Benchmarking a transformer-FREE model for ad-hoc retrieval

Tiago Almeida
IEETA
Universidade de Aveiro
Aveiro, Portugal
tiagomeloalmeida@ua.pt

Sérgio Matos
DETI/IEETA
Universidade de Aveiro
Aveiro, Portugal
aleixomatos@ua.pt

Abstract

Transformer-based "behemoths" have grown in popularity, as well as structurally, shattering multiple NLP benchmarks along the way. However, their real-world usability remains a question. In this work, we empirically assess the feasibility of applying transformer-based models in real-world ad-hoc retrieval applications by comparison to a "greener and more sustainable" alternative, comprising only 620 trainable parameters. We present an analysis of their efficacy and efficiency and show that considering limited computational resources, the lighter model running on the CPU achieves a 3 to 20 times speedup in training and 7 to 47 times in inference while maintaining a comparable retrieval performance. Code to reproduce the efficiency experiments is available on https://github.com/bioinformatics-ua/EACL2021-reproducibility/.

1 Introduction

The Natural Language Processing (NLP) field has been revolutionised by the simple, yet extremely successful idea of transfer learning allied to the transformer neural architecture capability of exploring long sequences, while being computationally more efficient than the recurrent counterparts. More precisely, Dai and Le (2015) and later Howard and Ruder (2018) introduced the transfer learning approach to the NLP field by pre-training a language model and then fine-tuning it to multiple NLP tasks. However, it was Radford (2018) who applied the same technique to a large transformer encoder, shattering the SOTA in 9 of 12 NLP tasks.

Since then, the NLP field became dominated by works that use, explore or improve, in some way, the transformer-based architecture. Although their performance is undeniable, these are large models that comprise millions of parameters and require large computational resources to use and maintain, making them inoperable for the majority of smaller institutions. Furthermore, it also gives the impression that the NLP field was reduced to throwing a lot of computation power (money) to continually achieve SOTA results in multiple benchmarks, leaving behind the careful process of designing a neural solution that follows the human intuition to solve some specific task. Moreover, given their running costs and the consequent impact on the environment, their real-world virtues remain questionable, i.e., "is it really feasible to use these models in real-world applications, such as search engines or question answering systems to aid a broader range of people in their day-to-day tasks?". As an example, biomedical experts need to routinely search an unprecedented amount of scientific literature to keep updated with their research, which could be facilitated by an intelligent system.

In contradiction with this trend, this work examines a lightweight interaction-based model, with only 620 trainable parameters, carefully designed by considering years of research in ad-hoc retrieval systems and interaction-based architectures. This system was evaluated in two ad-hoc retrieval competitions where it was able to compete with the state-of-the-art transformer-based models. In this paper, we demonstrate a "greener and more sustainable" alternative to the transformer-based architecture, by thoroughly testing such a system against the most popular transformer behemoths in a series of performance experiments.

Following this section, we frame our work in the current literature, we then present the lightweight model, and finally detail and discuss the evaluation performed.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 3343–3353

2 Related Work

The Information Retrieval (IR) field is nowadays considered as being divided into traditional IR (Baeza-Yates and Ribeiro-Neto, 2011) and neural IR (Mitra and Craswell, 2018). The former explores exact match signals, i.e., the co-occurrence of query terms in the document, to derive handcrafted rules and formulas to directly compute the query-document importance, with the BM25 (Robertson and Zaragoza, 2009) ranking function being the most popular example. On the other end, neural IR explores the increasing success of neural networks to approximate a (sub)optimal ranking function by exploiting labelled examples. In the literature, the most successful neural IR architectures are Interaction Based, which measure multiple matching signals to create a joint representation of the query and the documents. Moreover, considering that the transformer architecture is now widely adopted in interaction-based architectures, we further propose to subdivide the works into shallow interaction-based and transformer interaction-based models.

2.1 Shallow Interaction-Based Models

In this subsection, we address some relevant interaction-based neural models proposed before the transformer revolution, and briefly present their intuition to tackle the ad-hoc retrieval challenge. Guo et al. (2016) proposed the DRMM model, which was one of the first neural models to achieve improvements over strong traditional IR solutions, and showed the importance of individually weighing each query term's contributions. Likewise, Pang et al. (2016); Hui et al. (2017); Dai et al. (2018) showed that 2D convolutions directly applied over the interaction matrix are capable of extracting strong hierarchical n-gram matching patterns. Pang et al. (2017) built an end-to-end neural solution, inspired by the human process of selecting relevant documents. Allied to the semantic matching signals, Fan et al. (2018) added, in parallel, exact matching signals and discussed ways of combining them.

2.2 Transformer Interaction-Based Models

Despite representing a recent trend, there is already an extensive literature on transformer architectures. Here we briefly introduce this architecture and its most adopted variants.

Models that follow a transformer-based architecture are composed of a fixed set of stacked transformer blocks (Vaswani et al., 2017), hence their name, and learn useful word representations from large text corpora. Since the first appearance (Radford, 2018), multiple other variants emerged, arguably the most notable and widely adopted being BERT (Devlin et al., 2019), which uses bidirectional self-attention and requires being trained as a masked language model. However, this model has some limitations. One of them is its large size, which is addressed by distillation models, such as distilBERT (Sanh et al., 2020), or by factorisation tricks to reduce the number of trainable parameters, as in Lan et al. (2020). Another limitation is their short input length (512 tokens), which derives from the quadratic complexity and associated memory requirements imposed by the self-attention mechanism. Beltagy et al. (2020); Wang et al. (2020b); Katharopoulos et al. (2020); Zaheer et al. (2020) address this issue by reducing this complexity order, hence increasing the input length.

Regarding the information retrieval field, some works tried to apply these models to the ad-hoc retrieval task with varying approaches. Yang et al. (2019) proposed to perform "best" relevance sentence inference and then linearly interpolate with the original document score (BM25). Dai and Callan (2019b) discuss different strategies to approximate the document relevance by aggregating the relevance across sentences. Nogueira et al. (2019) proposed a three-stage retrieval pipeline and introduced monoBERT and duoBERT, two BERT models aimed at addressing pointwise and pairwise ranking problems, respectively. Finally, MacAvaney et al. (2019) proposed to use the context-aware embedding produced by these models as the input embedding to the well studied shallow interaction-based models. Furthermore, and also acknowledged by MacAvaney et al. (2019), these models are still not suitable to be used or implemented as real-world solutions given their slow inference times and expensive costs.

Some notable works address the issue of computational costs by introducing precomputation, i.e., compute and store for later reuse. Dai and Callan (2019a) propose to precompute term weights that are then used by efficient traditional retrieval systems such as BM25. MacAvaney et al. (2020), on the other hand, precompute the term representation of the document at index time to be later merged with the query representation at query time.

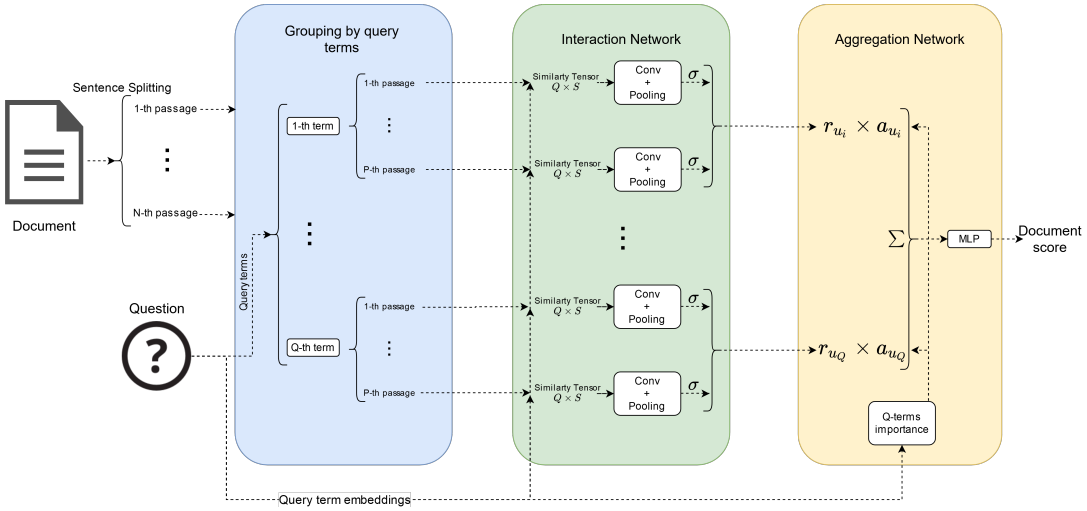

Figure 1: Lightweight neural model data and operation flow.

3 Proposed neural interaction based model

Our neural interaction-based model is an enhancement of Almeida and Matos (2020a) and was already used in the two international competitions, namely BioASQ (Almeida and Matos, 2020b; Tsatsaronis et al., 2015) and TREC-Covid (Almeida and Matos, 2020c; Roberts et al., 2020). However, in order to keep this paper self-contained, we will now introduce its insight and architecture.

From a general perspective, our model follows the successful ideas presented over years of research in the neural IR field, taking into its implementation key concepts of previous works. More precisely, it follows the DeepRank (Pang et al., 2017) intuition to mimic the human judgment process with neural networks; adopts the extraction capabilities, based on 2D convolution, of the PACRR (Hui et al., 2017) and MatchPyramid (Pang et al., 2016) models; explores the same assumption of independently scoring each sentence as addressed by McDonald et al. (2018); and finally, uses the DRMM (Guo et al., 2016) query term gating mechanism to consider the importance of different query terms. Overall, the model was designed with the intuition of weighting the importance of the document sentences in terms of the query by considering the context where the exact match occurs, i.e., this model produces a more refined judgment of the initial exact match signal evaluated by the BM25.

Regarding the architecture, the neural model is divided into three major blocks that can be observed in Figure 1. To clearly describe the model, let us first define a **query** as a sequence of tokens $q = \{u_0, u_1, ..., u_Q\}$, where u_i is the i-th token of the query and Q the size of the query; a **document passage** as $p = \{v_0, v_1, ..., v_S\}$, where v_j is the j-th token of the passage and S the size of the passage; and a document as a sequence of passages $D = \{p_0, p_1, ..., p_N\}$, where N is the total number of passages in the document. In the current implementation, a passage corresponds to a document sentence, as described below.

The first block aims to rearrange the input, query and document, to a suitable format for the downstream interaction model. More precisely, it splits the document into passages using the Punkt algorithm (Kiss and Strunk, 2006) and rearranges the passages by query-term, so that each query-term is associated with a set of passages where that term occurs. At this point, the document is represented by $D(u_i) = \{p_{i0}, p_{i1}, ..., p_{iP}\}$, where p_{ik} corresponds to the k-th passage with respect to the query term u_i, and P is the maximum number of passages that can be associated with each query-term. Note that it is possible to have repeated passages for different query-terms at this point since one passage can contain more than one query-term.

The second block, named Interaction Network, has the objective of scoring each passage according to its relevance for a given query. To accomplish this, we employed a shallow interaction-based model to capture the most relevant matching signals that are present in each query-passage combination. In more detail, for each query-term-passage pair, we construct an interaction matrix with dimension $Q \times S$, also designated similarity tensor, by computing the cosine similarity between the query and the

passage embedding. Then 3 by 3 convolutions are applied to learn n-gram relevance signals that are further extracted by multiple pooling layers (max, average, and k-max-average) applied to the filter dimension, creating a feature vector r per each query-term-passage pair. Consequently, $D(u_i)$ at this point is given by $D(u_i) = \{r_{i0}, r_{i1}, ..., r_{iP}\}$, where r_{ik} corresponds to the k-th passage feature vector with respect to the query term u_i. Finally, each resulting feature vector is linearly combined with a trainable vector, and a sigmoid function, σ, is applied to produce a sentence level relevance signal (1=relevant, 0=irrelevant).

The third block, the Aggregation Network, is responsible for producing the final document relevance score for the given query. This is accomplished by weighting the importance of each document passage score with respect to the importance of each query term. More precisely, this importance is given by a probabilistic distribution computed (softmax) over a linear combination of each query term embedding and a trainable vector and is defined as $A = \{a_{u_0}, a_{u_1}, ..., a_{u_Q}\}$, where a_{u_i} represents the probabilistic importance of the query-term u_i. Each passage score is then weighted by the corresponding query-term importance, a_{u_i}, and the resulting set of weighted scores is fed to a multi-layer perceptron to produce the final document score.

Our model is trained in a pairwise fashion and uses word-level pre-trained embeddings, specifically word2vec embeddings (Mikolov et al., 2013) in these experiments. More details on hyperparameters, configuration, and training can be found in previous work Almeida and Matos (2020b).

4 Evaluation

In this section, we aim to empirically demonstrate the efficacy of the proposed model and compare its efficiency against the state-of-the-art transformer-based models.

4.1 Efficacy

To compare the retrieval efficacy of the lightweight system and show that despite having only 620 trainable parameters, it is capable of competing with state-of-the-art models, namely transformer-based models, we reuse the results obtained in two ad-hoc retrieval competitions. Concretely, the results in both challenges derive from a two-stage retrieval pipeline, where our proposed neural model reranks the top-N documents previously retrieved by the BM25 (Robertson and Zaragoza, 2009) ranking function.

4.1.1 BioASQ - Challenge

The BioASQ challenge (Tsatsaronis et al., 2015) is an annual competition on document classification, retrieval, and question-answering, currently in the eighth edition. For the document retrieval task, the objective was to retrieve the most relevant articles from the PubMed/MEDLINE annual database.

Concerning the system, the BM25 filter was fine-tuned with the 2700 biomedical questions provided by the organisers as training data. Furthermore, the neural model was trained on the same data using a pairwise cross-entropy loss with cyclic learning rates. We also trained the word embeddings using word2vec (Mikolov et al., 2013) skip-gram algorithm directly on the PubMed/MEDLINE articles. For more details about our participation, we direct the readers to the system description paper (Almeida and Matos, 2020b).

System	Document Retrieval		
	Rank	MAP@10	GMAP@10
Batch 1 (total of 21 systems)			
Proposed Model	1	**38.23**	**1.63**
Top Competitor (Pappas et al., 2020)	4	36.48	1.14
Batch 2 (total of 26 systems)			
Proposed Model	3	37.19	**6.75**
Top Competitor (Kazaryan et al., 2020)	1	**39.45**	6.00
Batch 3 (total of 28 systems)			
Proposed Model	5	52.30	**7.63**
Top Competitor (Pappas et al., 2020)	1	**53.29**	6.25
Batch 4 (total of 26 systems)			
Proposed Model	3	48.10	**7.96**
Top Competitor (Pappas et al., 2020)	1	**49.92**	7.00
Batch 5 (total of 25 systems)			
Proposed Model	2	50.98	**6.52**
Top Competitor (Kazaryan et al., 2020)	1	**52.02**	6.34

Table 1: Summary of the results obtained in BioASQ. Values in bold represent the top score achieved during the competition.

The results summarised in Table 1 show that the system provided the best ranking overall in terms of GMAP@10 for all five batches and achieved top 3 results in all but one batch, in terms of MAP@10. Additionally, this challenge received an average of 25 submissions for each batch and, according to Nentidis et al. (2020), all the other top-performing systems used either BERT or a BERT variant.

4.1.2 TREC-Covid - Challenge

TREC-Covid (Roberts et al., 2020) was an initiative to rapidly promote the development of an automatic system capable of searching the growing literature about the 2019 novel coronavirus. The

challenge followed a TREC style format and relied on the CORD-19 dataset (Wang et al., 2020a) as the collection of scientific articles about the novel coronavirus. The objective was to retrieve the most relevant articles from this collection for each topic given by the organisers. The system results were evaluated in a residual manner since, with the exception of the first round, the rounds shared topics that had already been evaluated. The metrics adopted were P@5, NDCG@10, Brepf, and MAP. The organisers also allowed the use of the evaluation feedback of previous rounds as training data to tune/train the submitted systems. An important note is that for the first round no training data was available since no previous evaluation had been performed.

Concerning the system, we utilised the BioASQ system in the first round, which means this was a zero-shot approach, exploiting the proximity of the domains. The only change was on the embeddings used, which were trained on the PubMed/MEDLINE articles plus the CORD-19 dataset. For the remaining rounds, we kept a similar approach, but we also fine-tuned (trained) the model with the feedback data from previous rounds. A complete description of our participation is available in Almeida and Matos (2020c).

System	Rank	P@5	NCDG@10
Round 1 (total of 100 automatic runs)			
Proposed Model	9	63.33	52.98
Top Competitor*	1	78.00	60.80
Round 3 (total of 79 runs)			
Proposed Model	2	**86.50**	77.15
Top Competitor** (Zhang et al., 2020)	1	86.00	77.40

Runs description:
*https://ir.nist.gov/covidSubmit/archive/round1/sab20.1.meta.docs.pdf
**https://ir.nist.gov/covidSubmit/archive/round3/covidex.r3.t5_lr.pdf

Table 2: Summary of the two best results achieved on TREC-Covid.

Comparatively to BioASQ, TREC-Covid was more challenging given the high number of participating teams. As an example, the first round had a total of 53 teams and approximately 100 submissions in the same category as our submission and hence comparable. This large number also covers a variety of IR solutions, ranging from simple traditional IR to transformer interaction-based models. Table 2 shows our best results, a ninth-place out of 100 in the first round and a second-place out of 73 in the third round. In both rounds, our system was able to beat traditional IR techniques and more recent transformer-based architectures, such as BERT and its variants, T5 (Raffel et al., 2020), among others. Additionally, our round 3 submission differed from the overall system since, motivated by the residual nature of the evaluation, we ended up utilising a (pseudo)-relevance feedback baseline instead of our BM25 baseline.

4.2 Performance

In this section, we address the performance side of the proposed neural model by empirically comparing it to the transformer-based behemoths. We propose three dimensions of comparison, namely the number of parameters, time to infer document relevance, and time to train. The first dimension has the objective of giving an idea of the model's size. The second dimension is especially concerned with proving the feasibility of using these models in real-world applications. Finally, the third dimension is concerned with the impact and resources required to build systems with these models.

All the measures were performed on a machine with the characteristics presented in Table 3, which we consider a good representative of a generally accessible setup. Additionally, all the experiments were performed on TensorFlow 2.2.0 with CUDA 10.1, and for the transformer-based models, we adopted the HuggingFaces (Wolf et al., 2020) library with the TensorFlow version of the models. To keep the comparison fair we utilize the "tf.function"[1] decorator to convert the model to a static computation graph, significantly speeding up the transformer-based models.

CPU	2x Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz
GPU	Nvidia Tesla K80 with 12 GB
RAM	128 GB

Table 3: Specification of the hardware used during the experiments.

An important aspect is that we do not have access to the systems listed in Section 4.1. To keep a fair comparison, we implemented a simple minimal working example that uses a transformer-based model for ad-hoc retrieval. Pursuing the ideas of (Yang et al., 2019; Nogueira et al., 2019; Dai and Callan, 2019b), we fed to the model the following input "[CLS] query tokens [SEP] document tokens [SEP]", where "[CLS]" and "[SEP]" are special boundary tokens, with the first one being a classifier token that is further fed to a multi-layer perceptron to compute the final document relevance. We

[1] https://www.tensorflow.org/versions/r2.2/api_docs/python/tf/function

set the input size to the maximum that the model could handle and applied padding or truncation. This implementation gives us a best-case scenario in terms of computational performance for ad-hoc retrieval.

We compared the following transformer-based models against our proposed solution:

- BERT (x12): Corresponds to the original checkpoint "bert-base-uncased" of the BERT model and has 12 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 110 million trainable parameters.
- BERT (x24): Corresponds to the original checkpoint "bert-large-uncased" of the BERT model and has 24 layers, 1024 hidden dimension, 16 multi attention heads, totaling approximately 340 million trainable parameters.
- distilBert: Corresponds to the original checkpoint "distilbert-base-uncased" of the distilBERT model and has 6 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 66 million trainable parameters.
- ALBERT: Corresponds to the original checkpoint "albert-base-v2" of the ALBERT model and has 12 repeating layers, 128 size embeddings, 768 hidden dimension, 12 multi attention heads, totaling approximately 11 million trainable parameters.
- Longformer: Corresponds to the original checkpoint "allenai/longformer-base-4096" of the Longformer model and has 12 layers, 768 hidden dimension, 12 multi attention heads, totaling approximately 149 million trainable parameters.

We mostly use BERT models since these are widely adopted in the literature. Moreover, we also include distilBERT since it is currently one of the fastest transformer-based models available, and ALBERT, given its comparatively small number of trainable parameters. Furthermore, all the models had a maximum input size of 512, with the exception of Longformer, which had an input size of 4096.

Some final notes regarding our neural model implementation are also relevant. In the first place, this model was constructed to run on the CPU, and so we do not consider operations and data flow that would be easily accelerated on the GPU. Furthermore, our implementation is in a prototype phase, which means that we have not performed any optimisations. For example, in the current state, the model performs some redundant computation, and the amount of padding data reaches approximately 90%.

4.2.1 On Inference

Regarding the inference tests, these were performed both in CPU and GPU since it is well known that the transformer architecture can achieve a drastic speed up on the GPU. However, on the other side, this brings more operational costs and offers a new single point of failure that real-world systems would need to deal with.

With respect to the experiment, it consists of computing the document relevance score for a BioASQ test set that contains 100 queries and a pool of 250 documents for each question, making a total of 25000 query document pairs. We fed this to the models in a systematic way, only varying the batch size and measured the elapsed time in seconds. We discarded the time taken on the first and last batch since the first batch may perform some initialization operations and the last batch has fewer documents than the others. Moreover, in the column "Tokens seen", we show the number of tokens fed to the model per sample.

Model	**Tokens seen**	**Time elapsed** (on a batch of **16**)	
		CPU(s) **Mean ± std**	**GPU(s)** **Mean ± std**
Proposed model	4500	0.074 ± 0.006	0.824 ± 0.093
distilBERT	512	2.421 ± 0.038	0.551 ± 0.007
BERT (x12)	512	4.967 ± 0.072	1.133 ± 0.011
ALBERT	512	5.144 ± 0.081	1.198 ± 0.027
BERT (x24)	512	15.729 ± 0.185	3.511 ± 0.052
LongFormer	4096	102.051 ± 1.009	$*11.188 \pm 0.035$

*Measured on a batch size of 8

Table 4: Inference times in seconds measured over 25000 query document pairs using a batch size of 16 on a CPU and GPU.

Table 4 presents the measured times using a batch size fixed at 16. As illustrated, our simple model can run 32 times faster on the CPU compared to the fastest transformer-based solution while processing almost 9 times the amount of tokens. Compared with a model that processes almost the same amount of data, in our experiments the LongFormer, the time difference becomes more evident, with our proposed model being 1379 times faster.

Regarding the behaviour when running on the

GPU, our proposed model is not able to achieve a speedup when compared to the CPU time and to the transformer-based models. However, we believe the main reason for this is related to the time needed to transfer the data to the GPU since the proposed model needs to send 9 times more data, which also gives us an indication that it may not be worth to run on the GPU due to the small number of operations that the model performs compared to the amount of data that it needs to send. Furthermore, the results that we show for the LongFormer were obtained with a batch size of 8, since we could not fit the model with a batch of 16 on the GPU, due to memory constraints.

Figure 2 presents the models inference time for four different batch sizes, $\{16, 32, 64, 258\}$. On the left, which corresponds to the experiments on the CPU, the models seem to scale with the increase of the batch size linearly. Additionally, only our proposed model is capable of keeping an inference time under one second, which can be viewed as an acceptable query latency time. Similarly, on the right side, corresponding to the experiments performed on the GPU, we observe the same linear behaviour for the transformer models. However, the proposed model shows an almost constant time, which reinforces the idea that when executed in the GPU, the most time-consuming operation is data transfer.

4.2.2 On Training

The training experiments were performed on both CPU and GPU, similarly to the inference. Regarding this experiment, we follow a pairwise training approach, which may be a computationally heavier option when combined with the transformer architecture, since these are usually trained as a binary classification problem, i.e., a document is relevant to a question or not. However, we decided to keep the pairwise approach since the proposed model was designed to use it, this way keeping a more fair comparison and also demonstrating the burden of training transformer-based models in a pairwise setting. Moreover, the transformer-based model can be entirely finetuned, i.e., training the classification layer and all layers, or just training the classification layer, which we also evaluate, since it should speed up the training.

Concerning the training data, we used 100 questions from the BioASQ training dataset each associated with a list of relevant documents and a list of negative documents randomly sampled from the BM25 ranking order, producing a total of 1104 training samples.

After the sampling process, we store this data to ensure that every model processes exactly the same set of triplets (query, positive document and negative document). To further ensure homogeneity among the experiments, we fixed every random seed that we were aware of.

Model	**Tokens seen**	**Time elapsed**	
		(on a batch of **16**) **CPU(s) Mean $\pm$ std**	(on a batch of **1**) **GPU(s) Mean $\pm$ std**
Proposed model	4500	0.350 ± 0.011	1.204 ± 0.234
distilBERT	512	9.700 ± 0.055	1.116 ± 0.007
BERT (x12)	512	19.727 ± 0.280	0.484 ± 0.005
ALBERT	512	19.835 ± 0.414	0.451 ± 0.003
BERT (x24)	512	66.536 ± 0.724	1.582 ± 0.040

Table 5: Complete training times, i.e., training all the layers, in seconds measured over 1104 samples on a CPU and GPU.

Table 5 shows the time required to fully train the transformer-based models, i.e., training all the layers plus the classifier layer, in a pairwise setting. Similarly to the inference, we also adopted a batch of 16 for the CPU and a batch of 1 for the GPU, since we could not perform the training on the GPU due to memory issues for some models. Additionally, we did not measure results for the LongFormer, neither in CPU or GPU, since we were unable to store the model plus all the gradients on memory (128GB) for the given batch size.

Regarding the results, all the models seem to be four times slower at training time when compared with the inference times. As expected, the proposed model presented the fastest training time, being 10 times faster than the fastest transformer model. Also note that, for all the models, training with a batch of 1 on GPU does not bring any advantages over training on the CPU, which is approximately 3 times slower (per batch) but for a batch size that is 16 times larger. Finally, regarding the performance of the proposed model on the GPU, it also seems to support the previously enunciated problem related to data transfer to the GPU.

Table 6, presents the measured time, in seconds, required only to train the classification layer in the transformer-based models. In this case, both the CPU and GPU experiments could be done on a batch size of 16, since it is only necessary to store gradients for the classification layer.

Looking at the results, if we only update the classifier layers weights, the transformer-based models

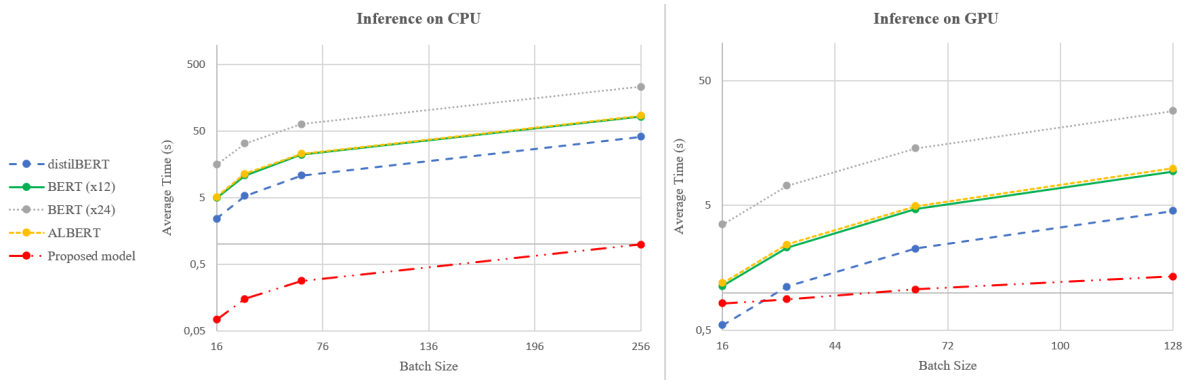

Figure 2: Inference times on the CPU, left side, and on the GPU, right side, as a function of the batch size.

Model	**Tokens seen**	**Time elapsed** (on a batch of **16**)	
		CPU(s) **Mean** $\pm$ **std**	**GPU(s)** **Mean** $\pm$ **std**
Proposed model	4500	0.350 ± 0.011	1.008 ± 0.094
distilBERT	512	3.696 ± 0.098	1.108 ± 0.011
BERT (x12)	512	7.513 ± 0.098	2.266 ± 0.012
ALBERT	512	7.867 ± 0.202	2.400 ± 0.011
BERT (x24)	512	24.699 ± 0.377	7.042 ± 0.033

Table 6: Training times, in seconds, only for the classification layer measured over 1104 samples on a CPU and GPU.

training becomes 2.5 times faster. Additionally, using the GPU further improves this speed up to 8.7 times. However, even using the GPU, the fastest transformer based model, distilBERT, is still 3.1 times slower than our proposed model.

5 Discussion

In this section, we take into consideration the presented results, both in terms of efficacy and performance, and present an overall overview of the comparison of our lightweight model and the transformer-based approaches.

First of all, we acknowledge some limitations of the proposed model, which are all related to the current implementation that processes a large portion of padding data, increasing the number of tokens that the model sees and thus injuring the performance of the current solution. However, this is something that can be addressed in future work by rearranging the model data flow and better understanding the required maximum input size of the query and document tokens. Moreover, to get a sense of speed up that we could achieve, we made a comparison with a transformer-based model that is able to encode up to 4096 tokens, which is close to the 4500 tokens of the proposed model. In this case, the lightweight model ends up being 1379 times faster, which gives an idea that the expected speedup may be in the order of a thousand.

Recalling the three proposed dimensions of comparison, the proposed model excels in all of them. First of all, it is a model with 10 thousand times fewer trainable parameters when compared to distilBERT. In the efficacy evaluation, the results show that the proposed model could achieve close to state of the art performance in both biomedical ad-hoc retrieval tasks. Finally, according to the presented inference times, more precisely, with the times presented in Figure 2, it is observable that the proposed model was the only one to perform inference under one second even with a batch size of 256, which is more than enough to adopt in real-world ad-hoc retrieval applications. Furthermore, this observation enables us to conclude that the transformer-based model solution is currently not suitable for a real-world ad-hoc retrieval application unless large scale computational resources are available.

Moreover, our results also show that the proposed model was always the fastest model by a significant margin, even though using only the CPU. This, in our view, ends up to be a preferable characteristic, since it facilitates the deployability and scalability of systems that implement this model.

6 Conclusion

In this paper, we discuss the feasibility of using transformer-based models in real-world ad-hoc retrieval applications and show an extreme lightweight solution, with only 620 trainable parameters.

We evaluate the solution against the transformer-based models, in terms of efficacy and performance, and show that this model is capable of matching the efficacy offered by the transformer-based models in two biomedical ad-hoc retrieval challenges while

being considerably faster by just using the CPU compared to the best GPU runs of the transformer-based models.

Acknowledgments

This work received support from EU/EFPIA IMI 2 JU under grant No 806968, and from National Funds through FCT - Foundation for Science and Technology, in the context of project UIDB/00127/2020 and grant 2020.05784.BD.

References

Tiago Almeida and Sérgio Matos. 2020a. Calling attention to passages for biomedical question answering. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part II*, volume 12036 of *Lecture Notes in Computer Science*, pages 69–77. Springer.

Tiago Almeida and Sérgio Matos. 2020b. BIT.UA at bioasq 8: Lightweight neural document ranking with zero-shot snippet retrieval. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Tiago Almeida and Sérgio Matos. 2020c. Frugal neural reranking: evaluation on the covid-19 literature. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. 2011. *Modern Information Retrieval: The Concepts and Technology behind Search*, 2nd edition. Addison-Wesley Publishing Company, USA.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3079–3087. Curran Associates, Inc.

Zhuyun Dai and Jamie Callan. 2019a. Context-aware sentence/passage term importance estimation for first stage retrieval.

Zhuyun Dai and Jamie Callan. 2019b. Deeper text understanding for ir with contextual neural language modeling. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. pages 126–134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Chengxiang Zhai, and Xueqi Cheng. 2018. Modeling diverse relevance patterns in ad-hoc retrieval. *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval*.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Croft. 2016. A deep relevance matching model for ad-hoc retrieval. pages 55–64.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. pages 328–339.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1049–1058, Copenhagen, Denmark. Association for Computational Linguistics.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Ashot Kazaryan, Uladzislau Sazanovich, and Vladislav Belyaev. 2020. Transformer-based open domain biomedical question answering at bioasq8 challenge. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Efficient document re-ranking for transformers by precomputing term representations. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ryan McDonald, George Brokos, and Ion Androutsopoulos. 2018. Deep relevance ranking using enhanced document-query interactions. pages 1849–1860.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval t. *Foundations and Trends® in Information Retrieval*, 13:1–126.

Anastasios Nentidis, Anastasia Krithara, Konstantinos Bougiatiotis, Martin Krallinger, Carlos Rodriguez-Penagos, Marta Villegas, and Georgios Paliouras. 2020. Overview of bioasq 2020: The eighth bioasq challenge on large-scale biomedical semantic indexing and question answering. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 194–214, Cham. Springer International Publishing.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with bert.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2016. Text matching as image recognition.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 257–266, New York, NY, USA. Association for Computing Machinery.

Dimitris Pappas, Petros Stavropoulos, and Ion Androutsopoulos. 2020. AUEB-NLP at bioasq 8: Biomedical document and snippet retrieval. In *Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*, volume 2696 of *CEUR Workshop Proceedings*. CEUR-WS.org.

A. Radford. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Kirk Roberts, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, Kyle Lo, Ian Soboroff, Ellen M. Voorhees, Lucy Lu Wang, and William R. Hersh. 2020. TREC-COVID: rationale and structure of an information retrieval shared task for COVID-19. *J. Am. Medical Informatics Assoc.*, 27(9):1431–1436.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael Alvers, Dirk Weißenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel-Cyrille Ngonga Ngomo, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, and Georgios Paliouras. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16:138.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020a. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online. Association for Computational Linguistics.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020b. Linformer: Self-attention with linear complexity.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple applications of bert for ad hoc document retrieval.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences.

Edwin Zhang, Nikhil Gupta, Raphael Tang, Xiao Han, Ronak Pradeep, Kuang Lu, Yue Zhang, Rodrigo Nogueira, Kyunghyun Cho, Hui Fang, and Jimmy Lin. 2020. Covidex: Neural ranking models and keyword search infrastructure for the covid-19 open research dataset.