Generative Dense Retrieval: Memory Can Be a Burden

Peiwen Yuan[1]*, Xinglin Wang[1]*, Shaoxiong Feng[2], Boyuan Pan[2], Yiwei Li[1]
Heda Wang[2], Xupeng Miao[3], Kan Li[1]†
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]Xiaohongshu Inc [3]Carnegie Mellon University
{peiwenyuan,wangxinglin,liyiwei,likan}@bit.edu.cn {xupeng}@cmu.edu
{shaoxiongfeng2023,whd.thu}@gmail.com {panboyuan}@xiaohongshu.com

Abstract

Generative Retrieval (GR), autoregressively decoding relevant document identifiers given a query, has been shown to perform well under the setting of small-scale corpora. By memorizing the document corpus with model parameters, GR implicitly achieves deep interaction between query and document. However, such a memorizing mechanism faces three drawbacks: (1) Poor memory accuracy for fine-grained features of documents; (2) Memory confusion gets worse as the corpus size increases; (3) Huge memory update costs for new documents. To alleviate these problems, we propose the **G**enerative **D**ense **R**etrieval (GDR) paradigm. Specifically, GDR first uses the limited memory volume to achieve inter-cluster matching from query to relevant document clusters. Memorizing-free matching mechanism from Dense Retrieval (DR) is then introduced to conduct fine-grained intra-cluster matching from clusters to relevant documents. The coarse-to-fine process maximizes the advantages of GR's deep interaction and DR's scalability. Besides, we design a cluster identifier constructing strategy to facilitate corpus memory and a cluster-adaptive negative sampling strategy to enhance the intra-cluster mapping ability. Empirical results show that GDR obtains an average of 3.0 R@100 improvement on NQ dataset under multiple settings and has better scalability[1].

1 Introduction

Text retrieval (Karpukhin et al., 2020; Zhao et al., 2022) is an essential stage for search engines (Brickley et al., 2019), question-answering systems (Liu et al., 2020) and dialog systems (Chen et al., 2017). Traditional retrieval methods include *sparse retrieval* (SR) and *dense retrieval* (DR). SR (Robertson and Zaragoza, 2009; Robertson and Walker, 1997) relies on the assumption that queries and relevant documents have a high degree of word overlap. However, such methods suffer from the zero-recall phenomenon when there is a lexical mismatch between queries and documents. DR (Ren et al., 2021; Zhang et al., 2022a) alleviates this issue by training dual-encoders for semantic matching instead of lexical matching, which brings a high hit rate. Nevertheless, most queries are semantically related to multiple documents that may not be close to each other in semantic space. Thus it is challenging to use a single query representation to recall all the relevant documents with matching mechanism (Zhang et al., 2022b).

Recently, *generative retrieval* (GR) (Zhou et al., 2022; Bevilacqua et al., 2022), which utilizes a language model to memorize document features and autoregressively decodes the identifiers of relevant documents given a query, is considered a promising paradigm. The model is served as a memory bank for candidate documents, and the memorizing process implicitly implements the deep interaction between queries and documents by attention mechanism, which has been proven to be effective in the small-scale corpus settings (Wang et al., 2022; Sun et al., 2023). Also, beam search, a diversity-promoting decoding strategy, is beneficial for the model to find relevant documents from multiple directions and thus can recall more relevant documents than DR (Tay et al., 2022).

However, after empirically comparing the performance of typical GR model NCI (Wang et al., 2022) and DR model AR2 (Zhang et al., 2022a), we found that the memorizing mechanism brings three problems: (1) *Poor memory accuracy for fine-grained features of documents*. We calculated the error rate of each position when decoding document identifiers (see Table 1). Compared with AR2, NCI performs well on the former part of the decoding process while poorly on the latter part.

*Equal contribution.
†Corresponding author.
[1]Our code have been released on https://github.com/ypw0102/GDR.

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics Volume 1: Long Papers, pages 2835–2845

Model	Error Rate of the ith Position					
	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}
NCI	**1.09**	**1.75**	**1.86**	5.77	14.91	12.66
AR2	1.19	1.77	2.11	**5.44**	**8.03**	**3.05**

Table 1: Error rate (%) on the ith position when decoding document identifiers. See Appendix A.1 for the detailed calculation method.

Settings	NCI	AR2
	R@1/100	R@1/100
334K-334K	14.7 - / 65.5 -	21.2 - / 69.0 -
1M-1M	11.1↓3.6 / 54.5↓11.0	20.3↓0.9 / 66.2↓2.8
1M-334K	12.3↓2.4 / 59.8↓5.7	21.2 - / 69.0 -

Table 2: Performance of NCI and AR2 on NQ validation set with different settings. For setting $x - y$, x denotes the training corpus size and y denotes the candidate corpus size during the inference phase. AR2 is only trained on the training set, thus is independent of x.

We argue that NCI aims to map queries to relevant document identifiers instead of real document content, which results in its lack of accurate memory for fine-grained document features. (2) *Memory confusion gets worse as the corpus size increases*. As shown in Table 2, we scaled both training and candidate corpus sizes from 334K to 1M and found that NCI decreased by 11.0 on R@100 while AR2 only decreased by 2.8. NCI trained on 1M training corpus is further tested on 334K candidate corpus. The results indicate that the burden of memorizing more documents causes 5.7 R@100 drop. (3) *Huge memory update costs for new documents*. When new documents come, the document cluster tree needs to be updated, and the model needs to be re-trained to re-memorize all the documents. Otherwise, the outdated mapping relationship, i.e., query to document identifiers and document identifiers to documents, will significantly degrade the retrieval performance (see Table 6).

Based on the above analysis, a natural idea is to employ memorizing-free matching mechanism from DR to alleviate the burden faced by the memorizing mechanism. However, it is challenging to realize complementary advantages of both mechanisms while ensuring retrieval efficiency. To this end, we propose a coarse-to-fine retrieval paradigm **G**enerative **D**ense **R**etrieval (GDR). Concretely, memorizing mechanism and matching mechanism are successively applied to achieve coarse-grained inter-cluster (query $\rightarrow$ document clusters) and fine-grained intra-cluster (document clusters $\rightarrow$ documents) matching. A shared query encoder is used to generate query representations that apply both mechanisms, thereby improving retrieval efficiency. We also explore the strategy of constructing a memory-friendly document cluster tree, including distinguishable document clusters and controllable cluster amounts, so as to further alleviate memory burden. Moreover, a cluster-adaptive negative sampling strategy is proposed to enhance the intra-cluster matching ability of GDR.

Overall, the coarse-to-fine process maintains the advantages of the memorizing mechanism while alleviating its drawbacks by introducing matching mechanism. Unlike GR, the limited memory volume of GDR is only responsible for memorizing the coarse-grained features of corpora. The fine-grained features of documents are extracted into dense representations, which promotes accurate intra-cluster mapping. When new documents come, GDR achieves scalability by adding documents to relevant clusters and extracting their dense representations by a document encoder, without reconstructing document identifiers and retraining the model.

Our contributions are summarized as follows:

- We revisit generative retrieval (GR) with a detailed empirical study, and discuss three key drawbacks that limit GR performance.
- We propose generative dense retrieval (GDR), a coarse-to-fine retrieval paradigm, that exploits the limited memory volume more appropriately, enhances fine-grained feature memory, and improves model scalability.
- Comprehensive experiments demonstrate that GDR obtains higher recall scores than advanced SR, DR and GR methods. And the scalability of GDR is also significantly improved.

2 Related Work

Given queries, text retrieval task aims to find relevant documents from a large corpus. In this section, we introduce typical paradigms DR and GR that are most related to our work.

2.1 Dense Retrieval

DR (Karpukhin et al., 2020; Xiong et al., 2021; Ren et al., 2021; Zhang et al., 2022b,a; Zhao et al., 2022) is the most widely studied retrieval paradigm in recent years. A dual-encoder architecture (query-encoder and document-encoder) is commonly used to extract the dense semantic representations of queries and documents. The similarities between them are computed through simple operations (*e.g.,* inner product) in Euclidean space and ranked to recall the relevant documents. By extracting features and constructing indexes for matching, DR does not have to memorize the corpus and attains good scalability. However, the upper bound of DR is constrained due to the limited interaction between queries and candidate documents (Li et al., 2022). GDR inherits the matching mechanism from DR in the fine-grained mapping stage, and introduces deep interaction through memorizing mechanism in the coarse-grained mapping stage, thus achieving better recall performance.

2.2 Generative Retrieval

Recently, a new retrieval paradigm named GR, which adopts autoregressive model to generate relevant document identifiers, has drawn increasing attention. Cao et al. (2021) proposes to retrieve documents by generating titles. Tay et al. (2022) utilizes BERT (Devlin et al., 2019) combined with the *K*-means algorithm to generate identifiers with hierarchical information. Bevilacqua et al. (2022) leverages n-grams to serve as identifiers. Wang et al. (2022) enhances the model's memory of candidate documents through query generation. Mehta et al. (2022) proposes retraining model with generated queries of old documents when new documents are added to reduce forgetting. Sun et al. (2023) suggests training the model to learn to assign document identifiers. However, all of these methods require models to memorize the whole corpus and inevitably face the problems we have discussed above, for which we propose GDR.

3 Methodology

Our task is to retrieve a candidate document set $\mathcal{D}_c$ from a large corpus $\mathcal{D}_l$ ($|\mathcal{D}_l| >> |\mathcal{D}_c|$) for a given query q, with the objective of including as many documents d from $\mathcal{D}_q$ as possible, where $\mathcal{D}_q$ is the set of documents relevant to q. In this section, we introduce the proposed Generative Dense Retrieval (GDR) paradigm (see Figure 1). To realize complementary advantages of memorizing mechanism and matching mechanism, we need to consider the following issues:

3.1 Order of Applying Two Mechanisms

Based on Table 1 and Table 2, we found that the coarse-grained semantic mapping between query and documents attained lower error rates when applying memorizing mechanism (NCI), while feature extraction and matching mechanism (AR2) was better suited for handling fine-grained features of numerous documents. Thus, we consider utilizing the advantage of memorizing mechanism in deep interaction between query and corpus memory bank to recall relevant document clusters. Afterwards, we leverage the superiorities of memorizing-free matching mechanism in fine-grained representation extracting and better scalability characteristics to further retrieve the most relevant documents from the recalled clusters.

Inter-cluster Matching The classic Encoder-Decoder architecture is used to achieve the inter-cluster mapping $f_{inter} : q \to \text{CID}^{1:k}$, where CID denotes document cluster identifiers. Given query $q^{1:|q|}$, GDR first leverages Query Encoder E_Q to encode it into query embeddings $e_q^{1:|q|} \in \mathbb{R}^d$ and takes the embedding of <CLS> token as query representation r_q. Based on this, the probability of generating CID^i can be written as follows:

$$p(\text{CID}^i|e_q, r_q, \theta) = \prod_{j=1}^{|\text{CID}^i|} p(\text{CID}_j^i|e_q, r_q, \text{CID}_{<j}^i, \theta) \quad (1)$$

where θ is the parameters of Cluster Decoder D_C. We denote this probability as inter-cluster mapping score $S_{inter}(q, \text{CID}^i)$, which characterizes the matching between q and $\mathcal{D}_l$ under coarse-grained features. For a training pair (q, d^+), we use CrossEntropy loss to train GDR to achieve inter-cluster matching correctly:

$$\mathcal{L}_{Inter} = -\log p(\text{CID}(d^+)|E_Q(q), \theta_{D_C}) \,. \quad (2)$$

Following NCI, we use the encoder of T5-base (Brown et al., 2020) to initialize E_Q and randomly initialized PAWA decoder (see Wang et al. (2022) for details) as D_C.

Intra-cluster Matching To further achieve the intra-cluster mapping $f_{intra} : \text{CID}^{1:k} \to d^{1:k}$, GDR applies the matching mechnism of calculating representation similarity for retrieval. Specifically, GDR leverages the Document Encoder E_D trained in section 3.2 to extract the fine-grained features of candidate documents $d^{1:|\mathcal{D}_l|}$ into semantic representations $r_d^{1:|\mathcal{D}_l|} \in \mathbb{R}^d$ in prior. Then we pick out the d^i belonging to the recalled clusters $\text{CID}^{1:k}$ in the previous stage and calculate the intra-cluster mapping score between them and q as follows:

$$S_{intra}(q, d^i) = \text{Sigmoid}(\text{sim}(r_q, r_d^i)). \quad (3)$$

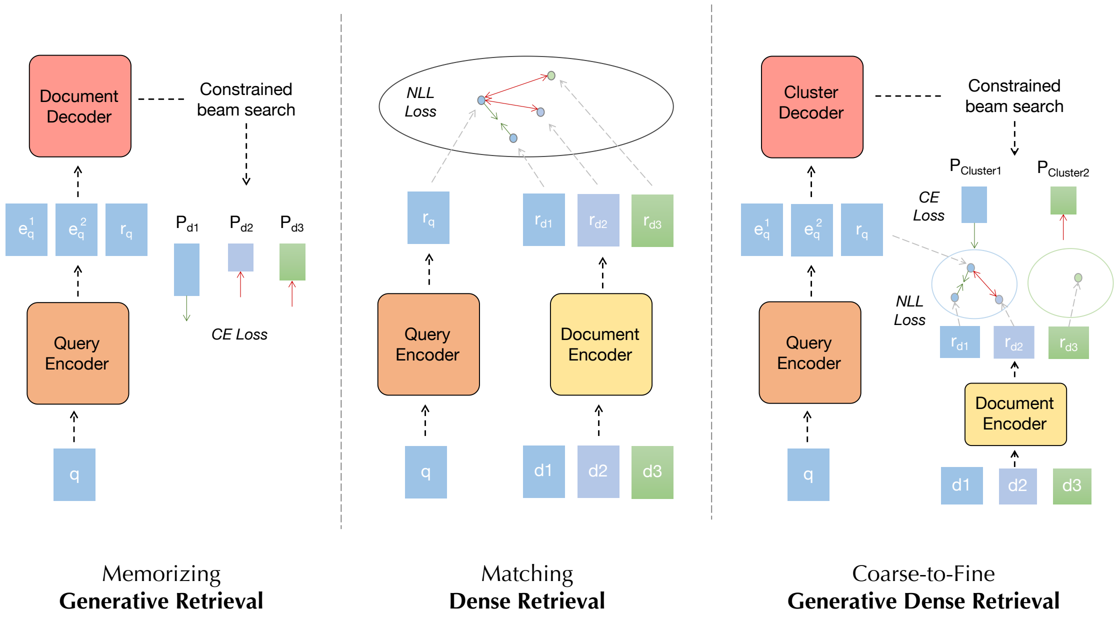

Figure 1: Illustration of Dense Retrieval, Generative Retrieval and Generative Dense Retrieval.

where sim(·) denotes the inner product function. The Sigmoid function is used to map S_{intra} into [0,1] to align with S_{inter}. NLL loss is used to train GDR for intra-cluster mapping ability:

$$\mathcal{L}_{Intra} = -\log \frac{e^{sim(q,d^+)}}{e^{sim(q,d^+)} + \sum_i^n e^{sim(q,d_i^-)}} \quad (4)$$

where d^+ and d^- refer to documents relevant and irrelevant to q respectively. On this basis, the overall mapping score of d^i is defined as:

$$S_{overall}(q, d^i) = S_{inter}(q, \text{CID}(d^i)) + \beta * S_{intra}(q, d^i) \quad (5)$$

where β is a hyperparameter which we set as 1 by default. In the end, we take the Top-k documents according to $S_{overall}$ as the final retrieval set $\mathcal{D}_c$.

3.2 Construction of Memory-friendly CIDs

Considering the limited memory volume of the model, we are supposed to construct memory-friendly CIDs to ease the mapping f_{intra}.

Ideally, we would like the CIDs corresponding to documents relevant to the same query to have similar prefixes. Such property can provide a mapping relationship between the query and CIDs with lower entropy, so as to alleviate the memorizing burden. What's more, the total number of document clusters should be determined by the memory volume (model size) rather than the size of $\mathcal{D}_l$ to avoid exceeding the memorizing volume. Based on these considerations, our strategy for generating CIDs is shown in Algorithm 1.

Algorithm 1 Generating document cluster identifiers (CIDs).

```
Require: Corpus d^{1:|D_l|}, Document Encoder E_D,
    Inter-cluster number k, Intra-cluster number c
Ensure: Document cluster identifiers CID^{1:|D_l|}
  Encode d^{1:|D_l|} with E_D to obtain document representations X^{1:|D_l|}
  function GENERATECIDS(X^{1:N})
      C^{1:k} ← Kmeans(X^{1:N})
      L ← ∅
      for i ← 1, k do
          L_current ← [i] * |C^i|
          if |C^i| ≥ c then
              L_rest ← GENERATECIDS(C^i)
          else
              L_rest ← [0] * |C^i|
          end if
          L_cluster ←Concat(L_current, L_rest)
          L ← L.Append(L_cluster)
      end for
      ReorderToOriginal(L, X^{1:N}, C^{1:k})
      Return L
  end function
  CID^{1:|D_l|} ← GENERATECIDS(X^{1:|D_l|})
```

To meet the first property, we finetuned ERNIE-2.0-base (Sun et al., 2020) model following Zhang et al. (2022a) on the training set [2] and then used the finetuned document encoder as E_D in Algorithm 1. Compared to previous studies (Tay et al., 2022;

[2]All experiments in this work were conducted on the Natural Questions dataset (Kwiatkowski et al., 2019)

Wang et al., 2022) using BERT (Devlin et al., 2019) as E_D, our strategy can fully leverage the knowledge in the training set. To analyse the qualities of CIDs generated with different E_D, we calculated the average prefix overlap O_{pre} of CIDs between the relevant documents for each query in the validation set S_{val} as follows:

$$O_{pre} = \frac{1}{|S_{val}|} \sum_{q \in S_{val}} \frac{1}{|\mathcal{D}_q|^2} \sum_{i=1}^{|\mathcal{D}_q|} \sum_{j=1}^{|\mathcal{D}_q|} o_{pre}(\mathrm{CID}_q^i, \mathrm{CID}_q^j)$$
$$o_{pre}(s_1, s_2) = |LCP(s_1, s_2)| / |s_1| \tag{6}$$

where CID_q^i is the cluster identifier of the i^{th} relevant document of q and $LCP(s_1, s_2)$ is the longest common prefix of string s_1 and s_2. The results show that the O_{pre} corresponding to the CIDs generated by our strategy (0.636) is significantly higher than the previous study (0.516), indicating that our CIDs is more distinguishable and can better meet the first property . To meet the second property, we consider adaptively changing c in Algorithm 1 to ensure the total number of clusters $|\mathrm{CID}|$ not to change with $\mathcal{D}_l$ as follows:

$$c = |\mathcal{D}_l| \;/\; Exp(|\mathrm{CID}|) \tag{7}$$

where $Exp(|\mathrm{CID}|)$ is the expected value of $|\mathrm{CID}|$ which we set as 5000 in our experiment for simplicity. Under different sizes of $\mathcal{D}_l$, the $|\mathrm{CID}|$ we obtained through this strategy is basically in the same order of magnitude (Appendix A.2), which meets the second properties.

3.3 Cluster-adaptive Negative Sampling

An important issue in calculating $\mathcal{L}_{Intra}$ is how to select d^- with effective training signals. Various negative sampling methods (e.g., static bm25-based sampling (Karpukhin et al., 2020), dynamic index-based sampling (Xiong et al., 2021)) have been proposed to pick up hard negatives. However, GDR needs to retrieve relevant documents within the candidate clusters instead of the entire corpus, which requires negative samples to offer more intra-cluster discriminative signals. To this end, we propose cluster-adaptive negative sampling strategy. For a training pair (q, d^+), we treat $d \in \mathrm{CID}(d^+)$ as intra-cluster negatives N_a and in-batch negatives (Henderson et al., 2017) as inter-cluster negatives N_r, and rewrite Eq. (4) as follows:

$$\mathcal{L}_{Intra} = -\log \frac{e^{\mathrm{sim}(q,d^+)}}{\gamma * \sum_{d \in N_a} e^{\mathrm{sim}(q,d)} + \sum_{d \in N_r} e^{\mathrm{sim}(q,d)}} \tag{8}$$

where γ is a hyperparameter we set as 2 to enhance intra-cluster discriminative training signals.

3.4 Training and Inference

Training Phase Given a corpus $\mathcal{D}_l$ and a training set $\mathcal{S}_{train} = \{(q^i, d^i) | i \in (1, ..., n)\}$, we use DocT5Query [3] to generate 5 pseudo queries through and randomly select 5 groups of 40 consecutive terms from the document as additional queries for each document. Compared with Wang et al. (2022) that augment each document with totally 26 queries, fewer augmented queries are required as GDR only needs to memorize coarse-grained semantics, thus saves training expenses. The augmented training set $\mathcal{S}_{aug}$ together with $\mathcal{S}_{train}$ are used to train GDR using the total loss:

$$\mathcal{L}_{GDR} = \mathcal{L}_{Inter} + \mathcal{L}_{Intra} \tag{9}$$

To accelerate the training process, we use E_D to calculate the representations of $\mathcal{D}_l$ in advance and freeze the parameters of E_D during training phase.

Inference Phase During inference, we first generate k relevant CIDs through beam search, and then retrieve the top-m documents with highest S_{intra} in each relevant cluster (m is the minimum value between the number of documents in the cluster and k). Finally, we reorder all these documents according to $S_{overall}$ to obtain the most relevant top-k documents. Following Tay et al. (2022), we pre-build a prefix tree to ensure only the valid CIDs can be generated. We conduct Approximate Nearest Neighbor Search (Li et al., 2020) in each cluster to accelerate the intra-cluster matching process.

4 Experiments

We empirically demonstrate the performance of GDR and effectiveness of various proposed strategies on text retrieval task in this section.[4] In the following, we will discuss the detailed experimental setups in 4.1, present empirical results in 4.2, verify the effectiveness of proposed modules in 4.3, and conduct specific analysis in 4.4, respectively.

4.1 Experimental settings

Datasets We choose classic text retrieval dataset Natural Questions [5] (NQ) (Kwiatkowski et al., 2019) for experiment, which consists of 58K

[3] https://github.com/castorini/docTTTTTquery

[4] we will release our code as soon as the paper is accepted

[5] We use the cleaned version of NQ downloaded from https://huggingface.co/Tevatron

(query, relevant passages) training pairs and 6K validation pairs along with 21M candidate passage corpus. Each query corresponds to an average of 7.5 relevant passages, which puts higher demands on the recall performance of the model. We gather all the relevant passages of queries included in NQ training and validation set, resulting in a 334K candidate passage corpus setting (NQ334K). We further build NQ1M, NQ2M, and NQ4M settings to evaluate the performance of GDR on larger corpus by adding the remaining passages from the full 21M corpus to NQ334K. For GDR, CIDs are generated separately for each dataset so as to prevent leakage of semantic information from larger candidate document corpus into smaller ones. GDR of different settings are trained on the training set together with corresponding augmented set, and evaluated on the validation set [6].

Evaluation metrics We use widely accepted metrics for text retrieval, including $\mathrm{R@}k$ (also denoted as $\mathrm{Recall@}k$) and $\mathrm{Acc@}k$, where $k \in \{20, 100\}$. Specifically, $\mathrm{R@}k$ calculates the proportion of relevant documents included in top-k retrieved candidates ($\#\,\mathrm{retr}_{q,k}$) among all the candidate relevant documents ($\#\,\mathrm{rel}_q$) (Eq. (10)), while $\mathrm{Acc@}k$ measures how often the correct document is hit by top-k retrieved candidates (Eq. (11)).

$$\mathrm{R@}k = \frac{1}{|\mathcal{S}_{val}|} \sum_{q \in \mathcal{S}_{val}} \frac{\#\,\mathrm{retr}_{q,k}}{\#\,\mathrm{rel}_q} \tag{10}$$

$$\mathrm{Acc@}k = \frac{1}{|\mathcal{S}_{val}|} \sum_{q \in \mathcal{S}_{val}} \mathbb{I}\left(\#\,\mathrm{retr}_{q,k} > 0\right) \tag{11}$$

Baselines We choose the following methods for detailed comparisons. BM25 (Anserini implementation (Yang et al., 2017)) is served as a strong **SR** baseline. As for **DR**, we select a strong baseline DPR [7] (Karpukhin et al., 2020) and state-of-the-art (SOTA) method AR2 [8] (Zhang et al., 2022a). As for **GR**, we select the SOTA method NCI [9] (Wang et al., 2022). To ensure the reliability of the experimental results, we reproduce all the baseline methods based on their official implementations.

Experimental details We implement GDR with python 3.8.12, PyTorch 1.10.0 and HuggingFace transformers 3.4.0. The learning rates are set as 2×10^{-4} for the Query Encoder and 1×10^{-4} for the Cluster Decoder with a batch size 256 per GPU. For inference, we apply the constraint beam search algorithm, and set the length penalty and the beam size as 0.8 and 100, respectively. All experiments are based on a cluster of NVIDIA A100 GPUs with 40GB memory. Each job takes 8 GPUs, resulting in a total batch size of 2048 (256×8). We train the GDR models for 60 epochs and pick the final checkpoint for evaluation.

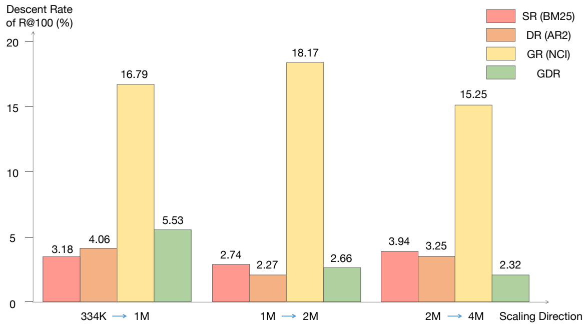

Figure 2: R@100 descent rate of different types of methods when scaling to larger corpus.

4.2 Main Results

Horizontal Comparison As shown in the Table 3, the performance of each method on $\mathrm{R@}k$ metrics is as follows: GDR (GDR-ours) > SR (BM25) > DR (AR2) > GR (NCI), while the ranking on $\mathrm{Acc@}k$ metrics is as follows: DR (AR2) > GDR (GDR-ours) > SR (BM25) > GR (NCI). Based on the characteristics of sparse lexical matching, SR can recall the majority of relevant documents ($2nd$ $\mathrm{R@}k$) when the query is accurate while may not even hit one target when there is a lexical mismatch ($3rd$ $\mathrm{Acc@}k$). On the contrary, DR can hit at least one relevant document in most situations by semantic representation matching ($1st$ $\mathrm{Acc@}k$). However, the semantic differences in relevant documents make it difficult to recall them all simultaneously ($3rd$ $\mathrm{R@}k$). GR (NCI) ranks last due to the difficulty in memorizing large-scale corpus we have discussed.

By conducting a coarse-to-fine retrieval process, GDR maximizes the advantages of memorizing mechanism in deep interaction and matching mechanism in fine-grained features discrimination, thus ranks $1st$ on $\mathrm{R@}k$ with an average of 3.0 improvement and $2nd$ on $\mathrm{Acc@}k$.

Scaling to Larger Corpus Memorizing mechanism has been proven to bring advanced retrieval

[6] The lack of relevant documents makes the test set inconvenient to partition different settings

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

[8] https://github.com/microsoft/AR2

[9] https://github.com/solidsea98/Neural-Corpus-Indexer-NCI

Paradigm	Method	NQ334K		NQ1M		NQ2M		NQ4M	
		Acc@20/100	R@20/100	Acc@20/100	R@20/100	Acc@20/100	R@20/100	Acc@20/100	R@20/100
SR	BM25	86.1 / 92.4	56.0 / 75.4	84.0 / 91.0	51.3 / 73.0	82.4 / 89.9	47.5 / 71.0	79.6 / 88.4	42.3 / 68.2
DR	DPR	93.9 / 97.3	49.8 / 60.2	91.5 / 96.3	46.7 / 56.6	90.4 / 95.5	45.2 / 54.9	88.4 / 94.6	42.9 / 52.8
	AR2	**96.3 / 98.6**	57.4 / 69.0	**94.9 / 98.0**	54.7 / 66.2	**94.3 / 97.7**	53.2 / 64.7	**93.4 / 97.2**	51.2 / 62.6
GR	NCI-bert	80.0 / 88.7	49.4 / 65.5	72.0 / 82.6	38.7 / 54.5	63.9 / 76.4	30.2 / 44.6	55.4 / 70.0	25.2 / 37.8
	NCI-ours	88.0 / 94.1	60.0 / 75.6	80.3 / 89.6	50.6 / 66.2	78.2 / 88.6	46.4 / 63.5	77.3 / 87.8	45.2 / 61.0
GDR	GDR-bert	87.5 / 91.2	59.3 / 71.2	84.8 / 88.8	54.8 / 66.0	83.3 / 88.0	51.9 / 64.8	82.1 / 87.7	49.7 / 63.8
	GDR-ours	91.1 / 95.3	**64.6 / 79.6**	88.2 / 93.6	**60.1 / 75.2**	87.4 / 92.8	**57.7 / 73.2**	87.0 / 92.2	**55.2 / 71.5**

Table 3: Experimental results on NQ document retrieval. The settings "-bert" and "-ours" denote using BERT and our finetuned E_D in section 3.2 to generate document embeddings for the generation of identifiers respectively. Bold numbers represent best performance. We run four random seeds and report the averaged result for each method.

performance under small corpus settings (Wang et al., 2022). However, when the corpus size that needs to be memorized exceeds the memory volume, it can instead become a burden. As shown in Figure 2, when the candidate corpus scaling to larger size, the descent rate of R@100 for both SR and DR keeps below 4.06%, while it astonishingly retains exceeding 15.25% for GR on all three scaling directions. As a comparison, GDR ensures the maximum utilization of memorizing mechanism by focusing memory content on fixed volume coarse-grained features of corpus to achieve inter-cluster matching. This strategy results in GDR achieving an average of 3.50% descent rate of R@100, which is almost the same as SR (3.29%) and DR (3.19%).

4.3 Ablation studies on Model training

To further understand how different paradigm options affect model performance, we conduct ablation experiments and discuss our findings below.

Cluster Identifiers We first analyse the influence of identifiers constructed with documents representations generated by different models. Specifically, the results are shown in Table 3, where "-bert" and "-ours" denotes using BERT and our finetuned model as E_D in Algorithm 1 respectively. Basically, both NCI and GDR trained with "-ours" perform significantly better than those trained with "-bert" across all the settings. The results empirically demonstrate that fully leveraging the knowledge in the training set to generate identifiers that characterizing a mapping from query to relevant documents with lower entropy can significantly release the memorizing burden thus leading to better retrieval performances. Considering that NCI has a heavier memory burden compared to GDR, this strategy has benefited NCI more (10.1 > 8.4 R@100 improvements on NQ334K).

Strategy	Acc@20	Acc@100	R@20	R@100
Random	87.1	91.4	60.8	76.0
BM25	90.2	94.6	63.1	78.5
Cluster-adaptive	**91.1**	**95.3**	**64.6**	**79.6**

Table 4: Comparison of the performance of GDR trained with different negative sample strategies on NQ334K dataset.

β	Acc@20	Acc@100	R@20	R@100
0	70.5	83.9	39.2	59.4
0.5	89.1	93.7	61.9	77.2
1	**91.1**	**95.3**	**64.6**	**79.6**
2	90.9	95.0	64.4	79.5
1e5	90.4	94.8	63.1	77.9

Table 5: Results of GDR with different β on NQ334K dataset.

Negative Sampling Strategy To verify the effectiveness of the proposed cluster-adaptive negative sample strategy, We evaluate the performance of GDR trained with different negative sampling strategies and summarize the results in Table 4. We notice that GDR trained with the cluster-adaptive strategy outperforms that with widely used BM25 strategy by 1.1 on R@100. This indicates that our proposed cluster-adaptive negative sampling strategy can indeed provide more intra-cluster discriminative training signals to strengthen the fine-grained matching ability.

4.4 Analysis

Combination of Mapping Scores We study the influence of different combination weights of S_{inter} and S_{intra} in Eq. (5) and choose the value of β from {0,0.5,1,2,1e5}. As the beta gradually increases (Table 5), the retrieval performance of GDR will experience a process of first increasing and then decreasing. Therefore, we take the best

$\mathcal{D}_l$	$\mathcal{S}_{val}$	NCI		GDR	
		Acc@100	R@100	Acc@100	R@100
Set A	Set A	90.7 -	71.2 -	94.9 -	77.7 -
All	Set A	80.7↓10.0	52.9↓18.3	93.4↓1.0	75.8↓1.9
All	Set B	56.5↓34.2	27.7↓43.5	86.6↓8.3	66.2↓11.5

Table 6: Comparison of scalability performance between NCI and GDR. Specifically, We divide the original NQ334K dataset into two parts: Set A (constructing identifiers and training on it) and Set B (served as new added dataset).

Method	Latency (ms)	Throughput (queries/s)	Index Refresh (mins)
BM25	56	22.8	2
AR2	35	589.0	5
NCI	232	6.3	-
GDR	195	7.2	7

Table 7: Efficiency analysis on NQ334K dataset with recall quantity as 100. NCI can not refresh indexes without retraining.

performing (β=1) as the default setting. When GDR only relies on S_{inter} for retrieval (β = 0), the ranking of documents within the same cluster will be the same, which will result in a significant performance degradation compared with the default setting. On the contrary, when GDR only relies on S_{intra} for retrieval (we set β = 1e5 to approximate this situation), the lack of matching information of coarse-grained semantic features will result in a decrease of 1.7 R@100. The above experimental results fully demonstrate the significance of S_{inter} and S_{intra} and the necessity of combining them.

Scalability of Model A common scenario in retrieval tasks is adding new documents to candidate corpus. To simulate this scenario, we split the NQ334K dataset into Set A and Set B, both of which contain half of the original training and validation set together with corresponding relevant documents. For both NCI and GDR, we first train and evaluate the model on Set A. After adding Set B to Set A, we further evaluate the model on validation subset of Set A and Set B respectively. As shown in Table 6, though NCI has already memorized the documents corresponding to Set A validation set, the situations where one document identifier corresponds to multiple documents caused by the new added documents led to a 18.3 R@100 drop. On the contrary, GDR only degraded 1.9 on R@100 thanks to the introduction of S_{intra}. When evaluating on Set B, NCI further significantly degraded 25.2 on R@100 as the model did not have a memory of documents corresponding to Set B validation set. As a comparison, GDR can quickly extract dense representations through E_D and assign cluster identifiers by searching for the nearest cluster representation in the semantic space for the added documents, so as to obtain inter-cluster and intra-cluster features. Although GDR also does not have a memory of added documents, its R@100 performance (66.2) still significantly surpassed NCI (27.7) on Set B.

Efficiency Analysis We use an NVIDIA A100-40G GPU to analyze the efficiency of AR2, NCI, and GDR. We use the Anserini implementation of BM25 and evaluate it on an Intel Xeon CPU. As shown in Table 7, BM25 and AR2 achieve fast retrieval by indexing the corpus in advance. Typical GR method NCI has lower efficiency due to the autoregressive generation of document identifiers with beam search. As a compromise, GDR uses autoregressive generation in inter-cluster matching and pre-indexes for retrieval in intra-cluster matching, thus achieves an efficiency that falls between DR and GR. We leave the research on improving the efficiency of GR and GDR for future work.

5 Conclusions

In this paper, we empirically demonstrate that the memorizing mechanism of Generative Retrieval (GR) brings deep interaction characteristics but also causes serious problems. To this end, we propose the Generative Dense Retrieval (GDR) paradigm, which subdivides the text retrieval task into inter-cluster and intra-cluster matching and achieves them by autoregressively generating cluster identifiers and calculating dense representation similarities respectively. GDR focuses the limited memory volume on the deep interaction between query and document cluster and conducts multi-directions decoding, thus maintaining the superiority of memorizing mechanism. Memorizing-free matching mechanism is further introduced to achieve intra-cluster mapping by fully leveraging fine-grained features of documents. Such a coarse-to-fine process can also bring better scalability, i.e., stable corpus expansion and low-cost document updates. We further propose a cluster identifier constructing strategy to release the memory burden and a cluster-adaptive negative sampling strategy to provide discriminative signals. Comprehensive experiments on the NQ dataset demonstrate the state-of-the-art R@k performance and better scalability of GDR.

Limitations

Despite the achievement of state-of-the-art $\text{R@}k$ performance and better scalability, the current implementation of GDR still suffers from the following limitations. Firstly, the inference speed of GDR needs to be further improved to be employed in real-time retrieval services. Secondly, GDR's performance on $\text{Acc@}k$ falls short compared to the state-of-the-art method (AR2 (Zhang et al., 2022a)). We suppose that this is because part of the Query Encoder's capacity is utilized to handle the inter-cluster matching task, thus affects the accuracy of GDR in intra-cluster mapping. Thirdly, due to the high training cost (70 hours on 8 NVIDIA A100 GPUs for NQ4M), the generalization of GDR on larger scale corpus has not been tested.

Ethics Statement

All of the datasets used in this study were publicly available, and no annotators were employed for data collection. We confirm that the datasets we used did not contain any harmful content and was consistent with their intended use (research). We have cited the datasets and relevant works used in this study.

Acknowledgments

This work is supported by Beijing Natural Science Foundation (No. 4222037, L181010).

References

Michele Bevilacqua, Giuseppe Ottaviano, Patrick S. H. Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. In *NeurIPS*.

Dan Brickley, Matthew Burgess, and Natasha F. Noy. 2019. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 1365–1375. ACM.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. *SIGKDD Explor.*, 19(2):25–35.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *CoRR*, abs/1705.00652.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

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.

Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Mingjie Li, Wenjie Zhang, and Xuemin Lin. 2020. Approximate nearest neighbor search on high dimensional data - experiments, analyses, and improvement. *IEEE Trans. Knowl. Data Eng.*, 32(8):1475–1488.

Zehan Li, Nan Yang, Liang Wang, and Furu Wei. 2022. Learning diverse document representations with deep query interactions for dense retrieval. *CoRR*, abs/2208.04232.

Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Daxin Jiang, Jiancheng Lv, and Nan Duan. 2020. Rikinet: Reading wikipedia pages for natural question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6762–6771. Association for Computational Linguistics.

Sanket Vaibhav Mehta, Jai Prakash Gupta, Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Jinfeng Rao, Marc Najork, Emma Strubell, and Donald Metzler. 2022. DSI++: updating transformer memory with new documents. *CoRR*, abs/2212.09744.

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, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 2825–2835. Association for Computational Linguistics.

Stephen E. Robertson and Steve Walker. 1997. On relevance weights with little relevance information. In *SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27-31, 1997, Philadelphia, PA, USA*, pages 16–24. ACM.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. 2023. Learning to tokenize for generative retrieval. *CoRR*, abs/2304.04171.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE 2.0: A continual pre-training framework for language understanding. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8968–8975. AAAI Press.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Prakash Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. 2022. Transformer memory as a differentiable search index. In *NeurIPS*.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. 2022. A neural corpus indexer for document retrieval. In *NeurIPS*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

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, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 1253–1256. ACM.

Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2022a. Adversarial retriever-ranker for dense text retrieval. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

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), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5990–6000. Association for Computational Linguistics.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2022. Dense text retrieval based on pretrained language models: A survey. *CoRR*, abs/2211.14876.

Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, Peitian Zhang, and Ji-Rong Wen. 2022. Ultron: An ultimate retriever on corpus with a model-based indexer. *CoRR*, abs/2208.09257.

Dataset	Cluster Counts
NQ334K	34337
NQ1M	33003
NQ2M	29000
NQ4M	28362

Table 8: Total number of CIDs included in datasets with different scale of candidate document corpus.

Method	Acc@20/100	R@20/100
BM25	95.7/98.6	70.5/93.2
AR2	97.9/99.3	71.8/92.8
NCI	90.4/95.7	71.6/92.3
GDR	96.8/98.9	74.6/95.0

Table 9: Experimental results on TriviaQA549K.

A Appendix

A.1 Calculation method of error rate

Considering that the AR2 (Zhang et al., 2022a) itself does not make predictions on identifiers, we select identifier corresponding to the predicted document as AR2's identifier prediction. We calculate the error rate of model's prediction on the ith position as follows: For each predicting document identifiers, we calculate the probability that, given its prefix up to the i-1th position belonging to a prefix of a relevant document identifier, the addition of the model's prediction for the ith position no longer belongs to any prefix of a relevant document identifier.

A.2 Magnitude of CIDs

We collect the total count of CIDs for datasets with different scales of candidate documents obtained through our proposed strategy introduced in section 3.2. As shown in Table 8, the results demonstrate that the proposed strategy can effectively control the total number of clusters, thus guarantee the memorizing volume of GDR. The reason why the magnitude of cluster counts in Table 8 (approximately 30000) is larger than the $Exp(|\text{CID}|)$ we set as 5000 is that, The constructed cluster tree is unbalanced, resulting in more clusters than the expected value. Our preliminary studies show that, setting $Exp(|\text{CID}|)$ in Algorithm 1 as 5000 can lead to a favorable budget between efficiency and performance.

A.3 Experiments on TriviaQA

We further verified the generalization of GDR on a subset of TriviaQA. We constructed the TriviaQA549K dataset following the procedure to construct NQ334K and compared GDR with other methods on it as shown in Table 9. The experimental results verified the good generalization of GDR.