Mitigating the Position Bias of Transformer Models in Passage Re-Ranking

Sebastian Hofstätter[1], Aldo Lipani[2], Sophia Althammer[1], Markus Zlabinger[1], and Allan Hanbury[1]

[1] TU Wien, Vienna, Austria
{*first.last*}@tuwien.ac.at
[2] University College London, United Kingdom,
aldo.lipani@ucl.ac.uk

Abstract. Supervised machine learning models and their evaluation strongly depends on the quality of the underlying dataset. When we search for a relevant piece of information it may appear anywhere in a given passage. However, we observe a bias in the position of the correct answer in the text in two popular Question Answering datasets used for passage re-ranking. The excessive favoring of earlier positions inside passages is an unwanted artefact. This leads to three common Transformer-based re-ranking models to ignore relevant parts in unseen passages. More concerningly, as the evaluation set is taken from the same biased distribution, the models overfitting to that bias overestimate their true effectiveness. In this work we analyze position bias on datasets, the contextualized representations, and their effect on retrieval results. We propose a debiasing method for retrieval datasets. Our results show that a model trained on a position-biased dataset exhibits a significant decrease in re-ranking effectiveness when evaluated on a debiased dataset. We demonstrate that by mitigating the position bias, Transformer-based re-ranking models are equally effective on a biased and debiased dataset, as well as more effective in a transfer-learning setting between two differently biased datasets.

1 Introduction

Datasets used to train neural network models are subject to a range of biases, which might constitute unwanted artefacts that should not be incorporated in the trained model [20]. Multiple studies showed that in the ad-hoc retrieval of full documents the text location is of relevant importance, such as the beginning in news articles [7, 50] or general web search [23]. In contrast, in this study we specifically probe positional bias in passage collections that are not linked to the previously studied full document relevance distributions. We operate on the assumption, based on the findings of the annotation study of TREC'19 Deep Learning data [10] by Hofstätter et al. [23], that inside a passage (made up of a few sentences) no word position is supposed to be explicitly favored when matching query and passage sequences.

Transformer-based neural re-ranking models, especially models based on the large-scale pre-trained BERT model [11], have shown a significant improvement in ad-hoc retrieval, where a natural language question is asked by the user and a set of passages

is retrieved [38, 35]. In this study we evaluate three state-of-the-art Transformer ranking models with varying characteristics: **1) BERT$_{\text{CAT}}$** [38] using BERT with query and passage concatenation, **2) BERT$_{\text{DOT}}$** [52] using a dot-product between query and passage BERT classification (CLS) vectors and **3) TK** [22], a lightweight Transformer-Kernel model that does not require pre-training. Each of the three architectures exhibits different strengths and weaknesses, which we describe in Section 2.

In the Transformer-architecture, positional information is induced through absolute position information provided by a positional encoding [48]. This positional encoding is added to each non-contextualized representation in a sequence before applying the self-attention. If a bias favoring certain positions in a text exists the Transformer may implicitly incorporate this bias in its word representation as Transformers tend to learn positional information [53]. To our knowledge, the connection between the explicit positional information of the Transformer and positional artefacts in common retrieval collections has not been studied before.

Traditional IR datasets contain relevance judgements for query-document pairs, where a single judgement covers the full document. In contrast to that, QA datasets contain exact location spans of the answer or an answer text that can be partly matched to a position in the document. In our work, we utilize two widely used QA datasets: MS MARCO [3] and SQuAD 2.0 [42]. Both datasets are converted to retrieval collections, by setting paragraphs that were selected to contain the answer as a relevant paragraph for a question. We observe that for the MS MARCO dataset the positions of the mapped answers strongly favor earlier positions in the paragraphs, while the SQuAD 2.0 dataset is more balanced although not completely bias free. The evaluation set is taken from the same distribution, therefore the evaluation is also biased and models overfitting to that bias overestimate their true effectiveness. In the case of MS MARCO this bias is especially concerning as it – because of its size – became the defacto standard collection in the neural re-ranking community, including as base retrieval training for transfer learning [27, 55].

We propose to create unbiased versions of the datasets by switching the first and second parts of a passage around a randomly selected position. This approach does not affect the relevance judgements, since they are on a passage level, and allows us to train unbiased re-ranking models as well as to measure the true effectiveness of re-ranking approaches, since relevant matches might now occur in every part of the passage.

We analyze passage term representations to study the position bias induced in Transformer based contextualization and answer ***RQ1*** *How can we measure the degree of position bias in the passage representations?* We propose a new metric to measure the mean average term similarity (MATS) per positional delta of all terms in the collection to investigate whether the term representations are independent of the positional encoding or not.

To understand the effects of our debias augmentation in conjunction with Transformer models we further study the following questions:
RQ2 *What effect has the debiasing on the evaluation of Transformers?*

We evaluate the effectiveness of our modifications on the original, as well as the debiased collections. We find that all three models perform better on the original (biased) evaluation, but their effectiveness drops substantially on a debiased evaluation set.

RQ3 *Does a debiased training result in better generalization?*

Training on an unbiased collection shows much more robust results across the evaluated collections and models, which we view as a a more accurate indicator for their actual effectiveness.

RQ4 *Do we observe differences in transfer-learning, based on debiased pre-training?*

We demonstrate the usefulness of mitigating bias in the learned representations in the scenario of transfer learning between differently biased collections. We use the larger MS MARCO to pre-train our model variants, before fine-tuning the models on SQuAD 2.0. The bias-mitigated pre-training shows more effective results in the fine-tuning, than starting with a biased pre-training.

The contributions of this work are as follows:

- We measure the positional bias of judgments in two popular Open-QA passage retrieval collections and propose a method to debias the collections;
- We show how three different Transformer-based re-ranking models learn to incorporate the position bias;
- We demonstrate the importance of mitigating the position bias with debiased evaluation sets and the benefit of debiasing in transfer learning between collections.
- We publish the source code of our work at:
  `github.com/sebastian-hofstaetter/transformer-kernel-ranking`

2 Background

In this section we first describe the Transformer architecture, followed by the three Transformer-based passage re-ranking models we employ in this study.

2.1 Transformer

The Transformer-layer [48] is a versatile building block for different architectures. In our work we use an encoder structure to encode a sequence and output contextualized representations of this sequence. The Transformer architecture incorporates a natural algorithmic bias on the position of a term in a sequence, because it adds a positional encoding to its input sequence. Vaswani et al. [48] use overlapping sinusoidal-waves per dimension, forming an equidistant relationship among neighbouring terms, whereas Devlin et al. [11] employ a trainable positional embedding for BERT. This positional encoding is important since the Transformer otherwise would be entirely invariant to sequence ordering. However, adding the positional encoding directly to the input means that absolute positional information is retained in the output sequence. Each encoding is unique to a position of the input sequence. Based on the provided training examples, the Transformer may tend to learn position-biased representations.

In this paper we define the Transformer as the sequential use of n Transformer-layers (TLs) as:

$$\begin{aligned} s^{(1)}_{1:m} &= \mathrm{TL}(s_{1:m}) \\ s^{(n)}_{1:m} &= \mathrm{TL}(s^{(n-1)}_{1:m}) \\ \mathrm{TF}(s_{1:m} + e_{1:m}) &= s^{(n)}_{1:m} \end{aligned} \tag{1}$$

where $s_{1:m}$ is the sequence of input embeddings, $e_{1:m}$ is the positional encoding. We call this sequence of recursive applications TF.

2.2 BERT_{CAT} Ranking Model

First proposed by Nogueira et al. [38] the BERT_{CAT} approach has become a common way of utilizing the BERT pre-trained Transformer model in a re-ranking scenario [35, 55]. It uses the capability of the BERT pre-training approach to compute the relationship of two concatenated sequences, separated by a special SEP token and depending on the BERT version a sequence embedding. The BERT architecture is a simple Transformer model (TF), the effectiveness comes from the masked language and next sentence prediction pre-training. In the BERT_{CAT} ranking model the query ($q_{1:m}$) and passage ($p_{1:n}$) sequences as well as BERT's special tokens are concatenated (where ; is the concatenation operator) and after the TF computation we select only the first vector of the output sequence (which has been initialized with the special CLS token) and score this pooled representation with a single linear layer (W_s):

$$\text{BERT}_{\text{CAT}}(q_{1:m}, p_{1:n}) = \text{TF}([\text{CLS}; q_{1:m}; \text{SEP}; p_{1:n}])_1 * W_s \tag{2}$$

BERT_{CAT} is the current state-of-the art in terms of effectiveness, however it requires substantial compute at query time and increases the query latency by seconds [21]. Therefore, we also feature additional models that provide a more balanced efficiency-effectiveness tradeoff.

2.3 BERT_{DOT} Ranking Model

In contrast to the full-interaction BERT_{CAT} model, that requires a full online computation of all selected passages, the BERT_{DOT} model only matches a single CLS vector of the query with a single CLS vector of a passage [52, 34]. This makes it possible to pre-compute contextualized representations for all passages in our index, as well as to employ a vector-based nearest neighbour retrieval approach.

The BERT_{DOT} model, with $\cdot$ as the dot product operator, is formalized by two independent TF computations (and their pooled representations by selecting the first vector output) as follows:

$$\text{BERT}_{\text{DOT}}(q_{1:m}, p_{1:n}) = \text{TF}([\text{CLS}; q_{1:m}])_1 \cdot \text{TF}([\text{CLS}; p_{1:n}])_1 \tag{3}$$

BERT_{DOT} brings strong query time improvements (a few milliseconds latency per query) over BERT_{CAT}, however it still requires the full BERT pre-computation of all indexed passages, which can be very costly depending on the collection size.

2.4 TK Ranking Model

The TK model [22], while also utilizing Transformers, is not based on BERT pre-training, rather it uses shallow Transformers atop word embeddings followed by an explicit term-by-term interaction matrix and scoring with kernel-pooling [51]. In contrast to the BERT approaches TK offers us great control to probe the individual term representations, as it splits the representation learning and their interactions in two distinct parts.

The first part of TK is learning contextualized representations. TK independently contextualizes query ($q_{1:m}$) and passage ($p_{1:n}$) sequences based on pre-trained word embeddings, where the intensity of the contextualization (with TF) is regulated by a gate (α):

$$\begin{aligned}\hat{q}_i &= q_i * \alpha + \mathrm{TF}(q_{1:m})_i * (1-\alpha) \\ \hat{p}_i &= p_i * \alpha + \mathrm{TF}(p_{1:n})_i * (1-\alpha)\end{aligned} \tag{4}$$

The two resulting sequences $\hat{q}_{1:m}$ and $\hat{p}_{1:n}$ interact in a match-matrix with a cosine similarity per term pair and each similarity is then activated by a set of RBF-kernels [51]:

$$K^k_{i,j} = \exp\left(-\frac{\left(\cos(\hat{q}_i, \hat{p_j}) - \mu_k\right)^2}{2\sigma^2}\right) \tag{5}$$

Kernel-pooling is conceptually a soft-histogram, which counts the number of occurrences of certain similarities. Each kernel focuses on a fixed similarity range with center μ_k and width of σ. Each kernel results in a matrix $K \in \mathbb{R}^{|q|\times|p|}$.

These kernel activations are then summed, first by the passage term dimension j, log-activated, and then the query dimension is summed resulting in a single score per kernel. The final score is calculated by a weighted sum using the linear layer W_s:

$$s = \left(\sum_{i=1}^{|q|} \log\left(\sum_{j=1}^{|p|} K^k_{i,j}\right)\right) W_s \tag{6}$$

The kernel-pooling technique is position-independent, as every activation for position j is summed without a weighting them, which allows us to isolate the positional analysis in the Transformer in Section 5.

3 Experiment Design

For the first stage indexing and retrieval we use the Anserini toolkit [54] to compute the initial ranking lists with BM25, which we use to generate training and evaluation inputs for the neural models. For our neural re-ranking training and inference we use PyTorch [39] and AllenNLP [15]. We tokenize the text with the fast BlingFire library[3]. As proposed for the MS MARCO dataset [3] we evaluate our neural re-ranking systems using mean reciprocal rank (MRR), normalized discounted cumulative gain (nDCG), and recall (Recall).

For the BERT-based models we use the 6-layer DistilBERT [45] pre-trained weights and the Adam [26] optimizer with a learning rate of $7*10^{-6}$. For TK we use pre-trained GloVe [40] word embeddings with 300 dimensions[4] and Adam with a learning rate of 10^{-4} for word embeddings and contextualization layers, 10^{-3} for the kernel-pooling weights.

For the Transformer layers in TK we evaluate 2 layers each with 16 attention heads with size 32 and a feed-forward dimension of 100. For kernel-pooling we set the number

[3] *github.com/microsoft/BlingFire*

[4] 42B CommonCrawl: *nlp.stanford.edu/projects/glove/*

Table 1. Collection statistics

Collection	# Docs.	# Queries		
		Train	Val.	Test
MS MARCO	8,841,823	502,939	6,980	48,598
SQuAD 2.0	20,239	86,821	5,000	5,928

of kernels to 11 with the mean values of the Gaussian kernels varying from -1 to $+1$, and standard deviation of 0.1 for all kernels. We use the same sinusoidal positional encodings as Vaswani et al. [48], for the document encodings we shift the start position by 500 to distinguish them from the query encodings.

We train all neural models with a pairwise hinge loss and a batch size of 32. The re-ranking depth for each model instance is tuned on the best mean nDCG@10 of the validation set, as part of an early stopping strategy. For MS MARCO we evaluate a re-ranking depth until 1000 and for SQuAD up to 100.

4 Dataset Analysis & Debiasing

To better understand the neural models, we first need to look at the source of the position bias of the training and evaluation data, specifically the distribution of answer positions in our QA-datasets.

4.1 Dataset Analysis

The question answering task is strongly linked to ad-hoc information retrieval, as IR provides the first stage of selecting potential candidate passages that contain the natural language answer, that should be presented to a user. In addition to traditional relevance judgements, that cover full documents, the QA datasets also contain short answer strings or exact spans pointing to the answer in a passage.

Using QA datasets to evaluate the retrieval portion of the QA pipeline offers us the unique opportunity of inspecting the answer position, which gives us an insight into the positional importance inside the relevant passages. For SQuAD 2.0 we follow the approach done for MS MARCO [3] and set a passage as relevant to a query if the passage is connected to the answer. We provide an overview of the size of our collections in Table 1, where we observe that MS MARCO is a much larger collection than SQuAD.

In Figure 1 we show the distribution of the QA-answer start positions in their respective relevant passages for the training sets of MS MARCO and SQuAD. To determine the answer positions, we matched the available answer tokens to the passage tokens of the selected passages for both collections and counted all matches. For MS MARCO we omitted answers that could not directly be matched in the passage. In this figure, it is evident that the answer positions in the MS MARCO dataset strongly favor earlier positions in the paragraphs. MS MARCO was created in a retrieval setting, where annotators were given a question and a list of 10 possible paragraphs to judge, which

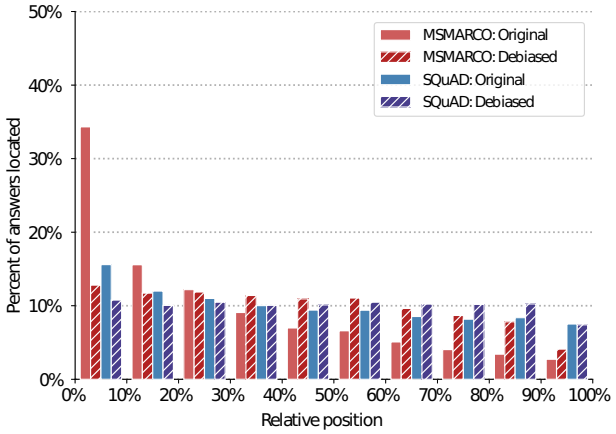

Fig. 1. QA collection in-passage relative answer positions

may have favoured passages with answers appearing early in the text. On the other hand SQuAD 2.0, for which annotators were asked to create questions based on a given passage, is relatively unbiased, as the distribution of answer spans in the paragraphs is more uniform.

4.2 Debiasing the Passage Datasets

We have established that MS MARCO answers excessively favor the beginning of a passage, while SQuAD does not. To explicitly study this phenomenon, Hofstätter et al. [23] conducted a fine-grained relevance position study. They found, that if annotators are shown only one query passage pair at a time, annotators select answers uniformly across passages. As we simply cannot re-annotate a collection of the size of MSMARCO with hundreds of thousands of queries, we apply an automatic debiasing method to the existing collections.

For each passage $p_{1:n}$ in the collection we create a *debiased* instance $\tilde{p}_{1:n}$, for which we generate a random number $r \in \{1, \ldots, n\}$, slice the word sequence at the r^{th} index, switch and concatenate the two sub-sequences again:

$$\tilde{p}_{1:n} = [p_{r:n} \,;\, p_{1:r-1}] \tag{7}$$

As shown in Figure 1 this approach produces near uniformly distributed relative answer positions for both collections. This approach is minimally invasive as it only breaks the contextualization at a single point per passage, without the need for additional annotations. In a pilot study we also experimented with sentence splitting based rotation, however we found that in the the MSMARCO web-page collection too many passages do not contain punctuation and therefore the sentence split approach does not produce uniform answer positions.

Table 2. MATS statistics for TK's contextualized passage vectors. *Lower MATS means less position bias.*

Training	MS MARCO		SQuAD	
	MATS	Std.dev.	MATS	Std.dev.
Original	0.176	0.046	0.056	0.014
Debiased	**0.021**	**0.006**	**0.007**	**0.002**

5 Transformer Bias Analysis

In this section we probe term-wise Transformer representations to determine their bias across positions. Both BERT model variants incorporate their scoring decision mechanism inside the Transformer layers and only use the CLS vector representation, hiding individual term interactions inside the model. The TK model on the other hand utilizes every passage term representation in the cosine match matrix, which allows us to decouple the Transformer layers from the relevance scoring and analyze the passage term representations of a trained model on their own.

We now discuss ***RQ1*** *How can we measure the degree of position bias in the passage representations?* by analyzing the implicit bias of the absolute position of a term in a sequence. If a contextualized vector contains enough information about the original position, then a bias is measurable when we compare different vectors of the same term. We propose to compare the cosine similarity of the contextualized representations r between occurrences of the same term t across different passages computing the average term similarity (ATS) at distance Δa for all terms in the collection $t \in \mathcal{T}$. This is formalized as follows:

$$\begin{aligned} \text{ATS}(\Delta a) &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{|C_{t,\Delta a}|} \sum_{(r^t_{a_1}, r^t_{a_2}) \in C_{t,\Delta a}} \cos(r^t_{a_1}, r^t_{a_2}) \\ C_{t,\Delta a} &= \left\{ (r^t_{a_1}, r^t_{a_2}) \middle| \Delta a = |a_1 - a_2|, (t_{a_1}, t_{a_2}) \in \mathcal{C} \right\} \end{aligned} \tag{8}$$

where $r^t_{a_1}$ is the representation of term t at absolute position a_1. The set $C_{t,\Delta a}$ is a set of all couples of representations of term t, which occur in the passages with a distance between their absolute positions of $\Delta a = |a_1 - a_2|$ in the collection $\mathcal{C}$. The mean ATS difference to the first point (MATS) is computed as:

$$\text{MATS} = \frac{1}{\max(\Delta a) - 1} \sum_{i=1}^{\max(\Delta a)} \text{ATS}(0) - \text{ATS}(i) \tag{9}$$

MATS aggregates ATS across all available positions in the passages and allows us to formally compare the different distributions. In Table 2 we show TK's MATS for both collections.

In Figure 2 we show the ATS for different (Δa) along the x-axis using TK passage term representations on the MS MARCO collection. The shaded area corresponds to the standard deviation. In this plot, an unbiased contextualization would result in a horizontal line, with a uniformly distributed standard deviation of the vectors. A set of

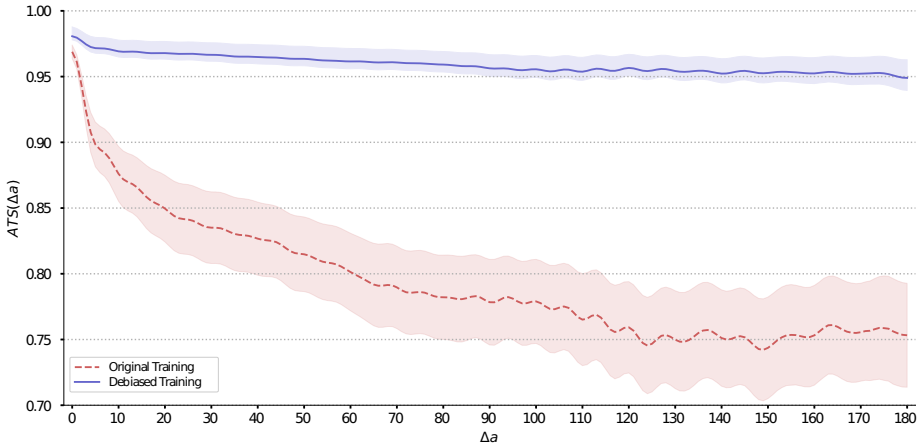

Fig. 2. ATS and standard deviation (y-axis) of same-term occurrences in different passages along positional Δa of each term pair (x-axis) trained and evaluated on the MS MARCO collection with TK passage term representations.

contextualized vectors naturally has a standard deviation, as each vector, even for the same term is influenced by different context terms.

It is evident from observing Figure 2 and Table 2, that the TK model incurs a strong positional bias, especially for deltas smaller than 20. This shows the influence of the bias in the training data, which conditions the contextualized vectors on their absolute position. Using a debiased training set improves the representations and makes them much less dependent on their position. The SQuAD collection, not pictured in Figure 2, exhibits a similar pattern, although dampened as the collection is less biased.

6 Retrieval Results

In this section we discuss our effectiveness related research questions with an emphasis on the differences in using the original vs. debiased training and evaluation, including the conclusion we can draw from them:

RQ2 *What effect has the debiasing on the evaluation of Transformers?*

We look at the two collections separately to answer this RQ. In Table 3 we have the results for the heavily-biased MS MARCO collection. We compare each measure by all possible training and evaluation approaches for all three Transformer models. The delta shows the relative difference between the original and debiased evaluation per training type. We can see that across all Transformer models we have a substantial drop in effectiveness when trained on the original training set and evaluated on the debiased set. This shows how the models learn to prioritize the beginning of the passages, and cannot generalize well to the scenario where answers are located in evenly distributed across the passage. The SQuAD results in Table 4 on the other hand offer a different picture with only minor differences between original and debiased evaluation sets. This is to be

Table 3. MSMARCO re-ranking results of original and debiased training sets (rows) on the original and debiased test sets (columns). Each measure uses a cutoff of 10 and the smallest absolute margin per block is marked in bold.

Model		MSMARCO - Test								
		nDCG			MRR			Recall		
	Training	Orig.	Deb.	Δ	Orig.	Deb.	Δ	Orig.	Deb.	Δ
BERT$_{CAT}$	Original	0.432	0.395	-9.4%	0.372	0.336	-10.7%	0.630	0.594	-6.1%
	Debiased	0.416	0.415	**-0.2%**	0.357	0.355	**-0.6%**	0.617	0.617	**0.0%**
BERT$_{DOT}$	Original	0.373	0.329	-13.4%	0.316	0.276	-14.5%	0.567	0.509	-11.4%
	Debiased	0.362	0.364	**+0.6%**	0.305	0.307	**+0.7%**	0.555	0.554	**-0.2%**
TK	Original	0.371	0.307	-20.8%	0.312	0.254	-22.8%	0.567	0.484	-17.1%
	Debiased	0.356	0.355	**-0.3%**	0.298	0.296	**-0.7%**	0.551	0.552	**+0.2%**

Table 4. Retrieval effectiveness results of original and debiased SQuAD training sets (rows) on the original and debiased SQuAD test sets (columns). Each measure uses a cutoff of 10 and the smallest absolute margin per block is marked in bold.

Model		SQuAD - Test								
		nDCG			MRR			Recall		
	Training	Orig.	Deb.	Δ	Orig.	Deb.	Δ	Orig.	Deb.	Δ
BERT$_{CAT}$	Original	0.908	0.902	-0.7%	0.892	0.884	**-0.9%**	0.957	0.956	**-0.1%**
	Debiased	0.910	0.905	**-0.6%**	0.894	0.885	-1.0%	0.959	0.956	-0.3%
BERT$_{DOT}$	Original	0.780	0.783	+0.4%	0.734	0.738	+0.5%	0.924	0.919	-0.5%
	Debiased	0.784	0.783	**-0.1%**	0.740	0.739	**-0.1%**	0.919	0.919	**0.0%**
TK	Original	0.846	0.840	-0.7%	0.818	0.811	-0.9%	0.933	0.930	-0.3%
	Debiased	0.848	0.844	**-0.5%**	0.820	0.816	**-0.5%**	0.932	0.931	**-0.1%**

expected, as we showed in Section 4 that the SQuAD collection is almost unbiased in its original form.

RQ3 *Does a debiased training result in better generalization?*

In contrast to the poor original training to debiased test set results on MSMARCO in Table 3, using the debiased training set we observe similar results on the two test sets with little delta across all three models. These debiased training results are better than those using original training to debiased test sets, leading us to the conclusion that these results represent the true generalized effectiveness of the models. For the SQuAD results in Table 4 we make an interesting observation, that some of the debiased trained models outperform those trained on the original training sets when applied to the original test sets.

RQ4 *Do we observe differences in transfer-learning, based on debiased pre-training?*

Finally, we look at a common transfer learning scenario: We utilize the large-scale MSMARCO as first retrieval pre-training and then transfer the trained model to a

Table 5. MS MARCO to SQuAD transfer learning results. Each measure uses a cutoff of 10. Significance is tested between training variants per model with Wilcoxon ($p < 0.05$).

Model	Train	Sig	SQuAD Original Test nDCG	MRR	Recall
BERT$_{CAT}$	SQuAD(Original)	a	0.908	0.892	0.957
	MS(Original) → SQuAD(Original)	b	**0.913**	**0.898**	0.957
	MS(Debiased) → SQuAD(Original)	c	0.911	0.896	**0.958**
BERT$_{DOT}$	SQuAD(Original)	a	0.780	0.734	0.924
	MS(Original) → SQuAD(Original)	b	0.788^{a}	0.744^{a}	0.922
	MS(Debiased) → SQuAD(Original)	c	$\mathbf{0.792}^{ab}$	$\mathbf{0.748}^{ab}$	$\mathbf{0.927}^{b}$
TK	SQuAD(Original)	a	0.846	0.818	0.933
	MS(Original) → SQuAD(Original)	b	0.854^{a}	0.827^{a}	0.936
	MS(Debiased) → SQuAD(Original)	c	$\mathbf{0.857}^{ab}$	$\mathbf{0.832}^{ab}$	**0.937**

smaller collection (SQuAD) and train it again. This is especially helpful in production scenarios that require efficient models and do not provide ample training data.

In Table 5 we show our transfer learning results. We recall that the original MS MARCO is heavily biased and SQuAD is not. The debiased MS MARCO is closer to the SQuAD answer distribution. In general, using the MS MARCO pre-training improves the SQuAD results. For the production scenario models, that enable query independent passage representation caching – BERT$_{DOT}$ and TK – we observe another significant increase in effectiveness on SQuAD using the debiased MS MARCO training. Only BERT$_{CAT}$ does not benefit from the debiased pre-training.

7 Related Work

Biases in datasets. Recent studies have observed a variety of artefacts (biases) in datasets of several NLP tasks. Gururangan et al. [20] demonstrate that for Natural Language Inference (NLI) datasets it is possible to identify the correct label by only looking at the hypothesis, without observing the premise based on superficial patterns generated while constructing the dataset. This is also confirmed by Poliak et al. [41] and Tsuchiya et al. [47]. McCoy et al. [36] shows that state-of-the-art models follow simple heuristics to identify the correct answer. Glockner et al. [18] show the deficiency of state-of-the-art NLI architecture by testing them in an unbiased dataset. Also QA and Visual QA (VQA) suffer from dataset artefacts. In fact, Jia and Liang [24] show that human-level performance on SQuAD can be achieved by only relying on superficial cues, and Chen et al. [8] show that in NewsQA, 73% of the answers can be predicted by simply identifying the single most relevant sentence. Formal et al. [14] studied the reliance of the ColBERT [25] model on exact term matches in IR.

Another form of bias affecting IR test collections is the pool bias [30, 32]. This bias is a side effect of the sampling method used to build these test collections called, the pooling method [29]. This is caused by the presence of non-annotated relevant documents in the collection which makes the evaluation of newly developed retrieval systems less reliable [31, 33].

Social biases are another form of bias manifesting in NLP and IR datasets [12, 17, 44]. In this case these biases are not generated by the way the datasets were constructed but by historical and cultural discriminations manifesting as a prejudice or unfair characterization of the members of a particular group.

Bias mitigation methods. The research on the mitigation of these biases has branched out into two directions. One defining methods to mitigate biases when constructing the datasets. The other devising mechanism to make models robust against the presence of bias in datasets. Agrawal et al. [1], Anand et al. [2], and Min et al. [37] develop methods to build unbiased datasets without a variety of biases. Other forms of bias removal consist in learning unbiased representations. Bolukbasi et al. [6] learned unbiased word embeddings to mitigate gender bias. Belinkov et al. [5] propose two probabilistic methods to build models that are more robust to biases and better transfer across datasets. Other methods to develop more robust NLP methods have been developed using adversarial methods [9, 28, 13, 4, 43, 19]. In the IR setting Gerritse et al. [16] studied and proposed methods to mitigate echo-chamber biases in personalised search.

Modeling relative position in Transformers. To overcome this limitation in machine translation tasks, Shaw et al. [46] developed a Transformer with a relation-aware self-attention, which induces the model to learn a relative positional encoding in a translation task. However, we have tested this Transformer-version and observed no improvement over the original version used in this paper. Also in translation tasks, Wang et al. [49] extend the transformer developed by Shaw et al. [46] to model hierarchies based on a dependency tree. We believe that these transformer-versions would benefit from our work, however we leave this to future work.

8 Conclusion

We observed a judgment bias towards the beginning of passages of selected answers in two popular QA datasets used for retrieval. Furthermore, the biased evaluation data hides the existence of this bias in the data. To overcome this problem, we proposed a dataset debiasing method, by switching two parts of a passage split at a random point, as the relevance of word matches in passage retrieval should be position independent.

We showed how the excessive focus on earlier positions in the data propagates through Transformer-based contextualization to form position-biased representations. Our results show that three different Transformer ranking models (BERT_{DOT}, BERT_{CAT}, and TK) trained on the original (biased) MS MARCO collection, substantially lose effectiveness on the debiased version. On the SQuAD collection, acting as an unbiased control dataset, the models do not show this behavior.

We demonstrate that by using a debiased training data transformation, the Transformer models achieve the same performance on biased and debiased datasets, showing the increased generalizability of the models. Finally, we also show that for production-scenario transfer-learning, the debiased pre-training is the most effective strategy. This leads us to the conclusion that going forward, the community should adopt the simple data-transformation for debiasing the MSMARCO pre-training in these transfer-learning scenarios.

References

1. Agrawal, A., Batra, D., Parikh, D., Kembhavi, A.: Don't just assume; look and answer: Overcoming priors for visual question answering. In: Proc. of CVPR (2018)
2. Anand, A., Belilovsky, E., Kastner, K., Larochelle, H., Courville, A.: Blindfold baselines for embodied qa. arXiv preprint 1811.05013 (2018)
3. Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., Mcnamara, A., Mitra, B., Nguyen, T.: MS MARCO : A Human Generated MAchine Reading COmprehension Dataset. In: Proc. of NeurIPS (2016)
4. Barrett, M., Kementchedjhieva, Y., Elazar, Y., Elliott, D., Søgaard, A.: Adversarial removal of demographic attributes revisited. In: Proc. of EMNLP-IJCNLP (2019)
5. Belinkov, Y., Poliak, A., Shieber, S., Van Durme, B., Rush, A.: Don't take the premise for granted: Mitigating artifacts in natural language inference. In: Proc. of ACL (2019)
6. Bolukbasi, T., Chang, K.W., Zou, J.Y., Saligrama, V., Kalai, A.T.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Proc. of NeurIPS (2016)
7. Catena, M., Frieder, O., Muntean, C.I., Nardini, F.M., Perego, R., Tonellotto, N.: Enhanced news retrieval: Passages lead the way! In: Proc. of SIGIR (2019)
8. Chen, D., Bolton, J., Manning, C.D.: A thorough examination of the CNN/daily mail reading comprehension task. In: Proc. of ACL (2016)
9. Clark, C., Yatskar, M., Zettlemoyer, L.: Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In: Proc. of EMNLP-IJCNLP (2019)
10. Craswell, N., Mitra, B., Yilmaz, E., Campos, D.: Overview of the trec 2019 deep learning track. In: TREC (2019)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. of NAACL (2019)
12. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608 (2017)
13. Elazar, Y., Goldberg, Y.: Adversarial removal of demographic attributes from text data. In: Proc. of EMNLP (2018)
14. Formal, T., Piwowarski, B., Clinchant, S.: A white box analysis of colbert (2020)
15. Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N.F., Peters, M., Schmitz, M., Zettlemoyer, L.S.: Allennlp: A deep semantic natural language processing platform. arXiv preprint arXiv:1803.07640 (2017)
16. Gerritse, E.J., Hasibi, F., de Vries, A.P.: Bias in conversational search: The double-edged sword of the personalized knowledge graph. In: Proc. of ICTIR (2020)
17. Gezici, G., Lipani, A., Saygin, Y., Yilmaz, E.: Evaluation metrics for measuring bias in search engine results. Information Retrieval Journal (2021)
18. Glockner, M., Shwartz, V., Goldberg, Y.: Breaking NLI systems with sentences that require simple lexical inferences. In: Proc. of ACL (2018)
19. Grand, G., Belinkov, Y.: Adversarial regularization for visual question answering: Strengths, shortcomings, and side effects. In: Proc. of the Workshop on Shortcomings in Vision and Language (2019)
20. Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., Smith, N.A.: Annotation artifacts in natural language inference data. In: Proc. of NAACL (2018)
21. Hofstätter, S., Hanbury, A.: Let's measure run time! Extending the IR replicability infrastructure to include performance aspects. In: Proc. of OSIRRC (2019)
22. Hofstätter, S., Zlabinger, M., Hanbury, A.: Interpretable & Time-Budget-Constrained Contextualization for Re-Ranking. In: Proc. of ECAI (2020)

23. Hofstätter, S., Zlabinger, M., Sertkan, M., Schröder, M., Hanbury, A.: Fine-grained relevance annotations for multi-task document ranking and question answering. In: Proc. of CIKM (2020)
24. Jia, R., Liang, P.: Adversarial examples for evaluating reading comprehension systems. In: Proc. of EMNLP (2017)
25. Khattab, O., Zaharia, M.: Colbert: Efficient and effective passage search via contextualized late interaction over bert. In: Proc. of SIGIR (2020)
26. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
27. Li, C., Yates, A., MacAvaney, S., He, B., Sun, Y.: Parade: Passage representation aggregation for document reranking. arXiv preprint arXiv:2008.09093 (2020)
28. Li, Y., Baldwin, T., Cohn, T.: Towards robust and privacy-preserving text representations. In: Proc. of ACL (2018)
29. Lipani, A., Losada, D.E., Zuccon, G., Lupu, M.: Fixed-cost pooling strategies. IEEE Transactions on Knowledge & Data Engineering (2019)
30. Lipani, A.: Fairness in information retrieval. In: Proc. of SIGIR (2016)
31. Lipani, A., Lupu, M., Hanbury, A.: Splitting water: Precision and anti-precision to reduce pool bias. In: Proc. of SIGIR (2015)
32. Lipani, A., Lupu, M., Hanbury, A.: The curious incidence of bias corrections in the pool. In: Proc. of ECIR (2016)
33. Lipani, A., Lupu, M., Kanoulas, E., Hanbury, A.: The solitude of relevant documents in the pool. In: Proc. of CIKM (2016)
34. Luan, Y., Eisenstein, J., Toutanova, K., Collins, M.: Sparse, dense, and attentional representations for text retrieval. arXiv preprint arXiv:2005.00181 (2020)
35. MacAvaney, S., Yates, A., Cohan, A., Goharian, N.: CEDR: Contextualized Embeddings for Document Ranking. In: Proc. of SIGIR (2019)
36. McCoy, T., Pavlick, E., Linzen, T.: Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In: Proc. of ACL (2019)
37. Min, S., Wallace, E., Singh, S., Gardner, M., Hajishirzi, H., Zettlemoyer, L.: Compositional questions do not necessitate multi-hop reasoning. In: Proc. of ACL (2019)
38. Nogueira, R., Cho, K.: Passage Re-ranking with BERT. arXiv preprint arXiv:1901.04085 (2019)
39. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: Proc. of NeurIPS-W (2017)
40. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: In Proc of EMNLP (2014)
41. Poliak, A., Naradowsky, J., Haldar, A., Rudinger, R., Van Durme, B.: Hypothesis only baselines in natural language inference. In: Proc. of the CLCS (2018)
42. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. In: Proc. of EMNLP (2016)
43. Ramakrishnan, S., Agrawal, A., Lee, S.: Overcoming language priors in visual question answering with adversarial regularization. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Proc. of NeurIPS (2018)
44. Rekabsaz, N., Schedl, M.: Do neural ranking models intensify gender bias? arXiv preprint arXiv:2005.00372 (2020)
45. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
46. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: Proc. of NAACL (2018)
47. Tsuchiya, M.: Performance impact caused by hidden bias of training data for recognizing textual entailment. In: Proc. of LREC (2018)

48. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Proc. of NIPS (2017)
49. Wang, X., Tu, Z., Wang, L., Shi, S.: Self-attention with structural position representations. In: Proc. of EMNLP-IJCNLP (2019)
50. Wu, Z., Mao, J., Liu, Y., Zhang, M., Ma, S.: Investigating passage-level relevance and its role in document-level relevance judgment. In: Proc. of SIGIR (2019)
51. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-End Neural Ad-hoc Ranking with Kernel Pooling. In: Proc. of SIGIR (2017)
52. Xiong, L., Xiong, C., Li, Y., Tang, K.F., Liu, J., Bennett, P., Ahmed, J., Overwijk, A.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020)
53. Yang, B., Wang, L., Wong, D.F., Chao, L.S., Tu, Z.: Assessing the ability of self-attention networks to learn word order. In: Proc. of ACL (2019)
54. Yang, P., Fang, H., Lin, J.: Anserini: Enabling the use of Lucene for information retrieval research. In: Proc. of SIGIR (2017)
55. Yilmaz, Z.A., Yang, W., Zhang, H., Lin, J.: Cross-domain modeling of sentence-level evidence for document retrieval. In: Proc. of EMNLP-IJCNLP (2019)