FAA: Fine-grained Attention Alignment for Cascade Document Ranking

Zhen Li[1], **Chongyang Tao**[2], **Jiazhan Feng**[1], **Tao Shen**[3], **Dongyan Zhao**[1,4*]
Xiubo Geng[2], **Daxin Jiang**[2*]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Microsoft Corporation
[3]FEIT, University of Technology Sydney
[4]State Key Laboratory of Media Convergence Production Technology and Systems
[1]{lizhen63,fengjiazhan,zhaody}@pku.edu.cn [2]{chotao,xigeng,djiang}@microsoft.com
[3]shentao@uts.edu.cn

Abstract

Document ranking aims at sorting a collection of documents with their relevance to a query. Contemporary methods explore more efficient transformers or divide long documents into passages to handle the long input. However, intensive query-irrelevant content may lead to harmful distraction and high query latency. Some recent works further propose cascade document ranking models that extract relevant passages with an efficient selector before ranking, however, their selection and ranking modules are almost independently optimized and deployed, leading to selecting error reinforcement and sub-optimal performance. In fact, the document ranker can provide fine-grained supervision to make the selector more generalizable and compatible, and the selector built upon a different structure can offer a distinct perspective to assist in document ranking. Inspired by this, we propose a fine-grained attention alignment approach to jointly optimize a cascade document ranking model. Specifically, we utilize the attention activations over the passages from the ranker as fine-grained attention feedback to optimize the selector. Meanwhile, we fuse the relevance scores from the passage selector into the ranker to assist in calculating the cooperative matching representation. Experiments on MS MARCO and TREC DL demonstrate the effectiveness of our method.

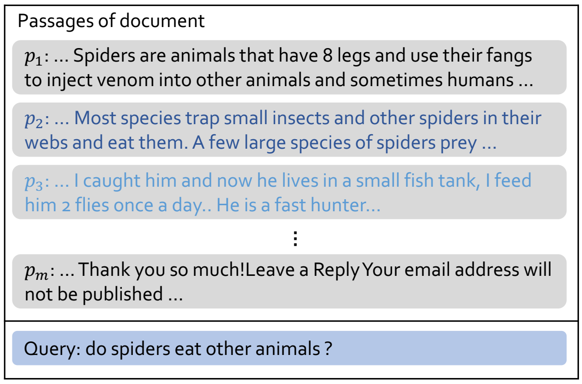

Figure 1: The case of scope hypothesis. In this example, p_2 is strongly relevant to the query, and p_3 is weakly relevant where other passages focus on other topics different from query.

1 Introduction

Document ranking aims at ranking the candidate documents according to their relevance to an input query, and it has been widely applied in many natural language processing (NLP) and information retrieval tasks, such as search engines (Hofstätter et al., 2021) and question answering (Chen and Yih, 2020). Due to the powerful representation ability of large-scale pre-trained language models (PLMs) (e.g., BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019)) that have achieved impressive performance in a large number of NLP tasks, several researchers have considered making use of pre-trained models for document ranking (MacAvaney et al., 2019; Li and Gaussier, 2021; Fu et al., 2022).

One major challenge in applying PLMs for neural document ranking is their difficulty in handling long texts due to high computational complexity and memory requirements, such as the 512 token limit for BERT. In fact, documents typically contain long text, for example, the mean length of documents in 2019 TREC Deep Learning Track Document Collection is 1600 (Hofstätter et al., 2021). To address this issue, various studies have been conducted to develop more efficient attention mechanisms in transformers (Beltagy et al., 2020; Hofstätter et al., 2020a), by simply truncating the document to meet the requirement for the relevance model (Boytsov et al., 2022), or by breaking down the long document into smaller segments or passages that can be processed individually by the pre-trained models (Dai and Callan, 2019; Rudra and Anand, 2020; Li et al., 2020; Chen et al., 2022).

Actually, long documents often contain a vari-

*Corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 1688–1700
July 9-14, 2023 ©2023 Association for Computational Linguistics

ety of subjects, as evidenced by the scope hypothesis (Robertson et al., 2009) from traditional information retrieval. An illustration from the MS MARCO dataset (Nguyen et al., 2016) is presented in Figure 1, and it is noted that only a small part of the document (e.g., p_2 and p_3) is relevant to the given query and different parts may be unequally informative to the query. Thus even though existing techniques for modeling long documents have been demonstrated to be effective and efficient, utilizing the entire document can result in high query latency and intensive query-irrelevant content can be a distraction and negatively impact performance. Consequently, some recent studies propose cascade document ranking models (Li et al., 2020; Hofstätter et al., 2021; Li and Gaussier, 2021) that extract relevant passages with an efficient selector before performing the ranking. However, their selection and ranking modules are almost independently optimized and deployed, leading to selecting error reinforcement and sub-optimal performance. Moreover, these models do not differentiate between the passages or segments taken from a document while matching with the query.

In fact, the document ranker can provide fine-grained supervision to enhance the generalizability and compatibility of the selector. Conversely, the selector, built upon a heterogeneous structure, can offer a distinct perspective to assist in document ranking. Taking inspiration from this, we propose a **F**ine-grained **A**ttention **A**lignment approach (FAA) to jointly optimize a cascade document ranking model. Specifically, we initialize the passage selector as an efficient dual encoder and the document ranker with an effective cross-encoder. To better optimize and make use of both worlds, we leverage the attention activations over the passages from the ranker as fine-grained attention feedback to optimize the selector. Simultaneously, we incorporate the relevance scores from the passage selector into the ranker to assist in calculating the final cooperative matching representation. We conduct experiments on three public benchmarks: MS MARCO (Nguyen et al., 2016), TREC-DL 2019 (Craswell et al., 2020), and TREC-DL 2020. The evaluation results show that our proposed model is better than several competitive baselines and our FAA can bring significant improvement to the cascade model. To sum up, our contribution is three-fold:

- We propose a **F**ine-grained **A**ttention **A**lignment approach to jointly optimize a cascade document ranking model.
- We explore fusing the passage-level relevance scores into the document ranker to produce the cooperative matching representation.
- We conduct extensive experiments and analysis on three benchmarks and the evaluation results show the effectiveness of our model.

2 Related Works

Neural models for document ranking In the early stages, traditional algorithms like BM25 (Robertson et al., 2009) and TF-IDF were commonly employed for ranking documents in information retrieval. With the development of neural network technology (Cho et al., 2014; Gu et al., 2018), some neural-based ranking models have been proposed (Huang et al., 2013; Guo et al., 2016; Hui et al., 2017, 2018; MacAvaney et al., 2020). Xiong et al. (2017) proposed a kernel-based neural ranking model (K-NRM) which used a kernel-pooling layer to combine word pair similarities with distributed representations. Dai et al. (2018) extended K-NRM to Conv-KNRM which used Convolutional Neural Networks to model n-gram embedding. Hofstätter et al. (2020b) proposed a Transformer-Kernel model which used a small number of transformer layers to contextualize query and document sequences independently and distilled the interactions between terms. Compared to traditional methods, neural ranking models produce a dense representation of the queries and documents which improves the ranking performance.

Pre-trained models for document ranking Recently a large number of transformer-based pre-trained models have been proposed (Devlin et al., 2019; Lewis et al., 2020; Radford et al., 2019; Raffel et al., 2020) and shown their effectiveness in natural language processing tasks. Therefore many works have utilized pre-trained models in document ranking tasks. Nogueira et al. (2019) used a sequence-to-sequence transformer model with document terms as input and produced the possible questions that the document might answer to expand document for document retrieval. Finally this work used BERT to re-rank these retrieved documents. Yan et al. (2019) used a pre-trained BERT model to classify sentences into three categories and then fine-tuned the model using a point-wise ranking method for ranking documents.

Passage-level document ranking Since the high demand for memory space and computing resources, pre-trained models usually have a limit on the input length, and the length of actual long documents is always beyond this limitation. To this end, some works proposed to split the long documents into multiple passages which satisfy the limitation of the input length of the pre-trained models (Li et al., 2020; Hofstätter et al., 2020a; Yang et al., 2019). The studies applied pre-trained models to each passage individually and then combined the relevance scores at the passage level to generate the relevance scores for the entire document. For example, Dai and Callan (2019) determined the relevance score of the document by utilizing the score of the first passage, the top-performing passage, and the summation of all passages, respectively. Fu et al. (2022) proposed a Multi-view inter-passage Interaction based Ranking model (MIR) with intra-passage attention and inter-passage attention, and used a multi-view aggregation layer to produce the document-level representation across multiple granularities. These works took all passages into document ranking which may introduce noise from the query-irrelevant passages and increase the query latency. To address this problem, some works proposed to pre-select query-relevant passages from all passages before aggregating. In this work, we propose a cooperative distillation and representation cascade ranking model which uses an efficient model as a passage selector to calculate passage-level relevance scores and select top-k passages, while uses an effective model as the document ranker to calculate the document-level relevance scores with the selected passages.

3 Methodology

In this section we first formalize the document ranking task, then we introduce the model architecture and the proposed Fine-grained Attention Alignment (FAA) approach for model optimization.

Task Formalization Given a query q and a set of candidate documents $\mathcal{C} = \{d_1, d_2, ..., d_m\}$ including both the ground-truth document and negative documents, where m is the number of the candidate documents, the task is to train a document ranking model $\mathcal{R}(q, d)$ with the training data $\mathcal{D}$. When provided with a new query and its corresponding candidate documents, the ranking model assesses the relevance between the query and each candidate document by computing relevance scores. Subsequently, it can arrange the documents in order based on these scores.

Model Overview Inspired by previous work on passage-level evidence for document ranking (Hofstätter et al., 2021; Li and Gaussier, 2021), in this paper we adopt the efficient and effective cascade document ranking paradigm which first extracts relevant passages with an efficient selector and then performs the ranking with a smart document ranker based on the pruned content. To better optimize and make use of both worlds, we propose a fine-grained attention alignment approach to jointly optimize a cascade document ranking model. Specifically, we utilize the attention activations over the passages from the ranker as fine-grained attention feedback to optimize the selector. Additionally, in the process of document ranking, the passage-level relevance scores in the selector are fused in the document ranker to produce the cooperative matching representation for calculating the final matching score. By this means, the document ranker can provide fine-grained supervision to make the selector more generalizable and compatible, and the selector built upon a heterogeneous structure can conversely offer a distinct view to help the ranker. Figure 2 presents the high-level architecture of the proposed method.

3.1 Passage Selector

To satisfy the input length limit of the pre-trained models, the candidate documents are first split into multiple passages with a sliding window in the size of l words and a stride in the size of s words. The set of passages of document d can be formalized as

$$\mathbb{P} = \{d_{0:\ l}, d_{s:\ s+l}, d_{2*s:\ 2*s+l}, ...\} \tag{1}$$

In the phase of passage selection, the passage selector identifies and extracts a subset of passages that are highly relevant to the given query. We adopt the simple and efficient dual-encoder structure built on a small pre-trained model as the passages selector which has a lower query latency. Given the query q and the set of passages $\mathbb{P} = \{p_1, p_2, \cdots, p_w\}$ where w is the number of passages, q and each $p_i \in \mathbb{P}$ are fed into the passage selector and encoded as d-dimensional vectors respectively which are denoted as $\mathsf{Enc}_{\mathsf{psg}}(q)$ and $\mathsf{Enc}_{\mathsf{psg}}(p_i)$. With the representative vectors of

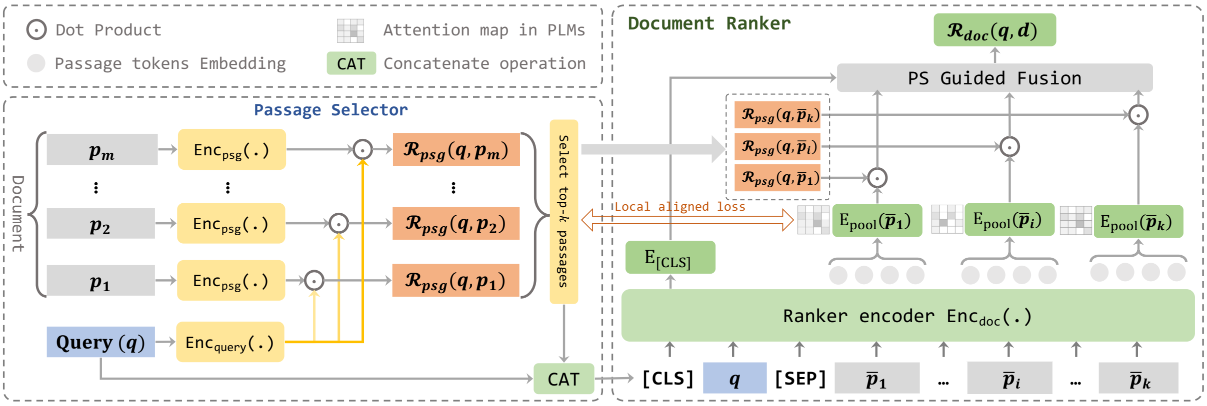

Figure 2: Overall architecture of our model (FAA). The selector encodes the query and passages with sharing parameters.

query and passages, the passage selector calculates the dot product between $\mathsf{Enc}_{\mathsf{psg}}(q)$ and $\mathsf{Enc}_{\mathsf{psg}}(p_i)$:

$$\mathcal{R}_{\mathsf{psg}}(q, p_i) = \frac{\mathsf{Enc}_{\mathsf{psg}}(q)^T \mathsf{Enc}_{\mathsf{psg}}(p_i)}{\sqrt{d}} \tag{2}$$

The passage-level relevance scores are scaled by dividing by $\sqrt{d}$. Next, the passage selector selects the k passages with the highest relevance score to form $\bar{\mathbb{P}}$, which is formalized as:

$$\bar{\mathbb{P}} = \mathop{\arg\max}_{\bar{\mathbb{P}} \subset \mathbb{P},\ \|\bar{\mathbb{P}}\| = k} \sum_{p_i \in \bar{\mathbb{P}}} \mathcal{R}_{\mathsf{psg}}(q, p_i) \tag{3}$$

Passages in $\bar{\mathbb{P}}$ contain informative content for query and are used for document ranking. By selecting the most relevant top-k passages $\bar{\mathbb{P}}$ from all the passages, the passage selector filters out a large number of irrelevant passages for document ranking processes, which can reduce the query latency and avoid the noise caused by irrelevant passages.

3.2 Document Ranker

We adopt a cross-encoder based on pre-trained models as the document ranker to calculate the document-level relevance score with $\bar{\mathbb{P}}$. The architecture performs full attention across the query and the extracted passages and has been proven to be effective for ranking (Hofstätter et al., 2021). Formally, all selected passages in $\bar{\mathbb{P}}$ are first spliced together as $\hat{\mathbb{P}} = \{\bar{p}_1; \bar{p}_2; \cdots; \bar{p}_k\}$, and then we concatenate query and the spliced passages $\hat{\mathbb{P}}$ as the input of the document ranker with [CLS] and [SEP] tokens, which is denoted as u:

$$u = \{[\mathsf{CLS}]; q; [\mathsf{SEP}]; \hat{\mathbb{P}}; [\mathsf{SEP}]\} \tag{4}$$

The document ranker performs semantic interaction through multi-layer attention blocks and outputs a sequence of contextualized representations. Typically, the output representation of the first token [CLS] is adpoted the encoded vector of u, namely $\mathsf{Enc}_{\mathsf{doc}}(u) = E_{[\mathsf{CLS}]}$. Then the vector is fed to a multilayer perceptron (MLP) to calculate the document-level relevance score:

$$\mathcal{R}_{\mathsf{doc}}(q, d) = \mathsf{MLP}(\mathsf{Enc}_{\mathsf{doc}}(u)) \tag{5}$$

Since the dataset provides the positive document for each query, the loss function we use to optimize the document ranker is defined below following the previous works (Wu et al., 2018; Oord et al., 2018):

$$\mathcal{L}_{\mathsf{rank}} = -\sum_{q \in \mathcal{D}} \log \frac{\exp(\mathcal{R}_{doc}(q, d^+))}{\sum_{d \in \mathcal{C}} \exp(\mathcal{R}_{doc}(q, d))} \tag{6}$$

where d^+ is the ground-truth document for the query q and $\mathcal{C}$ is a set of document candidates (including both the ground-truth document and negative documents) for q.

3.3 Cooperative Matching Representation

Considering the passage selector is based on heterogeneous dual-encoder architecture, we think that the selector can offer a distinct view to help document ranking. Therefore, different from traditional encoding which only uses the encoded vector of the first token [CLS] as the representation of sequence, we propose to fuse the selected passage-level relevance scores from the passage selector to produce the cooperative matching representation $\mathsf{Enc}_{doc}(u)$ of input sequence u. Specifically, we denote the embedding vector of [CLS] as $E_{[\mathsf{CLS}]}$ and denote the embedding vector of tokens in $\hat{\mathbb{P}}$ as $\{E_1^1, E_1^2, \cdots, E_i^j, \cdots\}$, where E_i^j represents the embedding vector of the j-th token in the i-th selected passage $\bar{p}_i$. To produce $\mathsf{Enc}_{doc}(u)$, we first calculate the average embedding vector of each selected passage:

$$\texttt{MeanPool}(\bar{p}_i) = \sum_{z=1}^{l} E_i^z / l \tag{7}$$

where l is the length of $\bar{p}_i$. We then calculate the product of the passage-level relevance scores from the selector and the average vector of the passage in the ranker, and take the summation of the results as the passage-selector guided vector E_{PGV}, formalized as:

$$E_{\mathsf{PGV}} = \sum_{t=1}^{k} \texttt{MeanPool}(\bar{p}_i) \cdot \mathcal{R}_{\mathsf{psg}}(q, \bar{p}_i) \tag{8}$$

Finally we fuse the passage-selector guided vector E_{PGV} with $E_{[\mathsf{CLS}]}$ to get the cooperative document-level matching representation:

$$\mathsf{Enc}_{doc}(u) = E_{[\mathsf{CLS}]} + \lambda \cdot E_{\mathsf{PGV}} \tag{9}$$

where λ is a parameter to control the weight of E_{PGV}. Then we can feed the above $\mathsf{Enc}_{doc}(u)$ into a multi-layer perceptron to calculate the final document-level relevance score, as formalized in Equation 5. We can find that the more relevant a passage is, the greater its proportion in the fusion, which causes the document ranker to pay more attention to it.

3.4 Fine-grained Attention Alignement

As mentioned above, the passage selector is initialized by dual-encoder architecture which is efficient but performance sub-optimal compared with cross-encoder. It is not so compatible in ranking model and need to be tuned. Besides, there are no passage-level labels in most document ranking tasks. Inspired by knowledge distillation (Hinton et al., 2015; Wang et al., 2020), we use the complicated and effective document ranker as the teacher model to provide fine-grained supervision for optimizing the passage selector which is regarded as a student model to make the selector more generalizable and compatible. To be specific, with the self-attention mechanism in the transformer-based model, we use fine-grained attention activation scores over the passage as the pseudo labels of passages for optimization. We consider that if one passage is more informative to query, the query will provide more attention to it when document ranking which results in a higher attention score for this passage.

For the input u, the representation output by the previous layer is denoted as $H \in R^{l_u \times d}$ where l_u is the length of u. The self-attention module produces queries Q, keys K, and values V matrices through linear transformations (Vaswani et al., 2017), and then the attention map can be calculated as:

$$M = \texttt{softmax}(\frac{QK^T}{\sqrt{d}}) \tag{10}$$

where d is the dimension of vectors in Q. We denote $\alpha_{i \to j} = M_{i,j}$ as the attention score from i-th token to j-th token in u. Following the calculation of the attention score from one token to another token, we calculate the attention activation score of each selected passage $\bar{p}_i$ ($\in \bar{\mathbb{P}}$) as the maximal attention score from all tokens in query q and all tokens in the $\bar{p}_i$:

$$\begin{aligned} \alpha_{q \to \bar{p}_i} &= \texttt{MaxPool}(\widetilde{M}), \\ \widetilde{M} &= M_{x:x+l_q, y:y+l_{p_i}} \end{aligned} \tag{11}$$

where x, y is the starting token of q and p_i, and l_q, l_{p_i} is the length of q and p_i respectively. $\widetilde{M}$ is the attention map between q and p_i, where $\widetilde{M}_{i,j}$ is the attention score from i-th token in q to j-th token in p_i. We also experimented with the mean-pooling operation to calculate attention scores and found that it performed worse than max-pooling. Following previous knowledge distillation methods based on pre-trained language models (Wang et al., 2020), we also calculate the attention score of $\bar{p}_i$ in the last self-attention layer of the document ranker. Taking into account the multi-head attention mechanism in the transformer-based model, we select the maximal attention score through all attention heads as the final scores.

We use KL-divergence between the relevance scores of passages output by the passage selector and the attention scores as the loss function of the passage selector:

$$\mathcal{L}_{\texttt{align}} = \sum_{q \in \mathcal{D}} \texttt{KL-Div}(\mathcal{H}^{psg}(q, \bar{\mathbb{P}}), \mathcal{A}^{doc}(q, \bar{\mathbb{P}})) \tag{12}$$

where $\mathcal{H}_{psg}(q, \bar{\mathbb{P}})$ is the output distribution over the relevance scores of passages in $\bar{\mathbb{P}}$ from the selector, $\mathcal{A}_{doc}(q, \bar{\mathbb{P}})$ is the distribution of the aggregated attention scores in ranker. $\mathcal{H}^{psg}(q, \bar{p}_k)$ and $\mathcal{A}^{doc}(q, \bar{p}_k)$ are the k-th item in $\mathcal{H}^{psg}$ and $\mathcal{A}^{doc}$ respectively, which can be calculated as below:

$$\mathcal{H}^{psg}(q, \bar{p}_k) = \frac{\exp(\mathcal{R}_{\mathsf{psg}}(q, \bar{p}_k)/\tau)}{\sum_{\bar{p} \in \bar{\mathbb{P}}} \exp(\mathcal{R}_{\mathsf{psg}}(q, \bar{p})/\tau)} \tag{13}$$

Algorithm 1 The proposed FAA

Require: Training set $\mathcal{D}$, selector parameters $\phi_{\mathtt{psg}}$, ranker parameters $\phi_{\mathtt{doc}}$
  Initialize parameters $\phi_{\mathtt{psg}}$, $\phi_{\mathtt{doc}}$
  repeat
    Sample a batch $\mathcal{B}$ from $\mathcal{D}$
    Compute passage relevance scores by Eq (2)
    Select top-k relevant passages $\bar{P}$ by Eq (3)
    Compute document relevance scores with $\bar{P}$
    Compute $\mathcal{L}_{\mathtt{rank}}$ on $\mathcal{B}$ and *optimize* $\phi_{\mathtt{doc}}$
    Compute attention score $P_{att}(\bar{p}_i)$ by Eq (14)
    Compute $\mathcal{L}_{\mathtt{align}}$ on $\mathcal{B}$ and *optimize* $\phi_{\mathtt{psg}}$
  until Convergence
  Return $\phi_{\mathtt{psg}}$, $\phi_{\mathtt{doc}}$

$$\mathcal{A}^{doc}(q, \bar{p}_k) = \frac{\exp(\alpha_{q \to \bar{p}_k}/\tau)}{\sum_{\bar{p} \in \bar{\mathbb{P}}} \exp(\alpha_{q \to \bar{p}}/\tau)} \tag{14}$$

where τ is the temperature hyper-parameter.

Above all, in our overall ranking model, the loss function can be described as the combination of the loss for the document ranker and the attention alignment loss:

$$\mathcal{L}_{\mathtt{final}} = \mathcal{L}_{\mathtt{align}} + \mathcal{L}_{\mathtt{rank}} \tag{15}$$

In this work, we tried to jointly train the passage selector and document ranker. Particularly, we update the ranker with only $\mathcal{L}_{\mathtt{rank}}$, and the gradient from $\mathcal{L}_{\mathtt{align}}$ is stopped. Algorithm 1 gives a pseudo-code of our training process.

4 Experiments

In this section, we first introduce the datasets we use, the evaluation metrics, the baselines, and the implementation details of our experiment. Then we introduce the evaluation results and further analysis of our method.

4.1 Datasets and Evaluation

In line with previous studies on this task (Hofstätter et al., 2021; Li and Gaussier, 2021), we conduct an evaluation of our proposed model on three publicly available document ranking datasets: MSMARCO (Nguyen et al., 2016), TREC-DL 2019 (Craswell et al., 2020), and TREC-DL 2020. The MS-MARCO dataset comprises 3.2 million documents and 367,013 training queries, sourced from web pages. For evaluation, we utilize the MS-MARCO DEV set, which consists of 5,193 queries. The evaluation metrics employed are NDCG@10, MAP, and MRR@10. Both the TREC-DL 2019 and TREC-DL 2020 datasets share the same document collection as MS-MARCO and include 43 and 45 queries, respectively. For both TREC-DL datasets, we employ NDCG@10 and MAP as the evaluation metrics. Across all datasets, we perform document re-ranking based on the top 100 documents retrieved by BM25.

4.2 Baselines

We compare our model with traditional and neural document ranking models:

- **BM25** (Robertson et al., 2009) is a widely-used unsupervised text-retrieval algorithm based on IDF-weighted counting.
- **BERT-MaxP** (Dai and Callan, 2019) uses BERT to encode passages split from the document to calculate the relevance score and choose the best passage-level score as the document-level score.
- **Sparse-Transformer** (Child et al., 2019) introduces several sparse factorizations of the attention matrix.
- **LongFormer-QA** (Beltagy et al., 2020) extends Sparse-Transformer by attaching two global attention tokens to the query and the document as their settings for QA.
- **Transformer Kernel Long** (Hofstätter et al., 2020a) proposes a local self-attention mechanism with the kernel-pooling strategy.
- **Transformer-XH** (Zhao et al., 2020) introduces an extra hop attention layer that can produce a more global representation of each piece of text.
- **QDS-Transformer** (Jiang et al., 2020) proposes a query-directed sparse transformer-based ranking model which uses sparse local attention to obtain high efficiency.
- **KeyBLD** (Li and Gaussier, 2021) proposes using local query-block pre-ranking to choose key blocks of a long document and aggregates blocks to form a short document which is further processed by BERT.

Models	MSMARCO DEV			TREC DL 2019		TREC DL 2020	
	NDCG@10	MAP	MRR@10	NDCG@10	MAP	NDCG@10	MAP
BM25 (Robertson et al., 2009)	0.311	0.265	0.252	0.488	0.234	-	-
BERT-MaxP (Dai and Callan, 2019)	-	-	-	0.642	0.257	0.630	0.420
Sparse-Transformer (Child et al., 2019)	-	-	-	0.634	0.257	-	-
LongFormer-QA (Beltagy et al., 2020)	-	-	-	0.627	0.255	-	-
Transformer Kernel Long (Hofstätter et al., 2020a)	0.403	0.345	0.338	0.644	0.277	0.585	0.381
Transformer-XH (Zhao et al., 2020)	-	-	-	0.646	0.256	-	-
QDS-Transformer (Jiang et al., 2020)	-	-	-	0.667	0.278	-	-
$\text{PARADE}_{\text{Max-Pool}}$ (Li et al., 2020)	0.445	-	-	0.679	0.287	0.613	0.420
$\text{PARADE}_{\text{TF}}$ (Li et al., 2020)	0.446	0.387	0.382	0.650	0.274	0.601	0.404
KeyBLD (Li and Gaussier, 2021)	-	-	-	0.707	0.281	0.618	0.415
IDCM (Hofstätter et al., 2021)	0.446	0.387	0.380	0.679	0.273	-	-
FAA	<u>0.453</u>	<u>0.397</u>	<u>0.390</u>	0.685	0.275	<u>0.647</u>	<u>0.424</u>

Table 1: Performance of different methods on the document ranking task in MSMARCO DEV and TREC-DL dataset. The best results are in underlined fonts.

- **PARADE** (Li et al., 2020) truncates a long document into multiple passages and uses different strategies to aggregate the passage-level relevance scores. $\textbf{PARADE}_{\textbf{Max-Pool}}$ uses max-pooling to obtain document-level relevance scores and $\textbf{PARADE}_{\textbf{TF}}$ uses a transformer encoder for passages aggregation.

- **IDCM** (Hofstätter et al., 2021) uses a fast model (ESM) for passage selection and a effective model (ETM) for document ranking, where optimizes the ESM with the knowledge distillation from ETM to ESM.

4.3 Implementation Details

Our proposed model is implemented by the transformer library provided by hugging face[1]. We use DistilBERT (Sanh et al., 2019) to initialize our passage selector which is more efficient and has comparable performance with BERT-base. For document ranking, we use the publicly trained model[2] to initialize our document ranker. We set the length of the sliding window and stride as 72. The query length is set as 30 and the number of selected passages is set as 3. We use Adam optimizer (Kingma and Ba, 2015) to train our model with batch size set as 4. The initial learning rate of the passage selector and document ranker are set as 5e-7 and 7e-6 respectively. We vary λ (Equation (9)) in $\{0.1, 0.2, 0.5, 1.0\}$ and find that 0.2 is the best choice. τ in Equation (13) and Equation (14) is set as 0.2.

4.4 Evaluation Results

The evaluation results of our proposed model and all baselines on MS MARCO, TREC-DL 2019 and TREC-DL 2020 are reported in Table 1. First, compared with the models with more efficient attention mechanisms in transformer (e.g. Sparse-Transformer, Transformer-XH, QDS-Transformer, and Transformer Kernel Long), our method and other cascade document ranking models (e.g. KeyBLD and IDCM) can achieve better performance on almost all metrics. The phenomenon indicates the superiority of the cascade document ranking paradigm. Second, compared with two previous cascade methods[3] (e.g. IDCM) that select passage before ranking, our model has better performance than them on MS-MARCO and TREC DL 2020, and shows comparable performance on TREC DL 2019. Different from these baselines which optimize the selector and ranker independently, our model jointly optimizes the selector and ranker with fine-grained attention alignment. Meanwhile, we utilize the passage-level relevance scores in document ranking to obtain cooperative fusion representation. The evaluation results demonstrate the effectiveness of our proposed methods.

4.5 Discussions

Ablation study Table 2 presents the findings from our ablation study, where we systematically remove specific components to assess their impact on performance. Firstly, we eliminate the fine-grained attention alignment for the passage selec-

[1] https://huggingface.co/docs/transformers/

[2] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

[3] As of now, a direct comparison with KeyBLD is not feasible due to the lack of reported results on the MS-MARCO DEV dataset and the unavailability of the source code.

Models	NDCG@10	MAP	MRR@10
FAA	**0.453**	**0.397**	**0.390**
w/o. $\mathcal{L}_{\texttt{align}}$	0.361	0.313	0.290
w/o. E_{PGV}	0.449	0.393	0.385
w/o. $\{\mathcal{L}_{\texttt{align}}$ & $E_{\text{PGV}}\}$	0.358	0.312	0.288
$\mathcal{R}_{\text{psg}}(q, \bar{p}_i) = 1/k$	0.449	0.394	0.386
$\alpha_{q \to \bar{p}_i} = \texttt{MeanPool}(\widetilde{M})$	0.436	0.380	0.352

Table 2: Ablation Study.

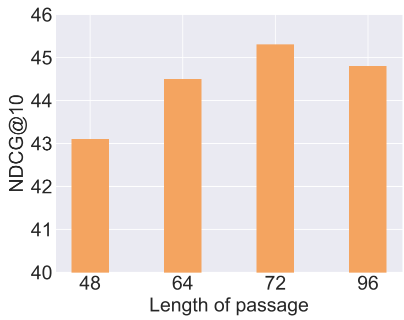

Figure 3: The impact of length of the split passage on MS-MARCO Dev.

tor, denoted as "w/o. $\mathcal{L}_{\texttt{align}}$". Next, we remove the passage-level multi-vector fusion during document ranking, denoted as "w/o. E_{PGV}". The results reveal that removing either $\mathcal{L}_{\texttt{align}}$ or E_{PGV} leads to a drop in performance, indicating the effectiveness of our fine-grained attention alignment approach and the importance of utilizing cooperative fusion representation to enhance the ranker's capabilities. Notably, removing both components simultaneously results in an even greater performance decrease. Furthermore, we examine the use of average pooling in representation fusion, denoted as "$\mathcal{R}_{\text{psg}}(q, \bar{p}_i) = 1/k$," which replaces $\mathcal{R}_{\text{psg}}$ in Eq. 8 with $\frac{1}{k}$. Our findings indicate that simply incorporating average pooling of passage representations does not yield substantial gains, as it only achieves comparable performance to the model without E_{PGV}. Notably, the performance of "$\mathcal{R}_{\text{psg}}(q, \bar{p}_i) = 1/k$" and the model without E_{PGV} are inferior to that of our model, illustrating the utility and superiority of cooperative fusion of relevance scores from selectors over independent representation fusion. Lastly, we explore the use of mean-pooling operation for calculating attention scores and observe that it performs worse than max-pooling.

The impact of passage length When constructing the training data, the length of the split passage plays a vital role as it also indirectly controls the number of passage candidates for each document. To investigate the impact of the passage length, we test the performance of our method across different passage lengths and the results are shown in Figure 3. We can find that the performance of our model improves until the passage length reaches 72, and then drops when the passage length keeps increasing. The reason might be that the selector needs to rank fewer candidates as the passage length increases at first and it could select more accurate passages that are relevant to the query for matching, but when the length of the passage becomes larger enough, the noise will be brought to matching as some content in each passage could be irrelevant to the query.

# PSG	NDCG@10	MAP	MRR@10
1	0.389	0.340	0.331
2	0.440	0.384	0.377
3	**0.453**	**0.397**	**0.390**
4	0.451	0.390	0.387

Table 3: The performance across different numbers of selected passages on MS-MARCO Dev.

Query: *how many mm is a nickel coin*

PID	Content	Rank / $\mathcal{R}_{\text{psg}}$
0	... Nickel United States Value 0.05 U. S. dollar Mass 5.000 g, Diameter 21.21 mm (0.835 in)...	1 / 0.954
2	... Its diameter is .835 inches (21.21 mm) and its thickness is .077 inches (1.95 mm)...	2 / 0.934
1	...War Nickels" (mid-1942 to 1945): 56% copper, 35% silver, 9% manganese Silver 1942 to 1945 Wartime Nickels only...	3 / 0.759
11	...The half dime was originally struck from 1794 until 1805, though none were dated 1798, 1799, or 1804....	20 / 0.468

Table 4: A case study from MS-MARCO dataset. The term "PID" refers to the position number of a passage that has been split from the test document. $\mathcal{R}_{\text{psg}}$ denotes relevance scores provided by the passage selector while "Rank" signifies the rank of the passage based on these relevance scores.

The impact of the number of selected passages We are also curious about the impact of the number of selected passages. We test the performance of our method with different numbers of selected passages and the evaluation results are illustrated in Table 3. We can observe that when the performance of our model was significantly improved as the number of selected passages increased at the beginning (≤ 3), and then began to drop when the number kept increasing. The results are rational because more passage entries can provide more useful information for response matching, but when the passage becomes enough, query-irrelevant noise will be brought to matching.

Case study To verify the effectiveness of our cascade model in document ranking, we show a ranker example from MS-MARCO dataset in Table 4. For the input query *how many mm is a nickel coin*, our FAA ranks the positive document at first and it is split into 24 passages. We show the top-3 passages selected by our passage selector and a random passage that is not selected. We can find that the top 2 passages harbor a significant amount of valuable query-relevant information, encompassing terms like "nickel" and "diameter." Conversely, the final passage, which displays lesser relevance to the query, receives a lower relevance score as determined by the passage selector. This case serves as an illustration of our model's proficiency in selecting pertinent content within the document and ranking it based on query relevance.

5 Conclusion

In this work, we propose FAA, a cascade ranking model with a fine-grained attention alignment and cooperative matching representation. Our model utilizes the fine-grained attention alignment approach to train the passage selector and fuses the passage-level relevance scores into document ranking to produce cooperative matching representation. The evaluation results on MS MARCO and TREC DL demonstrate the effectiveness of FAA.

6 Limitations

While our approach effectively mitigates query latency through a cascade ranking paradigm, it necessitates additional computational resources during training due to the need for attention score calculation and alignment in the optimization process. Additionally, our model incorporates passage-level relevance scores into the ranker, generating a cooperative matching representation during document ranking, which could marginally augment the inference time. In our future endeavors, we aim to explore more efficient methodologies that can further improve ranking efficiency. Furthermore, it is worth noting that our approach has been tested using specific backbone models. To fully evaluate the effectiveness of our method, it is essential to conduct experiments with a diverse range of backbone models, which remains an avenue for further exploration.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by the National Key Research and Development Program of China (No. 2021YFC3340304).

Ethical Statement

Our paper centers around the document ranking task, a well-established and widely applicable problem. In conducting our research, we have exclusively utilized queries and documents sourced from open public datasets, with proper citation and adherence to licensing agreements. We have taken great care to ensure that our experiments have no bearing on privacy security, discrimination, or bias. We affirm that our work aligns with ethical principles and regulations, and it does not infringe upon any established ethical codes or guidelines.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Leonid Boytsov, Tianyi Lin, Fangwei Gao, Yutian Zhao, Jeffrey Huang, and Eric Nyberg. 2022. Understanding performance of long-document ranking models through comprehensive evaluation and leaderboarding. *arXiv preprint arXiv:2207.01262*.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online. Association for Computational Linguistics.

Junying Chen, Qingcai Chen, Dongfang Li, and Yutao Huang. 2022. Sedr: Segment representation learning for long documents dense retrieval. *arXiv preprint arXiv:2211.10841*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger

Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 985–988.

Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 126–134.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chengzhen Fu, Enrui Hu, Letian Feng, Zhicheng Dou, Yantao Jia, Lei Chen, Fan Yu, and Zhao Cao. 2022. Leveraging multi-view inter-passage interactions for neural document ranking. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 298–306.

Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. 2018. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 55–64.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. 2021. Intra-document cascading: learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1349–1358.

Sebastian Hofstätter, Hamed Zamani, Bhaskar Mitra, Nick Craswell, and Allan Hanbury. 2020a. Local self-attention over long text for efficient document retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2021–2024.

Sebastian Hofstätter, Markus Zlabinger, and Allan Hanbury. 2020b. Interpretable & time-budget-constrained contextualization for re-ranking. In *ECAI 2020*, pages 513–520. IOS Press.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

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.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard De Melo. 2018. Co-pacrr: A context-aware neural ir model for ad-hoc retrieval. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 279–287.

Jyun-Yu Jiang, Chenyan Xiong, Chia-Jung Lee, and Wei Wang. 2020. Long document ranking with query-directed sparse transformer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4594–4605, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. Parade: Passage representation aggregation for document reranking. *arXiv preprint arXiv:2008.09093*.

Minghan Li and Eric Gaussier. 2021. Keybld: Selecting key blocks with local pre-ranking for long document information retrieval. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2207–2211.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–58.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 1101–1104.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPs*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Koustav Rudra and Avishek Anand. 2020. Distant supervision in bert-based adhoc document retrieval. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2197–2200.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 55–64.

Ming Yan, Chenliang Li, Chen Wu, Bin Bi, Wei Wang, Jiangnan Xia, and Luo Si. 2019. Idst at trec 2019 deep learning track: Deep cascade ranking with generation-based document expansion and pre-trained language modeling. In *TREC*.

Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple applications of bert for ad hoc document retrieval. *arXiv preprint arXiv:1903.10972*.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitation section

☒ A2. Did you discuss any potential risks of your work?
The topic of the paper deals only with document retrieval

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Introduction section

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

4 Experiments section

☑ B1. Did you cite the creators of artifacts you used?
4 Experiments section

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
MS-MARCO and Trec DL are open-source datasets

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Our use of MS-MARCO and Trec DL was consistent with their intended use.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4 Experiments section

C ☑ Did you run computational experiments?

4 Experiments section

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4 Experiments section

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4 Experiments section

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4 Experiments section

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.