Evaluating Extrapolation Performance of Dense Retrieval

Jingtao Zhan¹, Xiaohui Xie¹, Jiaxin Mao², Yiqun Liu¹*, Min Zhang¹, Shaoping Ma¹
1 Department of Computer Science and Technology, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China
2 Beijing Key Laboratory of Big Data Management and Analysis Methods, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing 100872, China
jingtaozhan@gmail.com, yiqunliu@tsinghua.edu.cn

ABSTRACT
A retrieval model should not only interpolate the training data but also extrapolate well to the queries that are rather different from the training data. While dense retrieval (DR) models have been demonstrated to achieve better retrieval performance than the traditional term-based retrieval models, we still know little about whether they can extrapolate. To shed light on the research question, we investigate how DR models perform in both the interpolation and extrapolation regimes. We first investigate the distribution of training and test data on popular retrieval benchmarks and identify a considerable overlap in query entities, query intent, and relevance labels. This finding implies that the performance on these test sets is biased towards interpolation and cannot accurately reflect the extrapolation capacity. Therefore, to evaluate the extrapolation performance of DR models, we propose two resampling strategies for existing retrieval benchmarks and comprehensively investigate how DR models perform. Results show that DR models may interpolate as well as complex interaction-based models (e.g., BERT and ColBERT) but extrapolate substantially worse. Among various DR training strategies, text-encoding pretraining and target-domain pretraining are particularly effective for improving the extrapolation capacity. Finally, we compare the extrapolation capacity with domain transfer ability. Despite its simplicity and ease of use, the extrapolation performance can reflect the domain transfer ability in some domains of the BEIR dataset, further highlighting the feasibility of our approaches in evaluating the generalizability of DR models.

1 INTRODUCTION
Ranking is essential for many IR-related tasks, such as Web search and open domain question answering [19, 25]. Recently, there has been a surge of research interest in applying Dense Retrieval (DR) for ranking [18, 21, 25, 30, 35, 52]. DR is trained to encode queries and documents into low-dimension embeddings [29]. It leverages efficient vector search algorithms during online serving [24, 48, 51, 53]. On standard retrieval benchmarks such as MS MARCO [1, 9, 11], DR has outperformed traditional term-based retrieval methods like BM25 [36] by a large margin and set test-set performance to new heights [17, 25, 35, 53], making it very appealing for practical use.

However, an overall evaluation metric score on the test set does not show the full picture of DR performance, especially its generalization ability. Ranking models like DR should not only interpolate the training data for handling similar test queries, but also extrapolate to novel test queries that are distinct from the training queries. The latter capacity, i.e., extrapolation, is arguably essential for DR because queries are arbitrary in ad hoc search. Despite the impressive test-set performance scores of DR, we know relatively little about its extrapolation performance. Recently, Thakur et al. [39] construct the BEIR dataset to evaluate the zero-shot out-of-domain performance of retrieval models. Although such domain adaption performance can, to some extent, reflect the extrapolation ability, both BEIR authors [39] and other researchers [46] have pointed out that the BEIR dataset has strong lexical annotation bias. It may substantially underestimate the DR performance and lead to unjustified comparisons.

To better interpret the empirical results (i.e., the evaluation metrics on the test set) reported by previous studies [16, 21, 30, 47, 52], we investigate the distribution of training and test data on TREC Deep Learning Track [9, 11] and MS MARCO dataset [1], both of which are popular retrieval benchmarks [10]. We find a substantial overlap in query entities, query intent, and relevance labels. For example, manual annotation shows a majority of test queries are similar to or even duplicate training queries in terms of intent. Therefore, the existing test-set scores largely reflect the interpolation ability of DR. Without the knowledge of its extrapolation ability, it is risky to apply DR in practical use because real-world ranking systems like Web search engines must be able to properly process new queries that are constantly emerging.

To shed light on this research question, we propose two resampling strategies to evaluate the interpolation and extrapolation performance based on the standard retrieval benchmarks¹. The two strategies are designed for small and large test query sets, respectively. For small test sets, we propose to ReSample Training Queries (ReSTrain), which simply removes the queries that are similar to the test queries from the training set to evaluate the extrapolation performance. For large test sets, we propose to ReSample Training and Test Queries (ReSTTest). We first cluster both the training and test queries into k buckets. When evaluating the extrapolation performance, we use a sampling strategy similar to the k-fold cross-validation to ensure the training and test queries are not from the same bucket. The two proposed methods to comprehensively study the generalization ability of several popular DR methods. First, we study the influence of model architectures by comparing DR models with interaction-based ranking models (BERT reranker [34] and ColBERT [26]) and learned sparse retrieval methods (SPLADE [15]).

*Corresponding author

¹Code and data are released at https://github.com/jingtaozhan/extrapolate-eval
Second, we investigate the efficacy of some popular DR training methods, including hard negative mining [47, 52], distillation [21, 30], and pretraining [16, 17]. In summary, we investigate the following two research questions.

- **RQ1**: How do dense retrieval models extrapolate when compared with interaction-based deep neural ranking models and learned sparse retrieval methods?
- **RQ2**: How do different training strategies of dense retrieval models affect the extrapolation performance?

Extensive experimental comparisons are performed to address these two questions. For RQ1, results suggest that interaction-based models extrapolate well, whereas DR and sparse retrieval suffer from severe effectiveness degeneration when transitioning from interpolation to extrapolation. For RQ2, we observe that hard negative mining and distillation hardly mitigate the performance gap between the two evaluation regimes. Instead, the gap is tightly related to the pretraining technique. Text-encoding pretraining and target-domain pretraining are particularly effective for improving extrapolation performance.

Finally, we connect our extrapolation evaluation with existing domain-transfer studies [23, 39, 46]. Domain transfer ability means how well a trained model performs in out-of-domain scenarios, which is particularly meaningful for low-resource domains with little supervision data. Such ability is also a reflection of generalization ability. Since it is tightly related to our extrapolation evaluation, we further investigate the relationship between the interpolation/extrapolation performance and domain transfer ability and try to answer RQ3:

- **RQ3**: How does the interpolation/extrapolation performance reflect domain transfer ability?

By measuring the correlations between the interpolation/extrapolation performance and the out-of-domain performance, we find that the extrapolation performance is a better indicator of the domain transfer ability than interpolation performance. This result indicates that we can test the domain transfer ability on the data-rich benchmark dataset by using our extrapolation evaluation. In this way, we can save the cost of annotating a new out-of-domain retrieval dataset.

The rest of the paper is organized as follows: We first recap related work in the next section. Then we show that the benchmarks are biased to interpolation in Sections 3 and 4. We propose extrapolation evaluation in Section 5 and use it to re-evaluate various ranking models in Section 6. Finally, we show extrapolation performance can potentially reflect domain transfer ability in Section 7. Section 8 concludes the paper.

## 2 RELATED WORK

In this section, we recap related work in dense retrieval and previous studies that investigate its generalization ability.

Compared with conventional lexical retrieval methods, DR embeds text with Transformer-based models [12, 31, 41] and retrieves items in the latent space [17, 25, 30, 47, 52]. It achieves remarkable ranking performance on popular retrieval benchmarks [1, 9, 11] and has been demonstrated to be at least as efficient as lexical methods with the help of approximate nearest neighbor search algorithms [48, 51, 53]. There are mainly three directions to improve the DR ranking effectiveness. First, Xiong et al. [47] and Zhan et al. [52] highlight the importance of using hard negatives in training DR models and investigate how to mine them during training. Second, Hofstätter et al. [21] and Lin et al. [30] explore using knowledge distillation methods to train DR models with expressive yet slow interaction-based models like BERT rerankers [34] and ColBERT [26]. Finally, Gao and Callan [16] and Lu et al. [32] study pretraining techniques tailored for DR. The three directions complement each other and can be combined in practice.

Recently, several researchers challenge the generalization ability of DR. Sciavolino et al. [37] observe poor DR performance on entity-rich questions. Concurrently, Thakur et al. [39] introduce the BEIR dataset to test the zero-shot out-of-domain ranking performance. They find DR substantially underperforms traditional methods like BM25 [36]. Nevertheless, the authors also note that the annotation has lexical bias and that the conclusion may not hold if more items are annotated.

Different from previous studies that evaluate how DR models generalize on a held-out test set or an out-of-domain dataset, we try to take a closer look at their generalization performance by analyzing their behavior in the interpolation and extrapolation regimes. Interpolation and extrapolation are two fundamental concepts used for investigating generalization performances in machine learning. While there are different definitions for the two concepts [2–4], it is commonly agreed that extrapolation is a more intricate task and that as an algorithm transitions from interpolation to extrapolation, its performance tends to decrease. In this paper, we define them based on the similarity between training and test data, which closely relates to the definitions of Barnard and Wessels [3]. They define interpolation as the case where the training data is fully representative of the test data and define extrapolation as the case where test data differs distinctly from the training data. Based on the two definitions, they investigate the efficacy of multilayer perceptrons (MLP) in the two scenarios and find MLP is unsuitable for applications that require extrapolation. In ad hoc search, a query issued by the user could be rather different from any queries in the training data. Therefore, an effective retrieval model should be able to extrapolate well on these novel queries.

## 3 BENCHMARKS FOR DR MODELS

MS MARCO dataset [1] and TREC Deep Learning Tracks (TREC DL) [9, 11] are widely-adopted ad hoc retrieval benchmarks for training and evaluating DR models and other neural IR models [10]. They have attracted considerable attention from the neural IR community. Lin et al. [29] even point out that the rapid progress in IR would not have been possible without them. In this paper, we investigate whether they properly evaluate the extrapolation performance in Section 4 and then apply two resampling strategies on them to answer RQ1 and RQ2 in Section 6.

The benchmarks consist of two tightly-related tasks: passage ranking and document ranking, which share very similar training and test queries. The MS MARCO labels of the document ranking task are transferred from the passage ranking task. This paper focuses on the passage ranking task, and the findings can also be generalized to the document ranking task. Next, we show the details of the passage ranking task.
MS MARCO and TREC DL share the same corpus and training queries. The corpus has approximately 8.8 million passages extracted from Web pages. The training queries are 0.5 million natural-language queries gathered from the Bing search engine’s log. They are shallowly annotated. On average, only one passage per query is marked relevant.

MS MARCO and TREC DL provide different test queries. MS MARCO has a development (dev) set and an evaluation set. The latter is hidden and is used for the leaderboard. Thus this paper uses the public dev set for investigation. It has 6,980 natural-language queries sampled from Bing’s search log. They are shallowly annotated, and most queries only have one relevant passage. The official evaluation metric is NDCG@10. As for TREC DL, it provides small test query sets. The query-passage pairs are judged on a four-point scale. A large annotation pool is constructed to identify a sufficiently comprehensive set of relevant passages. We use the query sets in the year 2019 and 2020. They have 43 and 54 test queries, respectively. The official evaluation metric is MRR@10.

4 TRAIN-TEST LABEL OVERLAP

To investigate whether the MS MARCO and TREC DL benchmarks evaluate the interpolation or extrapolation performance, we examine the overlap between the test and training sets. Inspired by Lewis et al. [28], we investigate the overlap from two aspects. Consider a query-passage pair \((q, P)\) from the test set where \(P\) consists of several relevant passages \(P = \{p_1, ..., p_n\}\). We can examine the query overlap where there exists some \((q', P')\) in the training set such that \(q'\) is very similar to \(q\). We can also examine passage overlap where there exists at least one \((q'', P'')\) in the training set such that \(P\) and \(P''\) share at least one passage.

4.1 Query Overlap

We investigate query overlap between training and test sets from two aspects, i.e., entity overlap and intent similarity. We first use BM25 [36, 49] and an embedding model\(^2\) to separately recall ten similar training queries for each test query. Then, three annotators are recruited to label the similarity scores between each test query and the recalled training queries. Judgments for entity overlap are on a three-point scale:

- **Full Overlap**: There exists at least one training query that shares exactly the same set of entities with the test query.
- **Partial Overlap**: There exists at least one training query that shares at least one entity with the test query but no training query that fully overlaps with the test query.
- **No Overlap**: There exists no training query that shares any entity with the test query.

Judgments for intent similarity are on a four-point scale:

- **Duplicate**: There exists at least one training query with exactly the same intent as the test query.
- **Similar**: There exists at least one training query with a similar intent as the test query but no duplicate query.
- **Dissimilar**: All recalled training queries are different in intent from the test query.

\(^2\)We use the open-sourced ADORE query encoder [52] to encode the queries.

| Entity Overlap       | TREC 19 DL | TREC 20 DL | Dev-samples |
|----------------------|------------|------------|-------------|
| Full overlap         | 63%        | 54%        | 45%         |
| Partial overlap      | 30%        | 39%        | 51%         |
| No overlap           | 7%         | 7%         | 4%          |

Table 1: Distribution of test queries in terms of entity overlap with training queries. The number shows the percentage of test queries that have exactly/partially/no (the) same entities with training queries. Dev-samples are 100 queries randomly sampled from MS MARCO dev set.

| Intent Similarity    | TREC 19 DL | TREC 20 DL | Dev-samples |
|----------------------|------------|------------|-------------|
| Duplicate            | 51%        | 37%        | 34%         |
| Similar              | 28%        | 22%        | 19%         |
| Dissimilar           | 21%        | 41%        | 47%         |

Table 2: Distribution of test queries in terms of intent similarity with training queries. The number shows the percentage of test queries that share duplicate/similar/dissimilar intent with training queries. Dev-samples are 100 queries randomly sampled from MS MARCO dev set.

Table 3: Examples of duplicate test and training queries.

| TREC DL Test Qry                          | Duplicate MS MARCO Train Qry                          |
|------------------------------------------|-----------------------------------------------------|
| average wedding dress alteration cost   | average cost for wedding dress alterations           |
| what does it mean if your tsh is low    | what does it mean if my tsh is low                   |
| meaning of shebang                      | what is a shebang                                     |
| do google docs auto save                | does google docs automatically save                  |

The average Cohen’s Kappa values [8] are 0.59 and 0.54 for entity and intent annotation, respectively. When there is disagreement among the annotators, we use the median score as the final label. Table 1 shows the train-test query entity overlap. We can see that more than 90% test query entities are fully or partially overlapped by one training query. The results indicate that the benchmarks may fail to evaluate how DR matches novel entities. Table 2 shows the train-test query intent similarity. We observe a surprisingly high overlap. More than 50% queries share duplicate or similar intent with training queries. Since we only annotate not more than 20 candidates for each test query, the similar training queries for some test queries are missing from the candidates and the true overlap proportions are expected to be higher. Among the three test collections, TREC 19 DL test set is most similar to the training set, where only 21% of test queries are novel. Table 3 shows several duplication examples. We can see that the training query is very similar to the test query.
We examine to what extent the relevant passages of TREC 19 & 20 DL overlap with the relevant passages in the training set. We convert the four-point scaled labels into binary labels using different thresholding values in order to construct the relevant passage set \( P \) on the two test sets. The training labels are already binary and thus do not need to be converted. Second, we mark \((q, P)\) as an overlapped pair if any passage in \( P \) is labeled relevant in the training set.

We show the proportion of overlapped test pairs with different relevance thresholds in Table 4. It is striking that near 80\% queries have at least one relevant passage that is also labeled as relevant in the training set. Such a considerable overlap makes it possible for the ranking models to achieve decent test-set scores through memorizing training labels.

One might argue that such considerable passage overlap does not rule out the situation where one passage is relevant to two dissimilar queries. In this case, memorization may not help improve test-set performance. Therefore, we investigate the relationship between query overlap and passage overlap. Recall that we have labeled whether each test query has a similar training query and whether its relevant passages overlap with the training set. We compute the confusion matrix between query overlap and passage overlap. We regard the query as ‘query intent overlapped’ as long as it has a duplicate or similar training query. We view the query as ‘passage overlapped’ as long as any of its perfectly, highly, or somewhat relevant passages is involved in the training set. The confusion matrix is shown in Table 5. We can see that if a query is ‘passage overlapped’, it is much more likely to be ‘query intent overlapped’ than ‘not overlapped’ (59\% vs. 19\%). What is more, since the ‘query intent overlap’ is annotated based on at most 20 recalled training query candidates, the similar training queries of those 19\% test queries may not be included in the pool. Thus the probability of ‘query intent overlap’ is underestimated. The actual probability that the query is ‘passage overlapped’ and ‘query intent overlapped’ is expected to be much higher than 59\%, and the probability that the query is ‘passage overlapped’ but not ‘query intent overlapped’ should be much smaller than 19\%. Therefore, the substantial passage overlap does allow memorization to play an unjustified role during evaluation.

It raises our concern about to what extent the benchmark measures the extrapolation performance, which is arguably important for ad hoc search. To address this issue, we attempt to propose two novel evaluation methods in the next section.

### 5 Resampling Evaluation Methods

Having shown that the evaluation metrics on benchmarks may be heavily biased towards the interpolation performance, we propose two resampling methods to evaluate extrapolation performance on existing benchmarks. In short, we resample training queries that are similar or dissimilar to the test queries to evaluate interpolation or extrapolation performance, respectively. Next, we first discuss how to compute query similarity and then elaborate on the two resampling methods.

There are two options for computing query similarity, i.e., term-based BM25 [36] and embedding-based retrieval [52]. We utilize the manual annotation results to decide which is better. Recall that Section 4.1 annotates similarities between test queries and recalled training queries. A better model should recall more similar queries, resulting in higher annotated similarity scores. According to the annotation results, the embedding-based method recalls more similar queries. Hence it is used to compute query similarity. With the embedding-based query similarity, we propose the following two resampling methods for small and large test sets, respectively.

#### 5.1 Resampling Training Queries

The first method is ReSampling Training Queries (ReSTrain). ReSTrain keeps the original test set unchanged. It selects similar or dissimilar training queries as a new training set for interpolation or extrapolation evaluation, respectively. Concretely, ReSTrain collects top-\( I \) similar training queries of each test query as the interpolation training set. It excludes top-\( E \) similar training queries of each test query to construct the extrapolation training set.
and $E$ to explore the influence of training set size. We separately train ranking models on the new training sets and evaluate them on the original test set. The gap in the evaluation scores reveals the performance loss when the model transitions from interpolation to extrapolation.

We visualize ReSTrain in Figure 1a. To enable clear visualization, we first use PCA to reduce dimension to two and then resample training queries. Blue and orange points indicate extrapolation and interpolation training queries, respectively. As the figure shows, memorization may help improve interpolation performance because the interpolation training queries are close to the test queries. But it hardly helps extrapolation due to the substantial distinction between training and test queries. Thus, models that do not rely on memorization can achieve similar effectiveness in the two evaluation regimes.

ReSTrain works well for small test sets like TREC 19 & 20 DL because it can find many distinct training queries to construct a new extrapolation training set. Nevertheless, if the test set is large (e.g., MS MARCO dev set), the test queries cover a broad range of topics and thus relate to too many training queries. In this scenario, ReSTrain struggles to evaluate extrapolation performance because only a few truly dissimilar training queries would remain in the extrapolation training set. Next, we will introduce another resampling method to address this limitation.

5.2 Resampling Training and Test Queries

The second evaluation method is ReSampling Training and Test Queries (ReSTTest), which is specially used for large test sets. ReSTTest clusters the large test set to several small ones so that we can evaluate extrapolation performance for a small proportion of the test queries each time.

Specifically, ReSTTest clusters training queries and test queries into $k$ buckets with k-means. We use training queries in $k-1$ buckets as a new training set. The test queries in those $k-1$ buckets are used for interpolation evaluation, and the test queries in the one remaining bucket are used for extrapolation evaluation. We train and evaluate models $k$ times to acquire full extrapolation results of all test queries. As for interpolation, we report the average evaluation metric score since each test query is judged $k-1$ times. In this paper, $k$ is set to 5.

We visualize ReSTTest in Figure 1b. Like Figure 1a, we also use PCA for dimension reduction to obtain clear visualization. As shown in the figure, we divide the vector space into five ($k=5$) parts. Each time we select one part. The training queries in the selected part are removed when we train the ranking models. Then, we use the test queries in this part to evaluate the extrapolation capacity and the remaining test queries to evaluate the interpolation performance. We repeat this process five times until all test queries have been used in the extrapolation evaluation.

6 EVALUATING EXTRAPOLATION

With the two resampling methods, we perform a deep analysis into how DR performs in the extrapolation regime. Specifically, we compare DR with models of different architectures to address RQ1 in Section 6.1, and then evaluate DR trained by different methods to answer RQ2 in Section 6.2.

6.1 Comparing Model Architectures

We compare DR with learned sparse retrieval method and interaction-based models to answer RQ1 (the influence of the model architectures). They are all evaluated with the proposed two resampling methods. We select SPLADE [15] to represent sparse retrieval methods, and opt for ColBERT [26] and BERT reranker [34] to represent interaction-based ranking models. We first introduce the three baselines, then brief the implementation details, and finally discuss the evaluation results.

6.1.1 Baselines.

Learned sparse retrieval models leverage pretrained language models to predict term weights and build inverted indexes for efficient retrieval. It can be regarded as encoding text to a sparse and vocabulary-sized vector. Therefore, it is also representation-based like DR. We select SPLADE [15] as a representative because its variant [14] achieves state-of-the-art results in domain transfer tasks (BEIR [39]). Unlike sparse retrieval and DR, interaction-based methods model the term-term interactions to predict relevance. For example, BERT reranker [34] takes the input of both the query and the passage. At each layer, BERT computes the attention scores between each term pair to contextualize the representations. ColBERT [26] is based on BERT and further leverages a late-interaction mechanism. It only models the interaction at the end of the last layer. Therefore, the early document term representations are query-independent and can be computed offline to speed up inference.

6.1.2 Implementation.

We use the authors’ open-sourced code for training ColBERT [26] and SPLADE [15]. We implement BERT reranker by using pairwise cross-entropy loss. For all three models, we experiment with different strategies to sample negatives including random negatives and BM25 top negatives. We find all three models perform the best with BM25 top-1000 negatives. As for DR, we adopt a base training setting, i.e., 1,024 random negatives per mini-batch and cross-entropy loss. We do not use BM25 negatives here because Zhan et al. [52] point out that BM25 negatives are harmful to DR. We consistently use the output of [cls] tokens as the text embedding in this paper. We denote the trained DR model as DR (base) and will explore other advanced training methods in the next section. All neural models are initialized with BERT-base-uncased [12] and are trained for 50 epochs. For evaluation settings, BERT reranker reranks the top-1000 passages retrieved by BM25, and SPLADE, ColBERT, and DR end-to-end retrieve top passages from the entire corpus. We evaluate the checkpoint every 2 epochs and select the best one according to the performance on the development set. The metric scores when the entire training data is used match the reported numbers of previous studies [15, 26, 34, 47, 52], which validates the correctness of our implementation. Besides the above neural models, we involve BM25 [49] as a traditional baseline and tune its parameters on the training data with grid search.

6.1.3 Evaluation Results and Discussion.

We utilize ReSTrain to sample two training sets of the same size for interpolation and extrapolation evaluation. We set the number of training queries to 14k, 45k, and 200k to explore the influence of training data. ReSTrain is for small test sets and we use it to evaluate models on TREC DL test sets. The results are shown in
Table 6 shows that DR and SPLADE behave differently when the size of training data increases. When the training data is limited (e.g., 14k or 45k queries), SPLADE substantially underperforms in terms of NDCG@10 compared with DR. It also loses much more when transitioning from interpolation to extrapolation (17% vs. 14%). However, when there is abundant training data (200k queries), SPLADE extrapolates much better. It manages to outperform DR in terms of NDCG@10 and only loses 5% performance when it extrapolates. On the contrary, DR is less sensitive to the size of training data. When there is more training data, the gap between interpolation and extrapolation slightly decreases in terms of NDCG@10 (14% vs 11%) but slightly increases in terms of R@100 (7% vs. 9%). We conjecture that this is because SPLADE represents text with a much higher dimensional vector than DR. The vector dimension of SPLADE equals the vocabulary size, whereas the vector dimension of DR is usually a few hundred. In our experiments, they are 30,522 and 768, respectively. The high dimensional vectors of SPLADE may easily overfit a small amount of training data. This problem is resolved when sufficient training data exists. We leave further investigating this phenomenon to future studies.

6.1.4 Answer to RQ1.

DR, as long as SPLADE, is much more vulnerable to the transition from interpolation to extrapolation than interaction-based models. If the test query is similar to the training queries, DR performs almost as well as interaction-based models. However, for those novel test queries, DR performs substantially worse. The results also suggest that the overall metric scores fail to show a complete picture of DR performance and that we need additional extrapolation evaluation to test how ranking models perform on novel test queries.

6.2 Comparing DR Training Methods

Having shown that DR models trained with a base setting extrapolate poorly, we evaluate some recently proposed DR training methods to answer RQ2. In particular, we consider hard negative mining [47, 52], distillation [21, 30], and pretraining [16, 17] methods in our experiments because previous studies have demonstrated that they effectively improve the overall retrieval performance on the test sets of MS MARCO and TREC DL. We will first introduce

Table 6: Interpolation and extrapolation ranking performance on TREC 2019 and 2020 Deep Learning Tracks evaluated with ReSTrain. Δ=Extra./Inter.-1. #Q denotes the number of training queries. Reranker and ColBERT are interaction-based models. SPLADE and DR are representation-based models.

| #Q Models       | TREC 19&20 DL (ReSTrain) | BM25 | BM25 | R@100 R@100 | BM25 | BM25 |
|-----------------|--------------------------|------|------|-------------|------|------|
|                 | NDCG@10                  | Inter. | Extra. | Δ          | Inter. | Extra. | Δ          |
| 14k Reranker    | 0.485                    | 0.490 | +1%  | 0.534       | 0.541 | +1%  |
| 14k ColBERT     | 0.597                    | 0.585 | -2%  | 0.567       | 0.605 | +7%  |
| 14k SPLADE      | 0.596                    | 0.488 | -18% | 0.588       | 0.547 | -7%  |
| 14k DR (base)   | 0.617                    | 0.529 | -14% | 0.576       | 0.534 | -7%  |
| 45k Reranker    | 0.692                    | 0.680 | -2%  | 0.659       | 0.650 | -1%  |
| 45k ColBERT     | 0.649                    | 0.633 | -2%  | 0.612       | 0.633 | +3%  |
| 45k SPLADE      | 0.615                    | 0.512 | -17% | 0.615       | 0.567 | -8%  |
| 45k DR (base)   | 0.647                    | 0.558 | -14% | 0.607       | 0.558 | -8%  |
| 200k Reranker   | 0.709                    | 0.705 | -1%  | 0.660       | 0.664 | +1%  |
| 200k ColBERT    | 0.676                    | 0.661 | -2%  | 0.633       | 0.649 | +3%  |
| 200k SPLADE     | 0.637                    | 0.607 | -5%  | 0.642       | 0.611 | -5%  |
| 200k DR (base)  | 0.664                    | 0.591 | -11% | 0.621       | 0.564 | -9%  |

Table 7: Interpolation and extrapolation ranking performance on MS MARCO development set evaluated with ReSTTest. Δ=Extra./Inter.-1. Reranker and ColBERT are interaction-based models. SPLADE and DR are representation-based models.

| Models | MMSMARCO Dev (ReSTTest) | MRR@10 | R@100 |
|--------|-------------------------|--------|-------|
|        | Inter. | Extra. | Δ     | Inter. | Extra. | Δ     |
| BM25   | 0.189  | 0.188  | -1%   | 0.671  | 0.669  | 0%    |
| Reranker | 0.369  | 0.346  | -6%   | 0.824  | 0.815  | -1%   |
| ColBERT | 0.358  | 0.324  | -9%   | 0.877  | 0.848  | -3%   |
| SPLADE | 0.324  | 0.282  | -13%  | 0.867  | 0.812  | -6%   |
| DR (base) | 0.313  | 0.277  | -12%  | 0.848  | 0.789  | -7%   |
the three training methods and the implementation details. Then we will discuss the experimental results.

6.2.1 Advanced DR Training Methods.

There are mainly three methods to improve the efficacy of DR models. First, Xiong et al. [47] and Zhan et al. [52] employ hard negatives during training. Xiong et al. [47] asynchronously sample hard negatives, and Zhan et al. [52] further propose synchronous sampling via freezing the document index. Zhan et al. [52] also theoretically explain that hard negatives change the optimization target and thus improve the top-ranking performance. Second, Hofstätter et al. [21] and Lin et al. [30] propose to distill BERT rankers and ColBERT, respectively. The two interaction-based models exhibit strong ranking performance and thus are suitable teachers for DR. Distilling ColBERT is more computationally friendly because ColBERT is more efficient than BERT ranker. Finally, pretraining methods tailored for DR are also effective [16, 17]. Condenser [16] utilizes an auto-encoding task, and coCondenser [17] leverages contrastive learning. Note that recently there is another research direction to improve the efficiency of DR via jointly optimizing encoders and index [51, 53]. We leave investigating these methods to future work.

6.2.2 Implementation.

We follow STAR and ADORE [52] to implement hard negative mining. And we follow TCT-ColBERT [30] to implement distillation. The metric scores when we use the full training set match the numbers reported in previous studies, demonstrating the correctness of our implementation. As for pretraining techniques, we explore a variety of pretrained language models, including BERT [12], RoBERTa [31], ERNIE 2.0 [38], Condenser [16], and coCondenser [17]. All models except coCondenser are trained on Wikipedia corpus. coCondenser has two variants. One denoted as coCon-wiki is pretrained on Wikipedia. The other denoted as coCon-marco is pretrained on Wikipedia and MS MARCO.

6.2.3 Comparing Pretraining and Finetuning.

This section investigates how the pretraining and finetuning methods affect DR’s extrapolation capacity. We vary the training methods and evaluate the DR models with our proposed two resampling methods. The results are shown in Table 8. We can see that initializing models with coCon-marco [16] is the most effective way to improve extrapolation capacity. It substantially reduces the interpolation-extrapolation performance gap, whereas distillation [30] and hard negative mining [52] cannot. It also remarkably improves the absolute extrapolation performance. For example, measured by R@100, its extrapolation performance nearly matches or even outperforms the other models’ interpolation performance. On the contrary, hard negative mining and distillation are effective for interpolation. For example, they interpolate better than coCondenser in MRR@10 on the MS MARCO dev set.

Therefore, the results imply that pretraining is more effective than finetuning in improving DR extrapolation ability. Considering the impressive performance of coCon-marco, we decide to perform a deep dive into how pretraining helps extrapolation.

6.2.4 Investigation of Pretraining.

We comprehensively study the efficacy of pretraining to gain insights into what pretraining techniques are effective for improving extrapolation performance. We initialize DR encoders with various pretrained language models and then finetune them using the base training setting, i.e., 1024 random negatives per training step and cross-entropy loss. We also directly finetune a randomly initialized model as a baseline. The results are shown in Table 9.

First, the results highlight the importance of pretrained language models. Finetuning a randomly initialized model leads to 30% and 16% performance gap on TREC DL and MS MARCO dev set, respectively, which is about twice the gap when we use RoBERTa [31] for initialization. Table 10 visualizes the passage representations of models with and without pretraining. Different colors indicate relevant passages for different TREC 20 DL test queries. We can see that finetuning a randomly initialized model (Rand) leads to poor extrapolation performance but decent interpolation performance. Instead, the pretrained language model (coCon-marco) is

---

**Table 8: Interpolation and extrapolation ranking performance of models trained with different methods. Δ=Extra./Inter.-1. The training set size for ReSTrain is 45k.**

| Type        | PLM          | Finetune | TREC 19&20 DL (ReSTrain) | MS MARCO Dev (ReSTTest) |
|-------------|--------------|----------|--------------------------|--------------------------|
|             |              |          | NDCG@10 Inter. | R@100 Inter. | MRR@10 | R@100 |
|             |              |          | Extra. Δ       | Extra. Δ       |       | Extra. Δ |
| DR (base)   | BERT         | RandNeg  | 0.647 0.558 -14% | 0.607 0.558 -8% | 0.313 | 0.277 -12% | 0.848 | 0.789 -7% |
| DR (advance)| BERT         | HardNeg  | 0.661 0.581 -12% | 0.620 0.558 -10% | 0.334 | 0.289 -13% | 0.866 | 0.801 -8% |
|             | BERT         | Distill  | 0.656 0.573 -13% | 0.538 0.554 -5%  | 0.329 | 0.291 -12% | 0.852 | 0.797 -6% |
|             | coCon-marco | RandNeg  | 0.668 0.626 -6%  | 0.641 0.644 0%   | 0.323 | 0.299 -7%  | 0.875 | 0.850 -3% |

**Table 9: Ranking performance of DR models when initialized by different PLMs. Δ=Extra./Inter.-1. The training set size for ReSTrain is 45k. TE and TD stand for text-encoding and target-domain pretraining, respectively.**

| PLM         | TE | TD | TREC 19&20 DL R@100 (ReSTrain) | MSMARCO Dev R@100 (ReSTTest) |
|-------------|----|----|-------------------------------|-------------------------------|
|             |    |    | Inter. | Extra. Δ | Inter. | Extra. Δ |
| Rand        | ✓  |    | 0.437 | 0.305 | 0.723 | 0.606 | -16% |
| RoBERTa     | ✓  | ✓  | 0.582 | 0.499 | -14% | 0.845 | 0.778 | -8% |
| BERT        | ✓  | ✓  | 0.607 | 0.558 | -8% | 0.848 | 0.789 | -7% |
| ERNIE2.0    | ✓  | ✓  | 0.610 | 0.568 | -7% | 0.856 | 0.812 | -5% |
| Condenser   | ✓  | ✓  | 0.615 | 0.580 | -6% | 0.859 | 0.811 | -5% |
| coCon-wiki  | ✓  | ✓  | 0.615 | 0.582 | -5% | 0.862 | 0.819 | -5% |
| coCon-marco | ✓  | ✓  | 0.641 | 0.644 | 0% | 0.875 | 0.850 | -3% |
Table 10: Passage representation distribution in different evaluation paradigms of a (finetuned) randomly-initialized model or coCondenser-marco model. Different colors indicate relevant passages for different queries.

| Init  | Zeroshot | Extrapolation | Interpolation |
|-------|----------|---------------|---------------|
| Rand  |          |               |               |
| coCon |          |               |               |

already equipped with the representation ability and gets better clustering results than the finetuned random-initialized model. It also substantially reduces the gap between interpolation and extrapolation as the clustering results barely change when moving from the interpolation to the extrapolation regime.

Second, it is important to adopt a text-encoding pretraining task. RoBERTa does not use this kind of task and only pretrains the model on word level. It suffers severe performance degeneration and the worst extrapolation capacity among all the pretrained language models. Others adopt different text-encoding tasks, such as next sentence prediction (BERT), sentence ordering prediction (ERNIE2.0), auto-encoding (Condenser), and contrastive learning (coCon-wiki & coCon-marco). Among them, contrastive learning is the most effective. Thanks to it, coCon-wiki only loses 5% performance when transitioning from interpolation to extrapolation.

Third, pretraining on the target domain is very effective. Both coCon-marco and coCon-wiki adopt contrastive learning for pretraining. The only difference is that coCon-marco is additionally pretrained on the MS MARCO corpus. The results show that coCon-marco extrapolates substantially better than coCon-wiki, demonstrating the importance of pretraining on the target corpus. Beneficial from text-encoding and target-domain pretraining, coCon-marco achieves the most consistent behavior in interpolation and extrapolation regimes. However, pretraining on the target domain has limitations because the target-domain corpus may not be available in some scenarios, e.g., cloud computing services. Therefore, how to further improve the extrapolation performance without the knowledge of the target corpus remains to be explored.

6.2.5 **Answer to RQ2.**

Pretraining has pronounced effect of improving DR extrapolation capacity. Among various pretraining techniques, text-encoding and target-domain pretraining are especially effective. On the contrary, finetuning techniques such as hard negative mining and distillation only marginally improve the extrapolation performance and hardly reduce the interpolation-extrapolation performance gap.

7 **RELATIONSHIP WITH TRANSFER ABILITY**

After evaluating the extrapolation performance of existing ranking models, we investigate the relationship between extrapolation performance and another generalization capacity, i.e., domain transfer ability. Domain transfer tasks evaluate how well a ranking model performs in out-of-domain scenarios. For example, the model is trained on Web search data and is transferred to medical domains. It has drawn considerable attention from the neural IR community [23, 39, 45, 46]. To provide a deeper understanding of extrapolation evaluation, we investigate how benchmark extrapolation results correlate domain transfer ability (RQ3). In the following, we first introduce the current domain transfer tasks and then discuss the correlation.

7.1 **Domain Transfer and BEIR dataset**

Domain transfer is important for low-resource domains such as finance and medicine where supervision data is hard to obtain. It is difficult to directly train a neural ranking model in such domains because neural models are data-hungry. To resolve this problem, a model is trained in data-rich domains and is then directly used in the target domain.

Evaluating the domain transfer ability is hard and costly. To construct a meaningful and trustworthy domain transfer task, one should select a low-resource domain and the annotation should be complete. However, annotation is costly for many low-resource domains such as medicine and is even prohibited due to privacy restrictions. To make matters worse, evaluating retrieval models requires annotating a large candidate pool per query so that almost all relevant pairs are labeled. Otherwise, the top-ranked documents may be false negatives and the evaluation results are not trustworthy. Therefore, annotating such a dataset is expensive. On the contrary, our resampling strategies do not require a new dataset. We reuse the existing ad hoc benchmarks [9, 11], which involve relatively comprehensive annotations. Researchers can easily test the extrapolation performance via retraining the models on the resampled training sets.

BEIR is an existing popular tool to evaluate domain transfer ability. It contains various retrieval datasets from different domains or tasks. The evaluation protocol is to test the zero-shot performance of ranking models. Despite its popularity, it is worth mentioning that it has two problems. First, the annotation of its datasets has strong lexical bias. The bias originates from the annotation pools, which are constructed by traditional lexical methods. Thus, the relevant documents not recalled by lexical methods become false negatives. The bias severely underestimates the performance of DR models because they do not utilize lexical matching signals and their top-ranked documents are always missing from the annotations [39, 46]. Therefore, BEIR is not suitable to compare models with and without lexical matching modules. The second issue of BEIR is the inconsistent definitions of relevance. Some of the datasets in BEIR are not typical ad hoc retrieval tasks. For example, some tasks are fact checking, argument retrieval, counter-argument retrieval, etc. Their definitions of relevance are different from that of ad hoc search. In contrast, our extrapolation evaluation has much less lexical bias due to relatively comprehensive annotations and the relevance definitions are consistent.
7.2 Correlation Investigation

After introducing the BEIR dataset, now we investigate the correlation between the extrapolation performance evaluated by re-sampling methods and the out-of-domain performance evaluated by the BEIR dataset. Note that BEIR is not perfect, i.e., lexical annotation bias and inconsistent definitions of relevance, as introduced in the previous section. To alleviate the effect of lexical annotation bias, we only use DR models for correlation investigation so that no models take advantage of the bias. But it is worth pointing out that our evaluation methods based on the benchmark data do not have such bias and can evaluate models of other architectures. As for inconsistent definitions of relevance in BEIR, we decide to compute correlation with each dataset in BEIR instead of with the average performance of all BEIR datasets. We check the associated dataset paper if the correlation results are abnormal. In the following, we first present the implementation details and then discuss the results.

7.2.1 Implementation.

We train DR models using the resampled training data with different settings to acquire various models. We evaluate their interpolation and extrapolation performance on retrieval benchmarks [9, 11]. We also evaluate the transfer performance using the BEIR dataset. Finally, we study the correlation between the interpolation/extrapolation performance and the transfer performance.

Additionally training many DR models may exceed our computational budget, but fortunately, we have trained many DR models in Section 6. Therefore, we reuse these trained DR models and only need to evaluate the performance on the BEIR dataset. In Section 6, ReSTrain samples three sizes of training data for each model setting. Thus, it produces many model variants and the computed correlation results are reliable. However, ReSTTest does not sub-sample the training data and the number of checkpoints is not enough for correlation computation. Therefore, we only compute the correlation between ReSTrain and BEIR dataset, but we believe the results also generalize to ReSTTest.

The reused DR models are from Table 8 and Table 9. In Table 8, we have trained models with hard negative and distillation. In Table 9, we have trained the models initialized by RoBERTa [31], BERT [12], ERNIE2.0 [38], Condenser [16], coCondenser-wiki [17], and coCondenser-marco [17]. We reuse 8 model settings in total. Considering that ReSTrain samples 3 sizes of training set, i.e., 14k, 45k, and 200k training queries, for both interpolation and extrapolation, we reuse 48 checkpoints of which half is trained on the interpolation training set and the other half is trained on the extrapolation training set. We evaluate their out-of-domain performance on BEIR and compute the spearman and kendall correlation results. Note that we exclude the randomly initialized DR model in Table 9. Since this weak model achieves the worst performance, including it simply increases the correlation numbers. But such an increase also hides the fact that the correlation with some BEIR datasets is extremely low. Therefore, we decide to exclude this model.

7.2.2 Discussion.

The correlation of DR performance evaluated by ReSTrain or BEIR datasets is shown in Table 11. For the majority of BEIR datasets, extrapolation performance on benchmarks is substantially more correlated to the out-of-domain performance. These datasets cover multiple domains, including bio-medicine [6, 42], wiki-QA [27, 50], entity search [20], duplicate question retrieval [22], citation prediction [7], and science fact-checking [44]. Among these BEIR datasets, TREC-Covid [42], NQ [27], and HotpotQA [50] are widely adopted, and the evaluation results on them are assumed to be reliable. The strong correlation to these datasets is a trustworthy signal that extrapolation has the potential to reflect domain adaption ability. The results further verify the importance of evaluating extrapolation performance on benchmarks as the interpolation performance is much less consistent with the transfer performance.

In Table 11, we only compute the correlation between interpolation/extrapolation performance and the transfer performance. To present the correlation more clearly, we plot interpolation and extrapolation performances on TREC DL with ReSTrain.

| Datasets   | Domain            | Spearman Inter. | Kendall Inter. | Spearman Extra. | Kendall Extra. |
|------------|-------------------|-----------------|----------------|-----------------|----------------|
| TREC-Covid | Bio-Medical       | 0.537           | 0.359          | 0.645           |
| NCFCorpus  | Bio-Medical       | 0.582           | 0.418          | 0.611           |
| NQ         | Wiki-QA           | 0.791           | 0.606          | 0.805           |
| HotpotQA   | Wiki-QA           | 0.579           | 0.416          | 0.635           |
| DBpedia    | Wiki-Entity       | 0.647           | 0.492          | 0.679           |
| CQADups    | StackEx. Query    | 0.577           | 0.403          | 0.730           |
| Quora      | Quora Query       | 0.655           | 0.493          | 0.647           |
| SciDocs    | Sci Citation      | 0.595           | 0.440          | 0.546           |
| SciFact    | Sci Fact-check    | 0.603           | 0.453          | 0.611           |
| Arguana    | Counter-argu.     | 0.612           | 0.435          | 0.375           |
| FQA        | Finance-QA        | 0.273           | 0.223          | 0.199           |
| FEVER      | Wiki Fact-check   | -0.284          | -0.179         | -0.142          |
| Climate-F. | Wiki Fact-check   | -0.515          | -0.419         | -0.260          |
| Touche-2020| Argument          | -0.307          | -0.216         | -0.278          |

Figure 2: DR performance (NDCG@10) on TREC-Covid (y-axis) and TREC DL (x-axis). Interpolation and extrapolation performances on TREC DL are measured with ReSTrain.
performance (TREC-Covid). We use the TREC-Covid [42] dataset as a transfer example because it features a typical low-resource domain and its relevance definition is consistent with that of ad hoc search. Previous work [46] regards it as the most reliable dataset in BEIR. The two figures are consistent with the correlation results in Table 11. The points in Figure 2b clearly follow an upward trend, whereas the points in Figure 2a do not. The results suggest that if a model achieves good in-domain extrapolation performance, it is also very likely to achieve good out-of-domain transfer performance.

We also find abnormal results in the correlation analysis for some BEIR datasets. Except for FiQA [33] whose paper contains little information about how the dataset is constructed, we conjecture the abnormal results for the rest datasets are caused by inconsistent definitions of relevance. Concretely, Arguana [43] and Touché-2020 [5] are both argument retrieval tasks. Arguana [43] requires retrieving counterarguments that should have the opposite stance. Touché-2020 [5] requires retrieving arguments that help users engage in an argumentative conversation. As for FEVER [40] and Climate-FEVER [13], they require retrieving evidence that supports given claims. The relevance definitions of the four datasets are different from the typical query-document relevance definitions in ad hoc search. Note that another fact-checking dataset, SciFact [44], is an exception: it has a strong correlation with benchmark performance. We find the claim-evidence pairs in this dataset are natural sentences and their cited papers. Therefore, it guarantees that the claims are representative of the evidence. In contrast, such relations do not exist in FEVER [40] and Climate-FEVER [13].

8 CONCLUSIONS

In this paper, we propose a simple yet effective method that evaluates the extrapolation performance of DR models, i.e., how DR models perform on queries that are distinct from the training queries. With the proposed evaluation method, we first revisit how existing evaluation protocols are both argument retrieval tasks. Arguana [43] requires retrieving counterarguments that should have the opposite stance. Touché-2020 [5] requires retrieving arguments that help users engage in an argumentative conversation. As for FEVER [40] and Climate-FEVER [13], they require retrieving evidence that supports given claims. The relevance definitions of the four datasets are different from the typical query-document relevance definitions in ad hoc search. Note that another fact-checking dataset, SciFact [44], is an exception: it has a strong correlation with benchmark performance. We find the claim-evidence pairs in this dataset are natural sentences and their cited papers. Therefore, it guarantees that the claims are representative of the evidence. In contrast, such relations do not exist in FEVER [40] and Climate-FEVER [13].
