Bootstrapped Pre-training with Dynamic Identifier Prediction for Generative Retrieval

Yubao Tang[1], Ruqing Zhang[1], Jiafeng Guo[1], Maarten de Rijke[2], Yixing Fan[1], Xueqi Cheng[1]
[1]CAS Key Lab of Network Data Science and Technology, ICT, CAS; University of Chinese Academy of Sciences
[2]University of Amsterdam
{tangyubao21b,zhangruqing,guojiafeng,fanyixing,cxq}@ict.ac.cn
m.derijke@uva.nl

Abstract

Generative retrieval uses differentiable search indexes to directly generate relevant document identifiers in response to a query. Recent studies have highlighted the potential of a strong generative retrieval model, trained with carefully crafted pre-training tasks, to enhance downstream retrieval tasks via fine-tuning. However, the full power of pre-training for generative retrieval remains underexploited due to its reliance on pre-defined static document identifiers, which may not align with evolving model parameters. In this work, we introduce BootRet, a bootstrapped pre-training method for generative retrieval that dynamically adjusts document identifiers during pre-training to accommodate the continuing memorization of the corpus. BootRet involves three key training phases: (i) initial identifier generation, (ii) pre-training via corpus indexing and relevance prediction tasks, and (iii) bootstrapping for identifier updates. To facilitate the pre-training phase, we further introduce noisy documents and pseudo-queries, generated by large language models, to resemble semantic connections in both indexing and retrieval tasks. Experimental results demonstrate that BootRet significantly outperforms existing pre-training generative retrieval baselines and performs well even in zero-shot settings.

Introduction

- **Document retrieval** aims to retrieve candidate documents from a huge document collection for a given query[1,2]
- **Dense retrieval** is the dominant implementation, which encodes the query and documents into dense embedding vectors to capture rich semantics [3,4]
- **Generative retrieval** employs a sequence-to-sequence (Seq2Seq) architecture to generate relevant document identifiers (docids) for queries[5,6]
  - **Indexing:** memorizing the entire corpus by associating each document with its docid
  - **Retrieval:** using the indexed corpus information to produce a ranked list of potentially relevant docids for a given query
- Using general language models, e.g., BART[7] and T5[8], as the base Seq2Seq model has become a popular choice in GR[9,10]
- Some work has designed pre-training objectives for GR.
  - Zhou et al. (2022)[11]: document pieces or pseudo-queries are used as input, and docids (e.g., product quantization code) are predicted as output with maximum likelihood estimation (MLE)
  - Chen et al. (2022)[12]: construct and learn pairs of pseudo-queries and docids (i.e., Wikipedia titles) from the corpus
- Applying **specialized pre-trained models** to GR **yields superior result**s compared to using general language models

Approach

BootRet: a general bootstrapped pre-training method for GR
Key idea: dynamically adjust docids in accordance with the evolving model parameters during pre-training

The human brain updates the organization of existing knowledge to better match updated goals or contents in learning[13]

Key steps:

1. Initial docid generation
2. Pre-training
   - Corpus indexing task（CI task）
     - Noisy document construction
       - synonym replacement
       - sentence removal
       - sentence shuffling
       - word masking
     - Semantic consistency loss $\sum_{i=1}^{N}\sum_{h=1}^{4} 1-\mathrm{sim}(\mathrm{Enc}(d_i),\mathrm{Enc}(\tilde{d}_i^h))$
     - Contrastive losses $-\sum_{i=1}^{N}\log\frac{\exp(P(id_i \mid d_i)/\tau)}{\sum_{j=1}^{N}\exp(P(id_j \mid d_i)/\tau)}$, $-\sum_{i=1}^{N}\sum_{h=1}^{4}\log\frac{\exp(P(id_i \mid \tilde{d}_i^h)/\tau)}{\sum_{j=1}^{N}\exp(P(id_j \mid \tilde{d}_i^h)/\tau)}$
   - Relevance prediction task (RP task)
     - Pseudo-query construction
     - Pre-training objective $-\sum_{i=1}^{N}\sum_{x=1}^{X}\log\frac{\exp(P(id_i \mid q_i^x)/\tau)}{\sum_{j=1}^{N}\exp(P(id_j \mid q_i^x)/\tau)}$
   - Joint learning

$$L_{Pre}(\mathcal{D},\tilde{\mathcal{D}},\mathcal{I}_{\mathcal{D}},\mathcal{Q},\mathcal{I}_{\mathcal{Q}};\theta^{t-1}) = \gamma L_{CI}(\cdot)+\rho L_{RP}(\cdot)+\underbrace{\lambda L_{ID}(\cdot)}_{-\sum_{i=1}^{|\mathcal{D}|}\log P(id_i \mid d_i)-\sum_{i=1}^{|\mathcal{D}|}\sum_{h=1}^{4}\log P(id_i \mid \tilde{d}_i^h)}+\underbrace{\lambda L_{RE}(\cdot)}_{-\sum_{i=1}^{|Q|}\log P(id_i \mid q_i)}$$

3. Enhanced bootstrapping
   - Docid update: Fixing θ^t , we use the encoder of θ^t to encode documents to update docids of the previous iteration I_D^{t-1} , to I_D^t
   - Retrain the model: To proceed to the next iteration, we retrain the model with I_D^t. After multiple iterations, we achieve continuous dynamic alignment and enhancement

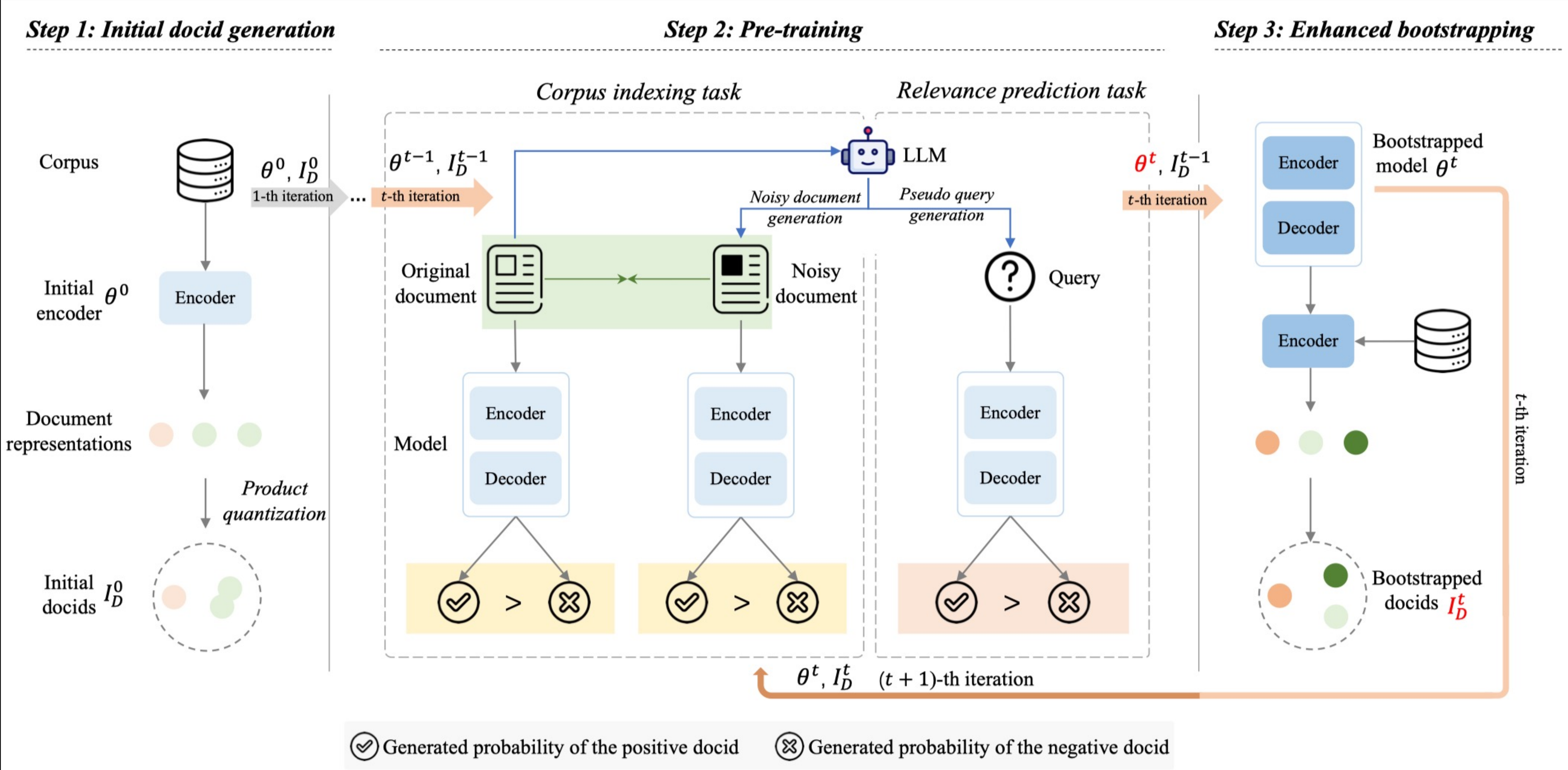


Figure 1. The bootstrapped pre-training pipeline of BootRet. (1) The initial docids I_D^0 are obtained with the initial model parameters θ^0. (2) To perform the t-th iteration, we design the corpus indexing task and relevance prediction task for pre-training. We construct noisy documents and pseudo-queries with a LLM, and design contrastive losses (the yellow and the orange rectangles) and a semantic consistency loss (the green rectangle) to learn the corpus and relevance information discriminatively. After pre-training, the model updates from θ^{t-1} to θ^t. (3) The bootstrapped θ^t is used to dynamically update the docids I_D^{t-1} to I_D^t, i.e., bootstrapped docids, which are further used in the next iteration. (Figure should be viewed in color).

Experimental settings

- **Pre-training corpus**
  - English Wikipedia[16]
  - MS MARCO Document Collection[17]
  - Sample 500K documents; Generate 4 noisy documents and 5 pseudo-queries, for each document (2.5M documents and 2.5M pseudo-queries for pre-training)
- **Downstream retrieval datasets**
  - MS MARCO Document Ranking dataset[17]: a subset of 300K documents (360K training queries, 6980 evaluation queries)
  - Natural Question (NQ)[18]: 228K documents (307K training queries, 7.8K test queries)
- **Baselines**
  - Sparse retrieval baselines: BM25[19] and DocT5Query[20]
  - Dense retrieval baselines: RepBERT[21], DPR[22], and ANCE[23]
  - Advanced GR baselines: DSI[6], GENRE[10], SEAL[9], DSI-QG[24], NCI[25], Ultron-PQ[11], Corpusbrain[12], GenRet, and NOVO[27]
- **Evaluation metrics**
  - Hits@K with K = { 1, 10 }; MRR@K with K = { 3, 20 }
- **Implementation details**
  - Pre-training:
    - noisy documents and pseudo-queries generation： LLaMA-13b[28]
    - Backbone: T5-base[8]
    - PQ: length 24; cluster 256; vector dimension 768 [12]
    - The max training step is 500K, with the first iteration occurring at step 100K, followed by iterations every 40K steps thereafter
  - Finetuning:
    - Use the pre-trained model obtained from the last iteration to generate docids
    - Models are further fine-tuned with document-docid pairs and labeled query-docid pairs with MLE[6]
    - Generate 10 pseudo-queries for each document to enhance training[24]

Experimental results

Table 1. Retrieval performance on MS 300k.

Method	MRR		Hits	
	@3	@20	@1	@10
BM25	22.57	26.67	24.78	40.73
DocT5query	27.38	29.63	30.13	46.93
RepBERT	31.47	33.68	33.16	55.83
DPR	34.84	36.79	36.52	58.68
ANCE	30.76	34.25	33.63	53.62
DSI	23.21	28.93	28.14	49.72
GENRE	31.12	33.49	33.18	53.56
SEAL	31.35	33.57	33.34	53.74
DSI-QG	33.64	35.81	34.96	58.62
NCI	33.86	36.20	35.02	59.21
Corpusbrain	34.72	37.25	36.14	60.32
Ultron-PQ	35.25	38.41	39.53	62.85
GenRet	37.26	40.53	41.68	64.92
NOVO	38.36	41.29	43.14	64.55
BootRet-Bs^{Wiki}	36.28*	39.25*	40.73*	63.78*
BootRet-Bs^{MS}	37.13*	40.48*	41.56*	64.89*
BootRet-Mt^{Wiki}	38.83*	41.36*	43.97*	65.83*
BootRet-Mt^{MS}	**39.35***	**42.79***	**44.21***	**66.73***

Table 2. Retrieval performance on NQ.

Method	Hits@1	Hits@10
BM25*	29.27	60.16
DocT5query*	39.13	69.72
RepBERT	50.20	78.12
DPR*	52.63	79.31
ANCE	45.42	72.75
DSI*	27.40	56.60
GENRE*	26.30	71.20
SEAL*	26.30	74.50
DSI-QG*	63.49	82.36
NCI	64.24	83.11
Corpusbrain	65.12	84.09
Ultron-PQ	64.61	84.45
GenRet	65.42	85.67
NOVO	66.13	86.24
BootRet-Bs^{Wiki}	66.71*	85.53*
BootRet-Bs^{MS}	65.88	85.04
BootRet-Mt^{Wiki}	**67.32***	**87.59***
BootRet-Mt^{MS}	66.15*	86.31*

Table 3. Ablation study of the pre-training components on Wikipedia corpus.

Method	MS 300K	NQ
	Hits@10	Hits@10
BootRet-Mt^{Wiki}	65.83	87.59
w/o dynamic identifiers	63.14	83.81
BootRet-Bs^{Wiki}	63.78	85.53
w/o pre-training	59.95	83.26
w/o retrieval prediction	63.01	83.82
w/o corpus indexing	63.28	83.91
w/o noisy documents	63.47	84.17
w/o contrastive losses	63.31	83.94

Figure 2. Results under zero- and low-resource settings.

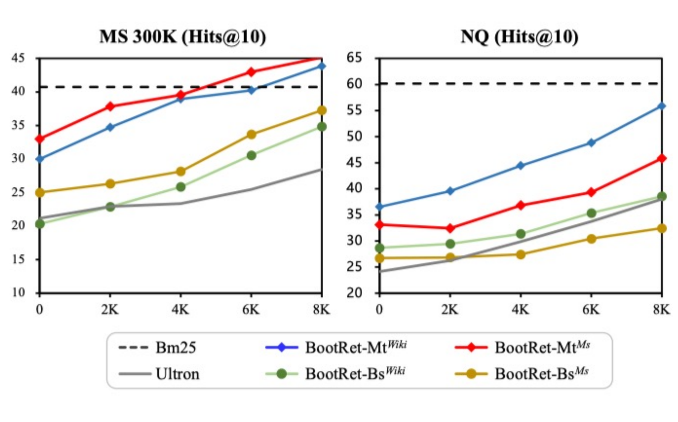


Figure 3. Retrieval performance of different number of iterations on MS 300k.

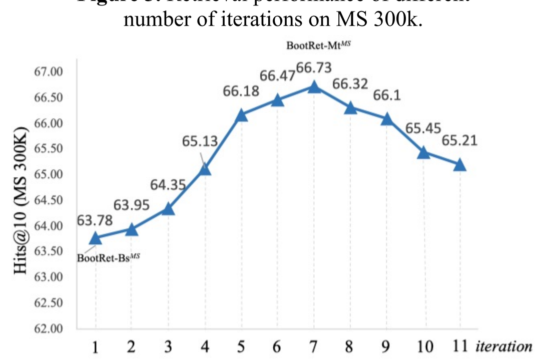


Figure 5. Impact of Noisy Documents.

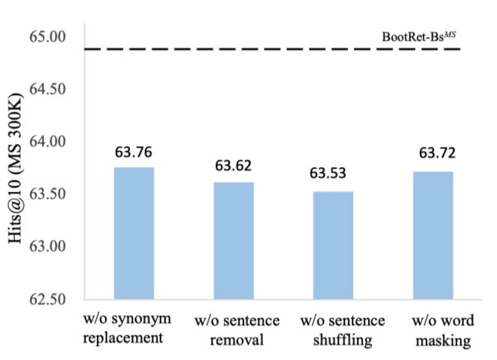
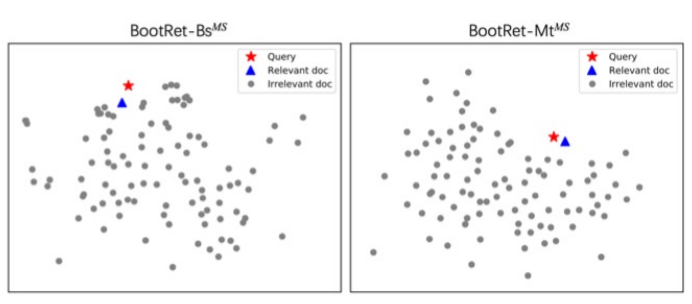


Table 4. Ablation study of the pre-training components on MS MARCO pre-training corpus.

Methods	MS 300K	NQ
	Hits@10	Hits@10
BootRet-Mt^{MS}	66.73	86.31
w/o dynamic identifiers	63.55	84.62
BootRet-Bs^{MS}	64.89	85.04
w/o pre-training	59.57	83.71
w/o retrieval prediction	63.02	84.51
w/o corpus indexing	63.46	84.76
w/o noisy documents	63.95	84.96
w/o contrastive losses	63.24	84.62

Figure 4. t-SNE plot of representations of a query (QID:1039861) from MS 300K validation set and documents corresponding to the generated top-100 docid list by BootRet-Bs^{MS} and BootRet-Mt^{MS}.



Conclusion & Limitations

- **Conclusion**:
  - We proposed BootRet, a bootstrapped pre-training method for GR, addressing the mismatch between pre-defined fixed docids and evolving model parameters in existing pre-training approaches
  - It dynamically adjusts docids based on the model pre-trained with two tasks
  - Extensive experiments validate that BootRet achieves superior performance compared to strong GR baselines on downstream tasks, even in the zero-shot setting
- **Limitations**:
  - Higher computational cost
  - Static incremental scenarios
  - Limited scalability
- Paper link: https://arxiv.org/pdf/2407.11504

References

[1] Unsupervised corpus aware language model pre-training for dense passage retrieval
[2] DC-BERT: Decoupling question and document for efficient contextual encoding
[3] Dcn+: Mixed objective and deep residual coattention for question answering
[4] Optimizing dense retrieval model training with hard negatives
[5] Recent advances in generative information retrieval
[6] Transformer memory as a differentiable search index
[7] BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension
[8] Exploring the limits of transfer learning with a unified text-to-text transformer
[9] Autoregressive search engines: Generating substrings as document identifiers
[10] Autoregressive entity retrieval
[11] Ultron: An ultimate retriever on corpus with a model-based indexer
[12] Corpusbrain: Pre-train a generative retrieval model for knowledgeintensive language tasks
[13] Dynamic updating of hippocampal object representations reflects new conceptual knowledge
[14] Optimized product quantization
[15] Jointly optimizing query encoder and product quantization to improve retrieval performance
[16] Wikipedia. 2022. Data dumps. https://dumps.wikimedia.org/enwiki/latest/ enwiki-latest-pages-articles.xml.bz2
[17] MS MARCO: A human generated machine reading comprehension dataset
[18] Natural questions: A benchmark for question answering research
[19] Okapi at TREC-3
[20] From doc2query to doctttttquery
[21] RepBERT: Contextualized text embeddings for first-stage retrieval
[22] Dense passage retrieval for open-domain question answering
[23] Approximate nearest neighbor negative contrastive learning for dense text retrieval
[24] Bridging the gap between indexing and retrieval for differentiable search index with query generation
[25] A neural corpus indexer for document retrieval
[26] Learning to tokenize for generative retrieval
[27] NOVO: Learnable and interpretable document identifiers for model-based IR
[28] LLaMA: Open and efficient foundation language models