwok, O. Etzioni, O. Etzioni, D. S. Weld, Scaling question answering to the web, ACM Transactions on Information Systems (TOIS) 19 (2001) 242–262.

[6] M. Tan, C. Dos Santos, B. Xiang, B. Zhou, Improved representation learning for question answer matching, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 464–473.

[7] N. K. Tran, C. Niederée, Multihop attention networks for question answer matching, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2018, pp. 325–334.

[8] S. Wang, M. Yu, J. Jiang, W. Zhang, X. Guo, S. Chang, Z. Wang, T. Klinger, G. Tesauro,

M. Campbell, Evidence aggregation for answer re-ranking in open-domain question answering, in: International Conference on Learning Representations, 2018.

[9] S. Wang, M. Yu, X. Guo, Z. Wang, T. Klinger, W. Zhang, S. Chang, G. Tesauro, B. Zhou, J. Jiang, R 3: Reinforced ranker-reader for open-domain question answering, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[10] Y. Lin, H. Ji, Z. Liu, M. Sun, Denoising distantly supervised open-domain question answering, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 1736–1745.

[11] D. Chen, A. Fisch, J. Weston, A. Bordes, Reading wikipedia to answer open-domain questions, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 1870–1879.

[12] T. Wang, X. Yuan, A. Trischler, A joint model for question answering and question generation, arXiv preprint arXiv:1706.01450 (2017).

[13] B. Kratzwald, S. Feuerriegel, Adaptive document retrieval for deep question answering, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 576–581. URL: https://www.aclweb.org/anthology/D18-1055. doi:10.18653/v1/D18-1055.

[14] P. Xu, X. Ma, R. Nallapati, B. Xiang, Passage ranking with weak supervision, in: International Conference on Learning Representations, 2019.

[15] W. Guo, X. Liu, S. Wang, H. Gao, A. Sankar, Z. Yang, Q. Guo, L. Zhang, B. Long, B.-C. Chen, D. Agarwal, Detext: A deep text ranking framework with bert, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, CIKM '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 2509–2516. URL: https://doi.org/10.1145/3340531.3412699. doi:10.1145/3340531.3412699.

[16] J. Zhan, J. Mao, Y. Liu, M. Zhang, S. Ma, An analysis of bert in document ranking, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 1941–1944. URL: https://doi.org/10.1145/3397271.3401325. doi:10.1145/3397271.3401325.

[17] M. E. Peters, S. Ruder, N. A. Smith, To tune or not to tune? adapting pretrained representations to diverse tasks, in: I. Augenstein, S. Gella, S. Ruder, K. Kann, B. Can, J. Welbl, A. Conneau, X. Ren, M. Rei (Eds.), Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019, Florence, Italy, August 2, 2019., Association for Computational Linguistics, 2019, pp. 7–14. URL: https://www.aclweb.org/anthology/W19-4302/.

[18] S. MacAvaney, A. Yates, A. Cohan, N. Goharian, Contextualized word representations for document re-ranking, arXiv preprint arXiv:1904.07094 (2019).

[19] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, 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, CIKM '14, ACM, 2014, pp. 101–110. URL: http://doi.acm.org/10.1145/2661829.2661935. doi:10.1145/2661829.2661935.

[20] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using clickthrough data, in: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, CIKM '13,

ACM, 2013, pp. 2333–2338. URL: http://doi.acm.org/10.1145/2505515.2505665. doi:10.1145/2505515.2505665.

[21] Y. Shen, X. He, J. Gao, L. Deng, G. Mesnil, Learning semantic representations using convolutional neural networks for web search, in: Proceedings of the 23rd International Conference on World Wide Web, WWW '14 Companion, ACM, 2014, pp. 373–374. URL: http://doi.acm.org/10.1145/2567948.2577348. doi:10.1145/2567948.2577348.

[22] C. Xiong, Z. Dai, J. Callan, Z. Liu, R. Power, 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, SIGIR '17, ACM, 2017, pp. 55–64. URL: http://doi.acm.org/10.1145/3077136.3080809. doi:10.1145/3077136.3080809.

[23] J. Guo, Y. Fan, Q. Ai, W. B. Croft, A deep relevance matching model for ad-hoc retrieval, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, ACM, 2016, pp. 55–64. URL: http://doi.acm.org/10.1145/2983323.2983769. doi:10.1145/2983323.2983769.

[24] B. Mitra, F. Diaz, N. Craswell, 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, 2017, pp. 1291–1299.

[25] L. Pang, Y. Lan, J. Guo, J. Xu, X. Cheng, A study of MatchPyramid models on ad-hoc retrieval, SIGIR workshop on Neural Information Retrieval (NeuIR-16) arXiv:1606.04648 (2016). URL: http://arxiv.org/abs/1606.04648. arXiv:1606.04648.

[26] Y. Nie, Y. Li, J.-Y. Nie, 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, ICTIR '18, ACM, 2018, pp. 59–66. URL: http://doi.acm.org/10.1145/3234944.3234954. doi:10.1145/3234944.3234954.

[27] Y. Nie, A. Sordoni, J.-Y. Nie, Multi-level abstraction convolutional model with weak supervision for information retrieval, in: the 41st International ACM SIGIR Conference, SIGIR '18, ACM, 2018, pp. 985–988. URL: http://doi.acm.org/10.1145/3209978.3210123. doi:10.1145/3209978.3210123.

[28] K. Hui, A. Yates, K. Berberich, G. de Melo, PACRR: A position-aware neural ir model for relevance matching, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 2017, pp. 1049–1058. URL: https://www.aclweb.org/anthology/D17-1110.

[29] K. Hui, A. Yates, K. Berberich, G. de Melo, Co-PACRR: A context-aware neural ir model for ad-hoc retrieval, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM '18, ACM, 2018, pp. 279–287. URL: http://doi.acm.org/10.1145/3159652.3159689. doi:10.1145/3159652.3159689.

[30] R. McDonald, G. Brokos, I. Androutsopoulos, Deep relevance ranking using enhanced document-query interactions, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, ACL, 2018, pp. 1849–1860. URL: http://aclweb.org/anthology/D18-1211.

[31] F. Diaz, B. Mitra, N. Craswell, Query expansion with locally-trained word embeddings, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 367–377. URL: http://aclweb.org/anthology/P16-1035.

doi:10.18653/v1/P16-1035.

[32] H. Zamani, W. B. Croft, Embedding-based query language models, in: Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval, ICTIR '16, ACM, 2016, pp. 147–156. URL: http://doi.acm.org/10.1145/2970398.2970405. doi:10.1145/2970398.2970405.

[33] H. Hashemi, M. Aliannejadi, H. Zamani, W. B. Croft, ANTIQUE: A non-factoid question answering benchmark, CoRR abs/1905.08957 (2019). URL: http://arxiv.org/abs/1905.08957.

[34] M. Feng, B. Xiang, M. R. Glass, L. Wang, B. Zhou, Applying deep learning to answer selection: A study and an open task, in: 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), IEEE, 2015, pp. 813–820.

[35] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng, Ms marco: A human-generated machine reading comprehension dataset (2016).

[36] I. Matveeva, C. Burges, T. Burkard, A. Laucius, L. Wong, High accuracy retrieval with multiple nested ranker, in: Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, ACM, 2006, pp. 437–444. URL: http://doi.acm.org/10.1145/1148170.1148246. doi:10.1145/1148170.1148246.

[37] R. Nogueira, K. Cho, Passage re-ranking with bert, arXiv preprint arXiv:1901.04085 (2019).

[38] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[39] C. Xiong, Z. Dai, J. Callan, Z. Liu, R. Power, 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, ACM, 2017, pp. 55–64.

[40] S. Goyal, A. R. Choudhury, S. Raje, V. Chakaravarthy, Y. Sabharwal, A. Verma, PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination, in: H. D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, volume 119 of *Proceedings of Machine Learning Research*, PMLR, 2020, pp. 3690–3699. URL: http://proceedings.mlr.press/v119/goyal20a.html.

[41] H. Holzmann, W. Nejdl, A. Anand, The dawn of today's popular domains: A study of the archived german web over 18 years, in: 2016 IEEE/ACM Joint Conference on Digital Libraries (JCDL), IEEE, 2016, pp. 73–82.

[42] H. Holzmann, W. Nejdl, A. Anand, Exploring web archives through temporal anchor texts, in: Proceedings of the 2017 ACM on Web Science Conference, 2017, pp. 289–298.

[43] J. Singh, W. Nejdl, A. Anand, 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, 2016, pp. 1105–1108.

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

[45] A. Anand, L. Cavedon, H. Joho, M. Sanderson, B. Stein, Conversational search (dagstuhl seminar 19461), in: Dagstuhl Reports, volume 9, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[46] Z. Zhang, K. Rudra, A. Anand, Explain and predict, and then predict again, in: WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual

Event, Israel, March 8-12, 2021, ACM, 2021, pp. 418–426. URL: https://doi.org/10.1145/3437963.3441758. doi:10.1145/3437963.3441758.

[47] J. Singh, J. Wallat, A. Anand, Bertnesia: Investigating the capture and forgetting of knowledge in bert, in: Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, 2020, pp. 174–183.

[48] Z. T. Fernando, J. Singh, A. Anand, A study on the interpretability of neural retrieval models using deepshap, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 1005–1008.

[49] J. Singh, A. Anand, Model agnostic interpretability of rankers via intent modelling, in: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, 2020, pp. 618–628.

[50] H. Holzmann, A. Anand, Tempas: Temporal archive search based on tags, in: Proceedings of the 25th International Conference Companion on World Wide Web, 2016, pp. 207–210.

[51] B. Fetahu, A. Anand, M. Koutraki, Tablenet: An approach for determining fine-grained relations for wikipedia tables, in: The World Wide Web Conference, 2019, pp. 2736–2742.