Hybrid and Collaborative Passage Reranking

Zongmeng Zhang[1], **Wengang Zhou**[2], **Jiaxin Shi**[3], **Houqiang Li**[2]
[1,2]University of Science and Technology of China
[3]Huawei Cloud Computing Technologies Co., Ltd.
[1]zhangzm@mail.ustc.edu.cn, [2]{zhwg, lihq}@ustc.edu.cn, [3]shijx12@gmail.com

Abstract

In passage retrieval system, the initial passage retrieval results may be unsatisfactory, which can be refined by a reranking scheme. Existing solutions to passage reranking focus on enriching the interaction between query and each passage separately, neglecting the context among the top-ranked passages in the initial retrieval list. To tackle this problem, we propose a ***Hyb**rid and Collaborative Passage **Rank**ing* (**HybRank**) method, which leverages the substantial similarity measurements of upstream retrievers for passage collaboration and incorporates the lexical and semantic properties of sparse and dense retrievers for reranking. Besides, built on off-the-shelf retriever features, HybRank is a plug-in reranker capable of enhancing arbitrary passage lists including previously reranked ones. Extensive experiments demonstrate the stable improvements of performance over prevalent retrieval and reranking methods, and verify the effectiveness of the core components of HybRank.[1]

1 Introduction

Information retrieval is a fundamental component within the field of natural language processing (Chen et al., 2017). Retrieval aims to search a set of candidate documents from a large-scale corpus, and thus high recall retrieval with efficiency is required to cover more relevant documents as far as possible. Traditionally, retrieval has been dominated by lexical methods like TF-IDF and BM25 (Robertson and Zaragoza, 2009), which treat queries and documents as sparse bag-of-words vectors and match them in token-level. Recently, neural networks have become prevalent to deal with information retrieval, where queries and documents are encoded into dense contextualized semantic vectors (Huang et al., 2020; Karpukhin et al., 2020; Ren et al., 2021a; Zhang et al., 2022), and then retrieval is performed with highly optimized vector search algorithms (Johnson et al., 2021).

Although numerous efforts have been dedicated to retrieval, the inherent efficiency requirement restricts the interaction between query and passage to a shallow level, leading to unsatisfactory retrieval results. Thus, in typical reranking (Nogueira and Cho, 2020; Sun et al., 2021), query and passage are concatenated and fed into a Transformer (Vaswani et al., 2017) pre-trained on large corpus, to estimate a more fine-grained relevance score and further enhance the retrieval results with richer interaction. These methods consider each passage in isolation, ignoring the context of the retrieved passage list. Some learning to rank (Rahimi et al., 2016; Xia et al., 2008) and pseudo-relevance feedback (Zamani et al., 2016; Zhai and Lafferty, 2001) methods utilize the ordinal relationship or listwise context of retrieved documents to further refine the retrieval. Moreover, the necessity of integrating listwise context is confirmed in multi-stage recommendation systems (Liu et al., 2022).

Inspired by the success of listwise modeling and collaborative filtering (Goldberg et al., 1992) in recommendation systems, we find that collaboration also exists among the passages in the retrieval list and has not been fully exploited. Intuitively, for a specific query, a set of passages relevant to the query tend to describe the same entities, events and relations (Lee et al., 2019), while irrelevant ones outside of this set involve multifarious objects. Therefore, a passage is more likely to be relevant with the query if most of other passages share similar content with it. Similarities between passages can be naturally derived from retrievers, like BM25 scores in sparse[2] retrievers and dot product of embeddings in dense retrievers.

In addition, the sparse and dense retrieval methods emphasize distinct linguistic aspects. Sparse

[1] Our code is available at https://github.com/zmzhang2000/HybRank

[2] To stand out in contrast to dense retrieval, lexical retrieval is referred to as term sparse retrieval in this paper.

Findings of the Association for Computational Linguistics: ACL 2023, pages 14003–14021
July 9-14, 2023 ©2023 Association for Computational Linguistics

retrieval relies on lexical overlap while dense retrieval focuses on semantic and contextual relevance. Several researchers have attempted to integrate the merits of these two types of methods. Karpukhin et al. (2020), Lin et al. (2020) and Luan et al. (2021) exploit the linear combination of these two types of retrieval scores. Seo et al. (2019), Khattab and Zaharia (2020) and Santhanam et al. (2022) index smaller units in sentence, *i.e.*, words or phrases, to obtain fine-grained similarity. Gao et al. (2021a) and Yang et al. (2021) retrain dense retriever from scratch with the supervision of sparse signals. Nevertheless, the linear score combination lacks sufficient interaction, indexing smaller units sacrifices efficiency due to tremendous amount of embeddings, while rebuilding of retrievers discards their origin ranking capability.

To fully exploit the context of retrieved passages list and explore more sufficient ensemble of heterogeneous retriever, we propose a ***Hyb**rid and Collaborative Passage **Rank**ing* (HybRank) method, which leverages the collaboration within retrieved passages and incorporates diverse properties of retrievers for reranking. Our method is a flexible plug-in reranker that can be applied to arbitrary passage lists, including those that have already been reranked by other methods. In this work, without loss of generality, we employ the two most representative types of retrievers: sparse and dense retriever. Given a query and an initial retrieval list, we first extract similarities between them and a set of anchor texts via both the sparse and dense retrievers. We project and group them to form a set of hybrid and collaborative sequences, each corresponding to a query or passage. Afterwards, the relevance scores between the query and these passages are evaluated in the light of these sequences.

Extensive experiments demonstrate the consistent performance improvement brought by HybRank over passage lists from prevalent retrievers and strong rerankers. We elaborate ablation studies on the collaborative information, feature hybrid, anchor-wise interaction and the number of anchor passages, verifying the impact and indispensability of these components in HybRank.

2 Method

In mainstream information retrieval systems, the first-stage retrieval is designed to fetch a coarse candidate list from a large corpus $\mathcal{C}$. Inevitably, false positives, *i.e.*, irrelevant passages in the retrieval list, are returned in the first-stage retrieval. To improve the precision of retrieval systems, the follow-up procedure reranking aims to distinguish the relevant passages from others in the retrieval list. This paper focuses on the reranking stage.

Formally, given a query q and an initial passage list $\mathcal{P} = [p_1, p_2, \dots, p_N]$ from upstream retriever, the reranking task is to reorder the passage list by reassigning scores $\mathcal{S} = [s_1, s_2, \dots, s_N]$ for each of these passages. We denote positive passages in the list as $\mathcal{P}^+$ and negative ones as $\mathcal{P}^-$. In this section, we will present the details of HybRank. The pipeline of HybRank is illustrated in Figure 1.

2.1 Preliminaries

Sparse Retrieval Traditionally, text retrieval is dominated by token-matching, where texts are encoded into high-dimensional sparse vectors using the statistic information of tokens. The most commonly-used sparse retrieval methods include TF-IDF, BM25 and so forth. We adopt BM25 score as the similarity metric of sparse retrieval due to its robustness and popularity.

Specifically, given the query q and the document d, the BM25 score between q and d is obtained by summing the BM25 weights over the terms co-occurred in q and d:

$$f^s(q,d) = \sum_{t \in q \cap d} w_t^{\mathrm{RSJ}} \frac{c_{t,d}}{k_1((1-b) + b\frac{|d|}{l}) + c_{t,d}}, \tag{1}$$

where t is a term, w_t^{RSJ} is t's Robertson-Spärck Jones weight, $c_{t,d}$ is the frequency of t in d, $|d|$ is the document length and l is the average length of all documents in the collection. k_1 and b are tunable parameters. Refer to Robertson and Zaragoza (2009) for more details about BM25.

Dense Retrieval Owning to the flexibility for a task-specific representation provided by learnable parameters, recent works leverage neural networks to encode text into dense vectors, and search similar documents for queries in vector space. Typically, the query and document are encoded separately, and the relevance score is measured by the similarity of their embeddings. Any neural architectures capable of encoding text into a single fixed-length vector are suitable for dense retrieval. We use the predominant Transformer (Vaswani et al., 2017) encoder and dot product similarity, formulated as

$$f^d(q,d) = \mathrm{T}_q(q)^\top \mathrm{T}_d(d), \tag{2}$$

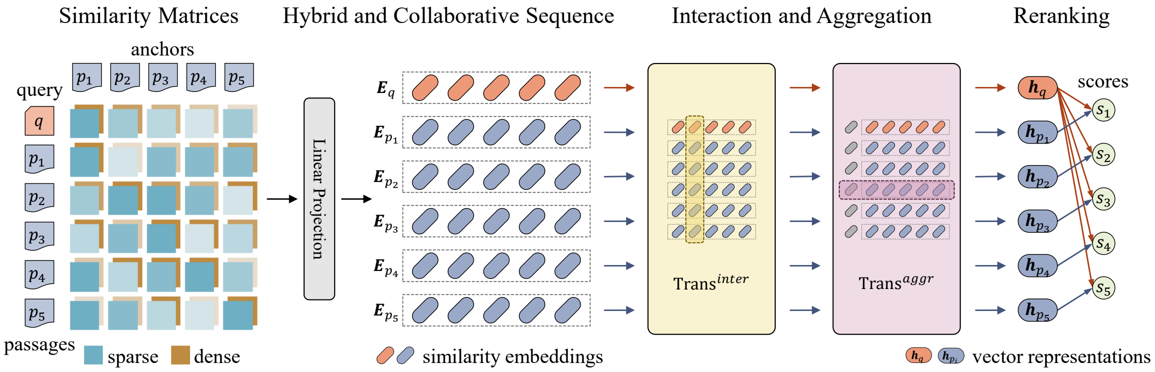


Figure 1: Illustration of HybRank pipeline. For a specific query, the passage list is initialized by an arbitrary retriever. The passage list may have been reranked by another reranker before HybRank. We display a 5-passage list as an example. First, similarities between query, passages and anchors are derived from sparse and dense retrievers. Then, these similarities are converted to hybrid and collaborative sequences as the representations of query and passages. Finally, these sequences are encoded into dense vectors via interaction and aggregation, and the reranking scores are obtained by dot product between the dense vectors of the query and each passage.

where $\mathrm{T}_q(\cdot)$ and $\mathrm{T}_d(\cdot)$ are Transformer encoders for queries and documents. Dot product similarity permits offline pre-encoding of large corpus and efficient retrieval via highly optimized vector nearest neighbor searching library (Johnson et al., 2021).

2.2 Hybrid and Collaborative Sequence

For a specific query, relevant passages tend to describe the same entities, events and relations from the query (Lee et al., 2019). In other words, most passages in the retrieval list would resemble to the true positive ones. Inspired by the success of collaborative filtering (Goldberg et al., 1992) in recommendation systems, we utilize the similarities between passages to distinguish the positive passages in the retrieval list.

Collaborative Sequence Similarity measurements can be naturally derived from retrievers. *e.g.*, BM25 score in sparse retriever and dot product in dense retriever as described in Section 2.1. We compute the similarity between each passage and a set of anchors, which are the top-L passages of the retrieval list in this work and will collaborate to distinguish the positive passages. These similarity scores between passages can be pre-computed, as HybRank utilizes off-the-shelf retrievers. Denoting similarity score between passage p_i and p_j as $f_{ij} \in \mathbb{R}$, the passage p_i can be represented as a sequence of similarity scalars $\boldsymbol{x}_{p_i} = [f_{i1}, f_{i2}, \ldots, f_{iL}] \in \mathbb{R}^L$.

Nevertheless, according to our observation, the similarity scalars within a retrieval list tend to concentrate on a small range. This is a reasonable phenomenon for that retrievers fetch relatively similar passages from the large corpus. To obtain more distinctive features, we employ a temperature $\mathrm{softmax}$ to stretch the distribution of similarities. After that, a $\mathrm{min\text{-}max}$ normalization is applied to scale them into range $[-1, 1]$. These two transforms are formulated as

$$\begin{aligned} \boldsymbol{x} &= \mathrm{softmax}(\boldsymbol{x}/t), \\ \boldsymbol{x} &= 2 \cdot \frac{\boldsymbol{x} - \min(\boldsymbol{x})}{\max(\boldsymbol{x}) - \min(\boldsymbol{x})} - 1, \end{aligned} \tag{3}$$

where t is the temperature. Subscripts of $\boldsymbol{x}_{p_i}$ are omitted for brevity.

Feature Hybrid Similarity metrics of sparse and dense retrievers concentrate on lexical overlap and semantic relevance, respectively. To combine the lexical and semantic properties embedded in sparse and dense retrievers, we mix their similarity scores[3] by stacking them in a channel manner. Formally, we substitute the similarity scalar f_{ij} in $\boldsymbol{x}_{p_i}$ with a vector $\boldsymbol{x}_{ij} = [f^s_{ij}, f^d_{ij}] \in \mathbb{R}^2$, where f^s_{ij} is the sparse similarity computed as Eqn. 1 and f^d_{ij} is the dense similarity computed as Eqn. 2. After that, the representation of passage p_i is turned into a sequence of similarity vectors $\boldsymbol{X}_{p_i} = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{iL}] \in \mathbb{R}^{L \times 2}$. Additionally, we map these similarity vectors in the sequence to D dimension with a trainable linear projection:

$$\boldsymbol{e}_{ij} = \boldsymbol{x}_{ij} \boldsymbol{W}, \tag{4}$$

[3] In this paper, we refer to similarity score from sparse and dense retrievers as sparse similarity and dense similarity, respectively.

where $\boldsymbol{W} \in \mathbb{R}^{2\times D}$ is a learnable parameter and $\boldsymbol{e}_{ij} \in \mathbb{R}^D$ are embedded similarities. Thereafter, passage p_i's representation becomes a sequence of similarity embeddings $\boldsymbol{E}_{p_i} = [\boldsymbol{e}_{i1}, \boldsymbol{e}_{i2}, \ldots, \boldsymbol{e}_{iL}] \in \mathbb{R}^{L\times D}$, which comprises the similarity information between p_i and anchor passages originating from both sparse and dense retrievers. These similarities deliver substantial information for the collaboration of passages and hold both the lexical and semantic properties from retrievers. With the same procedure, we compute the similarities between query and anchors, and derive the query representation $\boldsymbol{E}_q = [\boldsymbol{e}_{q1}, \boldsymbol{e}_{q2}, \ldots, \boldsymbol{e}_{qL}] \in \mathbb{R}^{L\times D}$. Noted that the similarities from sparse and dense retriever are stretched and normalized individually before linear projection, as described in Eqn. 3.

Consequently, we obtain $N + 1$ collaborative sequences in total, each representing a passage or a query and consisting of their lexical and semantic similarity information with L anchor passages.

2.3 Interaction and Aggregation

Following the prevalent sequence similarity learning paradigm in the field of natural language processing (Reimers and Gurevych, 2019; Gao et al., 2021b), we expect to measure the relevance of query and passage with their collaborative sequences in vector space. We obtain these vector representations by anchor-wise interaction and sequence aggregation in HybRank.

Anchor-wise Interaction The j-th elements $\boldsymbol{e}_{*j}$ in these collaborative sequences $\boldsymbol{E}_*$ indicate the similarities between retrieved passages and the j-th anchor passage. The importance of these anchors varies since they are picked with a single strategy. Specifically, an anchor is worthy of more consideration if showing strong correlation with a majority of retrieved passages, and vice versa.

To assess the quality of anchor passages, we conduct anchor-wise interaction. Concretely, for each position j, we collect the j-th similarity embedding $\boldsymbol{e}_{*j}$ from query sequence and every passage sequences and refine them with a Transformer encoder, denoted as

$$\begin{aligned} \boldsymbol{e}'_{qj}, &\boldsymbol{e}'_{1j}, \boldsymbol{e}'_{2j}, \ldots, \boldsymbol{e}'_{Nj} \\ &= \text{Trans}^{inter}(\boldsymbol{e}_{qj}; \boldsymbol{e}_{1j}; \boldsymbol{e}_{2j}; \ldots; \boldsymbol{e}_{Nj}), \end{aligned} \tag{5}$$

where $\boldsymbol{e}'_{*j} \in \mathbb{R}^D$. Position embeddings are added to $\boldsymbol{e}_{*j}$ according to its rank "$*$" for retaining the passage rank information. Subsequently, the similarity embedding sequences $\boldsymbol{E}_*$ are converted to $\boldsymbol{E}'_* = [\boldsymbol{e}'_{*1}, \boldsymbol{e}'_{*2}, \ldots, \boldsymbol{e}'_{*L}]$ and enhanced with the importance information of anchor passages.

Sequence Aggregation We encode these sequences into dense vectors by aggregating the enhanced similarity embeddings. To be specific, we prepend a [CLS] embedding to the collaborative sequence, feed the extended sequence into another Transformer encoder and use the output of [CLS] as the representation of p_i, formulated as

$$\boldsymbol{h}_{p_i} = \text{Trans}^{aggr}([\text{CLS}] \oplus \boldsymbol{E}'_{p_i})_{[\text{CLS}]}, \tag{6}$$

where $[\text{CLS}] \in \mathbb{R}^{1\times D}$, $\boldsymbol{E}'_{p_i} \in \mathbb{R}^{L\times D}$ and $\oplus$ denotes the concatenation operation. $\boldsymbol{h}_{p_i} \in \mathbb{R}^D$ is the vector representation of passage p_i. The query representation $\boldsymbol{h}_q \in \mathbb{R}^D$ is derived analogously.

Receptive Field and Complexity Interestingly, from another perspective, the anchor-wise interaction and sequence aggregation are equivalent to a column-wise and a row-wise attention applied on the matrix formulated by similarities of query, passages and anchors. Global receptive field is provided by these two axial-wise attentions (Ho et al., 2019). Consequently, similarity vector $\boldsymbol{x}_{ij}$ perceives with each other, and the vector representations of query and passages are aware of the collaborative information from others.

A more direct approach to obtain global receptive field is element-wise interaction. Concretely, we can feed the concatenation of all sequences $\boldsymbol{E}$ into a single Transformer encoder, and output representations for each passage and query via multiple separate [CLS] tokens. However, due to the self-attention operation in Transformer, the computational complexity of element-wise interaction achieves $O(N^2L^2)$. In contrast, our method reduce the complexity to $O(N^2L + NL^2)$, by decomposing the element-wise attention on the similarity matrix into axial-wise. Note that the complexity can be further reduced to $O(NL + NL)$ if leveraging linear Transformers (Katharopoulos et al., 2020; Wang et al., 2020) instead of vanilla Transformers.

2.4 Reranking and Training

Reranking Considering that query and passages have been converted into dense vectors encoded with collaborative information, we have several alternatives to judge the vector similarity as the relevance score between the query and passage. We use dot product in this work and thus the relevance score between query q and p_i is computed by

$$s_i = \boldsymbol{h}_q^\top \boldsymbol{h}_{p_i}. \tag{7}$$

Then passages are sorted in descending order of their relevance score s_i with query.

Training In order to assign high scores to relevant passages and low scores to irrelevant ones, HybRank needs to pull together the representation of relevant passages and query, while push the representation of irrelevant ones as apart from the query as possible. As there may exist more than one positive passage in the list, vanilla softmax loss fails to be directly applied to HybRank. We adopt the supervised contrastive loss (Khosla et al., 2020) to cope with multiple positives, which performs summation over positives outside the log function in softmax. The loss is formulated as

$$\mathcal{L}(q, \mathcal{P}) = -\frac{1}{|\mathcal{P}^+|} \sum_{p_i \in \mathcal{P}^+} \log \frac{\exp(s_i/\tau)}{\sum_{p_j \in \mathcal{P}} \exp(s_j/\tau)}, \tag{8}$$

where $|\mathcal{P}^+|$ is the number of positive passages in the retrieval list, and τ is a tunable temperature.

3 Experiments

3.1 Datasets

Natural Questions (Kwiatkowski et al., 2019) consists of real English questions from Google search engine with golden passages from English Wikipedia pages and answer span annotations. Following the settings from Karpukhin et al. (2020), we report the test set top-k accuracy (R@k), which evaluates the percentage of queries whose top-k retrieved passages contain the answers.

MS MARCO (Bajaj et al., 2018) includes English queries from Bing search logs and was originally designed for machine reading comprehension. Following previous works (Qu et al., 2021; Ren et al., 2021b), we evaluate the dev set R@k as well as Mean Reciprocal Rank (MRR), which means the average reciprocal of the first retrieved relevant passage rank.

TREC 2019/2020 (Craswell et al., 2020b,a) originate from TREC 2019/2020 Deep Learning (DL) Track. These two tracks provide additional Bing search queries and require to retrieve passages from the MS MARCO corpus. We use the official setting and evaluate the NDCG@10 of HybRank trained on MS MARCO with their test set.

3.2 Implementation Details

HybRank is a flexible plug-in reranker which can be applied on arbitrary passage lists including those that have already been reranked by other methods. Thus, we test HybRank against not only retrieval systems but also systems with other rerankers in it. We adopt dense retrievers which outperform sparse ones after elaborated pre-training (Chang et al., 2020; Gao and Callan, 2021, 2022) and fine-tuning (Sachan et al., 2021), as well as strong cross-encoder based rerankers, to initialize the passage list. We simply select all passages in the initial list as anchors. The impact of anchor passages will be discussed in Section 3.4. These methods are implemented using RocketQA toolkit[4] and Pyserini toolkit (Lin et al., 2021a) which is built on Lucene[5] and FAISS (Johnson et al., 2021).

The hyper-parameters in HybRank are as follows. The temperature t in the feature normalization is set to 100 and 10 for sparse and dense similarity, respectively. We randomly initialize a 2-layer Transformer encoder for Trans^{inter} and 1-layer for Trans^{aggr} using Huggingface Transformers (Wolf et al., 2020). The embedding dimension, MLP inner-layer dimension and number of heads are 64, 256 and 8, respectively. There are 0.22M parameters in total. The temperature τ in the loss function is 0.07. We adopt the Adam optimizer with an initial learning rate 1×10^{-3} with the warm-up ratio 0.1, followed by a cosine learning rate decay. We use gradient clipping of 2 and weight decay of 1×10^{-6}. We train the model for 100 epochs with batch size 32, which takes about 13 hours on Natural Questions and 4 days on MS MARCO. All experiments are conducted on a single NVIDIA RTX 3090 GPU.

3.3 Results

Table 1 and Table 2 summarize the performance of HybRank and baselines on the Natural Questions, MS MARCO and TREC 2019/2020 datasets. More detailed evaluation results are listed in Appendix B. Some of adopted retrieval baselines involve both sparse and dense similarity from different perspectives. DPR (Karpukhin et al., 2020) selects hard negative samples from passages returned by BM25; FiD-KD (Izacard and Grave, 2021) starts its iterative training with passages retrieved using BM25; TCT-ColBERT-v1 (Lin et al., 2020) proposes an

[4] https://github.com/PaddlePaddle/RocketQA.
[5] https://lucene.apache.org.

	Natural Questions Test		
	R@1	R@5	R@20
DPR-Multi + HybRank	45.82 → 51.99 (**+6.17**)	68.12 → 72.71 (**+4.59**)	80.31 → 83.24 (**+2.93**)
DPR-Single + HybRank	47.95 → 53.13 (**+5.18**)	69.39 → 73.05 (**+3.66**)	80.97 → 82.99 (**+2.02**)
FiD-KD + HybRank	50.36 → 52.85 (**+2.49**)	74.10 → 74.46 (**+0.36**)	84.27 → 84.49 (**+0.22**)
ANCE + HybRank	52.66 → 53.63 (**+0.97**)	72.66 → 73.57 (**+0.91**)	83.05 → 83.88 (**+0.83**)
RocketQA-retriever + HybRank	51.74 → 56.07 (**+4.33**)	74.02 → 77.04 (**+3.02**)	83.99 → 85.68 (**+1.69**)
RocketQA-reranker + HybRank	54.60 → 59.83 (**+5.23**)	76.59 → 78.73 (**+2.14**)	85.01 → 86.40 (**+1.39**)
RocketQAv2-retriever + HybRank	55.57 → 56.98 (**+1.41**)	75.98 → 76.65 (**+0.67**)	84.46 → 85.76 (**+1.30**)
RocketQAv2-reranker + HybRank	57.17 → 59.50 (**+2.33**)	75.98 → 78.34 (**+2.36**)	84.71 → 86.26 (**+1.55**)

Table 1: The reranking performance of HybRank on Natural Questions from a single run. We build HybRank upon DPR (Karpukhin et al., 2020), FiD-KD (Izacard and Grave, 2021), ANCE (Xiong et al., 2021), RocketQA (Qu et al., 2021) and RocketQAv2 (Ren et al., 2021b). The performance of these baselines and HybRank built upon them are on the left and right side of arrows, respectively. Improvements brought by HybRank are highlighted in bold.

	MS MARCO Dev	TREC 2019	TREC 2020
	MRR@10	NDCG@10	NDCG@10
DistilBERT-KD + HybRank	32.50 → 36.24 (**+3.74**)	69.23 → 72.55 (**+3.32**)	60.58 → 66.71 (**+6.13**)
ANCE + HybRank	33.01 → 36.44 (**+3.43**)	62.37 → 70.41 (**+8.04**)	60.00 → 63.70 (**+3.70**)
TCT-ColBERT-v1 + HybRank	33.49 → 36.23 (**+2.74**)	65.42 → 73.21 (**+7.79**)	61.03 → 66.91 (**+5.88**)
TAS-B + HybRank	34.44 → 36.38 (**+1.94**)	70.49 → 74.82 (**+4.33**)	63.89 → 66.53 (**+2.64**)
TCT-ColBERT-v2 + HybRank	35.85 → 37.55 (**+1.70**)	71.15 → 74.06 (**+2.91**)	64.32 → 66.35 (**+2.03**)
RocketQA-retriever + HybRank	35.77 → 36.97 (**+1.20**)	70.49 → 74.79 (**+4.30**)	63.74 → 67.25 (**+3.51**)
RocketQA-reranker + HybRank	40.51 → 40.98 (**+0.47**)	75.40 → 77.05 (**+1.65**)	67.66 → 69.85 (**+2.19**)
RocketQAv2-retriever + HybRank	37.28 → 38.74 (**+1.46**)	70.14 → 73.63 (**+3.49**)	63.04 → 67.87 (**+4.83**)
RocketQAv2-reranker + HybRank	41.15 → 41.40 (**+0.25**)	73.24 → 74.92 (**+1.68**)	69.47 → 70.71 (**+1.24**)

Table 2: The reranking performance of HybRank on MS MARCO and TREC 2019/2020 from a single run. We built HybRank upon DistilBERT-KD (Hofstätter et al., 2021a), ANCE (Xiong et al., 2021), TCT-ColBERT-v1 (Lin et al., 2020), TAS-B (Hofstätter et al., 2021b), TCT-ColBERT-v2 (Lin et al., 2021b), RocketQA (Qu et al., 2021) and RocketQAv2 (Ren et al., 2021b). The performance of these baselines and HybRank built upon them are on the left and right side of arrows, respectively. Improvements brought by HybRank are highlighted in bold.

alternative approximation for linear combination of dense and sparse retrieval; TCT-ColBERT-v2 (Lin et al., 2021b) further studies the dense-sparse hybrid in terms of quality, time and space. Besides, ANCE (Xiong et al., 2021) discovers new negatives via nearest neighbor search during model training; TAS-B (Hofstätter et al., 2021b) proposes balanced sampling strategies to compose informative training batches; DistilBERT-KD (Hofstätter et al., 2021a) leverages cross-architecture knowledge distillation for model-agnostic training.

From the results we can observe that HybRank shows a consistent improvements over upstream retrievers and even rerankers. In general, HybRank based on stronger baselines can produce better reranking results. For example, HybRank built upon the retriever of RocketQA outperforms HybRank built upon DPR-Multi on Natural Questions, and the same phenomenon can be observed on most retrievers. Additionally, HybRank built upon systems with reranker further improves the performance on both datasets. These results prove the advantage of reranking based on arbitrary off-the-shelf retrievers and even other reranked results, which distinguishes HybRank from other rerankers.

The most surprising aspect of these results is that, in spite of inferior reranking results, low-scoring retrievers gain more relative improvements from HybRank than high-scoring ones. This result may be explained by the fact that HybRank relies heav-

	R@1	R@5	R@10	R@20	R@50
retriever	45.82	68.12	75.24	80.30	84.57
r/d anchor	46.18	68.84	75.43	80.91	85.01
w/o *q*-*p*	47.12	69.17	75.54	80.47	85.07
w/o inter	49.92	69.61	76.32	81.02	84.99
w/o collab	50.78	**72.91**	**79.28**	83.10	85.79
HybRank	**51.99**	72.71	79.03	**83.24**	**85.93**

Table 3: The results of ablation study for collaborative features, anchor-wise interaction and anchor passages on the test set of Natural Questions.

list	feature	R@1	R@5	R@10	R@20	R@50
sparse	none	23.82	45.18	55.54	63.93	73.55
	sparse	30.50	50.39	59.00	67.26	75.24
	dense	47.01	64.68	**70.39**	**74.49**	**77.81**
	hybrid	**47.15**	**64.82**	69.78	74.32	77.65
dense	none	45.82	68.12	75.24	80.30	84.57
	dense	46.70	68.45	75.04	80.19	84.88
	sparse	50.89	71.86	78.98	83.16	85.90
	hybrid	**51.99**	**72.71**	**79.03**	**83.24**	**85.93**

Table 4: The results of ablation study for feature hybrid on the test set of Natural Questions.

ily on the complementary information provide by sparse similarity. Low-scoring retrievers receive relatively more valuable information from sparse similarity than high-scoring retrievers, and accordingly improve more performance over upstream retrievers. We will discuss more on sparse-dense hybrid in Section 3.4.

3.4 Analysis

In this section, we discuss the impact of core components of HybRank: the hybrid and collaborative features, the anchor-wise interaction and the number of anchor passages. All experiments are conducted on Natural Questions dataset with DPR-Multi retriever.

Collaborative Feature The main difference between HybRank and other works is, it leverages the collaborative information between retrieved passages. To verify the impact of passage collaboration on reranking, we omit the collaborative feature in "w/o collab" by substituting only query-passage similarities for collaborative sequences, *i.e.*, representing each passage as a one-token sequence according to its similarities with query. Besides, we exclude the query-passage similarity in "w/o *q*-*p*" by representing query via a learnable token rather than aggregated collaborative sequence. The results are presented in Table 3, where "retriever" denotes the assessment of initial passage list.

Table 3 indicates that "w/o collab" shows an appreciable gain over "retriever", demonstrating that query-passage similarity is an essential and indispensable feature for HybRank. The most remarkable phenomenon is, "w/o *q*-*p*" surpasses "retriever" by a large margin, despite the fact that "w/o *q*-*p*" is completely unaware of the query. Namely, HybRank has the ability to distinguish the positive even only with the collaborative information among passages. Furthermore, standing on the shoulder of query-passage similarity, HybRank achieves even better results than "w/o collab", which sufficiently substantiates the reranking capability of collaborative information.

Anchor-wise Interaction Apart from the collaborative sequence, anchor-wise interaction provides extra collaboration between sequences. We eliminate the Trans^{inter} and directly aggregate the linear projected collaborative sequence to study the effectiveness of anchor-wise interaction.

Table 3 shows that there is a noticeable drop of performance without anchor-wise interaction. The discrepancy could be attributed to the restricted receptive field. "w/o inter" individually encodes each collaborative sequences of query and passages into dense vectors without anchor-wise interaction. In this manner, the relevance of these sequences is evaluated only in vector space where sequence information are severely compressed and not expressive enough. In contrast, equipped with anchor-wise interaction, HybRank is capable of obtaining a global receptive field. Each element in these sequences captures the context of elements in all sequences, enabling more informative vector representation and fine-grained relevance estimation.

Feature Hybrid Despite the fact that the similarities of sparse and dense retriever reflect different aspect of linguistics, *i.e.*, lexical overlap and semantic relevance, both of them tend to have collaborative property. Hence, it is more natural and easier to mix sparse and dense retrieval from the perspective of collaboration. To illustrate the complementarity of sparse and dense features and the necessity of feature hybrid in HybRank, we separately validate the effect of the two individual features and their hybrid. The ablations are conducted not only on initial passage list retrieved by dense retriever, but also list retrieved by sparse retriever for integrity and comparison.

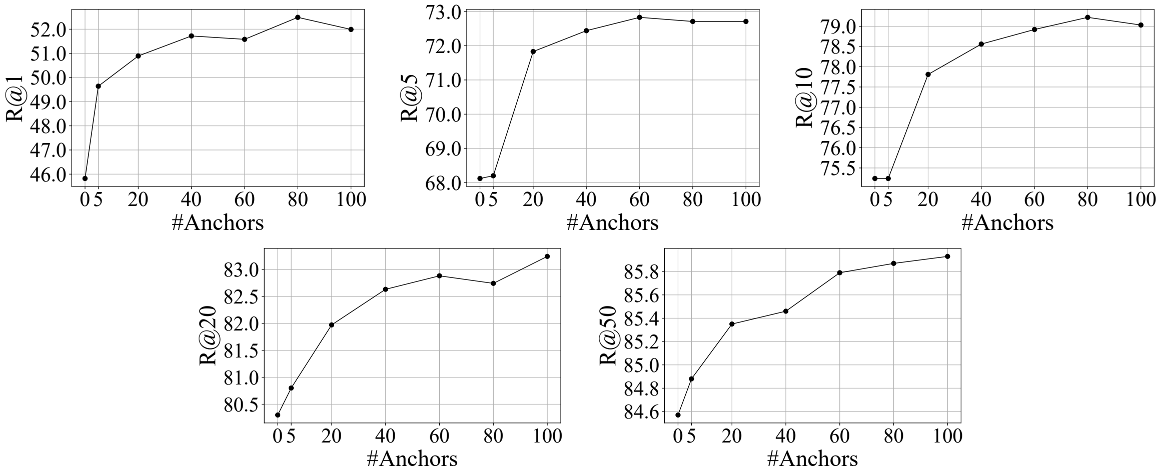


Figure 2: Impact study on the number of anchor passages. We conduct experiments on the test set of Natural Questions with anchor number $5, 20, 40, 60, 80, 100$. The metric of anchor number 0 denotes the assessment of initial retrieval list.

Identical trends can be observed from two settings of experiments in Table 4. The performance gains are limited when retrievers used for passage retrieval and similarity computation are same, but dramatically increase when they are different. Furthermore, additional slight improvements can be seen with the hybrid of the two features on both settings. These phenomena reveal that the main performance gains originate from the retriever different with that in retrieval stage, while the same type only plays an auxiliary role. Consequently, we draw the credible conclusion that different types of similarities provide additional complementary information over the initial passage list.

Moreover, regardless of feature used, HybRank achieves better results on passage list retrieved by dense retriever than sparse one, as more positives are contained in the dense retrieved list. This also corroborates the findings of Section 3.3 that superior initial passage list leads to better reranking results with HybRank.

Number of Anchor Passages We evaluate the performance of HybRank under different number of anchors to study its impact. What can be clearly seen in Figure 2 is a consistent growth of performance as the anchor number L increasing. The underlying philosophy is that, with more anchor passages the passage list can derive more agreement to facilitate the collaboration between passages and alleviate the distraction from noisy ones. The positive correlation between the performance and anchor number indicates the effect of collaborative information in the retrieval list.

Despite the consistent growth with anchor number, the rate of performance increase begins to slow down when the number of anchors is greater than 60. Anchor passages are used for deriving collaborative information, and thus with more diverse anchors we can obtain more distinctive collaborative features. As the anchor number approaches to 100, the diversity of passages levels off, leading to stable performance with larger anchor numbers.

As L increase to a very large number, the average relevance of anchors will degrade to a low level. A legitimate concern may be that poor quality anchor set would pollute the collaborative aspect. Due to the $O(L^2)$ computational complexity of sequence aggregation in HybRank, it is hard to directly perform experiments on large L. But we simulate the poor quality anchor set by randomly selecting anchor passages from corpus $\mathcal{C}$. “r/d anchor” in Table 3 indicates that random anchors slightly improves the performance but still lags far behind the relevant anchors, demonstrating the benefits of collaborative information and the predominance of the anchor quality.

Nevertheless, the selection of anchor passages is flexible. Ideally, more elaborated anchor passage selection, *e.g.*, clustering the passages from the corpus and selecting a fixed number of clustering centroids as anchors, would further enhance the performance and efficiency of HybRank. We leave the exploration of other anchor selecting strategy as a future work.

4 Related Work

4.1 Text Retrieval

Retrieval is the first stage of information retrieval which requires high recall to cover more relevant document in the retrieval list. Traditional sparse approaches like TF-IDF and BM25 (Robertson and Zaragoza, 2009) rely on lexical overlap between query and documents. Although having dominated the field of text retrieval for a long time, these sparse methods suffer from lexical gap (Berger et al., 2000), namely, the synonymy problem. To tackle this issue, earlier techniques (Nogueira et al., 2019; Dai and Callan, 2020) adopt neural networks to reinforce the sparse methods. Recently proposed dense retrieval approaches (Karpukhin et al., 2020; Xiong et al., 2021) directly encode the query and passages into dense vectors via dual-encoder, which capture semantic in text and enable low-latency search via highly optimized algorithms, *e.g.*, FAISS (Johnson et al., 2021).

These two types of methods are not mutually exclusive and one's weakness is the other's strength. Some researchers combine the sparse and dense methods by score ensemble, improved training or trade-off model between sparse and dense retriever. Karpukhin et al. (2020) samples hard negatives from sparse retriever for the training of dense retriever. Seo et al. (2019), Khattab and Zaharia (2020) and Santhanam et al. (2022) index terms or phrases instead of documents for more fine-grained similarity and higher efficiency. Lin et al. (2020) and Luan et al. (2021) explore the linear sparse-dense score combination and its alternatives. Gao et al. (2021a) and Yang et al. (2021) leverages the lexical matching or token-level interaction signals to train the dense retriever.

However, among these methods, score ensemble lacks sufficient interaction of sparse and dense methods, smaller units indexing sacrifices efficiency, and retraining one type of retriever with the help of the other discards its origin ranking capability. In contrast, our method can be applied to arbitrary passage list, incorporating the lexical and semantic properties of off-the-shelf retrievers and meanwhile ensuring the generality and flexibility.

4.2 Text Reranking

The second stage reranking is based on the results of retrieval system and aims to create a more fine-grained comparison within retrieval list. Typically, cross-encoder is utilized to capture the interactions between query and passage in token-level. Nogueira and Cho (2020) and Sun et al. (2021) adopt BERT (Devlin et al., 2019) to achieve token-level interactions with attention mechanism (Vaswani et al., 2017). To reduce the massive computation overhead (Reimers and Gurevych, 2019), Khattab and Zaharia (2020) and Gao et al. (2020) propose a lightweight interaction on dense representations from retrievers. While based on first-stage retrieval, these methods individually compute the relevance for each retrieved passage, omitting the extra information implied by the whole list and requiring multiple runs.

Several pseudo-relevance feedback approaches (He and Ounis, 2009; Zamani and Croft, 2016; Zamani et al., 2016) aim to refine the query model with the top-retrieved documents. Listwise context is also well explored in multi-stage recommendation systems (Liu et al., 2022), such as PRM (Pei et al., 2019), which regards each item as a token, learns the mutual influence between items using self-attention and reranks all items altogether. Different from prior approaches, we extract the collaborative feature from the retrieval list, represent the query and each passages as hybrid and collaborative sequences, and measure the relevance between query and passages using these sequences from the perspective of collaboration.

5 Conclusion

We introduce HybRank, a hybrid and collaborative passage reranking method. HybRank extracts the similarities between texts via off-the-shelf retrievers to form hybrid and collaborative sequences as the representations of query and passages. Efficient reranking is based on these sequences which incorporate the lexical and semantic properties of sparse and dense retrievers. Extensive experiments confirm the effectiveness of HybRank upon arbitrary passage list. Elaborated ablation studies investigate the impact of core components in HybRank. We hope our work could provide inspiration for researchers in the field of information retrieval, and steer more exploration on collaboration and correlation between texts.

Limitations

We evaluate HybRank on Natural Questions, MS MARCO and TREC 2019/2020 datasets, which focus on English Open-domain Question Answering. Although none of the components in HybRank are specifically designed for English, the verification of HybRank on other languages is limited. Otherwise, there are more general information retrieval tasks involving diversity or broader coverage in the returned results. Considering the possibility of lacking collaborative property, whether HybRank can generalize to these high-coverage retrieval tasks is still inconclusive.

As Transformer encoder architecture is adopted in the sequence interaction and aggregation, the computation cost would be unacceptable when the length of passage list or number of anchors is too large. This is also the reason why we only conduct experiments with anchor numbers no more than 100. Besides, HybRank only uses similarities computed by off-the-shelf retrievers as input features, and thus lacks sufficient interaction between raw inputs. The performance of HybRank may be limited by the capability of upstream retrievers. How to incorporate the interaction of raw inputs into HybRank while avoiding massive computation cost is still an open problem for further investigation.

Ethics Statement

This work focuses on improving the ranking results of passage retrieval systems. Retrieval is the fundamental component for many downstream tasks. However, it poses risks in terms of bias, misuse and misinformation due to the yet inaccurate results. Selection bias resulting from data collection, *e.g.*, lexical bias, may exist in the adopted datasets. Additionally, as the reranking approach in this work is built upon off-the-shelf retrievers, bias may ensue from upstream retrievers.

Acknowledgements

This work was supported by NSFC under Contract U20A20183 and 62021001.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A human generated machine reading comprehension dataset. *arXiv:1611.09268*.

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: Statistical approaches to answer-finding. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–199, Athens, Greece. ACM Press.

Wei-Cheng Chang, Felix X Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training Tasks for Embedding-based Large-scale Retrieval. In *Proceedings of the International Conference on Learning Representations*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2020a. Overview of the TREC 2020 deep learning track. In *Proceedings of the Twenty-Ninth Text REtrieval Conference*, volume 1266. National Institute of Standards and Technology.

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

Zhuyun Dai and Jamie Callan. 2020. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1533–1536. ACM.

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.

Luyu Gao and Jamie Callan. 2021. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853, Dublin, Ireland. Association for Computational Linguistics.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Modularized transfomer-based ranking framework. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4180–4190, Online. Association for Computational Linguistics.

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021a. Complementing lexical retrieval with semantic residual embedding. *arXiv:2004.13969*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70.

Ben He and Iadh Ounis. 2009. Finding good feedback documents. In *Proceedings of the ACM Conference on Information and Knowledge Management*, page 2011–2014. Association for Computing Machinery.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. 2019. Axial attention in multidimensional transformers. *arXiv:1912.12180*.

Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2021a. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv:2010.02666*.

Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021b. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–122. ACM.

Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2553–2561. ACM.

Gautier Izacard and Edouard Grave. 2021. Distilling Knowledge from Reader to Retriever for Question Answering. In *Proceedings of the International Conference on Learning Representations*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning*, pages 5156–5165. PMLR.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48. ACM.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021a. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2356–2362. ACM.

Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2020. Distilling Dense Representations for Ranking using Tightly-Coupled Teachers. *arXiv:2010.11386*.

Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021b. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 163–173, Online. Association for Computational Linguistics.

Weiwen Liu, Yunjia Xi, Jiarui Qin, Fei Sun, Bo Chen, Weinan Zhang, Rui Zhang, and Ruiming Tang. 2022. Neural Re-ranking in Multi-stage Recommender Systems: A Review. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 5512–5520. International Joint Conferences on Artificial Intelligence Organization.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage re-ranking with bert. *arXiv:1901.04085*.

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

Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, and Dan Pei. 2019. Personalized re-ranking for recommendation. In *Proceedings of the ACM Conference on Recommender Systems*, page 3–11. Association for Computing Machinery.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5835–5847, Online. Association for Computational Linguistics.

Razieh Rahimi, Azadeh Shakery, Javid Dadashkarimi, Mozhdeh Ariannezhad, Mostafa Dehghani, and Hossein Nasr Esfahani. 2016. Building a multi-domain comparable corpus using a learning to rank method. *Natural Language Engineering*, 22(4):627–653.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ruiyang Ren, Shangwen Lv, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021a. PAIR: Leveraging passage-centric similarity relation for improving dense passage retrieval. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2173–2183, Online. Association for Computational Linguistics.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021b. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Devendra Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6648–6662, Online. Association for Computational Linguistics.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States. Association for Computational Linguistics.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics.

Si Sun, Yingzhuo Qian, Zhenghao Liu, Chenyan Xiong, Kaitao Zhang, Jie Bao, Zhiyuan Liu, and Paul Bennett. 2021. Few-shot text ranking with meta adapted synthetic weak supervision. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5030–5043, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv:2006.04768*, abs/2006.04768.

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

Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*, page 1192–1199. Association for Computing Machinery.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the International Conference on Learning Representations*, page 16.

Yinfei Yang, Ning Jin, Kuo Lin, Mandy Guo, and Daniel Cer. 2021. Neural retrieval for question answering with cross-attention supervised data augmentation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 263–268, Online. Association for Computational Linguistics.

Hamed Zamani and W. Bruce Croft. 2016. Estimating embedding vectors for queries. In *Proceedings of the ACM International Conference on the Theory of Information Retrieval*, page 123–132. Association for Computing Machinery.

Hamed Zamani, Javid Dadashkarimi, Azadeh Shakery, and W. Bruce Croft. 2016. Pseudo-relevance feedback based on matrix factorization. In *Proceedings of the ACM International on Conference on Information and Knowledge Management*, pages 1483–1492. ACM.

Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, page 403–410. Association for Computing Machinery.

Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2022. Adversarial Retriever-Ranker for Dense Text Retrieval. In *Proceedings of the International Conference on Learning Representations*.

	Natural Questions	MS MARCO	TREC-DL 2019	TREC-DL 2020
# Passages in Corpus	20,015,324	8,841,823	-	-
Avg. Passage Length	100.0	56.58	-	-
Avg. Query Length	9.20	5.97	-	-
# Train Queries	58,880	502,939	-	-
# Dev Queries	6,515	6,980	-	-
# Test Queries	3,610	-	43	54
# Train Pairs	498,816	532,761	-	-
# Dev Pairs	55,121	7,437	-	-
# Test Pairs	-	-	9,260	11,386

Table 5: Statistics of Natural Questions, MS MARCO and TREC 2019/2020 datasets.

A Datasets Details

Dataset Natural Questions is under CC BY-SA 3.0 license. MS MARCO and TREC 2019/2020 are under CC BY-SA 4.0 license. The statistics of these datasets are presented in Table 5.

B Full Evaluation Results

We present the full evaluation results on Natural Questions, MS MARCO and TREC 2019/ 2020 in Table 6 and 7.

C Reranking Cases

We present reranking cases in Figure 3 and Figure 4. The first lines in these figures are the query sentence. We illustrate the distribution of positives in the passage list before and after reranking. Blue squares indicate positive passages while white squares stand for negative passages in the retrieval list. We only show top-50 out of 100 passages in these lists due to the space limitation. Following the positive distribution, we list several raw texts of reranked passages for the question.

Observed from the distribution visualization and rank changes of passages, the positive distributions shift toward the front of the lists as the quantitative analysis in Section 3.3. Ranks of many positive passages are raised by a large margin. Besides, it is apparent that positive passages tend to describe the same entities, events and relations as discussed in Section 1. Case 1 in Figure 3 involves “the king of England” while case 2 in Figure 4 is about “Where’s Waldo”.

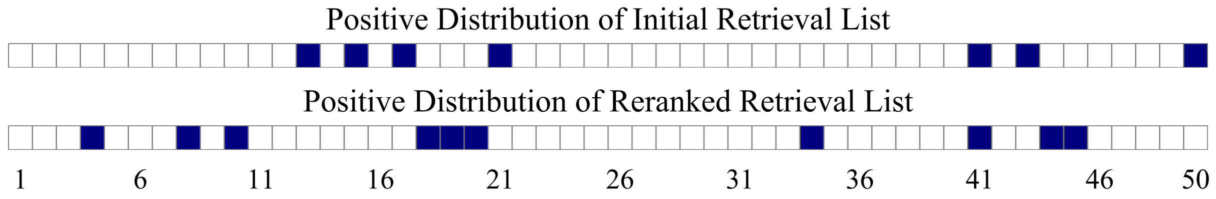
	Natural Questions Test				
	R@1	R@5	R@10	R@20	R@50
DPR-Multi	45.82	68.12	75.23	80.31	84.57
DPR-Multi + HybRank	51.99 (**+6.17**)	72.71 (**+4.59**)	79.03 (**+3.80**)	83.24 (**+2.93**)	85.93 (**+1.36**)
DPR-Single	47.95	69.39	75.93	80.97	84.90
DPR-Single + HybRank	53.13 (**+5.18**)	73.05 (**+3.66**)	78.84 (**+2.91**)	82.99 (**+2.02**)	85.93 (**+1.03**)
FiD-KD	50.36	74.10	79.78	84.27	87.90
FiD-KD + HybRank	52.85 (**+2.49**)	74.46 (**+0.36**)	80.50 (**+0.72**)	84.49 (**+0.22**)	88.06 (**+0.16**)
ANCE	52.66	72.66	78.70	83.05	86.29
ANCE + HybRank	53.63 (**+0.97**)	73.57 (**+0.91**)	79.28 (**+0.58**)	83.88 (**+0.83**)	87.12 (**+0.83**)
RocketQA-retriever	51.74	74.02	80.00	83.99	87.34
RocketQA-retriever + HybRank	56.07 (**+4.33**)	77.04 (**+3.02**)	82.30 (**+2.30**)	85.68 (**+1.69**)	88.17 (**+0.83**)
RocketQA-reranker	54.60	76.59	81.44	85.01	88.17
RocketQA-reranker + HybRank	59.83 (**+5.23**)	78.73 (**+2.14**)	82.83 (**+1.39**)	86.40 (**+1.39**)	88.42 (**+0.25**)
RocketQAv2-retriever	55.57	75.98	81.08	84.46	87.92
RocketQAv2-retriever + HybRank	56.98 (**+1.41**)	76.65 (**+0.67**)	81.94 (**+0.86**)	85.76 (**+1.30**)	88.61 (**+0.69**)
RocketQAv2-reranker	57.17	75.98	81.00	84.71	87.92
RocketQAv2-reranker + HybRank	59.50 (**+2.33**)	78.34 (**+2.36**)	83.24 (**+2.24**)	86.26 (**+1.55**)	88.75 (**+0.83**)

Table 6: The full evaluation of reranking results from HybRank on Natural Questions. We build HybRank upon DPR-Multi (Karpukhin et al., 2020), DPR-Single (Karpukhin et al., 2020), FiD-KD (Izacard and Grave, 2021), ANCE (Xiong et al., 2021), the retriever and reranker of RocketQA (Qu et al., 2021) and RocketQAv2 (Ren et al., 2021b). Improvements brought by HybRank are highlighted in bold.

	MS MARCO Dev			**TREC 2019**	**TREC 2020**
	MRR@10	R@10	R@50	NDCG@10	NDCG@10
DistilBERT-KD	32.50	58.77	79.24	69.23	60.58
DistilBERT-KD + HybRank	36.24 (**+3.74**)	64.40 (**+5.63**)	82.02 (**+2.78**)	72.55 (**+3.32**)	66.71 (**+6.13**)
ANCE	33.01	59.44	80.10	62.37	60.00
ANCE + HybRank	36.44 (**+3.43**)	64.63 (**+5.19**)	82.79 (**+2.69**)	70.41 (**+8.04**)	63.70 (**+3.70**)
TCT-ColBERT-v1	33.49	60.46	80.67	65.42	61.03
TCT-ColBERT-v1 + HybRank	36.23 (**+2.74**)	64.96 (**+4.50**)	83.44 (**+2.77**)	73.21 (**+7.79**)	66.91 (**+5.88**)
TAS-B	34.44	62.94	83.44	70.49	63.89
TAS-B + HybRank	36.38 (**+1.94**)	65.77 (**+2.83**)	84.71 (**+1.27**)	74.82 (**+4.33**)	66.53 (**+2.64**)
TCT-ColBERT-v2	35.85	63.64	83.31	71.15	64.32
TCT-ColBERT-v2 + HybRank	37.55 (**+1.70**)	66.39 (**+2.75**)	84.97 (**+1.66**)	74.06 (**+2.91**)	66.35 (**+2.03**)
RocketQA-retriever	35.77	64.01	83.41	70.49	63.74
RocketQA-retriever + HybRank	36.97 (**+1.20**)	65.67 (**+1.66**)	84.91 (**+1.50**)	74.79 (**+4.30**)	67.25 (**+3.51**)
RocketQA-reranker	40.51	69.81	86.46	75.40	67.66
RocketQA-reranker + HybRank	40.98 (**+0.47**)	70.40 (**+0.59**)	86.55 (**+0.09**)	77.05 (**+1.65**)	69.85 (**+2.19**)
RocketQAv2-retriever	37.28	65.72	84.04	70.14	63.04
RocketQAv2-retriever + HybRank	38.74 (**+1.46**)	68.12 (**+2.40**)	85.96 (**+1.92**)	73.63 (**+3.49**)	67.87 (**+4.83**)
RocketQAv2-reranker	41.15	69.99	86.55	73.24	69.47
RocketQAv2-reranker + HybRank	41.40 (**+0.25**)	70.37 (**+0.38**)	86.68 (**+0.13**)	74.92 (**+1.68**)	70.71 (**+1.24**)

Table 7: The full evaluation of reranking results from HybRank on MS MARCO and TREC 2019/2020. We built HybRank upon DistilBERT-KD (Hofstätter et al., 2021a), ANCE (Xiong et al., 2021), TCT-ColBERT-v1 (Lin et al., 2020), TAS-B (Hofstätter et al., 2021b), TCT-ColBERT-v2 (Lin et al., 2021b), the retriever and reranker of RocketQA (Qu et al., 2021) and RocketQAv2 (Ren et al., 2021b). Improvements brought by HybRank are highlighted in bold.

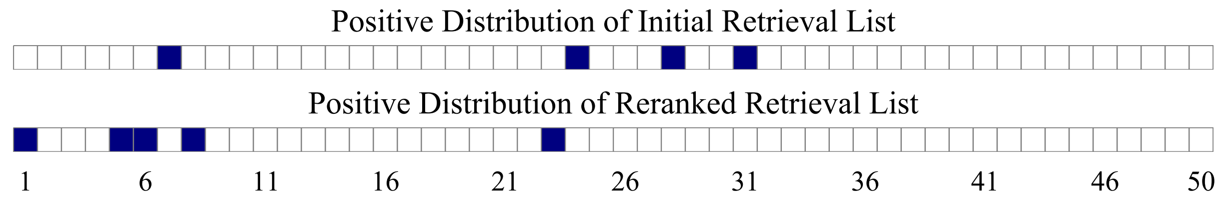
Query: ***Who was the king of England in 1756?***



Positive Passages	Rank Changes
George II of Great Britain. **George II** of Great Britain **George II** (George Augustus; ; 30 October / 9 November 1683 – 25 October 1760) was King of Great Britain and Ireland, Duke of Brunswick-Lüneburg (Hanover) and a prince-elector of the Holy Roman Empire from 11 June 1727 (O.S.) until his death in 1760. George was the last British monarch born outside Great Britain: he was born and brought up in northern Germany. His grandmother, Sophia of Hanover, became second in line to the British throne after about 50 Catholics higher in line were excluded by the Act of Settlement 1701 and the Acts of	15 → 4 (11 ↑)
George II of Great Britain. by his grandson, George III. For two centuries after **George II**'s death, history tended to view him with disdain, concentrating on his mistresses, short temper, and boorishness. Since then, most scholars have reassessed his legacy and conclude that he held and exercised influence in foreign policy and military appointments. George was born in the city of Hanover in Germany, and was the son of George Louis, Hereditary Prince of Brunswick-Lüneburg (later King George I of Great Britain), and his wife, Sophia Dorothea of Celle. His sister, Sophia Dorothea, was born when he was three years old. Both of George's parents	74 → 8 (66 ↑)
Monarchy of the United Kingdom. Britain was now in personal union. Power shifted towards George's ministers, especially to Sir Robert Walpole, who is often considered the first British prime minister, although the title was not then in use. The next monarch, **George II**, witnessed the final end of the Jacobite threat in 1746, when the Catholic Stuarts were completely defeated. During the long reign of his grandson, George III, Britain's American colonies were lost, the former colonies having formed the United States of America, but British influence elsewhere in the world continued to grow, and the United Kingdom of Great Britain and Ireland was created	17 → 10 (7 ↑)
Duke of Cumberland. of Wales, the eldest son and heir apparent of King **George II** and the father of King George III. He died without legitimate issue, when the dukedom again became extinct. This double dukedom, in the Peerage of Great Britain, was bestowed on Prince Ernest Augustus (1771–1851) (later King of Hanover), the fifth son and eighth child of King George III of the United Kingdom and King of Hanover. In 1919 it was suspended under the Titles Deprivation Act 1917 and has not been restored to its titular heir. A historic fixed bridge hand is known as the Duke of Cumberland	67 → 18 (49 ↑)
George II of Great Britain. the Hanoverian quarter differenced overall by a label of three points argent. The crest included the single arched coronet of his rank. As king, he used the royal arms as used by his father undifferenced. Caroline's ten pregnancies resulted in eight live births. One of their children died in infancy, and seven lived to adulthood. **George II** of Great Britain **George II** (George Augustus; ; 30 October / 9 November 1683 – 25 October 1760) was King of Great Britain and Ireland, Duke of Brunswick-Lüneburg (Hanover) and a prince-elector of the Holy Roman Empire from 11 June 1727 (O.S.) until	13 → 19 (6 ↓)

Figure 3: Reranking case 1. Blue squares indicate positive passages and white squares stand for negative passages. The titles of passages are bold and put in front of passages. These blue texts are the answers for the question.

Query: ***What kind of book is Where's Waldo?***



Positive Passages	Rank Changes
Where's Waldo? (video game). Where's Waldo? (video game) Where's Waldo? (Where's Wally? in the UK, Australia and South America) is a **puzzle** game developed by Bethesda Softworks and published in 1991 by THQ for the Nintendo Entertainment System. It was the first video game loosely based on Martin Handford's book of the "same name". Mostly similar to the books, players must help Waldo get to the moon by finding him in each of the eight levels in the game. The game was panned by critics, who criticized the game for its graphics, which made it more difficult to find Waldo in each of the	24 → 1 (23 ↑)
Where's Waldo? (video game). takes advantage of superior pointer-based motion controls to easily locate search targets and supports versus multiplayer. Where's Waldo? (video game) Where's Waldo? (Where's Wally? in the UK, Australia and South America) is a **puzzle** game developed by Bethesda Softworks and published in 1991 by THQ for the Nintendo Entertainment System. It was the first video game loosely based on Martin Handford's book of the "same name". Mostly similar to the books, players must help Waldo get to the moon by finding him in each of the eight levels in the game. The game was panned by critics, who criticized the	31 → 5 (26 ↑)
Activity book. and does not fall neatly into one of these more specific categories. Activity books are typically centred around a particular theme. This may be a generic theme, e.g. dinosaurs, or based on a toy, television show, book, or game. For example, the Where's Wally? series of books (known as Where's Waldo? in the USA) by Martin Handford consists of both **puzzle** books, wherein the reader must search for characters hidden in pictures, and activity books such as "", which include a wider range of games and activities as well as **puzzles**. In 2018, Nintendo announced its intention to publish activity	28 → 6 (22 ↑)
Where's Waldo? The Fantastic Journey (video game). Where's Waldo? The Fantastic Journey (video game) Where's Waldo? The Fantastic Journey is a video game published by Ubisoft and developed by Ludia based on the book of the same name. It is a **puzzle** adventure game released for the Nintendo DS, Wii, Microsoft Windows, and the iPhone, and is also a remake of "The Great Waldo Search", released in 1992. Like the other games in the series, the object of the game is to search for hidden characters and items within a time limit. Hints are awarded to the player through Woof, Waldo's pet dog. Woof alerts the players	7 → 8 (1 ↓)
Where's Wally? was turned into a Sunday newspaper comic/**puzzle**, distributed by King Features Syndicate. The comics were also released in book form in the US, using the regional name 'Waldo'. In the early 1990s Quaker Life Cereal in the US carried various "Where's Wally?" scenes on the back of the boxes along with collector's cards, toys and send-away prizes. This was shown in "The Simpsons" episode "Hello Gutter, Hello Fadder" where Homer shouts "WALDO, WHERE ARE YOU?!" after looking at the scene on the cereal box as Waldo walks by the kitchen window. On 1 April 2018 Google Maps added a minigame	61 → 23 (38 ↑)

Figure 4: Reranking case 2. Blue squares indicate positive passages and white squares stand for negative passages. The titles of passages are bold and put in front of passages. These blue texts are the answers for the question.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section Limitations

☑ A2. Did you discuss any potential risks of your work?
Section Ethics Statement

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section Abstraction and Section 1 Introduction

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

B ☑ Did you use or create scientific artifacts?

Section 4.1 Datasets and Section 4.2 Implementation Details

☑ B1. Did you cite the creators of artifacts you used?
Section 4.1 Datasets and Section 4.2 Implementation Details

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Appendix C Datasets Details

☑ 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)?
Section Ethics Statement

☑ 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?
Section Ethics Statement

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.1 Datasets

☑ 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.
Appendix C Datasets Details

C ☑ Did you run computational experiments?

Section 4 Experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4.2 Implementation Details

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?
Section 4.2 Implementation Details

☑ 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?
Section 4.3 Results

☑ 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.)?
Section 4.2 Implementation Details

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.?
No response.

□ 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)?
No response.

□ 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?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.