Noisy Pair Corrector for Dense Retrieval

Anonymous ACL submission

Abstract

Most dense retrieval models contain an implicit assumption: the training query-document pairs are exactly matched. Since it is expensive to annotate the corpus manually, most training pairs in real-world applications are automatically collected, which inevitably introduces mismatched-pair noise. In this paper, we explore an interesting and challenging problem in dense retrieval, how to train an effective model with mismatched-pair noise. To solve this problem, we propose Noisy Pair Corrector (NPC), which consists of a detection module and a correction module. The detection module estimates noise pairs by calculating the perplexity between the annotated positive and easy negative documents. The correction module provides a soft supervised signal via an exponential moving average (EMA) model. We conduct experiments on text-retrieval benchmarks Natural Question and TriviaQA, code-search benchmarks StaQC and SO-DS. Experimental results show that NPC achieves excellent performance in handling both synthetic and realistic noise.

Correctly Matched Pair

Query: Check that variable is a lambda function

Code:

```python
def is_lambda_function(obj):
    return isinstance(obj,types.LambdaType)\
        and obj.__name__ == "<lambda>"
```

Mismatched Noise Pair

Query: Sending a Dictionary using Sockets in Python

Code:

```python
import json
#data serialized
data_string = json.dumps(data)
#data loaded
data_loaded = json.loads(data)
```

Figure 1: Two examples from StaQC training set. In the bottom example, the given code is mismatched with the query, since it can not answer the query.

1 Introduction

With the advancements in pre-trained language models (Devlin et al., 2019; Liu et al., 2019), dense retrieval has developed rapidly in recent years. It is essential to many applications including search engine (Brickley et al., 2019), open-domain question answering (Karpukhin et al., 2020a), and code intelligence (Guo et al., 2021). A typical dense retrieval model maps both queries and documents into a low-dimensional vector space, and measures the relevance between them by the similarity between their respective representations (Shen et al., 2014). During training, the model utilizes query-document pairs as labeled training data (Xiong et al., 2021) and samples negative documents for each pair. Then the model learns to minimize the contrastive loss for obtaining a good representation ability (Zhang et al., 2022b; Qu et al., 2021).

Recent studies on dense retrieval have achieved promising results with hard negative mining (Xiong et al., 2021), pretraining (Gao and Callan, 2021a), distillation (Yang and Seo, 2020), and adversarial training (Zhang et al., 2022a). All methods contain an implicit assumption: each query is exactly aligned with the given positive documents in the training set. However, this assumption is hard to satisfy in real applications. Especially when the corpus is automatically collected from the internet, it is inevitable that mismatched pairs are mixed in the training data. As shown in Fig. 1, the examples are from StaQC benchmark (Yao et al., 2018), which is automatically collected from StackOverflow. The document, i.e., code solution, can not answer the query but is incorrectly annotated as a positive document. Such noisy pairs are widely present in automatically constructed datasets, which will limit the performance of dense retrievers.

One related work is Noisy Label which mainly focuses on the classification task (Wang et al., 2019; Bai et al., 2021; Han et al., 2020). An important difference is that, dense retrieval adopts a ranking object for training which aims to push the sim-

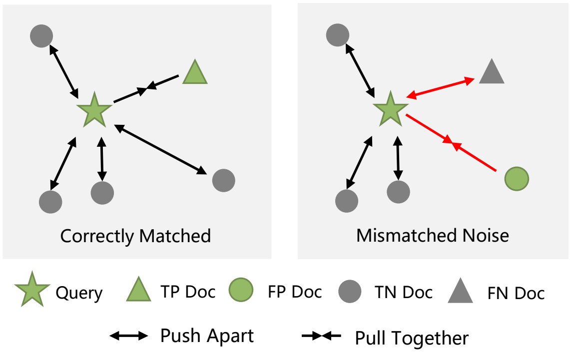


Figure 2: Effect of matched & mismatched pair for training. Green objects refer to annotated pairs, while pentagram and triangle are actually aligned pairs. In the left case, retrieval models are required to push the query with true-positive document (TP Doc) together and pull the query with true-negative documents (TN Doc) apart. In the right case, the retrieval models are misled by the mismatched data pair, where the false-positive document (FP Doc) and the false-negative document (FN Doc) are wrongly pulled and pushed, respectively.

ilarity between queries and positive documents greater than the negative documents. As shown in Fig. 2, the mismatched-pair noise will mislead the retriever to update in opposite direction. Some previous works focus on denoising false negatives, e.g., RocketQA filters the false negatives with a cross-encoder (Qu et al., 2021); AR2 adopts an adversarial framework to mitigate the effects of false negatives (Zhang et al., 2022a). So far, the mismatched-pair noise (false positive problem) in dense retrieval has not been well studied.

Based on these observations, we propose Noisy pair corrector (NPC) framework to solve the false-positive problem. NPC consists of noise detection and correction modules. At each epoch, the detection module estimates whether a query-document pair is mismatched by the perplexity between the annotated document and easy negative documents. Then the correction module provides a soft supervised signal for both estimated noisy data and clean data via an exponential moving average (EMA) model. Both modules are plug-and-play, which means NPC is a general training paradigm that can be easily applied to almost all retrieval models.

The contributions of this paper are as follows: 1) We reveal a long-neglected problem in dense retrieval, i.e., mismatched-pair noise, which is ubiquitous in the real world. 2) To address this problem, we propose a simple yet effective method for training dense retrieval models with mismatched-pair noise. 3) Extensive experiments on four datasets verify the effectiveness of our method against synthetic and realistic noise. Our method achieves new state-of-the-art performance on realistic-noisy dataset StaQC.

2 Preliminary

Before describing our model in detail, we first introduce the basic elements of dense retrieval, including problem definition, model architecture, and model training.

Given a query q, and a document collection $\mathbb{D}$, dense retrieval aims to find document d^+ relevant to q from $\mathbb{D}$. The training set consists of a collection of query-document pairs, donated as $C = \{(q_1, d_1^+), ..., (q_N, d_N^+)\}$, where N is the data size. Typical dense retrieval models adopt a dual encoder architecture to map queries and documents into a dense representation space. Then the relevance score $f(q, d)$ of query q and document d can be calculated with their dense representations:

$$f_\theta(q, d) = sim\left(E(q; \theta), E(d; \theta)\right), \qquad (1)$$

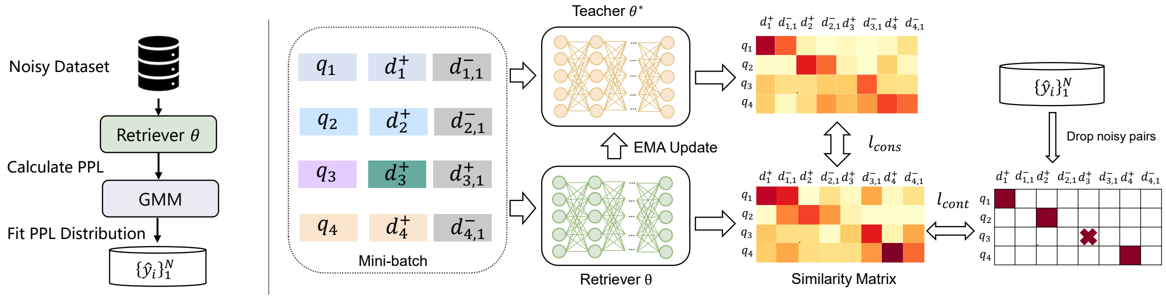
where $E(\cdot; \theta)$ denotes the encoder module parameterized with θ, and sim is the similarity function, e.g., euclidean distance, cosine distance, inner-product. Based on the embeddings, existing methods generally utilize ANN technique (Johnson et al., 2019) for efficient search.

For training dense retrievers, the contrastive loss is widely applied (Karpukhin et al., 2020a; Zhang et al., 2022b). Specifically, for each training pair $(q_i, d_i) \in C$, we sample m negative irrelevant documents $\{d_{i,1}^-, ..., d_{i,m}^-\}$ from document collection $\mathbb{D}$. To push the similarity of positive pairs higher than negative pairs, the retriever θ tends to minimize the loss function :

$$L_{cont} = -\log \frac{e^{\tau f_\theta(q_i, d_i)}}{e^{\tau f_\theta(q_i, d_i)} + \sum_{j=1}^{m} e^{\tau f_\theta(q_i, d_{i,j}^-)}}, \qquad (2)$$

where τ is a hyper-parameter to control the temperature. Previous work (Xiong et al., 2021) has verified the effectiveness of negative sampling strategy. The two most common strategies are "In-Batch Negative" and "Hard Negative" (Karpukhin et al., 2020a; Qu et al., 2021).

The above training paradigm assumes that the query-document pairs in training set C are correctly aligned. We argue that this assumption is difficult to satisfy. Since most training data in real-world applications are collected automatically without



(a) Noise Detection (b) Noise Correction

Figure 3: Overview of noise detection and noise correction. (a) Procedure of Noise Detection. At each epoch, we first calculate the perplexity of all training query-document pairs using the retriever θ; next fit the perplexity distribution with Gaussian Mixture Model to get the correctly matched probability of each pair; finally estimate the flag set $\{\hat{y}_i\}_{i=1}^N$ by setting the threshold. (b) Framework of Noise Correction. Given a batch of data pairs, where $d_{i,1}^-$ is the hard negative of q_i and $\{q_3, d_3^+\}$ is the estimated noisy pair, the retriever θ and teacher θ^* compute similarity matrices S_θ and S_{θ^*} for all queries and documents, respectively. The retriever learns to minimize (1) L_{cont}: the negative likelihood probability of true positive documents; (2) L_{cons}: the KL divergence between S_θ and the rectified soft label S_{θ^*} after normalization.

manual inspection, which will unavoidably contain some mismatched pairs.

3 Method

We propose NPC framework to learn retrievers with mismatched-pair noise. As shown in Fig. 3, NPC consists of two parts: (a) the noise detection module as described in Sec. 3.1, and (b) the noise correction module as described in Sec. 3.2.

3.1 Noise Detection

The noise detection module is meant to detect mismatched pairs in the training set. Previous works have shown that neural networks tend to first learn clean samples and then gradually fit noisy samples (Arazo et al., 2019; Arpit et al., 2017). Motivated by this, we hypothesize that: dense retrievers will first learn to distinguish correctly matched pairs from easy negatives, and then gradually overfit the mismatched pairs. Therefore, we determine whether a training pair is mismatched by the perplexity between the annotated document and easy negative documents.

Specifically, given a retriever θ and an uncertain pair (q_i, d_i), we calculate the perplexity as follows:

$$PPL_{(q_i,d_i,\theta)} = -\log \frac{e^{\tau f_\theta(q_i,d_i)}}{e^{\tau f_\theta(q_i,d_i)} + \sum_{j=1}^{m} e^{\tau f_\theta(q_i,d_{i,j}^-)}}, \quad (3)$$

where τ is a hyper-parameter, $d_{i,j}^-$ is the negative document randomly sampled from the document collection $\mathbb{D}$. Note that $d_{i,j}^-$ is a random easy negative, not a hard negative. We discuss this further in Appendix C. In practice, we adopt "In-Batch Negative" strategy for efficiency.

After obtaining the perplexity of each pair, we need an automated method to divide the noise and clean data. Motivated by Li et al. (2019), we fit the perplexity distribution over all training pairs by a two-component Gaussian Mixture Model (GMM):

$$p\left(PPL \mid \theta\right) = \sum_{k=1}^{K} \pi_k \phi\left(PPL \mid k\right), \quad (4)$$

where π_k and $\phi\left(PPL \mid k\right)$ are the mixture coefficient and the probability density of the k-th component, respectively. We optimize the GMM with the Expectation-Maximization algorithm (Dempster et al., 1977).

Based on the above hypothesis, we treat training pairs with higher PPL as noise and those with lower PPL as clean data. So the estimated clean flag can be calculated as follows:

$$\hat{y}_i = \mathbb{I}\left(p(\kappa \mid PPL_{(q_i,d_i,\theta)}) > \lambda\right), \quad (5)$$

where $\hat{y}_i \in \{1, 0\}$ denotes whether we estimate the pair (q_i, d_i) to be correctly matched or not, κ is the GMM component with the lower mean, λ is the threshold. $p(\kappa \mid PPL_{(q_i,d_i,\theta)})$ is the poster probability over the component κ, which can be intuitively understood as the correctly annotated confidence. We set λ to 0.5 in all experiments.

3.2 Noise Correction

Next, we will introduce how to reduce the interference of noise pairs after obtaining the estimated flag set $\{\hat{y}_i\}_{i=1}^{N}$. One quick fix is to discard the noise data directly, which is sub-optimal since it wastes the query data in noisy pairs. Motivated by semi-supervised methods (Tarvainen and Valpola, 2017), we adopt a self-ensemble teacher to provide rectified soft labels for noisy pairs. The teacher is an exponential moving average (EMA) of the retriever, and the retriever is trained with a weight-averaged consistency target on noisy data.

Specifically, given a retriever θ, the teacher θ^* is updated with an exponential moving average strategy as follows:

$$\theta_t^* = \alpha \theta_{t-1}^{\star} + (1 - \alpha)\theta_t, \tag{6}$$

where α is a momentum coefficient. Only the parameters θ are updated by back-propagation.

For a query q_i and the candidate document set D_{q_i}, where $D_{q_i} = \{d_{i,j}\}_{j=1}^{m}$ could consist of annotated documents, hard negatives and in-batch negatives, we first get teacher's and retriever's similarity scores, respectively. Then, the retriever θ is expected to keep consistent with its smooth teacher θ^*. To achieve this goal, we update the retriever θ by minimizing the KL divergence between the student's distribution and the teacher's distribution.

To be concrete, the similarity scores between q_i and D_{q_i} are normalized into the following distributions:

$$p_\phi(d_{i,j}|q_i; D_{q_i}) = \frac{e^{\tau f_\phi(q_i, d_{i,j})}}{\sum_{j=1}^{m} e^{\tau f_\phi(q_i, d_{i,j})}}, \phi \in \{\theta, \theta^*\}, \tag{7}$$

Then, the consistency loss L_{cons} can be written as:

$$L_{cons} = KL(p_\theta(.|q_i; D_{q_i}), p_{\theta^*}(.|q_i; D_{q_i})), \tag{8}$$

where $KL(\cdot)$ is the KL divergence, $p_\theta(.|q_i; D_{q_i})$ and $p_{\theta^*}(.|q_i; D_{q_i})$ denote the conditional probabilities of candidate documents D_{q_i} by the retriever θ and the teacher θ^*, respectively.

For the estimated noisy pair, the teacher corrects the supervised signal into a soft label. For the estimated clean pair, we calculate the contrastive loss and consistency loss. So the overall loss is formalized:

$$L = \hat{y}_i L_{cont} + L_{cons}, \tag{9}$$

where $\hat{y}_i \in \{1, 0\}$ is estimated by the noise detection module.

Algorithm 1 Noisy Pair Corrector (NPC)

Require: Retriever θ; Noisy Training dataset C.
Warmup the retriever θ on noisy dataset C by optimizing Eq.2;
Initial EMA model θ^* with θ;
for $i = 1 : num\_epoch$ **do**
  Calculate PPL of training pairs with random negatives using Eq.3;
  Fit PPL distribution with GMM;
  Get the estimated flag set $\{\hat{y}_i\}$ using Eq.5;
  for $i = 1 : num\_batch$ **do**
    Sample negatives with "In-Batch Negative" or "Hard Negative" strategy;
    Calculate rectified soft labels with EMA model θ^*;
    Train θ by optimizing Eq.9;
    Update EMA model θ^* using Eq.6;
  end for
end for

3.3 Overall Procedure

NPC is a general training framework that can be easily applied to almost all retrieval methods. Under the classical training process of dense retrieval, we add the noise detection module before training each epoch and the noise correction module during training. The detail is presented in Algorithm 1.

4 Experiments

4.1 Datasets

To verify the effectiveness of NPC in robust dense retrieval, we conduct experiments on four commonly-used benchmarks, including Natural Questions (Kwiatkowski et al., 2019), Trivia QA (Joshi et al., 2017), StaQC (Yao et al., 2018) and SO-DS (Heyman and Van Cutsem, 2020).

StaQC (Stack Overflow Question-Code pairs)is a large dataset that collects real query-code pairs from Stack Overflow[1]. The dataset has been widely used on code summarization (Peddamail et al., 2018) and code search (Heyman and Van Cutsem, 2020). SO-DS mines query-code pairs from the most upvoted Stack Overflow posts, mainly focuses on the data science domain. Following previous works (Heyman and Van Cutsem, 2020; Li et al., 2022), we resort to Recall of top-k (R@k) and Mean Reciprocal Rank (MRR) as the evaluation metric. StaQC and SO-DS are mined automatically

[1] https://stackoverflow.com/

Methods	StaQC			SO-DS		
	R@3	R@10	MRR	R@3	R@10	MRR
$BM25_{desc}$ (Heyman and Van Cutsem, 2020)	8.0	13.3	7.5	23.8	32.3	21.6
NBOW (Heyman and Van Cutsem, 2020)	10.9	16.6	9.5	27.7	38.0	24.7
USE (Heyman and Van Cutsem, 2020)	12.8	20.3	11.7	33.3	48.5	30.4
CodeBERT (Feng et al., 2020)	-	-	23.4	-	-	23.1
GraphCodeBERT (Guo et al., 2021)	-	-	24.1	-	-	25.2
CodeRetriever (In-Batch Negative) (Li et al., 2022)	-	-	25.5	-	-	27.1
CodeRetriever (Hard Negative) (Li et al., 2022)	-	-	24.6	-	-	31.8
UniXcoder (In-Batch Negative) (Guo et al., 2022)	29.98	47.47	28.04	31.90	51.21	28.29
UniXcoder (Hard Negative) (Guo et al., 2022)	31.18	48.38	28.63	33.42	53.37	29.97
NPC (In-Batch Negative)	33.07	50.35	30.39	35.58	54.54	30.96
NPC (Hard Negative)	**34.38**	**52.20**	**31.36**	**38.00**	**56.51**	**32.49**

Table 1: Retrieval performance on StaQC and SO-DS, which are realistic-noisy datasets. The results of the first block are borrowed from published papers (Heyman and Van Cutsem, 2020; Li et al., 2022). If the results are not provided, we mark them as "-".

without human annotation. Therefore, there are numerous mismatched pairs in training data.

Natural Questions (NQ) collects real queries from the Google search engine. Each question is paired with an answer span and golden passages from the Wikipedia pages. Trivia QA (TQ) is a reading comprehension corpus authored by trivia enthusiasts. In NQ and TQ, the goal of the retrieval stage is to find positive passages given queries from a large collection. Following Karpukhin et al. (2020a), we report Recall of top-k (R@k) as the evaluation metric. As NQ and TQ are well annotated by humans, we simulate the mismatched-pair noise with reference to the setting in the noisy classification task (Natarajan et al., 2013). Specifically, we randomly select a specific percentage of training queries and pair random documents to them.

4.2 Implementation Details

NPC is a general training paradigm that can be directly applied to almost all retrieval models. For StaQC and SO-DS, we adopt UniXcoder (Guo et al., 2022) as our backbone, which is the SoTA model for code representation. Following Guo et al. (2022), we adopt the cosine distance as similarity function and set temperature λ to 20. We update model parameters using the Adam optimizer and perform early stopping on the development set. The learning rate, batch size, warmup epoch, and training epoch are set to 2e-5, 256, 5, and 10, respectively. In the "Hard Negative" setting, we adopt the same strategy as Li et al. (2022). For a fair comparison, we implement UniXcoder with the same hyperparameters.

For NQ and TQ, we adopt BERT (Devlin et al., 2019) as our initial model. Following Karpukhin et al. (2020a), we adopt inner-product as the similarity function and set temperature λ to 1. The max sequence length is 16 for query and 128 for passage. The learning rate, batch size, warmup epoch, and training epoch are set to 2e-5, 512, 10, and 40, respectively. We adopt "BM25 Negative" and "Hard Negative" strategies as described in the DPR toolkit [2]. For a fair comparison, we implement DPR (Karpukhin et al., 2020a) with the same hyperparameters.

All the experiments are run on 8 NVIDIA Tesla A100 GPUs. The implementation code of NPC is based on Huggingface (Wolf et al., 2020).

4.3 Results

Results on StaQC and SO-DS: Table 1 shows the results on the realistic-noisy datasets StaQC and SO-DS. Both datasets contain a large number of real noise pairs. The first block shows the results of previous SoTA methods. $BM25_{desc}$ is a traditional sparse retriever based on the exact term matching of queries and code descriptions. NBOW is an unsupervised retriever that leverages pretrained word embedding of queries and code descriptions for retrieval. USE is a simple dense retriever based transformer. CodeBERT, GraphCodeBERT are pretrained models for code understanding using large-scale code corpus. CodeRetriever is a pretrained model dedicated to code retrieval, which is pretrained with unimodal and bimodal

[2] https://github.com/facebookresearch/DPR

Noisy	Methods	Natural Questions				Trivia QA			
		R@1	R@5	R@20	R@100	R@1	R@5	R@20	R@100
0	BM25∗	-	-	59.1	73.7	-	-	66.9	76.7
	DPR∗	-	-	78.4	85.4	-	-	79.4	85.0
	DPR (BM25 Negative)	45.02	66.95	79.61	86.08	53.14	71.31	79.79	85.19
	NPC (BM25 Negative)	45.55	68.22	80.20	86.62	52.37	70.91	79.43	84.86
	DPR (Hard Negative)	51.88	73.56	82.96	87.74	**56.58**	73.10	**80.85**	**85.74**
	NPC (Hard Negative)	**51.94**	**73.64**	**83.08**	**88.11**	56.36	**73.22**	80.74	85.68
20	DPR (BM25 Negative)	27.07	47.79	63.36	75.69	35.73	52.88	64.05	74.16
	DPR-C (BM25 Negative)	43.69	66.62	79.07	86.12	52.10	70.52	79.05	85.08
	NPC (BM25 Negative)	45.22	68.42	79.76	86.56	52.34	70.22	79.10	84.86
	DPR (Hard Negative)	37.61	60.73	71.68	79.56	43.39	60.67	70.34	77.88
	DPR-C (Hard Negative)	51.66	72.40	81.50	87.80	55.35	72.36	80.33	85.34
	NPC (Hard Negative)	**51.85**	**73.06**	**82.47**	**87.80**	**56.03**	**72.54**	**80.59**	**85.58**
50	DPR (BM25 Negative)	16.12	33.88	49.70	63.38	20.09	34.63	47.42	61.04
	DPR-C (BM25 Negative)	41.29	65.21	78.48	85.70	49.61	68.81	78.00	84.23
	NPC (BM25 Negative)	42.87	65.65	78.37	85.76	50.80	68.98	78.21	84.43
	DPR (Hard Negative)	23.87	42.34	55.12	67.06	28.47	45.12	56.88	67.62
	DPR-C (Hard Negative)	**48.87**	70.52	**81.44**	87.17	53.07	**70.36**	79.02	84.69
	NPC (Hard Negative)	48.81	**70.60**	81.17	**87.20**	**53.09**	70.27	**79.31**	**84.96**

Table 2: Retrieval performance on Natural Questions and Trivia QA under the noise ratio of 0%, 20%, and 50%, respectively. The results of BM25∗ and DPR∗ are borrowed from Karpukhin et al. (2020a). If the results are not provided, we mark them as "-".

contrastive learning on a large-scale corpus. The second block shows the results of UniXcoder with two negative sampling strategies. UniXcoder is also a pretrained model that utilizes multi-modal data, including code, comment, and AST, for better code representation. The results are implemented by ourselves for a fair comparison with NPC. The bottom block shows the results of NPC using two negative sampling strategies.

From the results, we can see that our proposed NPC consistently performs better than the evaluated models across all metrics. Compared with the strong baseline UniXcoder which ignores the mismatched-pair problem, NPC achieves a significant improvement with both "in-batch negative" and "hard negative" sampling strategies. It indicates that the mismatched-pair problem greatly limits the performance of dense retrieval models, and NPC, a general training paradigm, can mitigate this negative effect.

Results on NQ and TQ: Table 2 shows the results on the synthetic-noisy datasets NQ and TQ under the noise ratio of 0%, 20%, and 50%. We compare NPC with BM25 (Yang et al., 2017) and DPR (Karpukhin et al., 2020a). BM25 is an unsupervised sparse retriever that is not affected by noisy data. DPR (Karpukhin et al., 2020a) is a widely used method for training dense retrievers. We implement NPC and DPR using two negative sampling strategies. Besides, we evaluate DPR on clean datasets by discarding the synthetic-noisy pairs, denoted by DPR-C. DPR-C is a strong baseline that is not affected by mismatched pairs.

We can observe that (1) With the increase of the noise ratio, DPR shows severe performance degradation. When the noise rate is 50%, the performance of supervised DPR is lower than unsupervised BM25. (2) Under the noise-free setting, NPC achieves competitive results compared to DPR, even though NPC is designed to combat mismatched-pair noise. (3) When the training data contains noisy pairs, NPC outperforms the DPR method by a large margin, with only a slight performance drop when the noise increases. Even comparing DPR-C, which is trained on clean data, NPC still achieves competitive results.

4.4 Analysis

In this section, we conduct a set of detailed experiments on analyzing the proposed NPC training framework to help understand its pros and cons.

Ablations of Noise Detection and Noise Correction: To get a better insight into NPC, we conduct ablation studies on the realistic-noisy dataset StaQC and the synthetic-noisy dataset NQ under the noise ratio of 50%. The result are shown in

Methods			NQ				StaQC			
De	Co	HN	R@1	R@5	R@20	R@100	R@1	R@3	R@5	MRR
-	-	-	16.84	33.06	48.22	62.31	18.08	31.09	47.94	27.93
-	✓	-	21.66	40.83	55.90	69.33	18.51	31.01	48.98	28.34
✓	-	-	39.08	62.18	75.19	83.31	20.05	32.71	51.14	30.09
✓	✓	-	42.57	65.47	77.50	84.79	20.70	33.55	52.71	30.66
-	-	✓	23.46	42.42	54.63	65.54	18.66	31.74	48.63	28.64
-	✓	✓	25.42	46.07	58.63	69.06	19.35	32.09	49.71	29.21
✓	-	✓	44.55	66.49	77.59	85.03	20.93	33.55	51.52	30.86
✓	✓	✓	**50.07**	**69.93**	**80.07**	**85.89**	**21.93**	**34.51**	**52.87**	**31.91**

Table 3: Ablation studies on StaQC dev set and NQ dev set under noise ratio of 50%.

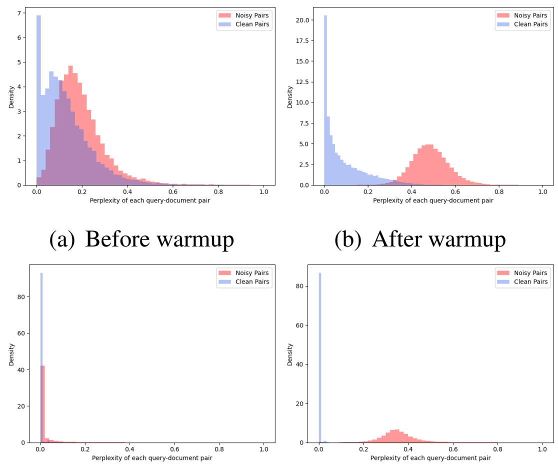
Setting	R@1	R@5	R@20	R@100
n=5	50.03	69.64	80.17	85.76
n=10	50.07	69.93	80.07	85.89
n=20	38.09	60.31	72.00	80.07
n=40	32.98	55.89	68.50	77.67

Table 4: Performance of NPC on NQ dev set with different warmup epoch number n.

Table 3. "De" and "Co" refer to noise detection and noise correction, respectively. "HN" indicates whether to perform "Hard Negative" strategy. For both synthetic noise and realistic noise, we can see that the noise detection module brings a significant gain, no matter which negative sampling strategy is used. Correction also enhances the robustness of the retriever since it provides rectified soft labels which can lead the model output to be smoother. The results show that combining the two obtains better performance compared with only using the detection module or correction module.

Impact of Warmup Epoch: According to the foregoing, NPC first warms up the retriever on the noisy dataset for initialization. In table 4, we show the performance of NPC with different warmup epoch number n. In this experiment, we adopt "Hard Negative" sampling strategy. We observe the performance degradation when increasing n from 5 to 30. According to the memorization effect of neural networks, we believe that warming up too long can cause the retriever overfits noisy pairs. Even if iterative detection is used in NPC, it is difficult to eliminate this effect.

Impact of Iterative Detection: In the training of NPC, we perform iterative noise detection every epoch. A straightforward approach is to detect the noise only once after warmup and fix the estimated flag set $\{\hat{y}_i\}$. To study the effectiveness of



(a) Before warmup (b) After warmup

(c) DPR (d) NPC

Figure 4: Perplexity distribution of training pairs under different settings.

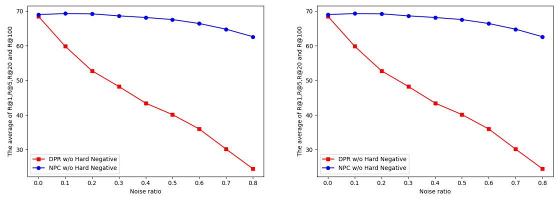
Setting	R@1	R@5	R@20	R@100
NPC	50.07	69.93	80.07	85.89
-w/o iterative detection	47.29	68.39	78.79	85.38
-ppl with HN	42.81	65.06	75.22	83.09

Table 5: Ablation studies of iterative noise detection and perplexity variants

iterative detection, we conducted an ablation study. The results are shown in Table 5. We can see that the model performance degrades after removing iterative detection.

Ablations of PPL: We distinguish noise pairs according to the perplexity between the annotated positive document and easy negatives. When calculating the perplexity, "Hard Negative" will cause trouble for detection. We construct ablation experiments to verify this, and the results are shown in Table 5. We can see that the perplexity with "Hard Negative" results in performance degradation.

Visualization of Perplexity Distribution: In Fig. 4, we illustrate the perplexity distribution of



(a) w/o Hard Negative (b) w/ Hard Negative

Figure 5: Retrieval performance of DPR and NPC on NQ dev set under different noise ratios.

training pairs before and after warmup, after training with DPR, and after training with NPC. The experiment is on NQ under the noise ratio of 50%. We can see that the perplexity of most noisy pairs is larger than the clean pairs after warmup, which verifies our hypothesis in Sec. 3.1. Comparing Fig. 4(c) and Fig. 4(d), we find that the retriever trained with DPR will overfit the noise pairs. However, NPC enables the retriever to correctly distinguish clean and noisy pairs because it avoids the dominant effect of noise during network optimization.

Visualization of Generalizability Fig. 5 shows the performance of DPR and NPC under the noise ratio ranging from 0% to 80%. We can see that as the noise ratio increases, the performance degradation of DPR is much larger than that of NPC, which demonstrates the generalizability of NPC.

5 Related Work

5.1 Dense Retrieval

Dense retrieval has shown better performance than traditional sparse retrieval methods (Lee et al., 2019; Karpukhin et al., 2020a). The studies of dense retrieval can be divided into two categories, (1) unsupervised pre-training to get better initialization (2) more effective fine-tuning on labeled data. In the first category, Some researchers focus on how to generate contrastive pairs automatically from a large unsupervised corpus (Lee et al., 2019; Chang et al., 2019; Ma et al., 2022; Li et al., 2022). Another line of research enforces the model to produce an information-rich CLS representation (Gao and Callan, 2021a,b; Lu et al., 2021). As for effective fine-tuning strategies, recent studies show that negative sampling techniques are critical to the performance of dense retrievers. DPR (Karpukhin et al., 2020b) adopts in-batch negatives and BM25 negatives; ANCE (Xiong et al., 2021), RocketQA (Qu et al., 2021), and AR2 (Zhang et al., 2022a) improve the hard negative sampling by iterative replacement, denoising, and adversarial framework, respectively. Several works distill knowledge from ranker to retriever (Izacard and Grave, 2020; Yang and Seo, 2020; Ren et al., 2021; Zeng et al., 2022).

Although the above methods have achieved promising results, they are highly dependent on correctly matched data, which is difficult to satisfy in real scenes. When the corpus is automatically mined, some mismatched pairs will inevitably be mixed in the training set. Previous works about denoising dense retrieval mainly focus on the false-negative problem (Qu et al., 2021; Zhang et al., 2022a), while the mismatched-pair noise problem has seldom been considered.

5.2 Denoising Techniques

Label noise is a common problem in real-world applications. Numerous methods have been proposed to solve this problem, and almost all of them focus on the classification task (Han et al., 2020). Some works design robust loss functions to learn models under label noise (Ghosh et al., 2017; Ma et al., 2020). Another line of work aims to identify noise from the training set with the memorization effect of neural networks (Arazo et al., 2019; Han et al., 2018; Bai et al., 2021), i.e., the deep neural network always learns clean samples before fitting noisy samples (Arpit et al., 2017).

The studies mentioned above mainly focus on classification. This paper studies the mismatched noise problem in dense retrieval, i.e., the mismatched errors in paired data rather than the errors in category annotations, which is more complex to handle. Different from classifiers the training target of dense retrievers aims to bring representations of positive pairs closer together and negative pairs further apart. It is challenging to adopt denoising methods in classification tasks directly.

6 Conclusion

This paper explores a neglected problem in dense retrieval, i.e., mismatched-pair noise. To solve this problem, we propose NPC, which iteratively detects noisy pairs per epoch and then provides rectified soft labels via an EMA model. We conduct experiments on four benchmarks. Experimental results show the excellent performance of NPC in handling synthetic and realistic mismatched-pair noise. We believe this work points out the long-neglected problems in dense retrieval and has great practical value.

Limitations

This work mainly focuses on training the dense retrieval models with mismatched noise. There may be two possible limitations in our study.

1) Due to the limited computing infrastructure, we only verified the robustness performance of NPC based on the classical retriever training framework. We leave experiments to combine NPC with more effective retriever training methods such as distillation (Ren et al., 2021), AR2 (Zhang et al., 2022a), as future work.

2) Mismatched-pair noise may also exist in other tasks, such as recommender systems. In future work, we will consider extending NPC to more tasks.

References

Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. 2019. Unsupervised label noise modeling and loss correction. In *ICML*.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *ICML*.

Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. 2021. Understanding and improving early stopping for learning with noisy labels. In *NIPS*.

Dan Brickley, Matthew Burgess, and Natasha Noy. 2019. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *WWW*.

Wei-Cheng Chang, X Yu Felix, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2019. Pre-training tasks for embedding-based large-scale retrieval. In *International Conference on Learning Representations*.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. Codebert: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*.

Luyu Gao and Jamie Callan. 2021a. Is your language model ready for dense representation fine-tuning? *arXiv preprint arXiv:2104.08253*.

Luyu Gao and Jamie Callan. 2021b. Unsupervised corpus aware language model pre-training for dense passage retrieval. *arXiv preprint arXiv:2108.05540*.

Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. 2017. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*.

Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. 2022. Unixcoder: Unified cross-modal pre-training for code representation. In *ACL*.

Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, Michele Tufano, Shao Kun Deng, Colin B. Clement, Dawn Drain, Neel Sundaresan, Jian Yin, Daxin Jiang, and Ming Zhou. 2021. Graphcodebert: Pre-training code representations with data flow. In *International Conference on Learning Representations*.

Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. 2020. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NIPS*.

Geert Heyman and Tom Van Cutsem. 2020. Neural code search revisited: Enhancing code snippet retrieval through natural language intent. *arXiv preprint arXiv:2008.12193*.

Gautier Izacard and Edouard Grave. 2020. Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020a. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020b. Dense passage retrieval for open-domain question answering. In *EMNLP*.

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

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

Junnan Li, Richard Socher, and Steven CH Hoi. 2019. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*.

Xiaonan Li, Yeyun Gong, Yelong Shen, Xipeng Qiu, Hang Zhang, Bolun Yao, Weizhen Qi, Daxin Jiang, Weizhu Chen, and Nan Duan. 2022. Coderetriever: Unimodal and bimodal contrastive learning. *arXiv preprint arXiv:2201.10866*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Shuqi Lu, Di He, Chenyan Xiong, Guolin Ke, Waleed Malik, Zhicheng Dou, Paul Bennett, Tie-Yan Liu, and Arnold Overwijk. 2021. Less is more: Pretrain a strong siamese encoder for dense text retrieval using a weak decoder. In *Empirical Methods in Natural Language Processing*.

Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. 2020. Normalized loss functions for deep learning with noisy labels. In *ICML*.

Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, and Xueqi Cheng. 2022. Pre-train a discriminative text encoder for dense retrieval via contrastive span prediction. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. 2013. Learning with noisy labels. *Advances in neural information processing systems*, 26.

Jayavardhan Reddy Peddamail, Ziyu Yao, Zhen Wang, and Huan Sun. 2018. A comprehensive study of staqc for deep code summarization. In *KDD*.

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In *NAACL-HLT*.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*, pages 373–374.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30.

Hao Wang, Bing Liu, Chaozhuo Li, Yan Yang, and Tianrui Li. 2019. Learning with noisy labels for sentence-level sentiment classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP*.

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

Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*.

Sohee Yang and Minjoon Seo. 2020. Is retriever merely an approximator of reader? *arXiv preprint arXiv:2010.10999*.

Ziyu Yao, Daniel S Weld, Wei-Peng Chen, and Huan Sun. 2018. Staqc: A systematically mined question-code dataset from stack overflow. In *WWW*.

Hansi Zeng, Hamed Zamani, and Vishwa Vinay. 2022. Curriculum learning for dense retrieval distillation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*.

Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2022a. Adversarial retriever-ranker for dense text retrieval. In *International Conference on Learning Representations*.

Shunyu Zhang, Yaobo Liang, Ming Gong, Daxin Jiang, and Nan Duan. 2022b. Multi-view document representation learning for open-domain dense retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

A Qualitative Analysis

Table 7 lists some mismatched pairs detected by NPC in StaQC training set. We can see that these mismatched pairs are almost irrelevant and can be correctly detected by NPC. These examples are not well aligned, mainly due to the low-quality answers of the open community (cases 2 and 4), inappropriate data preprocessing in the collection phase (cases 2 and 3), and other reasons. It is well known that collecting and cleaning training data is expensive and complex work. Automatically constructed datasets in real-world applications often contain such mismatched-pair noise. Our method can mitigate the impact caused by such noise during training.

B Statistics of Datasets

Dataset	Train	Dev	Test	Corpus size
StaQC	203.7K	2.6K	2.7K	14.6K
SO-DS	12.1K	0.9K	1.1K	12.1K
NQ	79.2K	8,8K	3.6K	21 M
TQ	78.8K	8.8k	11.3K	21 M

Table 6: The statistics of datasets. Corpus size means the size of document corpus for evaluation.

C Discussion about Perplexity

We calculate the perplexity between the annotated document and easy negative documents during noise detection. We emphasize that the negative documents are randomly selected from the document collection $\mathbb{D}$. Unlike Eq. 2, we can not adopt "Hard Negative" sampling strategy when calculating the perplexity. Although hard negatives are important to train a strong dense retriever, they will cause trouble during noise detection. Specifically, it is expected that the retriever is confused only between false positive and negative documents and can confidently distinguish true positive and negative documents. But if we adopt "Hard Negative" when calculating the perplexity, the retriever will also be confused between true positive and hard negative documents, which will affect noise detection. We construct ablation experiments to verify this, and the results are shown in Table 5.

	Question	Code
1	Split words in a nested list into letters	» [list(l[0]) for l in mylist]
2	Dictionary in python problem	» s = problem.getSuccessors(getStartState())
3	Find the Common first name from Django Auth user Model	» import operator
4	Find all text files not containing some text string	» lst = [1,2,4,6,3,8,0,5] » for n in lst[:]: »» if n % 2 == 0: »»» lst.remove(n) » lst

Table 7: Some noisy pairs detected by NPC in StaQC training set.