Transfer-Aware Data Selection for Domain Adaptation in Text Retrieval

Linzhu Yu[1,2], Huan Li[1,2], Ke Chen[1,2], Lidan Shou[1,2*]
[1]State Key Laboratory of Blockchain and Data Security, Zhejiang University
[2]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
{linzhu, lihuan.cs, chenk, should}@zju.edu.cn

Abstract

Domain adaptation is widely adopted in text retrieval scenarios where large labeled data is unavailable. To improve model adaptability, existing methods try to expand more source datasets. However, we found from experiments that indiscriminately using a large amount of source data from various text tasks does not guarantee improved adaptability, but may negatively impact ranking model performance. To tackle this issue, we propose Trait, a framework that can effectively improve model adaptability by selecting beneficial data without evaluating all source data. Specifically, we first divide multiple source datasets into data chunks of the same size as the minimum selection unit to form the whole selection space. Then we devise an iterative process that includes Bayesian optimization-based selection and transfer-aware chunk evaluation to incrementally select beneficial chunks. To reduce unnecessary evaluation costs, we also design backtracking and pruning actions to adjust the selection subspace. Extensive experimental results show that Trait not only achieves average state-of-the-art for few-shot on nine target datasets by evaluating only 4% of BERRI source data, but is also highly competitive for zero-shot compared with LLM-based rankers.

1 Introduction

In many realistic text retrieval tasks, acquiring large labeled training data is challenging due to the high cost of manual annotations (Xu et al., 2022; Yu et al., 2022; Fang et al., 2024). To tackle this issue, a technique called domain adaptation is widely adopted (Wang et al., 2022; Xin et al., 2022), which trains the model on large labeled source datasets from related tasks (e.g., MS MARCO (Nguyen et al., 2016) and NQ (Kwiatkowski et al., 2019)), and then adapts it to the target dataset.

Recently, some efforts (Su et al., 2023; Lin et al., 2023) attempt to facilitate domain adaptation by extending source datasets. For example, BERRI (Asai et al., 2023), a set of 37 source datasets covering various text tasks, is used to build adaptive retrieval models. Driven by the expansion of source datasets, a question arises: ***Can the source data be used indiscriminately to promote model adaptability?***

Indeed, recent studies (Yu et al., 2022; Fang et al., 2024) found that significant distribution differences between source and target data could diminish adaptability, but the effects of source data from various tasks remain to be investigated. As shown in Figure 1, we finetune the `monoT5-`*base* ranking model (Nogueira et al., 2020) on 640,000 BERRI examples, spanning 8,000 steps with 80 non-overlapping examples per step. During training, we periodically monitor its transfer performance on the BEIR target datasets (Thakur et al., 2021). In subplots (1)-(6) of Figure 1, the clear downward trends of model performance indicate that merely enriching source data does not ensure promoted adaptation effectiveness. Consistent with prior research (Yu et al., 2022), our findings confirm that some source data induce negative transfer (Rosenstein et al., 2005) to the model, as evidenced by notable declines in nDCG@10 between two adjacent monitored steps. Conversely, certain source data can facilitate adaptation (e.g., subplots (9)). Therefore, selecting source data that can effectively improve adaptability is crucial in the domain adaptation of text retrieval. In addition, we note that there are data selection methods for adaptation (Ruder and Plank, 2017; Liu et al., 2019; Qu et al., 2019), which usually require repeated evaluation of each data sample. However, when faced with source data containing millions of samples, for example BERRI, these methods are impractical due to excessive computational cost.

In this paper, we propose **Trait**, a **Tr**ansfer-**A**ware

*Corresponding author

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 17504–17519
November 4-9, 2025

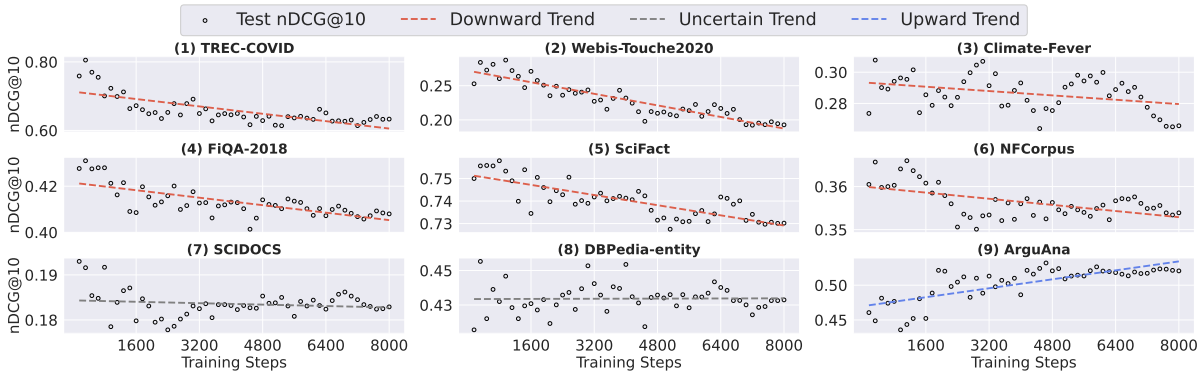


Figure 1: Transfer performance of the monoT5 ranker at different training steps on BEIR datasets. We finetune the ranker across 8,000 steps with 80 source examples per step. Periodly tested nDCG@10 values exhibit distinct trends: clear downward trends in (1)-(6); uncertain trends with slight declines in (7)-(8); only an upward trend in (9).

Incremental **I**terative data selection framework, which can effectively improve model adaptability by selecting beneficial data without evaluating all source data. The core idea of Trait is to divide source datasets into numerous chunks and select beneficial chunks through an iterative selection-evaluation process. Specifically, Trait features three main modules: (1) **Selection Space Structuring**. A notable limitation of existing methods is the need to evaluate each sample, even for million-level datasets. In this module, we will divide multiple source datasets into data chunks of the same size as the minimum selection unit to construct the selection subspace and map this space into a vector space. (2) **Chunk Selection**. This module further formalizes the chunk selection as a black-box optimization problem. In each iteration, we will employ Bayesian optimization to select the most valuable chunk in the selection subspace based on the existing evaluation records. (3) **Transfer-Aware Chunk Evaluation**. The third module then evaluates the selected chunk to obtain a new evaluation record that will also be added to the existing evaluation records for the next iteration of chunk selection. In addition, this module will also perform backtracking and pruning actions to adjust the selection subspace and reduce unnecessary evaluation costs. The whole selection-evaluation process iterates until the candidate subspace is empty or a maximum iteration limit is reached before scanning all data. Ultimately, our framework outputs the model, called Trait-ranker, that is finetuned on selected source chunks to rerank the documents from an initial BM25 retrieval.

To evaluate the effectiveness of Trait, we conducted extensive domain adaptation experiments on nine target datasets for both few-shot and zero-shot settings. Trait-ranker achieves state-of-the-art average performance for few-shot by evaluating only 4% of BERRI source data. For zero-shot, our Trait-ranker with only 737M parameters is still remarkably competitive with or superior to powerful ranking models based on large language models. Overall, our contributions are summarized as follows:

- We reveal the distinct effects of various source data on model adaptability to target retrieval tasks, and first propose the data selection problem for domain adaptation in text retrieval.
- We present Trait, a framework that can effectively improve adaptability by selecting beneficial data without evaluating all source data.
- We demonstrate the superiority of Trait for the few-shot and zero-shot settings through extensive experiments on the BEIR datasets.

2 Related Work

Domain Adaptation in Text Retrieval. There are various domain adaptation techniques in text retrieval that use source datasets as training data, including retrieval-oriented pretraining (Xu et al., 2022; Izacard et al., 2021), domain representation learning (Yu et al., 2022; Xin et al., 2022), source data expansion (Asai et al., 2023), model scale-up (Ni et al., 2022; Rosa et al., 2022), and knowledge distillation (Lin et al., 2023). Another direction is to generate numerous queries as pseudo-relevant labels by feeding the target corpus to LLMs (Bonifacio et al., 2022; Saad-Falcon et al., 2023; Fang et al., 2024), which then serve as train-

ing data. A recent trend is to use LLMs as zero-shot rankers (Zhuang et al., 2024a,b; Sun et al., 2023; Qin et al., 2024). These models take each query with documents retrieved by the BM25 retriever as input and return the reranking results. Although they require no further finetuning, the online computation costs are notoriously high.

Data Selection for Domain Adaptation. Traditional approaches rank source data by domain evaluation metrics (Cross-entropy (Moore and Lewis, 2010), Jensen-Shannon divergence (Plank and Van Noord, 2011; Remus, 2012; Ruder and Plank, 2017), Rényi divergence (Van Asch and Daelemans, 2010), sentiment graph-based metric (Wu and Huang, 2016)) to select top-k samples. However, these works are limited to designing metrics almost in isolation and focusing on a single task.

Recently, some studies (Liu et al., 2019; Qu et al., 2019; Dong and Xing, 2019) focus on reinforcement learning to select source data close to a given target domain for adaptation. They adopt an agent to make a selection decision for each example and then update the agent with feedback from the model trained with the selected examples. This procedure requires several scans of the entire source dataset for final selection. The rapid growth of source datasets collected for domain adaptation in text retrieval (now spanning millions of examples) renders conventional example-based evaluation methods prohibitively expensive for large-scale source data selection. We solve this challenge by not completely evaluating all data while removing the reliance on those domain evaluation metrics.

3 Problem Statement

Let $D = \{d_1, ..., d_M\}$ denote documents and $q \in Q$ a query. **Text Retrieval** requires a retrieval model to learn a ranking function $R : D \times Q \to \mathbb{R}$ that induces a ranked list of $L \in D$, such that for any $d_i, d_j \in L (i < j)$, there is:

$$R(q, d_i; \theta) \geq R(q, d_j; \theta), \quad (1)$$

where $R(q, d_i; \theta)$ parameterized by θ assigns a relevance score to document d for query q.

In many realistic retrieval scenarios, it is hardly possible to obtain large labeled training data. To solve it, traditional domain adaptation in text retrieval uses all available source datasets from related tasks for training (see Fig. 2 (a)): Given K source datasets $\{S_1, S_2, \ldots, S_K\}$ and a target retrieval dataset T, each S_k $(1 \leq k \leq K)$ contains query-document pairs (q, d) with relevance label l, the goal of traditional domain adaptation is to improve the adaptation of a model $R(\cdot, \cdot; \theta(S))$ to T, which is trained on the aggregated source pool $S = \bigcup_{k=1}^{K} S_k$. However, as shown in Figure 1, various source data can lead to different effects on model adaptability. Therefore, identifying a subset of source data beneficial to adaptability is the research problem of this paper (i.e., Fig. 2 (b)), named **Data Selection for Domain Adaptation in Text Retrieval**, which is formulated as follows:

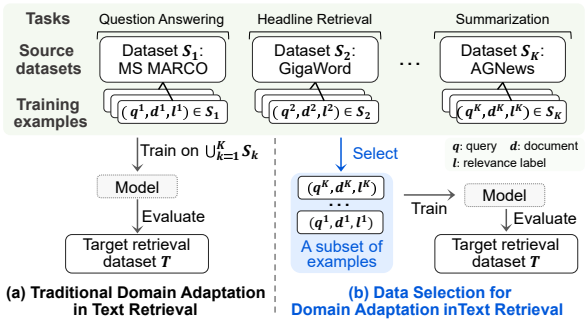


Figure 2: In (b), we offer a new scheme that selects a subset of examples helpful for model adaptation on the target dataset T, instead of using all source data in (a).

Problem 1 *Given a set of source data* $S = \bigcup_{k=1}^{K} S_k$*, a target retrieval dataset* T*, the goal of Data Selection for Domain Adaptation in Text Retrieval is to identify a subset* $\tilde{S} \subset S$ *with* $\left|\tilde{S}\right| \ll |S|$*, such that the model* $R(\cdot, \cdot; \theta(\tilde{S}))$ *trained on* $\tilde{S}$ *achieves enhanced performance on* T*.*

4 Methodology

We introduce Trait, a framework that efficiently selects beneficial source data to enhance model adaptation without computationally expensive evaluation of the entire source data. As depicted in Figure 3, it begins by initializing the selection subspace (Section 4.1). Then, the chunk selection module leverages Bayesian optimization to select a high-value data chunk from the subspace (Section 4.2.1). Thereafter, the transfer-aware module evaluates the selected chunk, which generates a new evaluation record and adjusts the subspace dynamically (Section 4.2.2). These updates are returned to facilitate the next selection. This selection-evaluation process iterates until the stopping conditions are met, outputting a Trait-ranker trained on a subset of selected chunks.

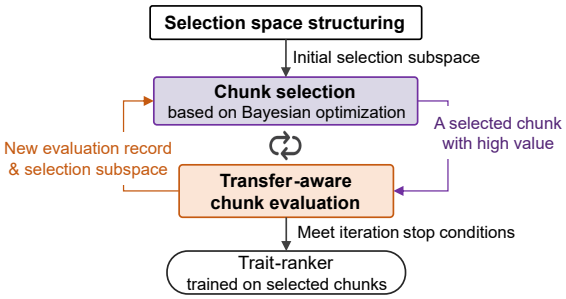
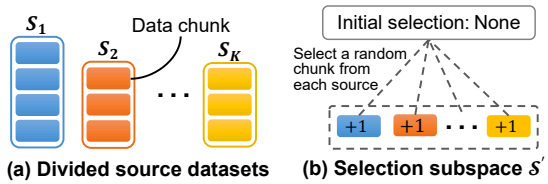


Figure 3: Overview of Trait



Chunk selection	Vector s
None (initial)	$(0, 0, \ldots, 0)$
None +	$(1, 0, \ldots, 0)$
None +	$(0, 1, 0, \ldots, 0)$
...	...
None +	$(0, \ldots, 0, 1)$

(c) Vectors represented candidates in $\mathcal{S}'$

Figure 4: Selection space structuring includes: (a) Divide source datasets; (b) Specify the selection subspace $\mathcal{S}'$; (c) Design vector to represent candidates in $\mathcal{S}'$.

4.1 Selection Space Structuring

Data Chunks. As shown in Figure 4 (a), each source dataset S_k is randomly divided into N_k non-overlapping data chunks, each uniformly sized at c, to form the whole selection space $\mathcal{S}$. To avoid repeated evaluation and reduce computational cost, each data chunk, as the minimum selection unit, can only be evaluated once. Experimental results in Section 5.4 confirm that our method is robust to variations in the random chunk division.

Selection Subspace. Driven by distribution characteristics: chunks from the same source dataset exhibit inherent distribution similarities, while cross-source chunks may exhibit significant divergence. This motivates us first to perform coarse-grained source selection to focus on a chunk (see Section 4.2.1) and then perform transfer-aware evaluation on this chunk to decide whether to keep it (see Section 4.2.2). To achieve coarse-grained source selection, we construct a selection subspace $\mathcal{S}'$ of size K via dynamic expansion. As shown in Figure 4 (b), starting with an empty selection set, we increase the current selected chunk collection by introducing a chunk randomly sampled from the candidate source S_k, with $k \in [1, K]$. This subspace construction mechanism ensures progressive refinement of chunk quality while maintaining distributional diversity during subsequent iterations.

Vector Representations. To distinguish each candidate in $\mathcal{S}'$, we formulate a simplified representation using a K dimensional vector $\mathbf{s} = [n_1, n_2, \ldots, n_K]$, where each dimension represents the number of chunks from a certain source dataset and each $n_k \in [0, N_k]$. Figure 4 (c) exemplifies this vector representation. These vectors play a role in the following chunk selection module, which has been demonstrated to be effective in Section 5.3.

4.2 Iterative Selection-Evaluation

Each iteration involves two steps: (1) the chunk selection module identifies a candidate $\mathbf{s} \in \mathcal{S}'$, followed by (2) the transfer-aware module evaluates the newly added chunk in $\mathbf{s}$. We will formally describe these candidates as *nodes* during iterations.

4.2.1 Chunk Selection

To efficiently find beneficial chunks within limited iterations, selecting the most valuable $\mathbf{s} \in \mathcal{S}'$ is significant but costly if it were traversed manually. Therefore, we frame this chunk selection process as a black-box function f that can be optimized. The input of f is $\mathbf{s}$ and the output is the adaptability score of the model finetuned on the chunks represented by $\mathbf{s}$, denoted by $f(\mathbf{s})$.

Bayesian Optimization has emerged as an efficient methodology to optimize black-box functions, proven promising in automatic hyperparameter tuning and neural architecture search (Kandasamy et al., 2018; Falkner et al., 2018). Despite this, we explore its application in chunk selection for domain adaptation in text retrieval for the first time. Formally, given our objective function $f : \mathcal{S} \to \mathbb{R}$, the goal is to find an input $\mathbf{s} \in \mathcal{S}'$ that globally maximizes f. The optimization relies on two concepts: a surrogate model $\hat{f}$ and an acquisition function $\mathcal{A} : \mathcal{S}' \to \mathbb{R}$. We employ the Gaussian process as $\hat{f}$ due to its computational convenience (Snoek et al., 2012) and the Expected Improvement function as $\mathcal{A}$ due to its efficiency (Jones et al., 1998).

Selection Process. Since the surrogate model $\hat{f}$ requires initial data observed from the objective f

to approximate the prior distribution of f, we observe each node $\mathbf{s}$ in the initial selection subspace before iterating. Specifically, we finetune a ranking model R using a chunk randomly sampled from the source S_k as described in Section 4.3. Then, a test interface is applied to score the model without updating it, outputting an adaptability score P_k^{init}. The interface requires only a small amount of data, which can be either query-document pairs generated by LLMs or a few examples from the target domain. Both cases are effective in experiments (see Section 5.1). The pair $\left(\mathbf{s}, P_k^{init}\right)$ is then added as an initial evaluation record in the set $\mathcal{O}$, thus initiating the optimization.

Firstly, the surrogate $\hat{f}$ analyzes all pairs in $\mathcal{O}$ to construct a probabilistic representation of the objective function f by predicting its posterior distribution. The predictions include both expected performance and confidence intervals of f in a given subspace $\mathcal{S}'$. Subsequently, the acquisition function $\mathcal{A}\left(\cdot; \hat{f}\right)$ assess all nodes $\mathbf{s} \in \mathcal{S}'$ by calculating their *utility* values. These values aim to balance exploration of uncertain nodes and exploitation of promising nodes. The candidate node $\mathbf{s}_i$ with the highest *utility* is selected in the current i-th iteration. The evaluation record of $\mathbf{s}_i$ will be added to $\mathcal{O}$ at the end of the i-th iteration, thus updating the $\hat{f}$ in the next iteration. This data-driven selection progressively helps to discover which sources may be worthy of further exploitation, thereby reducing the exploration of low-value ones.

4.2.2 Transfer-Aware Chunk Evaluation

New Evaluation Record. This module further evaluates the selected $\mathbf{s}_i$ through the feedback from the ranking model. The father node of $\mathbf{s}_i$, denoted by $\mathbf{s}_j$ $(j < i)$, carries an evaluation record $(\mathbf{s}_j, P(R(\cdot, \cdot; \theta(\mathbf{s}_j))))$, where $R(\cdot, \cdot; \theta(\mathbf{s}_j))$ denotes the model finetuned on the selected chunks in $\mathbf{s}_j$. Given that $\mathbf{s}_i$ extends $\mathbf{s}_j$ by adding one chunk, we further finetune $R(\cdot, \cdot; \theta(\mathbf{s}_j))$ exclusively on the newly added chunk for E epoch to obtain a new model $R(\cdot, \cdot; \theta(\mathbf{s}_i))$, thereby reducing computational overhead. Details of model finetuning are presented in Section 4.3. Similar to initial evaluation records, the adaptability score $P(R(\cdot, \cdot; \theta(\mathbf{s}_i)))$ is obtained through the test interface to form a new evaluation record $o = (\mathbf{s}_i, P(R(\cdot, \cdot; \theta(\mathbf{s}_i))))$ as a signal for adjusting the selection subspace. Besides, the new record o is added to the set $\mathcal{O}$ for the next optimization in the chunk selection module.

Selection Subspace Adjustment. As depicted in Figure 5, we design three actions to adjust the selection subspace based on the new evaluation record o. **(1) Forward.** If $P(R(\cdot, \cdot; \theta(\mathbf{s}_i))) > P(R(\cdot, \cdot; \theta(\mathbf{s}_j)))$, then we forward to $\mathbf{s}_i$ as the next expanded node for constructing selection subspace. **(2) Backtrack.** If $P(R(\cdot, \cdot; \theta(\mathbf{s}_i))) \leq P(R(\cdot, \cdot; \theta(\mathbf{s}_j)))$, then we backtrack to $\mathbf{s}_j$ as the next expanded node, explicitly excluding $\mathbf{s}_i$ from the generated subspace. **(3) Prune.** If $P(R(\cdot, \cdot; \theta(\mathbf{s}_i))) \leq P(R(\cdot, \cdot; \theta(\mathbf{s}_j)))$ and all the child nodes of $\mathbf{s}_j$ are evaluated, then we backtrack to the father node of $\mathbf{s}_j$. The subspace expanded from $\mathbf{s}_j$ is viewed as a **failed subspace** since adding a chunk to $\mathbf{s}_j$ cannot finetune a new model with a higher adaptability score. After identifying a failed subspace, this action prunes the subspace generated later. Specifically, nodes within this failed subspace are ranked based on their adaptability scores in descending order. Then, we mark the source datasets corresponding to the nodes sorted in the bottom α proportion and prune the nodes generated from these marked source datasets when expanding a new subspace later. When only β nodes remain in the subspace, no more pruning will be performed. The hyperparameter α controls the ratio of nodes to be pruned in the current subspace, while β defines the minimum ratio of nodes to be retained among all nodes. This iteration process continues until the **stop conditions** are met: either the maximum number of iterations M is reached or the new selection subspace can no longer be expanded. Trait then outputs the ranker with the best evaluation record.

4.3 Neural Retrieval Method

We adopt a multi-stage retrieval approach (Chen et al., 2017; Nogueira and Cho, 2019), first using efficient BM25 (Robertson et al., 1995) for document retrieval, then ranking these documents by a neural model. Following prior works (Bonifacio et al., 2022; Ni et al., 2022), we initialize our rank model with a `monoT5` backbone (Nogueira et al., 2020). The ranker $R(q, d; \theta)$ takes a formatted query-document pair as input and outputs its relevance score. The cross-entropy loss is employed during finetuning as follows:

$$\begin{aligned}\mathcal{L} = &-\sum\nolimits_{d^+ \in D^+} \log\left(R(q, d^+; \theta)\right) \\ &-\sum\nolimits_{d^- \in D^-} \log\left(1 - R(q, d^-; \theta)\right),\end{aligned} \tag{2}$$

where D^+ and D^- denote the sets of relevant and irrelevant documents, respectively.

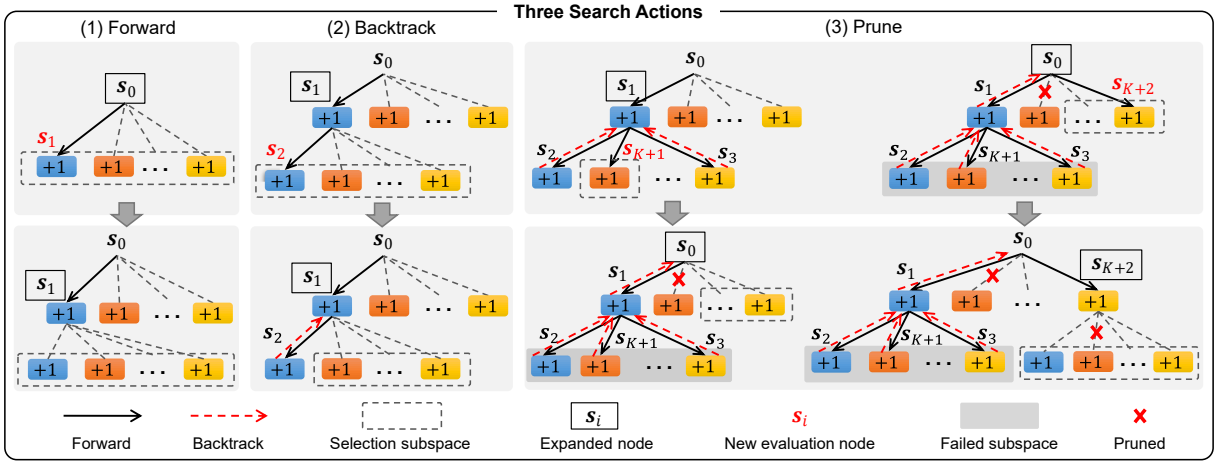


Figure 5: Transfer-aware chunk evaluation module is designed with three actions, including forward, backtrack, and prune, to adjust the selection subspace based on the new evaluation record in each iteration.

5 Experiments

5.1 Experimental Setups

Datasets and Metrics. We employ the BERRI collection (Asai et al., 2023) as source data and nine public BEIR datasets (Thakur et al., 2021) as target domains, reporting standard nDCG@10 scores (See Appendix A.1 for details). Following the research (Dai et al., 2022), we handle each target dataset with customized source data selection.

Baselines. Trait supports both zero-shot and few-shot adaptation through the data used in its test interface: (1) synthetic in-domain query-document pairs from InPars-v2 (Jeronymo et al., 2023) for zero-shot; (2) a few target pairs following the research (Dai et al., 2022) for few-shot. Trait-rankers rerank the top 1000 documents retrieved by BM25 for evaluation. We compare with three groups of baselines: (1) **zero-shot LLM-based rankers** directly rank documents retrieved by BM25 without additional finetuning; (2) **zero-shot finetuned rankers** are finetuned on source data without target examples; (3) **few-shot rankers** require a few target examples. (see Appendix A.2 for details.)

Implementation Details. We finetune Trait-ranker initialized from `monoT5-`*Large* (737M) by AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of 5e-5. Main hyperparameters comprise: data chunk size ($c = 640$), training epochs ($E = 4$), maximum iterations ($M = 150$), and pruning ratios ($\alpha = 25\%, \beta = 25\%$). Ablation studies of the two main modules in Trait (Section 5.3) and analysis of hyperparameters (Section 5.4) are conducted in the few-shot setting.

5.2 Results

Comparison with Baseline Methods. The main results are shown in Table 1. The following observations can be made: (1) As shown in rows 9-11, zero-shot Trait-ranker, learned from a selected subset of BERRI (under $60k$ instances), outperforms on average two larger TART-full rankers (1.5B) trained on random BERRI $640k$ examples. This highlights that selecting beneficial source data can enhance model adaptation more effectively than indiscriminately using source data. Details of selected data are listed in Appendix A.3 (2) Zero-shot Trait-ranker (737M) matches to or even exceeds existing LLM-based rankers (rows 1-4), despite being at least 6.3× smaller in model size. For instance, it outperforms SetwiseLLMRanker (11B) on 4 out of 5 tasks (e.g., DBPedia and Scifact), which relies on costly generation for document ranking. It is worth noting that due to the high computational cost, these LLM-based models only rerank the top 100 documents retrieved by BM25, limiting their adaptability to other target tasks. (3) Compared to rankers in rows 5-8, Trait-ranker (zero) shows mixed performance overall: outperforming RankT5-Enc and PROMPTAGATOR++ (zero), but lagging behind larger rankers in rows 6-7. Although RankLLaMA (7B) achieves the best average performance, it ranks the top 200 documents retrieved by dense retriever RepLLaMA (7B), which is costly for online inference. (4) Few-shot PROMPTAGATOR++ (rows 12) first generates and filters out a relevant query for each document by FLAN-137B with a few tar-

Table 1: nDCG@10 on nine target datasets. TRECC and CLI are short for TREC-COVID and Climate-FEVER. **AVG** is the average performance. The "Source" column shows labeled data for finetuning: 'n.a.' indicates no source data; † indicates synthetic in-domain data. Optimal results are reported for baseline rankers from their original papers. (*zero*) and (*few*) indicates zero-shot and few-shot. Results of LLM-based rankers not available are denoted −. ∗ indicates the data selection method of Plank et al. using the same source and selection size as Trait-Ranker. For each target dataset, the best results are marked in **bold** while the second-best results are underlined.

		TRECC	NFCorpus	FiQA	ArguAna	Touché	DBPedia	SCIDOCS	CLI	SciFact	**AVG**	Source
	BM25	65.6	32.5	23.6	31.5	36.7	31.3	15.8	21.3	66.5	36.1	n.a.
	Zero-shot LLM-based ranker											
(1)	$\text{RankGPT}_{3.5}$	76.7	35.6	-	-	36.2	44.5	-	-	70.4	-	n.a.
(2)	RG-S(0, 4)	80.5	**39.0**	-	-	27.6	41.9	-	-	75.2	-	n.a.
(3)	SetwiseLLMRanker	76.8	34.6	-	-	38.8	42.4	-	-	75.4	-	n.a.
(4)	PRP-Sliding-10	79.5	-	-	-	37.9	46.5	-	-	73.3	-	n.a.
	Zero-shot finetuned ranker											
(5)	RankT5-Enc	80.7	38.1	44.5	33.0	**44.0**	44.2	18.1	21.5	75.0	44.3	MS MARCO
(6)	RankLLaMA	**85.2**	30.3	46.5	56.0	40.1	48.3	17.8	28.0	73.2	47.3	MS MARCO
(7)	InPars-v2	84.6	38.5	**50.9**	36.9	29.1	**49.8**	**20.8**	32.3	77.4	46.7	$\text{BEIR}^{\dagger}$
(8)	PROMPTAGATOR++ (*zero*)	76.0	36.0	45.9	52.1	27.8	41.3	19.1	22.6	73.2	43.7	$\text{BEIR}^{\dagger}$
(9)	TART-full (FLAN-T5)	72.8	33.4	41.8	51.5	24.9	46.8	18.7	35.4	77.7	44.8	BERRI
(10)	TART-full (T0-3B)	71.7	34.0	42.3	49.8	31.2	45.1	17.5	30.0	75.8	44.1	BERRI
(11)	**Trait-Ranker (*zero*)**	79.7	36.9	44.0	45.5	26.5	47.1	19.6	31.2	76.3	45.2	BERRI ($< 60k$)
	Zero-shot data selection method											
(*)	Plank et al. (Plank and Van Noord, 2011)	73.1	28.9	42.2	43.9	24.5	38.0	19.2	25.1	73.6	40.9	BERRI ($< 60k$)
	Few-shot ranker											
(12)	PROMPTAGATOR++ (*few*)	76.2	37.0	49.4	**63.0**	38.1	43.4	20.1	24.1	73.1	47.2	$\text{BEIR}^{\dagger}$
(13)	**Trait-Ranker (*few*)**	84.9	37.5	46.6	59.5	33.5	47.8	20.3	**37.4**	**78.4**	**49.5**	BERRI ($< 50k$)

get examples as prompts, resulting in 1 million training data for each target task. Due to the learning from numerous synthetic data and the powerful reasoning ability of LLMs, PROMPTAGATOR++ performs well, especially on ArguAna. In contrast, our few-shot Trait-ranker learns fewer than $50k$ selected source examples without powerful LLMs. Trait-ranker surpasses PROMPTAGATOR++ on 6 of 9 tasks (e.g., TREC-COVID, DBPedia), and it achieves the state-of-the-art average performance across nine tasks in few-shot adaptation.

Comparison with Practical Selection Methods. To further demonstrate Trait's effectiveness in zero-shot settings, we compare it side-by-side with the domain metric-based data selection method by Plank et al. (Plank and Van Noord, 2011). To this end, we compute the Jensen-Shannon divergence for each BERRI query-document pair to measure source-target similarity. This domain similarity metric is computationally feasible. Thus, we subsequently select the top-t most similar source examples for training, where t is set to match Trait's selection size. As shown in Table 1, Trait consistently outperforms this traditional method across nine BEIR tasks, underscoring its clear advantage.

5.3 Ablation Studies

Effectiveness of Chunk Selection Module. We replace the chunk selection module based on Bayesian optimization (BO) in Trait with two simple strategies: (1) Random-S randomly chooses a node from the selection subspace; (2) Polling-S treats all nodes equally, employing a round-robin approach to sequentially select nodes from the subspace without prioritization. As illustrated in Figure 6, the BO-based module outperforms both strategies with a notable +4.0 gain in the average nDCG@10, which highlights its critical role in improving ranker adaptability. This BO-based module employs existing evaluation records to choose the most worthwhile node in each iteration, rather than blindly exploring indefinitely.

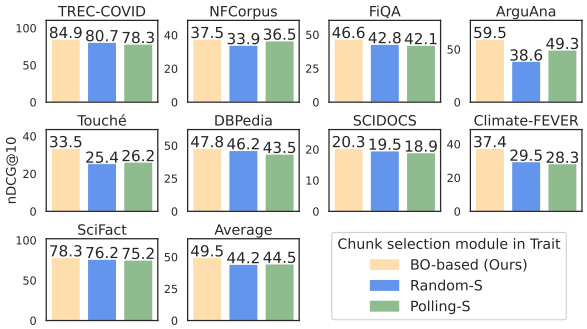


Figure 6: Comparison of the BO-based chunk selection module with two simple selection methods.

Impact of Search Actions. We conduct ablative analysis on the transfer-aware module's three actions for adjusting the selection subspace by evaluating two Trait variants: (a) None: adjust the subspace without actions until reaching the maximum iterations M; (b) W/O Pruning: subspace adjustment excluding pruning until meeting a failed

Table 2: Ablation results of three actions for adjusting the selection subspace: (1) Forward, (2) Backtrack, and (3) Prune. Two variants of Trait are included: (a) None: no actions taken; (b) W/O Pruning: abandon the pruning action.

Method	Action (1) F.	(2) B.	(3) P.	Iteration numbers	BEIR (nDCG@10) TRECC	NFCorpus	FiQA	ArguAna	Touché	DBPedia	SCIDOCS	CLI	SciFact
(a) None	-	-	-	150	84.0	37.3	45.4	49.8	32.6	47.5	19.7	36.2	76.7
(b) W/O Pruning	√	√	-	≤ 150	83.6	37.4	45.3	56.8	**33.5**	47.4	19.9	37.2	77.9
Trait	√	√	√	≤ 150	**84.9**	**37.5**	**46.6**	**59.5**	**33.5**	**47.8**	**20.3**	**37.4**	**78.4**

Table 3: Experimental verification of Trait's robustness to source data imbalance.

Trait-ranker	TRECC	NFCorpus	SCIDOCS	SciFact
imbalanced source	79.7	36.9	19.6	76.3
original source	79.7	36.9	19.6	76.3

Table 4: Analysis of maximum iteration number. The percentage in brackets refers to the ratio of the data evaluated in iterations to the total source data.

	Maximum iterations M			
BEIR (nDCG@10)	150 (**4%**)	200 (5.3%)	400 (10.6%)	1000 (26.5%)
TREC-COVID	85.7	86.0	86.3	**86.4**
NFCorpus	**37.3**	**37.3**	**37.3**	37.1
FiQA	**46.1**	45.3	45.9	45.9
ArguAna	53.0	55.9	**56.7**	55.3
Touché	33.5	33.7	**34.9**	34.2
DBPedia	47.9	47.6	**48.3**	47.8
SCIDOCS	20.2	**20.5**	**20.5**	20.3
Climate-FEVER	35.7	36.2	38.7	**39.5**
SciFact	**77.8**	76.8	76.9	**77.8**
Average	48.5	48.8	**49.5**	49.4

subspace or reaching M. As presented in Table 2, the results reveal that no actions or only removing the pruning action leads to a decrease in the performance of the final ranker. This is primarily because, in each iteration, the backtracking discards the chunk that is considered to be unable to improve model adaptability. In addition, when adjusting the selection subspace, the pruning neglects sources that may cause significant negative transfer to the model based on historical evaluation results.

Robustness to Source Data Imbalance.

To examine the impact of imbalanced source data on Trait, we create an intentional imbalance by replicating the CNN/DailyMail dataset (from BERRI) 10 times, making it 52.36% of the total source data. We then conduct zero-shot experiments on four BEIR target datasets. As shown in Table 3, Trait-rankers maintain performance comparable to the original setting. This robustness stems from Trait's selection mechanism (see Section 4.2.1), which prioritizes sources that are predicted to improve adaptation performance based on previous evaluation records, regardless of source data proportions. Specifically, it employs Bayesian optimization to select a promising source dataset, then randomly chooses a fixed-size chunk from it.

5.4 Hyperparameters

Impact of Maximum Iterations. We test the maximum iteration number $M \in [150, 1000]$, which affects Trait-ranker's adaptability by determining the upper limit of chunks we can select and evaluate. In this section, we keep the training epoch $E = 1$ to save computational cost. As shown in Table 4, further increasing M beyond 150 does not lead to significant improvement in model adaptability, especially on the FiQA and NFCorpus target datasets. Truly useful source data could be limited, making selection gradually difficult. Besides, once the model has acquired relevant knowledge from some source data, further learning of similar data yields little gains in adaptability.

Training Epoch. We analyze Trait-ranker's adaptability when varying the training epoch E. As illustrated in Figure 7, as the epoch E increases, the ranking performance in 9 target datasets either declines or improves. For instance, when E increases from 1 to 3, the performance of Trait-ranker improves significantly on datasets (4), (5), (7), and (8). These observations suggest that proper learning of source data can enhance domain adaptability. As E increases from 4 to 5, the nDCG@10 values decrease on 6 out of 9 datasets, indicating that excessive training on source data may impair transfer performance. This decline is possibly due to the ranker becoming overly aligned with the source domain's distribution, which differs to varying degrees from the distributions of the nine target domains. Overall, the average performance grows gradually and becomes stable when $E = 4$.

Data Chunk. To study the sensitivity of source data chunk division, we perform 20 experiments on the TREC-COVID task with different random seeds (keeping E=1 and M=100). Trait-rankers achieve stable nDCG@10 scores

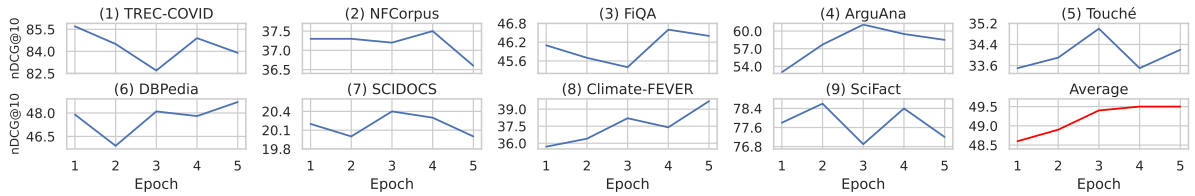


Figure 7: Effect of training epoch E. The last red line chart shows the average performance across these datasets.

Table 5: Comparison of experimental configurations of Ruder et al. and Trait.

Method	Task	Iterations	Evaluated Data/Iteration	Total Evaluated Data	Source Data	Evaluation Ratio	Repetition
Ruder et al.	Sentiment Analysis	$I_2 = 300$	$k = 1,600$	480,000	6,000	80×	Includes repeated evaluations
Trait	Text Ranking	$I_1 = 150$	$c = 640$	96,000	2,395,950	4%	No repeated evaluations

(mean=0.8447 ± 0.0044, 95% CI [0.8422, 0.8474]), confirming its low sensitivity to random chunk division. In addition, when testing different chunk sizes $c \in \{160, 320, 480, 640, 960\}$, we observed comparable nDCG@10 scores of $\{0.8417, 0.8424, 0.8417, 0.8472, 0.8425\}$, respectively, which indicates that reducing the chunk size did not improve adaptation effectiveness.

Pruning Ratio. During pruning, the hyperparameter α controls the pruning ratio each time, while β sets the minimum ratio of nodes to be retained among all nodes. We find that further adjusting these parameters minimally impacts Trait-ranker effectiveness (see Appendix A.4 for details).

5.5 Computational Cost Analysis

Our experiments used 8 NVIDIA A100 40GB GPUs with a batch size of 8 per GPU, consuming 37 GB of memory per GPU. Execution time varied by test data volume: Touché (1,766 instances) required about 14 hours, while ArguAna (20 instances) took about 7 hours. The entire process across all nine target tasks took about 118 hours.

Such costs motivated a formal efficiency analysis of Trait and previous selection methods. Trait performs incremental training on a selected chunk of size c and evaluates the ranker in each iteration. Given I_1 iterations, its total cost is $C_{Trait} = I_1 \times (C_{train}(c) + C_{test} + b)$, where $C_{train}(c)$ is the training cost on c examples, C_{test} is the testing cost, and b is the overhead of Bayesian optimization. This cost includes both data selection and model training. In contrast, many existing data selection methods, such as Ruder et al. (Ruder and Plank, 2017), rely on full-dataset reevaluation each iteration. Their approach involves scoring all source examples by domain metrics, selecting the top-t examples, training a model, and testing it. Given I_2 iterations, their total cost is $C_{Ruder} = I_2 \times (C_{train}(t) + C_{test} + b')$, where b' is the cost from updating similarity metric weights.

Theoretical Comparison. A key difference is that $t \gg c$: Ruder's method selects t examples per iteration, while Trait accumulates small chunks of size c incrementally. Thus, Trait avoids repeated evaluation of previously used data. Each iteration only processes new data, while Ruder's method repeatedly processes the entire selected set. Besides, the costs b and b' are negligible compared to training.

Empirical Comparison. Based on the experimental configuration, we list Table 5 to concretely analyze computational efficiency. Specifically, Trait evaluates only 4% of the source data, compared to 8000% evaluated by Ruder's method, substantially reducing redundant computation. It further shows that methods like Ruder's become prohibitively expensive to handle millions of source data in text ranking. Traditional data selection methods are computationally cheaper than Trait, but they are significantly less effective (see Section 5.1).

6 Conclusion

We show that indiscriminately using various source data may reduce model adaptability to the target retrieval task. We propose Trait, a data selection framework that accumulates beneficial source data by selectively evaluating data chunks in limited iterations, thus effectively improving model adaptability. Extensive experiments demonstrate the effectiveness of Trait for domain adaptation in both zero-shot and few-shot settings.

Limitations

We list the limitations of this work as follows:

1. Our research focuses on ranking models based on monoT5-*Large*, which is one of the best and computationally inexpensive open-source rankers. While we posit that our findings and the proposed Trait framework can be generalized to other types or sizes of ranking models (e.g., monoT5 or FLAN-T5 at diverse sizes), without evaluation on currently used monoT5-*Large*, our results may remain speculative.

2. In the zero-shot setting, we used the in-domain query-document pairs synthesized by GPT-J (6B) from the Inpars-V2 work to build the test interface. We did not specifically use more powerful large models, such as FLAN 137B used by PROMPTAGATOR++, to generate in-domain training data. Future work may generate in-domain data with better quality to build a test interface that enables zero-shot Trait-rankers to outperform rankers with high computational cost, such as zero-shot LLM-based rankers, in terms of adaptability.

Ethics Statement

All datasets and language models used in this work are publicly available. We tried our best to be fair and honest when analyzing the results of our experiments and to ensure that our work did not cause harm to any individual.

Acknowledgement

This work was supported by the Pioneer R&D Program of Zhejiang (No. 2024C01021) and the Zhejiang Province 'Leading Talent of Technological Innovation Program' (No. 2023R5214).

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, and 109 others. 2023. Palm 2 technical report. *Preprint*, arXiv:2305.10403.

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2023. Task-aware retrieval with instructions. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3650–3675.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392.

Ruey-Cheng Chen, Luke Gallagher, Roi Blanco, and J Shane Culpepper. 2017. Efficient cost-aware cascade ranking in multi-stage retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 445–454.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Zhuyun Dai, Vincent Y Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B Hall, and Ming-Wei Chang. 2022. Promptagator: Few-shot dense retrieval from 8 examples. *arXiv preprint arXiv:2209.11755*.

Nanqing Dong and Eric P Xing. 2019. Domain adaption in one-shot learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*, pages 573–588. Springer.

Stefan Falkner, Aaron Klein, and Frank Hutter. 2018. Bohb: Robust and efficient hyperparameter optimization at scale. In *ICML*, pages 1437–1446.

Yan Fang, Qingyao Ai, Jingtao Zhan, Yiqun Liu, Xiaolong Wu, and Zhao Cao. 2024. Combining multiple supervision for robust zero-shot dense retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17994–18002.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

Vitor Jeronymo, Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, Jakub Zavrel, and Rodrigo Nogueira. 2023. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *arXiv preprint arXiv:2301.01820*.

Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492.

Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing. 2018. Neural architecture search with bayesian optimisation and optimal transport. *NeurIPS*, 31.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton

Lee, and 1 others. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6385–6400.

Miaofeng Liu, Yan Song, Hongbin Zou, and T. Zhang. 2019. Reinforced training data selection for domain adaptation. In *Annual Meeting of the Association for Computational Linguistics*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Robert C Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 conference short papers*, pages 220–224.

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.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and 1 others. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718.

Barbara Plank and Gertjan Van Noord. 2011. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1566–1576.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, and 1 others. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518.

Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W Bruce Croft. 2019. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 699–707.

Robert Remus. 2012. Domain adaptation using domain similarity-and domain complexity-based instance selection for cross-domain sentiment analysis. In *2012 IEEE 12th international conference on data mining workshops*, pages 717–723. IEEE.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, and 1 others. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Guilherme Rosa, Luiz Bonifacio, Vitor Jeronymo, Hugo Abonizio, Marzieh Fadaee, Roberto Lotufo, and Rodrigo Nogueira. 2022. In defense of cross-encoders for zero-shot retrieval. *arXiv preprint arXiv:2212.06121*.

Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. 2005. To transfer or not to transfer. In *NIPS 2005 workshop on transfer learning*, volume 898, page 4.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (ACL).

Jon Saad-Falcon, Omar Khattab, Keshav Santhanam, Radu Florian, Martin Franz, Salim Roukos, Avirup Sil, Md Sultan, and Christopher Potts. 2023. Udapdr: Unsupervised domain adaptation via llm prompting and distillation of rerankers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11265–11279.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *NeurIPS*, pages 2960–2968.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937.

Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. 2023. Ul2: Unifying language learning paradigms. *Preprint*, arXiv:2205.05131.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A

heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Vincent Van Asch and Walter Daelemans. 2010. Using domain similarity for performance estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 31–36.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022. Gpl: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360.

Fangzhao Wu and Yongfeng Huang. 2016. Sentiment domain adaptation with multiple sources. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 301–310.

Ji Xin, Chenyan Xiong, Ashwin Srinivasan, Ankita Sharma, Damien Jose, and Paul Bennett. 2022. Zero-shot dense retrieval with momentum adversarial domain invariant representations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4008–4020.

Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. 2022. Laprador: Unsupervised pretrained dense retriever for zero-shot text retrieval. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3557–3569.

Yue Yu, Chenyan Xiong, Si Sun, Chao Zhang, and Arnold Overwijk. 2022. Coco-dr: Combating distribution shift in zero-shot dense retrieval with contrastive and distributionally robust learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1462–1479.

Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Bendersky. 2024a. Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 358–370.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023. Rankt5: Fine-tuning T5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 2308–2313. ACM.

Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2024b. A setwise approach for effective and highly efficient zero-shot ranking with large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 38–47.

A Appendix

A.1 Datasets

Source Datasets. BERRI, a collection of 37 text datasets from diverse domains with distinct expert-annotated instructions (Asai et al., 2023), contains the TART-full set for ranking models and the TART-dual set for retrieval models. We use the BERRI TART-full set with 3 million query-document pairs as the source data for selection. Since all instances in BERRI are randomly mixed, we classify each sample in its original dataset according to the instruction texts. As a result, 28 datasets are correctly classified, of which the MS MARCO dataset is removed because it was used in the pre-finetuning of `monoT5`, the initialization of our ranker. The remaining data are classified into an "unknown" dataset because their instruction texts do not identify the dataset to which they belong, and we asked the original authors but received no response. Finally, we conducted experiments on a total of 28 different source datasets, as shown in Table 6.

Target Datasets. For domain adaptation, we choose nine publicly available BEIR datasets (Thakur et al., 2021) as target domains. Following previous work (Dai et al., 2022; Asai et al., 2023), we exclude Natural Questions, MS MARCO, HotpotQA, FEVER, and CQADupStack from the evaluation for fair comparison since they are included either in the LLMs' pretraining datasets (such as FLAN-T5 using NQ in pretraining (Chung et al., 2024)) or in the source data of BERRI, leading to reduction in target datasets not caused by our method.

A.2 Baselines

In the main results, we consider three groups of strong ranking models as baselines, including zero-shot LLM-based rankers, zero-shot finetuned rankers, and few-shot rankers.

Zero-shot LLM-based rankers.

- RankGPT$_{3.5}$ (Sun et al., 2023) directly uses the gpt-3.5-turbo API for permutation generation to rank the top 100 documents from BM25.
- RG-S(0, 4) (Zhuang et al., 2024a) belongs to pointwise zero-shot LLM rankers. It uses FLAN PaLM2S (Anil et al., 2023) to generate fine-grained relevance labels for ranking the top 100 documents from BM25.
- SetwiseLLMRanker (Zhuang et al., 2024b) proposes a setwise prompt and utilizes the Flan-t5-xxl to generate ranking results of the top 100 documents retrieved by BM25.
- PRP-Sliding-10 (Qin et al., 2024), based on the Flan-UL2 (Tay et al., 2023) model with 20B parameters, ranks top 100 documents retrieved by BM25 in a pairwise manner.

Zero-shot finetuned rankers:

- This research (Zhuang et al., 2023) investigates the use of T5-based architecture for text ranking and has proposed RankT5-Enc, which is initialized from the encoder-only T5-based and finetuned on the MS MARCO dataset with the Softmax loss.
- RankLLaMA (Ma et al., 2024), a pointwise ranker based on LLaMA-2-7B, learns from MS MARCO with 500k training examples. It ranks the top 200 documents retrieved by the dense retriever RepLLaMA (7B).
- InPars-v2 (Jeronymo et al., 2023) uses the open-source GPT-J with 6B parameters to generate synthetic queries for each target document, and then applies monoT5-3B to filter out more relevant query-document pairs for finetung a ranking model.
- Promptagator++ (zero-shot) (Dai et al., 2022) applies the prompt without in-domain examples to prompt FLAN (137B) to generate synthetic relevant queries for each given target document, which are used to finetune the zero-shot Promptagator++ ranker.
- TART-full (T0-3B) (Asai et al., 2023) and TART-full (FLAN-T5) (Asai et al., 2023) are initialized from the T0-3B and FLAN-T5 encoder, respectively. These models, finetuned on about 640k BERRI instances, rank the top 100 documents retrieved by dense retriever TART-dual, also learned from BERRI.

Few-shot rankers:

- Promptagator++ (few-shot) (Dai et al., 2022) design prompts with a few query-document pairs from the target task. Then, these dataset-specific prompts are fed to FLAN (137B) to generate synthetic relevant queries for each given target document, which are used to finetune the Promptagator++. Besides, the few-

Table 6: BERRI source datasets statistics.

BERRI				
Index	Dataset	Domain	Task	Number
1	AGNews	news	summarization	118,587
2	Altlex	Wikipedia	sentence paraphrase	110,318
3	CNN Daily Mail	news	summarization	237,310
4	Coco captions	image captions	caption generations	98,466
5	ELI5	web	QA	121,713
6	FEVER	Wikipedia	fact verification	87,111
7	Gigaword	web	headline retrieval	21,498
8	HotpotQA	Wikipedia	QA	81,742
9	MedMCQA	medical	QA	22,661
10	medical simplification	medical	sentence simplification	6,431
11	NPR	news	summarization	118,602
12	NQ	Wikipedia	QA	22,568
13	OQA	Wikipedia	duplicated questions	164,741
14	PubMedQA	medical&science	QA	73,702
15	Qrecc	Wikipedia	conversational QA	26,067
16	Quora Duplicated Questions	community forum	duplicated questions	121,437
17	ReCoRD	news	QA	23,219
18	SciTLDR	science	summarization	2,853
19	SearchQA	web	QA	118,461
20	Sentence Compression	misc.	sentence compression	121,081
21	SQuAD	Wikipedia	QA	103,719
23	StackExchange (query→answer)	community forum	QA	121,158
22	StackExchange (title→title)	community forum	duplicated questions	128,020
24	TriviaQA	Wikipedia	QA	31,232
25	WikiHow	community forum	QA	118,367
26	WOW	Wikipedia	knowledge-grounded dialogue	94,737
27	XSUM	news	summarization	778
28	nan	misc.	misc.	99,371

Table 7: Statistics of the selected BERRI source data for each target task in the few-shot setting.

	Target Task								
BERRI	TREC-COVID	NFCorpus	FiQA	ArguAna	Touché	DBPedia	SCIDOCS	Climate-FEVER	SciFact
AGNews	0	0	1280	640	0	0	0	0	0
Altlex	0	0	0	0	0	0	0	0	0
CNN Daily Mail	640	640	0	0	0	0	640	0	640
Coco captions	640	0	0	0	0	0	640	0	640
ELI5	0	0	0	0	0	0	0	0	0
FEVER	640	640	0	0	0	0	0	0	0
Gigaword	0	0	0	0	0	1920	0	0	0
HotpotQA	0	0	0	1920	0	640	1280	0	0
MedMCQA	0	0	0	0	0	0	0	0	0
medical simplification	640	0	0	2560	0	0	0	0	0
NPR	0	1280	0	0	0	0	0	0	0
NQ	0	0	0	0	0	0	0	0	640
OQA	0	0	0	1920	0	0	0	0	640
PubMedQA	640	0	0	1280	0	640	0	0	0
Qrecc	0	0	0	0	0	0	640	0	640
Quora Duplicated Questions	0	0	0	640	0	1280	0	640	0
ReCoRD	0	0	0	0	0	0	0	0	640
SciTLDR	0	0	0	1280	0	0	0	0	0
SearchQA	0	0	0	640	0	0	0	0	0
Sentence Compression	0	0	0	0	0	0	0	1280	0
SQuAD	0	0	0	0	0	0	0	0	0
StackExchange (query→answer)	0	0	640	1280	1920	0	0	0	640
StackExchange (title→title)	0	1280	640	0	0	640	0	0	640
TriviaQA	640	0	0	0	0	0	0	640	640
WikiHow	640	0	0	0	0	0	0	0	0
WOW	0	0	0	2560	0	0	0	1280	0
XSUM	0	0	0	0	0	0	1280	0	640
nan	1280	0	0	0	0	0	0	0	0
Total	5760	3840	2560	14720	1920	5120	4480	3840	6400

shot Promptagator++ ranks the top 200 documents retrieved by the Promptagator retriever that is developed in the same few-shot setting.

A.3 Selected BERRI training examples

As shown in Table 8, we show the number of instances of each BERRI dataset selected for adaptation to each target task in the few-shot setting.

First, we find that the selection of source datasets varies widely. For example, the CNN Daily Mail source dataset contributes 640 instances to the TREC-COVID, NFCorpus, SCIDOCS, and SciFact target tasks, while being unused in other tasks. In contrast, 4 of the 28 source datasets (e.g., Altex and EL15) have zero utilization across all target tasks, suggesting that they are inherently incompatible with the target domain. Secondly, we observe that the distribution of the amount of data selected for target tasks is highly variable and concentrated on individual target tasks. For example, 61.8% of the instances were selected for only three tasks: ArguAna (14,720), SciFact (6,400), and DBPedia (5,120). This skewed distribution suggests that specific tasks, such as argument analysis containing miscellaneous domains, require more cross-domain knowledge transfer than other tasks. Overall, our results empirically validate our finding that prioritizing beneficial source data over piling up large amounts of source data can improve the effectiveness of domain adaptation.

A.4 Analysis of the pruning action

We perform experiments on the TREC-COVID target task with different settings of α and β (keeping $E = 1$ and $M = 150$). As shown in Table 8, the basis configuration with $\alpha = 25\%$ (the pruning ratio) and $\beta = 25\%$ (the minimum retention ratio) achieves the highest nDCG@10 score of 85.74. However, further adjusting these hyperparameters has no beneficial effect on the performance. In row 2, increasing β to 50%, which means keeping more nodes in the selection subspace, lowers the score to 84.53. In row 3, raising α to 50% to prune more nodes each time results in the same degraded score (84.53). Simultaneously setting both $\alpha = 50\%$ and $\beta = 50\%$ yields 84.98 (in row 4), still underperforming the basis setting in row 1.

Table 8: Analysis of the pruning action with varying α and β on TREC-COVID target task.

	Pruning ratio α	Minimum retention ratio β	nDCG@10
(1)	25%	25%	85.74
(2)	25%	50%	84.53
(3)	50%	25%	84.53
(4)	50%	50%	84.98