Fine Tuning Query Representation using Iterative Negative Feedback

Mudit Chaudhary
University of Massachusetts Amherst

Dhawal Gupta
University of Massachusetts Amherst

ABSTRACT

Dense retrieval methods such as DPR [6] are becoming increasingly popular due to their superior performance over traditional retrieval methods. Furthermore, dense retrieval methods are able to model the semantic meaning of the query and the documents. One of the ways to improve the retrieval performance of these methods is to use a re-ranker. Often, the re-rankers (e.g., cross-encoders) are computationally expensive and impractical for fast re-ranking on a large list of documents. Alternative ways to improve the performance is to use query representation refinement methods to re-rank the documents. Majority of the research focuses on using positive relevance feedback for improving the dense query representation. Work on using negative relevance feedback for dense query refinement is severely limited. In this paper, we propose a simple and computationally inexpensive model for query refinement using negative relevance feedback to re-rank documents. We perform experiments on the MS-Marco passage re-ranking task and aim to refine dense query representations from DPR.

KEYWORDS

Information Retrieval, Dense Retrieval, Query Refinement, Negative Feedback, Neural Networks

ACM Reference Format:
Mudit Chaudhary and Dhawal Gupta. 2022. Fine Tuning Query Representation using Iterative Negative Feedback.

1 INTRODUCTION

Online query representation refinement offers a way to increase user engagement and, at the same time, provide more personalized recommendations and information to the user. Often due to the magnitude of the models, it is prohibitive to modify the query representation online from a single user interaction feedback session. The reasons for the same are two-fold. Firstly, the amount of data available during a single user session is too scarce to perform useful finetuning on such a large representation model. Secondly, storing and updating a very large query representation model can be prohibitive on a small user device. It might require constant communication between a server and user device, which adds an extra overhead of communication and a centralized point of computation at the same time.

Negative feedback has often been looked at as a way to do better query finetuning because of the availability of large amounts of data. We often have a single / or couple of relevant passages for a given query, which is not useful enough to train a single query representation. Rather than that, because of the size of the corpus and word similarity between documents and queries, we have a plethora of document/query pairs that might have similar representation semantically but might not be relevant, i.e., be hard negatives. Using negative documents to refine query is believed to be more scalable and robust. Still, there is no well-known way to get the set of negative documents from a query based on the corpus.

In this project, we aim to answer both the above questions jointly up to some extent. We train a very small dense query representation refinement model, which can, in theory, be used for inferencing and finetuning online that too on a user device (because of being small enough), and at the same time, investigate different negative documents mining strategies to identify which strategies might aid our formulation of query refinement to improve document re-ranking in an iterative user-interaction scenario. We perform our experiments over MS-Marco Passage Re-ranking task [12] and refine DPR-generated query representations. The main components of our project are as follows :

- Train a simple and computationally inexpensive function approximator to perform online query refinement
- Use a variant of Rocchio's algorithm to define a multi-objective loss function [13]
- Investigate different negative document mining schemes to see which ones can be used to improve iterative feedback document re-ranking.

2 RELATED WORKS

The related works about the ideas can be broadly broken down into the following categories, we start with a discussion about how negative feedback is usually used.

2.1 Query Representation

Most of the existing retrieval systems use a dense representation for queries and documents, which are often encoded using a BERT-based embedding [6, 7]. Previous works have done fine-tuning for query representation for dense retrieval using pseudo relevance feedback [18] at training time. Recent works have looked into doing query refining at test time [15], on PRF can show increased performance but still uses the full transformer model and PRF data.

2.2 Negative Feedback Refinement

Relevance feedback has been a known way to improve the retrieval performance of systems through query expansion, document expansion, and query finetuning. Rocchio's algorithm is commonly used for this and there exist three forms of relevance feedback, positive, negative, and pseudo-relevance feedback [11]. PRF is the most commonly used feedback for dense models [9, 16]. Negative sampling is often performed through random batch sampling [17], sampling from BM25 top negatives [4, 8], sampling from the index of hard negatives from exiting Dense retrieval models [19], or re-ranking models [5, 10].

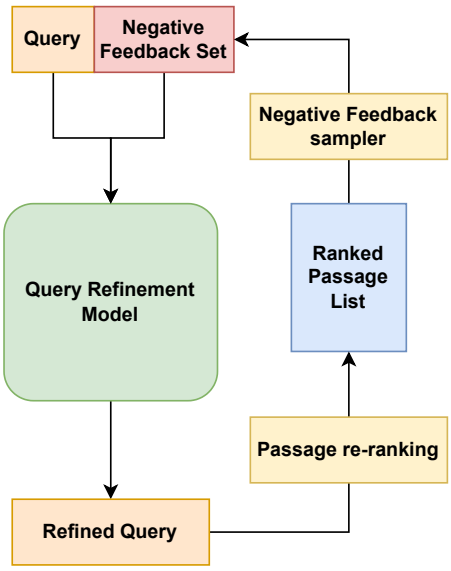

Figure 1: Query-refinement process at test time.

2.3 Interactive Feedback Refinement/ Active Learning

Recent work has tried to address the problem of doing query refinement by having interactive feedback with the user [3], usually through a multi-turn simulated user. Previous systems have also made use of user explicit user feedback for result ranking via having exploratory policy [14]. Other systems have posed this as an interactive question-answer-based system to clearly understand the user intent behind queries. [2]. The Natural Language Processing (NLP) community also poses a similar problem into the active learning problem [20], where we want to adapt queries with minimal data and training quickly.

3 METHODOLOGY

3.1 Dataset and Evaluation Metrics

We perform our experiments on the MS-Marco Passage Re-ranking dataset, which provides queries with their top-1000 retrieved passages using BM25 [1]. Due to computational resource and time constraints we train our model only on a subset of the actual training set (~8000 queries). Due to the unavailability of the test set, we evaluate our model on the development set (~5300 queries). We filter out queries that have less than 1000 retrieved passages or queries where the relevant passage is not present in the top-1000 BM25 retrieved passages.

We run Dense Passage Retrieval (DPR) on the provided top-1000 passages per query to get a DPR generated ranked list. This ranked list forms our baseline and is also used for negative sampling, which is explained in more detail in the later sections.

For evaluation, we use Mean Reciprocal Rank (MRR) as our metric. MRR is the standard metric for the MS-Marco passage re-ranking task.

3.2 Dense Retrieval Model

We use Dense Passage Retrieval (DPR) model for our initial passage ranking and to get the query & passage representations [6]. We use the `multiset-base` configuration for both passage and query encoder. To avoid re-calculating the representations during training and evaluation, we cache the representations and load them lazily. DPR outputs 768-dimensional representations of query and passages. The query representations are then refined using our query refinement model. DPR computes similarity between a query and a passage using inner product. However, to use cosine similarity we normalize the query and passage representations.

3.3 Query Refinement Model

3.3.1 Model Description. We use a simple 2-fully-connected layer neural network with ReLU activation between the two layers. The input of the model is the query and the negative feedback set. For the input, we average the representations of the negative feedback set and concatenate it with the query representations. The model outputs a 768-dimensional refined query representation.

3.3.2 Loss Formulation. We define a query fine-tuning model as $f_\theta : \mathbb{R}^{2d} \rightarrow \mathbb{R}^d$. We propose a multi-objective loss to achieve the following: i) refined query should be similar to the initial query, ii) refined query should be similar to the set of relevant documents, and iii) refined query should be dissimilar to the set non-relevant documents.

Let $q \in \mathbb{R}^d$ be the query representation, and let D_{NR}, D_R represent its set of non-relevant and relevant document representations i.e. $D_{NR} \subset \mathbb{R}^d, d_{nr} \in D_{NR}$ and $D_R \subset \mathbb{R}^d, d_r \in D_R$. We also represent the mean non-relevant document representation as $\bar{d}_{nr}$. The formulated loss function becomes:

$$\begin{aligned}\mathbb{L}(\theta) = \alpha_1(1 - cos(q, f_\theta([q, \bar{d}_{nr}])))\\ + \alpha_2 \frac{1}{|D_R|} \sum_{d_r \in D_R} (1 - cos(d_r, f_\theta([q, \bar{d}_{nr}])))\\ + \alpha_3 \frac{1}{|D_{NR}|} \sum_{d_{nr} \in D_{NR}} max(0, cos(d_{nr}, f_\theta([q, \bar{d}_{nr}])))\end{aligned}$$

where, $cos(x, y)$ is the cosine similarity function. $\alpha_1, \alpha_2, \alpha_3$ are hyper-parameters to control the effect of each objective in the loss function.

3.3.3 Training Methodology. For training, we perform a single iteration refinement of the query representation using the sampled negative feedback set, followed by a model parameter update. We use warmup steps to improve the training performance. We perform a small scale Bayesian hyper-parameter tuning on a held-out 100 samples from the MS-Marco dev set. Using the results of the tuning, we set $\alpha_1 = 4, \alpha_2 = 10, \alpha_3 = 4$, learning_rate= 0.00172, and warmup steps= 0.29×total training steps.

3.3.4 Inference Methodology. For inference, we try to simulate a scenario where user provides negative feedback iteratively to improve document ranking. We perform multiple iterations of query representation refinement as shown in Fig 1. After each refinement step, the passage list is re-ranked and we sample a new negative feedback set from the re-ranked passage list. At the initialization, we use the passage list generated from DPR. Furthermore, we do

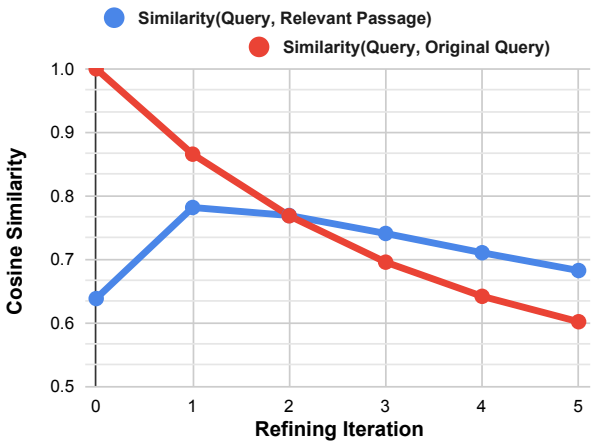

Figure 2: Average cosine similarity between the (original query, refined query) and (query, relevant passages) at each query refinement step. Step 0 represents no refinement.

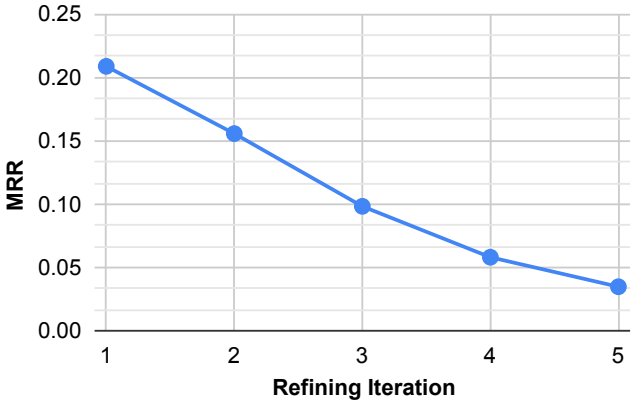

Figure 3: MRR performance at each refinement step.

not select the relevant passage in our negative feedback set as we assume that a real user will not mark it as negative.

3.3.5 Negative Sampling Strategy. We set the following parameters to find the negative feedback set during training and evaluation:

- **Negative Sampling Range**: It specifies the range of ranks from which we will sample the negatives. For example, the range 1-100 specifies that we will sample negative feedback set from the top-100 ranked documents.
- **Negative Sampling List Source**: It specifies which ranked list use for negative sampling. We perform experiments on two types of ranked lists – i) generated using DPR, ii) generated using BM25. The generation of these ranked lists is discussed in 3.1. During evaluation, we only use DPR generated ranked passage list.
- **Number of Negative Samples**: It specifies the number of negative samples to use for query refinement per iteration. It is to be noted that we only perform one iteration of refinement during training.

Furthermore, we do not use the relevant passages in the negative feedback set during training and evaluation.

4 EXPERIMENT RESULTS

We evaluate our models on MS-Marco Passage Re-ranking development set. The reported MRR are after a single step refinement. We present the effect of negative sampling range and list source on MRR in Table 1. Table 2 shows the effect of the number of negative samples on MRR performance.

4.1 Analysis

We observe that negative sampling using DPR sourced ranked list with sampling range 900-1000 and 20 negative samples performs the best. The results underperform the baseline. However, we can derive important insights from the results. We perform result analysis and explain the insights in detail below:

4.1.1 Effect of negative sampling list source. In Table 1 we observe that using negative samples from DPR generated ranked list consistently outperforms the BM25 generated ranked list for ranges 1-100 and 900-1000. We observe that BM25 generated list works better for higher-ranked ranges (1-25). We note that BM25 generated ranked list has a lower MRR (0.1932) than DPR ranked list (0.2316). Hence, DPR's ranked list contains more highly ranked documents. So, we hypothesize that as the DPR has more relevant documents in rank 1-25 in comparison to BM25, the model wrongfully learns to push away (make dissimilar to the refined query) the relevant documents along with the negative samples.

4.1.2 Effect of negative sampling range. Table 1 shows that negative sampling from low ranked documents gives better results. This is because pushing away negative documents sampled from low ranks are less likely to negatively affect the highly ranked relevant documents.

4.1.3 Effect of number of negative samples. Table 2 shows that the performance increases with the increase in the number of negative samples. We believe it happens because the model has more information and can refine the query better to make it dissimilar to a higher number of non-relevant documents.

4.1.4 Effect of number of refining iterations. Fig. 3 shows that the performance consistently decreases with the increasing number of refining iterations. In Fig. 2, we can observe that the average cosine similarity between the query and the relevant passage increases after first refinement and then consistently decreases in the subsequent iterations. We also observe that the refined query becomes increasingly dissimilar to the original query with more iterations. We believe that this performance deterioration is because after few iterations the refined query in order to be dissimilar to the negative samples becomes increasingly dissimilar to the relevant passages as well. This is one of the limitations of the system as it is not able to iteratively improve the passage re-ranking. This can be caused due to the fact that we only perform single refinement step during training. We can explore iterative refinement during the training phase in future work.

Model	Negative Sampling Range	Negative Sampling List Source	MRR
DPR (baseline)	-	-	**0.2316**
Ours	0-25	DPR	0.1891
Ours	0-100	DPR	0.1987
Ours	900-1000	DPR	**0.2063**
Ours	0-25	BM25	0.1987
Ours	0-100	BM25	0.1963
Ours	900-1000	BM25	0.2047

Table 1: Reported MRR on MS-Marco Passage Re-ranking dev set. Negative Sampling Rank Range specifies the rank range of documents from which the negative feedback set was sampled. Negative Sampling List Source specifies the method used to generate the ranked list for negative sampling. Number of negative samples=5.

Number of Negative Samples	MRR
1	0.1996
3	0.2039
10	0.20544
20	**0.20865**

Table 2: Effect of number of negative samples on the MRR. Negative Sampling Range=900-1000 and Negative Sampling List Source=DPR.

Subset of queries with relevant passage in range	Baseline MRR (DPR)	Post-refinement MRR
1-10	0.4815	0.4088 ↓
1-50	0.3208	0.2856 ↓
100-1000	0.00495	0.00917 ↑
500-1000	0.00157	0.002605 ↑
800-1000	0.00112	0.001172 ↑

Table 3: MRR Performance before and after refinement on query subsets. Subsets of MS-Marco dev set based on the rank of relevant document in the initial DPR-ranked list.

4.1.5 Effect of query refinement on high and low performing queries. We divide the DPR ranked list into 5 subsets shown in Table 3. The subsets are based on the rank of relevant passage of a query. For example, 1-10 subset contains queries whose relevant documents are ranked between 1 to 10. We do this to identify the queries on which our model underperforms and outperforms. In Table 3 observe that our model underperforms on high performing query subsets (1-10, 1-50) and outperforms on low performing query subsets (100-1000, 500-100, 800-1000). This means that our refiner model is able to improve the performance of queries which were previously low performing but it decreases the performance of previously high performing queries. This analysis helps us localize the possible cause of performance loss. Our query refinement model is wrongfully pushing away the highly ranked relevant documents along with the non-relevant documents. One potential way to alleviate it could be only to perform re-ranking of documents with ranks more than 100. It would keep the highly ranked documents intact and, at the same time, improve the ranks of low-ranked relevant documents.

5 CONCLUSION

In this work, we look at two facets of dense passage retrieval, i.e., how negative sampling should be done and whether we can use a small model to do iterative query refinement using the negative feedback from a user. For the first part, we identify that sampling negative documents from the lower ranks can help boost performance when compared to sampling documents from the higher ranks. This occurs potentially due existence of more relevant documents in the higher ranking, using which as negative samples might hurt the query representation. Secondly, we observe that this negative feedback gives us the maximal performance gain for queries where the relevant passages don't exist in the highly ranked list of documents and can help us improve on the low-performing queries.

We also see that the small models can be used for query refinement at test time, but they cannot do multi-step query refinement due to the limited training procedure. This possibly requires us to define our training objective as a multiple-step decision problem, where the query refinement model takes care of not destroying the representation to be away from the original query and relevant docs. Future work can use Reinforcement Learning because RL addresses this exact problem. We also identify the potential of using our refinement model for pseudo-relevance feedback.

6 EXTRA CREDIT

In this section we present our case for extra credit for the project. We accomplish the following:

- Propose a simple yet novel paradigm for using negative relevance feedback for dense query refinement.
- Perform a thorough set of experiments to identify different factors that can affect the performance of the model.
- Perform a detailed error analysis to localize the causes of our system's performance loss.
- Provide potential ways to address the model's current limitations.
- Solve various engineering issues such as:
  - Due to the presence of large number of dense passage representations (~7000×1000 for training) that won't be able to fit in memory, we implement lazy loading of cached representation into memory from the hard disk. However, this makes the inference slow because we need to lazily load 1000 representations per query. So, we implement parallelization to speedup the loading process.
  - Generating and efficiently storing of the cached representations on hard disk.

REFERENCES

[1] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. https://doi.org/10.48550/ARXIV.1611.09268

[2] Keping Bi, Qingyao Ai, and W. Bruce Croft. 2021. Asking Clarifying Questions Based on Negative Feedback in Conversational Search. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. 157–166. https://doi.org/10.1145/3471158.3472232 arXiv:2107.05760 [cs]

[3] Pierre Erbacher, Ludovic Denoyer, and Laure Soulier. 2022. Interactive Query Clarification and Refinement via User Simulation. arXiv:2205.15918 [cs]

[4] Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. 2021. Complementing Lexical Retrieval with Semantic Residual Embedding. https://doi.org/10.48550/arXiv.2004.13969 arXiv:2004.13969 [cs]

[5] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. https://doi.org/10.48550/arXiv.2104.06967 arXiv:2104.06967 [cs]

[6] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. https://doi.org/10.48550/arXiv.2004.04906 arXiv:2004.04906 [cs]

[7] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. https://doi.org/10.48550/arXiv.2004.12832 arXiv:2004.12832 [cs]

[8] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. https://doi.org/10.48550/arXiv.1906.00300 arXiv:1906.00300 [cs]

[9] Canjia Li, Yingfei Sun, Ben He, Le Wang, Kai Hui, Andrew Yates, Le Sun, and Jungang Xu. 2018. NPRF: A Neural Pseudo Relevance Feedback Framework for Ad-hoc Information Retrieval. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4482–4491. https://doi.org/10.18653/v1/D18-1478

[10] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2020. Distilling Dense Representations for Ranking Using Tightly-Coupled Teachers. https://doi.org/10.48550/arXiv.2010.11386 arXiv:2010.11386 [cs]

[11] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York.

[12] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. (Nov. 2016).

[13] J Rocchio. 1965. RELEVANCE FEEDBACK IN INFORMATION RETRIEVAL. (1965), 18.

[14] Vikram Singh and Ajay Singh. 2018. Learn-As-You-Go: Feedback-Driven Result Ranking and Query Refinement for Interactive Data Exploration. *Procedia Computer Science* 125 (Jan. 2018), 550–559. https://doi.org/10.1016/j.procs.2017.12.071

[15] Mujeen Sung, Jungsoo Park, Jaewoo Kang, Danqi Chen, and Jinhyuk Lee. 2022. Refining Query Representations for Dense Retrieval at Test Time. https://doi.org/10.48550/arXiv.2205.12680 arXiv:2205.12680 [cs]

[16] Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021. Pseudo-Relevance Feedback for Multiple Representation Dense Retrieval. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. 297–306. https://doi.org/10.1145/3471158.3472250 arXiv:2106.11251 [cs]

[17] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. https://doi.org/10.48550/arXiv.2007.00808 arXiv:2007.00808 [cs]

[18] HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving Query Representations for Dense Retrieval with Pseudo Relevance Feedback. arXiv:2108.13454 [cs]

[19] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. Learning To Retrieve: How to Train a Dense Retrieval Model Effectively and Efficiently. https://doi.org/10.48550/arXiv.2010.10469 arXiv:2010.10469 [cs]

[20] Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A Survey of Active Learning for Natural Language Processing. https://doi.org/10.48550/arXiv.2210.10109 arXiv:2210.10109 [cs]