GPL: Generative Pseudo Labeling for Unsupervised Domain Adaptation of Dense Retrieval

Anonymous ACL submission

Abstract

Dense retrieval approaches can overcome the lexical gap and lead to significantly improved search results. However, they require large amounts of training data which is not available for most domains. As shown in previous work (Thakur et al., 2021b), the performance of dense retrievers severely degrades under a domain shift. This limits the usage of dense retrieval approaches to only a few domains with large training datasets.

In this paper, we propose the novel unsupervised domain adaptation method Generative Pseudo Labeling (GPL), which combines a query generator with pseudo labeling from a cross-encoder. On six representative domain-specialized datasets, we find the proposed GPL can outperform an out-of-the-box state-of-the-art dense retrieval approach by up to 8.9 points nDCG@10. GPL requires less (unlabeled) data from the target domain and is more robust in its training than previous methods.

We further investigate the role of six recent pre-training methods in the scenario of domain adaptation for retrieval tasks, where only three could yield improved results. The best approach, TSDAE (Wang et al., 2021) can be combined with GPL, yielding another average improvement of 1.0 points nDCG@10 across the six tasks. The code is available.1

1 Introduction

Information Retrieval (IR) is a central component of many natural language applications. Traditionally, lexical methods (Robertson et al., 1994) have been used to search through text content. However, these methods suffer from the lexical gap (Berger et al., 2000) and are not able to recognize synonyms and distinguish between ambiguous words.

Recently, information retrieval methods based on dense vector spaces have become popular to address these challenges. These dense retrieval methods map queries and passages2 to a shared, dense vector space and retrieve relevant hits by nearest-neighbor search. Significant improvement over traditional approaches has been shown for various tasks (Karpukhin et al., 2020; Xiong et al., 2021). This method is also adapted increasingly by industry to enhance the search functionalities of various applications (Choi et al., 2020; Huang et al., 2020).

However, as shown in Thakur et al. (2021b), dense retrieval methods require large amounts of training data to work well.3 Most importantly, dense retrieval methods are extremely sensitive to domain shifts: Models trained on MS MARCO perform rather poorly for questions for COVID-19 scientific literature (Wang et al., 2020; Voorhees et al., 2021). The MS MARCO dataset was created before COVID-19, hence, it does not include any COVID-19 related topics and models did not learn how to represent this topic well in a vector space.

In this work, we present Generative Pseudo Labeling (GPL), an unsupervised domain adaptation technique for dense retrieval models (see Figure 1). For a collection of paragraphs from the desired domain, we use an existing pre-trained T5 encoder-decoder to generate suitable queries. For each generated query, we retrieve the most similar paragraphs using an existing dense retrieval model which will serve as negative passages. Finally, we use an existing cross-encoder to score each (query, passage)-pair and train a dense retrieval model on these generated, pseudo-labeled queries using MarginMSE-Loss (Hofstätter et al., 2020).

We use publicly available models for query generation, negative mining, and the cross-encoder, which have been trained on the MS MARCO dataset.

1Anonymous link.

2We use passage to refer to text of any length.

3For reference, the popular MS MARCO dataset (Bajaj et al., 2018) has about 500k training instances; the Natural Questions dataset (Kwiatkowski et al., 2019) has more than 100k training instances.
Figure 1: Generative Pseudo Labeling (GPL) for training domain-adapted dense retriever. First, synthetic queries are generated for each passage from the target corpus. Then, the generated queries are used for mining negative passages. Finally, the query-passage pairs are labeled by a cross-encoder and used to train the domain-adapted dense retriever. The output at each step is marked with dashed boxes.
predicted queries, generated by a trained encoder-decoder model, and uses traditional BM25 lexical search. This performed well in the zero-shot retrieval benchmark BeIR (Thakur et al., 2021b). Ma et al. (2021) propose QGen, that uses a query generator trained on general domain data to synthesize domain-targeted queries for the target corpus, on which a dense retriever is trained from scratch. Following this idea, Thakur et al. (2021b) views QGen as a post-training method to adapt powerful MS MARCO retrievers to the target domains.

Despite the success of QGen, previous methods only consider the cross-entropy loss with in-batch negatives, which provides coarse-grained relevance and thus limits the performance. In this work, we show that extending this approach by using pseudo-labels from a cross-encoder together with hard negatives can boost the performance by several points nDCG@10.

**Other Methods.** Recently, Xin et al. (2021) proposes MoDIR to use Domain Adversarial Training (DAT) (Ganin et al., 2016) for unsupervised domain adaptation of dense retrievers. MoDIR trains models by generating domain invariant representations to attack a domain classifier. However, as argued in Karouzos et al. (2021), DAT trains models by minimizing the distance between representations from different domains and such learning objective can result in bad embedding space and unstable performance. For sentiment classification, Karouzos et al. (2021) proposes UDALM based on multiple stages of training. UDALM first applies MLM training on the target domain; and it then applies multi-task learning on the target domain with MLM and on the source domain with a supervised objective. However, as shown in section 5, we find this method cannot yield improvement for retrieval tasks.

**Pseudo Labeling and Cross-Encoders:** Bi-encoders map queries and passage independently to a shared vector space from which the query-passage similarity is computed. In contrast, cross-encoders (Humeau et al., 2020) work on the concatenation of the query and passage and predict a relevance score using cross-attention between query and passage. This can be used in a re-ranking setup (Nogueira and Cho, 2019), where the relevance is predicted for all query-passage-pairs for a small candidate set. Previous work has shown that cross-encoders achieve much higher performances (Thakur et al., 2021a; Hofstätter et al., 2020; Ren et al., 2021) and are less prone to domain shifts (Thakur et al., 2021b). But cross-encoders come with an extremely high computational overhead, making them less suited for a production setting. Transferring knowledge from cross-encoder to bi-encoders have been shown previous for sentence embeddings (Thakur et al., 2021a) and for dense retrieval: Hofstätter et al. (2020) predict cross-encoder scores for (query, positive)-pairs and (query, negative)-pairs and learns a bi-encoder to predict the margin between the two scores. This has been shown highly effective for in-domain dense retrieval.

## 3 Method

This section describes our proposed Generative Pseudo Labeling (GPL) method for the unsupervised domain adaptation of dense retrievers. Figure 1 illustrates the idea of GPL.

For a given target corpus, we generate for each passage three queries (cf. Table 3) using an T5-generator model (Raffel et al., 2020). For each of the generated queries, we use an existing retrieval system to retrieve 50 negative passages. Dense retrieval with a pre-existing model was slightly more effective than BM25 lexical retrieval (cf. Appendix A). For each (query, positive, negative)-tuple we compute the margin \( \delta = CE(Q, P^+) - CE(Q, P^-) \) with CE the score as predicted by a cross-encoder, \( Q \) the query and \( P^+/P^- \) the positive / negative passage.

We use the synthetic dataset \( D_{GPL} = \{(Q_i, P_i, P^-_i, \delta_i)\} \) with the MarginMSE loss (Hofstätter et al., 2020) for training a domain-adapted dense retriever that maps queries and passages into the shared vector space.

Our method requires from the target domain just an unlabeled collection of passages. Further, we use pre-existing T5- and cross-encoder models that have been trained on the MS MARCO passages dataset.

**Query Generation:** To enable supervised training on the target corpus, synthetic queries can be generated for the target passages using a query generator trained on a different, existing dataset like MS MARCO. Previous work QGen (Ma et al., 2021) used the simple MultipleNegativesRanking (MNRL) loss (Henderson et al., 2017; van den Oord et al., 2018) with in-batch negatives to train the model:
\[ L_{\text{MNRL}}(\theta) = -\frac{1}{M} \sum_{i=0}^{M-1} \log \frac{\exp(\tau \cdot \sigma(f_0(Q_i), f_0(P_i)))}{\sum_{j=0}^{M-1} \exp(\tau \cdot \sigma(f_0(Q_i), f_0(P_j)))} \]

where \( P_i \) is a relevant passage for \( Q_i \); \( \sigma \) is a certain similarity function for vectors; \( \tau \) controls the sharpness of the softmax normalization; \( M \) is the batch size.

**MarginMSE loss**: MultipleNegativesRanking loss considers only the coarse relationship between queries and passages, i.e. the matching passage is considered as relevant while all other passages are considered irrelevant. However, the query encoder is not without flaws and might generate queries that are not answerable by the passage. Further, other passages might actually be relevant as well for a given query, which is especially the case if training is done with hard negatives as we do it for GPL.

In contrast, MarginMSE loss (Hofstätter et al., 2020) uses a powerful cross-encoder to soft-label (query, passage) pairs. It then teaches the dense retriever to assign a large distance between the corresponding embeddings. In section 6.3, we show that GPL solves two major issues from the previous QGen method: A badly generated query for a given passage will get a low score from the cross-encoder, hence, we do not expect the dense retriever to put the query and passage close in the vector space. A false negative will lead to a high score from the cross-encoder, hence, we do not force the dense retriever to assign a large distance between the corresponding embeddings. In section 6.3, we show that GPL is a lot more robust to badly generated queries than the previous QGen method.

### 4 Experiments

In this section, we describe the experimental setup, the datasets used and the baselines for comparison.

#### 4.1 Experimental Setup

We use the MS MARCO passage ranking dataset (Bajaj et al., 2018) as the data from the source domain. It has 8.8M passages and 532.8K query-passage pairs labeled as relevant in the training set. As Table 1 shows, a state-of-the-art dense retrieval model, achieving an MRR@10 of 33.2 points on the MS MARCO passage ranking dataset, performs poorly on the six selected domain-specific retrieval datasets when compared to simple BM25 lexical search.

We use the DistilBERT (Sanh et al., 2019) for all the experiments. We use a maximum sequence length of 350 with mean pooling and dot-product similarity. For QGen, we use the default setting in Thakur et al. (2021b): 1-epoch training and batch size 75. For GPL, we train the models with 140k training steps and batch size 32.

To generate queries for both QGen and GPL, we use the docT5query (Nogueira et al., 2019a) generator trained on MS MARCO and generate 3 queries per passage using nucleus sampling with temperature 1.0, \( k = 25 \) and \( p = 0.95 \). To retrieve hard negatives for both GPL and the zero-shot setting of MS MARCO training, we use two dense retrievers using cosine-similarity trained on MS MARCO: msMarco-distilbert-base-v3 and msMarco-MiniLM-L-6-v3 from Sentence-Transformers\(^5\). The zero-shot performance of these two dense retrievers are available in Appendix B.

We retrieve 50 negatives using each retriever and uniformly sample one negative passage and one positive passage for each training query to form one training example. For pseudo labeling, we use the ms-marco-MiniLM-L-6-v2\(^6\) cross-encoder. For TSDAE pre-training, we train the models with 100K training steps and batch size 8.

#### 4.2 Evaluation

As our methods focus on domain adaptation to specialized domains, we selected six domain-specific text retrieval tasks from the BeIR benchmark (Thakur et al., 2021b): FiQA (financial domain) (Maia et al., 2018), SciFact (scientific papers) (Wadden et al., 2020), BioASQ (biomedical Q&A) (Tsatsaronis et al., 2015), TREC-COVID

---

\(^5\)We use the script from BeIR at https://github.com/UKPLab/beir.

\(^6\)https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2
(scientific papers on COVID-19) (Roberts et al., 2020), CQA DupStack (12 StackExchange sub-forums) (Hoogeveen et al., 2015) and Robust04 (news articles) (Voorhees, 2005). These selected datasets each contain a corpus with a rather specific language and can thus act as a suitable test bed for domain adaptation.

The detailed information for all the target datasets is available at Appendix C. We make modification on BioASQ and TREC-COVID. For efficient training and evaluation on BioASQ, we randomly remove irrelevant passages to make the final corpus size to 1M. In TREC-COVID, the original corpus has many documents with a missing abstract. The retrieval systems that were used to create the annotation pool for TREC-COVID often ignored such documents, leading to a strong annotation bias for these documents. Hence, we removed all documents with a missing abstract from the corpus. The evaluation results on the original BioASQ and TREC-COVID are available at Appendix C. Evaluation is done using nDCG@10.

4.3 Baselines

Zero-Shot Models: We apply supervised training on MS MARCO or PAQ (Lewis et al., 2021) and evaluate the trained retrievers on the target datasets. (a) MS MARCO represents a distilbert-base dense retrieval model trained with MarginMSE on the MS MARCO dataset with batch-size 75 for 70k steps. (b) PAQ (Oguz et al., 2021) represents MNRL training on the PAQ dataset. (c) PAQ + MS MARCO represents MNRL training on PAQ followed by MarginMSE training on MS MARCO. (d) TSDAE + MS MARCO pre-training on MS MARCO followed by MarginMSE training on MS MARCO. (e) BM25 system based on lexical matching from Elasticsearch.

Previous Domain Adaptation Methods: We include two previous unsupervised domain adaptation methods, UDALM (Karouzos et al., 2021) and MoDIR (Xin et al., 2021). For UDALM, we apply MLM training on the target corpus and then apply the multi-task training of MarginMSE training on MS MARCO and MLM training on the target corpus. For MoDIR, it starts from the ANCE checkpoint and apply domain adversarial training on MS MARCO and the target dataset. As of writing, the training code of MoDIR is not public, but domain adapted models for 5 out of 6 datasets have been released by the authors.

Pre-Training based Domain Adaptation: We follow the setup proposed in Wang et al. (2021) on domain-adapted pre-training: We pre-train the dense retrievers with different methods on the target corpus and then continue to train the models on MS MARCO with MarginMSE loss. The pre-training methods consist of: (a) CD (Gao and Callan, 2021) extracts the hidden representations from an intermediate layer and applies MLM on the CLS token representation and these extracted hidden representations. (b) SimCSE (Gao et al., 2021b; Liu et al., 2021) simply encode the same text twice with different dropout masks in combination with MNRL loss. (c) CT (Carlsson et al., 2021) is similar to SimCSE but it uses two independent encoders to encode a pair of text. (d) MLM (Devlin et al., 2019) uses the default setting in original paper, where 15% tokens in a text are sampled to be masked and are needed to be predicted. (e) ICT (Lee et al., 2019) uniformly samples one sentence from a passage as the pseudo query to that passage and uses MNRL loss on the synthetic data. We follow the setting in Lee et al. (2019) and masked out the selected sentence 90% of the time. (f) TSDAE (Wang et al., 2021) uses a denoising autoencoder to pre-train the dense retrievers with 60% random tokens deleted in the input texts.

Generation-based Domain Adaptation: We use the training script from Thakur et al. (2021b) to train QGen models with the default setting. For each passage, 3 queries are generated using the same sampling strategy as for GPL. Cosine similarity is used and the models are fine-tuned for 1 epoch with MNRL. The default QGen is trained with in-batch negatives. For a fair comparison, we also test QGen with hard negatives as used in GPL, noted as QGen (w/ Hard Negatives). Further, We we test the combination of TSDAE and QGen (TSDAE + QGen).

Re-Ranking with Cross-Encoders: We also include results of the powerful but inefficient re-ranking methods for reference. Three retrievers for the first-phrase retrieval are tested: BM25 from Elasticsearch, the zero-shot MS MARCO retriever and the enhanced GPL retriever by TSDAE pre-training. We use the cross-

9CD can only be applied with CLS pooling.
9https://github.com/UKPLab/beir
Method | Dataset | FiQA | SciFact | BioASQ | TRECC. | CQADup. | Robust04 | Avg.  
--- | --- | --- | --- | --- | --- | --- | --- | ---  
Zero-Shot Models  
MS MARCO | 26.7 | 57.1 | 52.9 | 66.1 | 29.6 | 39.0 | 45.2  
PQ | 15.2 | 53.3 | 44.0 | 23.8 | 24.5 | 31.9 | 32.1  
PQ + MS MARCO | 26.7 | 57.6 | 53.8 | 63.4 | 30.6 | 37.2 | 44.9  
TSDAE_{MS MARCO} | 26.7 | 55.5 | 51.4 | 65.6 | 30.5 | 36.6 | 44.4  
BM25 | 23.9 | 66.1 | 70.7 | 60.1 | 31.5 | 38.7 | 48.5  
Previous Domain Adaptation Methods  
UDALM | 23.3 | 33.6 | 33.1 | 57.1 | 24.6 | 26.3 | 33.0  
MoDIR | 29.6 | 50.2 | 47.9 | 66.0 | 30.6 | 37.2 | –  
Pre-Training based Domain Adaptation: Target → MS MARCO  
CT | 28.3 | 55.6 | 49.9 | 63.8 | 30.5 | 35.9 | 44.0  
CD | 27.0 | 62.7 | 47.7 | 65.4 | 30.6 | 34.5 | 44.7  
SimCSE | 26.7 | 55.0 | 53.2 | 68.3 | 29.0 | 37.9 | 45.0  
ICT | 27.0 | 58.3 | 55.3 | 69.7 | 31.3 | 37.4 | 46.5  
MLM | 30.2 | 60.0 | 51.3 | 69.5 | 30.4 | 38.8 | 46.7  
TSDAE | 29.3 | 62.8 | 55.5 | 76.1 | 31.8 | 39.4 | 49.2  
Generation-based Domain Adaptation (Previous State-of-the-Art)  
QGen | 28.2 | 61.7 | 60.0 | 72.8 | 33.6 | 38.5 | 49.1  
QGen (w/ Hard Negatives) | 26.0 | 59.6 | 57.7 | 65.0 | 33.2 | 36.5 | 46.3  
TSDAE + QGen (Ours) | 30.3 | 64.7 | 60.5 | 73.8 | 35.1 | 38.4 | 50.5  
Proposed Method: Generative Pseudo Labeling  
GPL | 33.1 | 65.2 | 61.6 | 71.7 | 34.4 | 42.1 | 51.4  
TSDAE + GPL | 33.3 | 67.3 | 62.8 | 74.0 | 35.1 | 42.1 | 52.4  
Re-Ranking with Cross-Encoders (Upper Bound, Inefficient at Inference)  
BM25 + CE | 33.1 | 67.6 | 72.8 | 71.2 | 36.8 | 46.7 | 54.7  
MS MARCO + CE | 33.0 | 66.9 | 57.4 | 65.1 | 36.9 | 44.7 | 50.7  
TSDAE + GPL + CE | 36.4 | 68.1 | 68.0 | 71.4 | 38.1 | 48.3 | 55.1  

Table 1: Evaluation using nDCG@10. The best results of the single-stage dense retrievers are bold. TRECC. and CQADup. are short for TREC-COVID and CQADupStack. Our proposed GPL significantly outperforms other domain adaptation methods. For the first time, we investigate the TSDAE pre-training in domain adaptation for dense retrieval and find it can significantly improve both QGen and GPL.

encoder *ms-marco-MiniLM-L-6-v2* from Sentence-Transformers, which is also for pseudo labeling for GPL.

### 5 Results

#### Pre-Training based Domain Adaptation:

The results are shown in Table 1. Compared with the zero-shot MS MARCO model, TSDAE, MLM and ICT can improve the performance if we first pre-train on the target corpus and then perform supervised training on MS MARCO. Among them, TSDAE is the most effective method, outperforming the zero-shot baseline by 4.0 points nDCG@10 on average. CD, CT and SimCSE are not able to adapt to the domains in a pre-training setup and achieve a performance worse than the zero-shot model.

To ensure that TSDAE actually learns domain specific terminology, we include TSDAE_{MS MARCO} in our experiments: Here, we performed TSDAE pre-training on the MS MARCO dataset followed by supervised learning on MS MARCO. This performs slightly weaker than the zero-shot MS MARCO model.

We also tested the pre-training methods without any supervised training on MS MARCO. We find all of them fail miserably compared as shown in Appendix D.

#### Previous Domain Adaptation Methods:

We test MoDIR on the datasets except Robust04\(^{10}\). MoDIR performs on-par with our zero-shot MS MARCO model on FiQA, TREC-COVID and CQADupStack, while it performs much weaker on SciFact and BioASQ. An improved training setup with MoDIR could improve the results.

We also test UDALM, which first does MLM pre-training on the target corpus, and then runs multitask learning with MLM objective and supervised training on MS MARCO. The results show that UDALM in this case greatly harms the performance.

\(^{10}\)The original author did not train the model on Robust04 and the code is also not available.
mance by 12.2 points in average, when compared with the MLM-pre-training approach. We suppose this is because unlike text classification, the dense retrieval models usually do not have an additional task head and the direct MLM training conflicts with the supervised training.

**Generation-based Domain Adaptation:** The results show that the previous best method, QGen, can successfully adapt the MS MARCO models to the new domains, improving the performance on average by 3.9 points. It performs on par with TSDAE-based domain-adaptive pre-training. Combining TSDAE with QGen can further improve the performance by 1.4 points.

When using QGen with hard negatives instead of random in-batch negatives, the performance decreases by 2.8 points in average. QGen is sensitive to false negatives, i.e. negative passage that are actually relevant for the query. This is a common issue for hard negative mining. GPL solves this issue by using the cross-encoder to determine the distance between the query and a passage. We give more analysis in Appendix F.

**Generative Pseudo Labeling (GPL, proposed method):** We find GPL is significantly better on almost all the datasets compared to the other tested method, outperforming QGen by up to 4.9 points (on FiQA) and in average by 2.3 points. One exception is TREC-COVID, but as this dataset has just 50 test queries, so this difference can be due to noise.

As a further enhancement, we find that TSDAE-based domain-adaptive pre-training combined with GPL (i.e. TSDAE + GPL) can further improve the performance on all the datasets, achieving the new state-of-the-art result of 52.4 nDCG@10 points in average. It outperforms the out-of-the-box MS MARCO model 7.2 points on average.

**Re-ranking with Cross-Encoders:** Cross-encoders perform well in a zero-shot setting and outperform dense retrieval approaches significantly (Thakur et al., 2021b), but they come with a significant computational cost at inference. TSDAE and GPL can narrow but not fully close the performance gap. Due to the much lower computational costs at inference, the TSDAE + GPL model would be preferable in a production setting.

### 6 Analysis

In this section, we analyze the influence of training steps, corpus size, query generation and choices of starting checkpoints on GPL.

#### 6.1 Influence of Training Steps

We first analyze the influence of the number of training steps on the model performance. We evaluate the models every 10K training steps and end the training after 140K steps. The results for the change of averaged performance on all the datasets are shown in Figure 2. We find the performance of GPL begins to be saturated after around 100K steps. With the TSDAE pre-training, the performance can be improved consistently during the whole training stage. For reference, training a distilbert-base model for 100k steps takes about 9.6 hours on a single V100 GPU.

#### 6.2 Influence of Corpus Size

We next analyze the influence of different corpus sizes. We use Robust04 for this analysis, since it has a relatively large size. We sample 1K, 10K, 50K and 250K passages from the whole corpus independently to form small corpora and train QGen and GPL on the same small corpus. The results are shown in Table 2. We find with more than 10K passages, GPL can already significantly outperform the zero-shot baseline by 2.3 nDCG@10 points; with more than 50K passages, the performance begins to saturate. On the other hand, QGen falls...
behind the zero-shot baseline for each corpus size.

### 6.3 Robustness against Query Generation

Next, we study how the query generation influences the model performance. First, we train QGen and GPL on SciFact and generate 1 up to 10 queries per passage. The results are shown in Table 3. Generating 3 queries per passage appears to be optimal, generating more queries per passage does not yield further improvements.

The temperature plays an important role in nucleus sampling, higher values make the generated queries more diverse, but of lower quality. We train QGen and GPL on FiQA with different temperatures: 0.1, 1, 1.3, 3, 5 and 10. Examples of generated queries are available in Appendix E. We generated 3 queries per passage. The results are shown in Figure 3. We find the performance of QGen and GPL both peaks at 1.0. With a higher temperature, the next-token distribution will be flatter and more diverse queries, but of lower quality, will be generated. With high temperatures, the generated queries have nearly no relationship to the passage. QGen will perform poorly in these cases, worse than the zero-shot model. In contrast, GPL performs still well even when the generated queries are of such low quality.

### 6.4 Sensitivity to Starting Checkpoints

We also analyze the influence of initialization on GPL. In the default setting, we start from a distilbert-model supervised on MS MARCO using MarginMSE loss. We also evaluate to directly fine-tune a distilbert-model using QGen, GPL and TSDAE + GPL. The performance averaged on all the datasets are shown in Table 4. We find the MS MARCO training has relatively small effect on the performance of GPL (with 0.9-point difference in average), while QGen highly relies on the choice of the initialization checkpoint (with 3.7-point difference in average).

<table>
<thead>
<tr>
<th>Method</th>
<th>Init.</th>
<th>Distilbert</th>
<th>MS MARCO</th>
</tr>
</thead>
<tbody>
<tr>
<td>QGen</td>
<td>45.4</td>
<td>49.1</td>
<td></td>
</tr>
<tr>
<td>GPL</td>
<td>50.5</td>
<td>51.4</td>
<td></td>
</tr>
<tr>
<td>TSDAE + GPL</td>
<td>50.9</td>
<td>52.4</td>
<td></td>
</tr>
<tr>
<td>Zero-shot</td>
<td>–</td>
<td>45.2</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Influence of initialization checkpoint on performance in average. GPL yields similar performance when starting from different checkpoints.

## 7 Conclusion

In this work we propose GPL, a novel unsupervised domain adaptation method for dense retrieval models. It generates queries for a target corpus and pseudo labels these with a cross-encoders. Pseudo-labeling overcomes two important short-comings of previous methods: Not all generated queries are of high quality and pseudo-labels efficiently detects those. Further, training with mined hard negatives is possible as the pseudo labels performs efficient denoising.

We observe GPL performs well on all the datasets and significantly outperforms other approaches. As a limitation, GPL requires a relatively complex training setup and future work can focus on simplify this training pipeline.

In this work, we also evaluated different pre-training strategies in a domain-adaptive pre-training setup: We first pre-trained on the target domain, then performed supervised training on MS MARCO. ICT and MLM were able to yield a small improvement, while TSDAE was able to yield a significant improvement of 4 points. Other approaches degraded the performance.
References


A Performance of Using Different Retrievers for Negative Mining in GPL

The performance of using different retrievers (BM25, dense and BM25 + dense) for mining hard negatives in GPL is shown in Table 5. The results show GPL performs best when using hard negatives mined by dense retrievers.

<table>
<thead>
<tr>
<th>Method</th>
<th>FiQA</th>
<th>SciFact</th>
<th>BioASQ</th>
<th>TRECC.</th>
<th>CQADup.</th>
<th>Robust04</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPL (w/ BM25 + dense)</td>
<td>32.9</td>
<td>64.4</td>
<td>61.1</td>
<td>68.6</td>
<td>33.8</td>
<td>41.3</td>
<td>50.4</td>
</tr>
<tr>
<td>GPL (w/ BM25)</td>
<td>31.1</td>
<td>60.9</td>
<td>57.8</td>
<td>67.5</td>
<td>33.5</td>
<td>35.9</td>
<td>47.8</td>
</tr>
<tr>
<td>GPL (w/ dense)</td>
<td>33.1</td>
<td>65.2</td>
<td>61.6</td>
<td>71.7</td>
<td>34.4</td>
<td>42.1</td>
<td>51.4</td>
</tr>
<tr>
<td>MS MARCO</td>
<td>26.7</td>
<td>57.1</td>
<td>52.9</td>
<td>66.1</td>
<td>29.6</td>
<td>39.0</td>
<td>45.2</td>
</tr>
</tbody>
</table>

Table 5: Performance of using different retrievers for hard-negative mining in GPL. The scores of the baseline MS MARCO and the scores of GPL with dense retrievers are copied from Table 1.

B Performance of the Zero-Shot Retrievers in Hard-Negative Mining

The performance of directly using the zero-shot retrievers for hard-negative mining in GPL is shown in Table 6. Compared with the strong baseline (MS MARCO in Table 6) trained with MarginMSE, msmarco-distilbert-base-v3 and msmarco-MiniLM-L-6-v3 are much worse in terms of zero-shot generalization on each dataset. This comparison supports GPL can indeed train powerful domain-adapted dense retrievers with minimum reliance on choices of the retrievers for hard-negative mining.

<table>
<thead>
<tr>
<th>Method</th>
<th>FiQA</th>
<th>SciFact</th>
<th>BioASQ</th>
<th>TRECC.</th>
<th>CQADup.</th>
<th>Robust04</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>msmarco-distilbert-base-v3</td>
<td>24.0</td>
<td>52.3</td>
<td>45.6</td>
<td>61.1</td>
<td>24.3</td>
<td>30.6</td>
<td>39.7</td>
</tr>
<tr>
<td>msmarco-MiniLM-L-6-v3</td>
<td>23.3</td>
<td>48.8</td>
<td>41.9</td>
<td>57.9</td>
<td>24.3</td>
<td>28.5</td>
<td>37.5</td>
</tr>
<tr>
<td>MS MARCO</td>
<td>26.7</td>
<td>57.1</td>
<td>52.9</td>
<td>66.1</td>
<td>29.6</td>
<td>39.0</td>
<td>45.2</td>
</tr>
</tbody>
</table>

Table 6: Performance of different zero-shot retrievers. msmarco-distilbert-base-v3 and msmarco-MiniLM-L-6-v3 are used in GPL for hard-negative mining. The scores of the baseline MS MARCO are copied from Table 1.

C Target Datasets

FiQA is for the task of opinion question answering over financial data. It contains 648 queries and 5.8K passages from StackExchange posts under the Investment topic in the period between 2009 and 2017. The labels are binary (relevant or irrelevant) and there are 2.6 passages in average labeled as relevant for each query.

SciFact is for the task of verifying scientific claims using evidence from the abstracts of the scientific papers. It contains 300 queries and 5.2K passages built from S2ORC (Lo et al., 2020), a publicly-available corpus of millions of scientific articles. The labels are binary and there are 1.1 passages in average labeled as relevant for each query.

BioASQ is for the task of biomedical question answering. It originally contains 500 queries and 15M articles from PubMed11. The labels are binary and it has 4.7 passages in average labeled as relevant for each query. For efficient training and evaluation, we randomly remove irrelevant passages to make the final corpus size to 1M.

TREC-COVID is an ad-hoc search challenge for scientific articles related to COVID-19 based on the CORD-19 dataset (Wang et al., 2020). It originally contains 50 queries and 171K documents. The original corpus has many documents with only a title and an empty body. We remove such documents and the final corpus size is 129.2K. The labels in TREC-COVID are 3-level (i.e. 0, 1 and 2) and there are 430.8 passages in average labeled as 1 or 2 in the clean-up version.

CQADupStack is a dataset for community question-answering, built from 12 StackExchange subforums: Android, English, Gaming, Gis, Mathematica, Physics, Programmers, Stats, Tex, Unix, Webmasters

and WordPress. The task is to retrieve duplicate question posts with both a title and a body text given a post title. It has 13.1K queries and 457.2K passages. The labels are binary and there are 1.4 passages in average labeled as relevant for each query. As in Thakur et al. (2021b), the average score of the 12 sub-tasks is reported.

**Robust04** is a dataset for news retrieval focusing on poorly performing topics. It has 249 queries and 528.2K passages. The labels are 3-level and there are in average 69.9 passages labeled as relevant for each query.

The detailed statistics of these target datasets are shown in Table 7.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Domain</th>
<th>Title</th>
<th>Relevancy</th>
<th>#Queries</th>
<th>#Passages</th>
<th>PPQ</th>
<th>Query Len.</th>
<th>Passage Len.</th>
</tr>
</thead>
<tbody>
<tr>
<td>FiQA</td>
<td>Financial</td>
<td>✗</td>
<td>Binary</td>
<td>648</td>
<td>57.6K</td>
<td>2.6</td>
<td>10.8</td>
<td>132.2</td>
</tr>
<tr>
<td>SciFact</td>
<td>Scientific</td>
<td>✓</td>
<td>Binary</td>
<td>300</td>
<td>5.2K</td>
<td>1.1</td>
<td>12.4</td>
<td>213.6</td>
</tr>
<tr>
<td>BioASQ</td>
<td>Bio-Medical</td>
<td>✓</td>
<td>Binary</td>
<td>500</td>
<td>1.0M</td>
<td>4.7</td>
<td>8.1</td>
<td>204.1</td>
</tr>
<tr>
<td>BioASQ*</td>
<td>Bio-Medical</td>
<td>✓</td>
<td>Binary</td>
<td>500</td>
<td>14.9M</td>
<td>4.7</td>
<td>8.1</td>
<td>202.6</td>
</tr>
<tr>
<td>TREC-COVID</td>
<td>Bio-Medical</td>
<td>✓</td>
<td>3-Level</td>
<td>50</td>
<td>129.2K</td>
<td>430.8</td>
<td>10.6</td>
<td>210.3</td>
</tr>
<tr>
<td>TREC-COVID*</td>
<td>Bio-Medical</td>
<td>✓</td>
<td>3-Level</td>
<td>50</td>
<td>171.3K</td>
<td>493.5</td>
<td>10.6</td>
<td>160.8</td>
</tr>
<tr>
<td>CQADupStack</td>
<td>Forum</td>
<td>✓</td>
<td>Binary</td>
<td>13,145</td>
<td>457.2K</td>
<td>1.4</td>
<td>8.6</td>
<td>129.1</td>
</tr>
<tr>
<td>Robust04</td>
<td>News</td>
<td>✓</td>
<td>3-Level</td>
<td>249</td>
<td>528.2K</td>
<td>69.9</td>
<td>15.3</td>
<td>466.4</td>
</tr>
</tbody>
</table>

Table 7: Statistics of the target datasets used in the experiments. Column **Title** indicates whether there is (✓) a title for each passage or not (✗). Column **PPQ** represents number of Passages Per Query. Query/passage lengths are counted in words. Symbol * marks the original version from the BeIR benchmark (Thakur et al., 2021b).

We also evaluate the models trained in this work on the original version of BioASQ and TREC-COVID datasets from BeIR (Thakur et al., 2021b). The results are shown in Table 8.

<table>
<thead>
<tr>
<th>Method</th>
<th>BioASQ</th>
<th>TRECC.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPL</td>
<td>42.5</td>
<td>71.8</td>
</tr>
<tr>
<td>TSDAE + GPL</td>
<td>42.6</td>
<td>73.7</td>
</tr>
<tr>
<td>QGen</td>
<td>37.8</td>
<td>43.1</td>
</tr>
</tbody>
</table>

Table 8: Performance on the original version of BioASQ and TREC-COVID in BeIR (Thakur et al., 2021b).

**D Performance of Unsupervised Pre-Training**

The performance of the unsupervised pre-training methods without access to the MS MARCO data is shown in Table 9. We find ICT is the best method, achieving highest scores on all the datasets. However, all the unsupervised pre-training methods cannot directly yield improvement in performance compared with the zero-shot baseline.

<table>
<thead>
<tr>
<th>Method</th>
<th>FiQA</th>
<th>SciFact</th>
<th>BioASQ</th>
<th>TRECC.</th>
<th>CQADup.</th>
<th>Robust04</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>6.6</td>
<td>0.6</td>
<td>0.3</td>
<td>9.8</td>
<td>8.1</td>
<td>3.8</td>
<td>4.9</td>
</tr>
<tr>
<td>CT</td>
<td>0.2</td>
<td>0.7</td>
<td>0.0</td>
<td>2.5</td>
<td>0.9</td>
<td>0.0</td>
<td>0.7</td>
</tr>
<tr>
<td>MLM</td>
<td>5.4</td>
<td>27.8</td>
<td>4.7</td>
<td>16.0</td>
<td>8.5</td>
<td>6.1</td>
<td>11.4</td>
</tr>
<tr>
<td>TSDAE</td>
<td>7.8</td>
<td>37.2</td>
<td>6.9</td>
<td>9.4</td>
<td>14.3</td>
<td>10.1</td>
<td>14.3</td>
</tr>
<tr>
<td>SimCSE</td>
<td>5.5</td>
<td>25.0</td>
<td>13.1</td>
<td>26.0</td>
<td>14.6</td>
<td>9.8</td>
<td>15.7</td>
</tr>
<tr>
<td>ICT</td>
<td>10.2</td>
<td>42.6</td>
<td>39.0</td>
<td>47.5</td>
<td>23.0</td>
<td>16.5</td>
<td>29.8</td>
</tr>
<tr>
<td>MS MARCO</td>
<td>26.7</td>
<td>57.1</td>
<td>52.9</td>
<td>66.1</td>
<td>29.6</td>
<td>39.0</td>
<td>45.2</td>
</tr>
</tbody>
</table>

Table 9: Performance of unsupervised pre-training methods with only access to the target corpus as the training data. The scores of the zero-shot baseline MS MARCO are copied from Table 1.

**E Examples of Generated Queries under Different Temperatures**

The generation temperature controls the sharpness of the next-token distribution. The examples for one passage from FiQA are shown in Table 10 Higher temperature results in longer and less duplicate queries.
under more risk of generating non-sense texts.

<table>
<thead>
<tr>
<th>Item</th>
<th>Text</th>
<th>Pseudo Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Passage</td>
<td>You can never use a health FSA for individual health insurance premiums. Moreover, FSA plan sponsors can limit what they are willing to reimburse. While you can’t use a health FSA for premiums, you could previously use a 125 cafeteria plan to pay premiums, but it had to be a separate election from the health FSA. However, under N. 2013-54, even using a cafeteria plan to pay for individual premiums is effectively prohibited.</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 0.1</td>
<td>can you use a cafeteria plan for premiums 9.1</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 0.1</td>
<td>can you use a cafeteria plan for premiums 9.1</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 0.1</td>
<td>can you use a cafeteria plan for premiums 9.1</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 0.1</td>
<td>can i use my fsa to pay for a health plan 9.7</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 3.0</td>
<td>what type of benefits do the health savings accounts cover 7.2</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 3.0</td>
<td>when applying for medical terms health insurance 6.0</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 5.0</td>
<td>which one does not apply after an emergency medical 11.1</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 10.0</td>
<td>will employer limit premiums reimbursement on healthcare expenses with cafeteria and capetarians account on my employer ca. plans and deductible accounts 2.5</td>
<td>–</td>
</tr>
<tr>
<td>Temperature 10.0</td>
<td>what is allowed as personal health account or ca 10.2</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 10: Examples of generated queries under different temperature value for a passage from FiQA.

<table>
<thead>
<tr>
<th>Item</th>
<th>Text</th>
<th>GPL</th>
<th>QGen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query</td>
<td>what is futures contract</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Positive</td>
<td>Futures contracts are a member of a larger class of financial assets called derivatives ...</td>
<td>10.3</td>
<td>1</td>
</tr>
<tr>
<td>Negative 1</td>
<td>... Anyway in this one example the s&amp;p 500 futures contract has an “initial margin” of $19,250, meaning ...</td>
<td>2.0</td>
<td>0</td>
</tr>
<tr>
<td>Negative 2</td>
<td>... but the moment you exercise you must have $5,940 in a margin account to actually use the futures contract ...</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>Negative 3</td>
<td>... a futures contract is simply a contract that requires party A to buy a given amount of a commodity from party B at a specified price ...</td>
<td>8.2</td>
<td>0</td>
</tr>
<tr>
<td>Negative 4</td>
<td>... A futures contract commits two parties to a buy/sell of the underlying securities, but ...</td>
<td>6.9</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 11: Examples of the labels assigned to different query-passage pairs in FiQA by GPL and QGen. The key term “futures contract” are marked in bold. QGen uses only 0-1 scores. GPL uses raw logits, which can be any value between positive and negative infinity.

F Case Study: Fine-Grained Labels

GPL uses continuous pseudo labels from a cross-encoder, which can provide more fine-grained information and is more informative than the simple 0-1 labels as in QGen. In this section, we give a more detailed insight into it by a case study.

One example from FiQA is shown in Table 11. The generated query for the positive passage asks for the definition of “futures contract”. Negative 1 and 2 only mention futures contract without explaining
the term (with low GPL labels below 2.0), while Negative 3 gives the required definition (which high
GPL label 8.2). As an interesting case, Negative 4 gives a partial explanation of the term (with medium
GPL label 6.9). GPL assigns suitable fine-grained labels to different negative passages. In contrast, QGen
simply labels all of them as 0, i.e. as irrelevant. Such difference explains the advantage of GPL over QGen
and why using hard negatives harms the performance of QGen in Table 1.