Exploring the Practicality of Generative Retrieval on Dynamic Corpora

Chaeeun Kim[1,5*†] **Soyoung Yoon**[2*] **Hyunji Lee**[1]
Joel Jang[3] **Sohee Yang**[4] **Minjoon Seo**[1,5]

[1]KAIST AI [2]Seoul National University [3]University of Washington [4] UCL [5]LBox
chaeeun@lbox.kr minjoon@kaist.ac.kr

Abstract

Benchmarking the performance of information retrieval (IR) is mostly conducted with a fixed set of documents (static corpora). However, in realistic scenarios, this is rarely the case and the documents to be retrieved are constantly updated and added. In this paper, we focus on Generative Retrievals (GR), which apply autoregressive language models to IR problems, and explore their adaptability and robustness in dynamic scenarios. We also conduct an extensive evaluation of computational and memory efficiency, crucial factors for real-world deployment of IR systems handling vast and ever-changing document collections. Our results on the StreamingQA benchmark demonstrate that GR is more adaptable to evolving knowledge (4 – 11%), robust in learning knowledge with temporal information, and efficient in terms of inference FLOPs (×2), indexing time (×6), and storage footprint (×4) compared to Dual Encoders (DE), which are commonly used in retrieval systems. Our paper highlights the potential of GR for future use in practical IR systems within dynamic environments.

1 Introduction

Transformer-based information retrieval (IR) models play a vital role in advancing the field of semantic document search for information-seeking queries. Notably, *Generative Retrieval* (GR) (Petroni et al., 2019; De Cao et al., 2020; Wang et al., 2022; Bevilacqua et al., 2022; Tay et al., 2022; Zhou et al., 2022; Lee et al., 2022b,a; Sun et al., 2023; Li et al., 2023b) has recently gained a significant amount of recognition from the research community for its simplicity and high performance. However, *Dual Encoder* (DE) (Gillick et al., 2018; Karpukhin et al., 2020a; Ni et al., 2021; Gao et al., 2022; Izacard et al., 2022; Ram et al., 2022) continues to hold sway in practical IR systems. This contrast underscores the need for an investigation into their practical applicability. There is a lack of comprehensive comparison between GR and DE in real-world scenarios where knowledge is continually evolving and efficiency is crucial.

To this end, we establish Dynamic Information Retrieval (DynamicIR), a setup designed to simulate realistic scenarios for corpus updates in IR. This DynamicIR setup includes two distinct update strategies, (1) indexing-based updates: updating only the index without any further pretraining or finetuning and (2) training-based updates: continually pretraining the parameters on new corpora in addition to updating the index (See Figure 1). Within this experimental setup, we evaluate the adaptability of recent state-of-the-art retrieval models: SEAL (Bevilacqua et al., 2022), MINDER (Li et al., 2023b), and LTRGR (Li et al., 2023a) for GR, and SPIDER (Ram et al., 2022), CONTRIEVER (Izacard et al., 2022), and DPR (Karpukhin et al., 2020a) for DE. Furthermore, we perform extensive comparison for the *efficiency* of each method, considering factors such as floating-point operations (FLOPs) (Kaplan et al., 2020) required for the inference, indexing time, inference latency, and storage footprint.

The findings of our study underscore the strength of GR compared to DE across three key aspects: adaptability, robustness, and efficiency. (1) *GR exhibits superior adaptability to evolving corpora* (Section 5). GR outperforms DE, showcasing *4 – 11%* greater adaptability in both indexing-based and training-based updates. Notably, GR not only acquires new knowledge more effectively but also shows no sign of forgetting; rather, training with new corpora appears to enhance its existing knowledge. (2) *GR is more robust without inducing undesired bias from data characteristics* (Section 5). DE reveals a bias towards lexical overlap of times-

*Co-first authors.

†Most of the work was done while Chaeeun was an intern at KAIST AI.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 13616–13633
November 12-16, 2024 ©2024 Association for Computational Linguistics

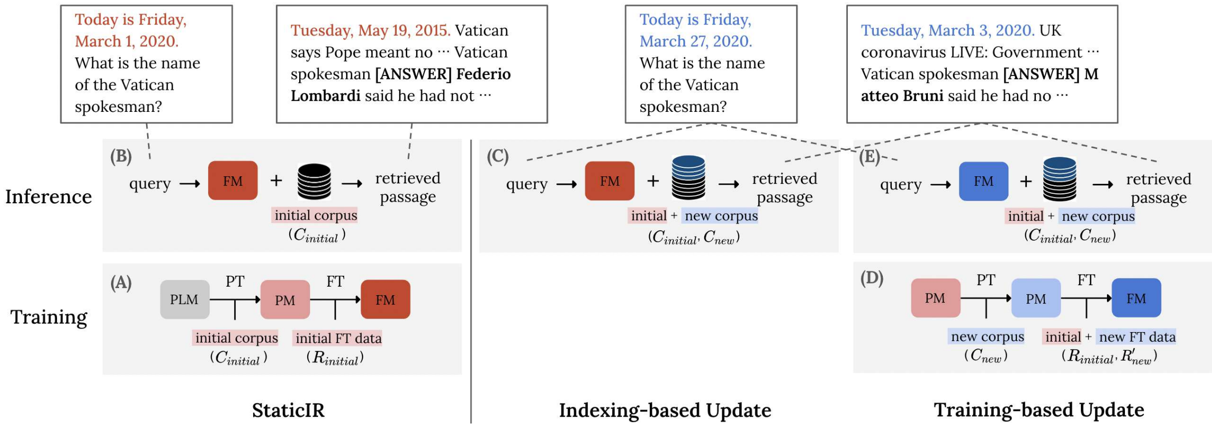

Figure 1: Structure of DynamicIR. This figure shows the training and inference processes for three setups in DynamicIR. We differentiate each model by color. First, in StaticIR, **(A)** retrieval models are pretrained on $C_{initial}$ and finetuned on the query-document pairs $R_{initial}$. **(B)** During inference, they perform retrieval only with the indexed $C_{initial}$. Second, in Indexing-based Update, **(C)** we use the same retriever developed from StaticIR and conduct an inference with the indexed $C_{initial}$ and C_{new}. Lastly, in Training-based Update, **(D)** we take the pretrained model on $C_{initial}$ in StaticIR and continually pretrain it on C_{new}. Subsequently, it is finetuned on the combination of $R_{initial}$ and R'_{new}. **(E)** Using the updated retrieval model, we conduct an inference with the indexed $C_{initial}$ and C_{new}.

tamps inserted into queries and documents, showing significant degradation (52.23% $\rightarrow$ 17.40%) when the timestamps are removed. Whereas, GR shows robust retrieval performance over temporal data. (3) *GR requires lower inference flops, reduced indexing costs, and a smaller storage footprint* (Section 6). For inference flops, GR has $\mathcal{O}(1)$ complexity with respect to the corpus size, requiring *2 times* less computation per query compared to DE which has $\mathcal{O}(N)$ complexity, where N represents the corpus size. Regarding indexing, DE necessitates re-indexing each time whenever the model is updated. To make matter worse, the indexing time itself is *6 times* longer than GR. In terms of storage footprint, GR requires *4 times* less storage by effectively compressing the knowledge in its internal parameters.

2 Related Work

Temporal Information Retrieval. While recent advancements have focused on the temporal updating of *language models* (Dhingra et al., 2022), the attention on temporal *information retrieval* (IR) (Kanhabua and Anand, 2016) has diminished somewhat with the rise of robust contextualized transformer-based models (Devlin et al., 2019). However, temporal considerations in IR remain crucial, especially with the widespread use of Retrieval-Augmented Generation (RAG) in many chat models. It is also worthwhile to examine whether emerging IR approaches are effective in adapting to evolving knowledge. Unlike previous works on document updates (Chen et al., 2023; Mehta et al., 2023), which pose disjoint questions from existing ones when retrieving updated documents and require parameter updates, our approach conducts experiments on the retrieval of distinct documents for the same query across varying timestamps. Similar to the text box in Figure 1, users often want to search for information based on a specific time period (e.g., laws, national curriculum, etc.). We also consider two realistic update scenarios with and without parameter updates and use a large-scale corpus of 50 million passages.

Generative Retrieval (GR). GR initially emerged with the work of GENRE (De Cao et al., 2020), in which an encoder-decoder model retrieves a document by generating the title of the document from a given query. (Tay et al., 2022) introduces DSI that produces a document ID as the output sequence. NCI (Wang et al., 2022) and DSI-QG (Zhuang et al., 2022) apply query generation, significantly improving DSI's performance. Recent methods that uses document ID, such as RIPOR (Zeng et al., 2023) and PAG (Zeng et al., 2024), also demonstrate superior performance. Instead of mapping to IDs for document identifiers, other works explore generating content directly from documents as identifiers. For instance, SEAL utilizes spans (Bevilacqua et al.,

Type	Split	Count
Query-Doc pairs	$R_{initial}$ (2007 – 2019)	99,402
	R'_{new} (2020)	90,000
Evaluation	$Q_{initial}$ (2007 – 2019)	2,000
	Q_{new} (2020)	3,000
	Q_{total} (2007 – 2020)	5,000
Corpus	$C_{initial}$ (2007 – 2019)	43,832,416
	C_{new} (2020)	6,136,419
	C_{total} (2007 – 2020)	49,968,835
# Tokens	Initial (2007 – 2019)	7.33B
	New (2020)	1.04B
	Total (2007 – 2020)	8.37B
# Tokens per passage	Initial (2007 – 2019)	169.7
	New (2020)	167.1
	Total (2007 – 2020)	167.5

Table 1: Statistics of the StreamingQA dataset modified for our setup. # Tokens is the total number of words separated by space in each passage.

2022), and MINDER (Li et al., 2023b) and LTRGR (Li et al., 2023a) leverage a combination of titles, pseudo-queries, and spans. Other works focus on the broader application of GR, such as multi-hop reasoning (Lee et al., 2022b), contextualization of token embeddings (Lee et al., 2022a), auto-encoder approach (Sun et al., 2023), and giving ranking signals (Li et al., 2023a).

Dual Encoder (DE). DE (Lee et al., 2019; Karpukhin et al., 2020b) refers to a set of model architectures where we project the query and document individually into a fixed sized embedding. Through contrastive learning, the projected embeddings of positive documents are learned to be close to the query and negative documents to be far away. Some works try to train the model in an unsupervised fashion with contrastive learning (Izacard et al., 2022; Lee et al., 2019; Sachan et al., 2023). While FAISS (Johnson et al., 2019) and ANCE (Xiong et al., 2020) can improve efficiency during inference, these models still face the limitation of requiring asynchronous creation of model-dependent embedding dumps.

3 Dynamic Information Retrieval

3.1 DynamicIR Task Setup

Adapting the retrieval models to evolving corpora is crucial for providing user with up-to-date knowledge. In order to evaluate the adaptability of retrievers, we create a setup called **Dynamic I**nformation **R**etrieval (DynamicIR). As depicted in Figure 1, our experimental setup includes three approaches: (1) *StaticIR*, where the retriever is trained on the initial corpus, (2) *Indexing-based updates*, incorporating the index of newly arrived documents into the existing index without further training on the new corpus; and (3) *Training-based updates*, where the retriever is continually pretrained on the new corpus, along with updating the index.

To conduct these experiments, we assume that we have an initial corpus $C_{initial}$ and a newly introduced corpus C_{new}, and datasets of query-document pairs $R_{initial}$ and R'_{new} from $C_{initial}$ and C_{new}, respectively. Unlike $R_{initial}$, R'_{new} consists of pseudo-queries, which are generated from C_{new} using docT5query (detailed explanation is in Section 3.2). These query-document pairs are used for supervised learning. Moreover, we assess the retrieval performance with two types of evaluation sets, $Q_{initial}$ and Q_{new}, where the answers to the questions are within $C_{initial}$ and C_{new}, respectively. Some questions between $Q_{initial}$ and Q_{new} are identical except for the timestamps, which necessitates the retrieval of different passages. Each set is employed to assess the forgetting of initial knowledge (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017) and the acquisition of new knowledge.

StaticIR. In this part, we focus on retrieving documents only from $C_{initial}$. The training process begins with pretraining the model on $C_{initial}$ using self-supervised learning, followed by finetuning it with $R_{initial}$ using supervised learning. We evaluate it only on $Q_{initial}$ with pre-indexed $C_{initial}$.

Indexing-based Update. In this update setup, we incorporate the new corpus to the retrieval models by updating only the index without any parameter updates. Since we utilize a retrieval model trained in StaticIR, this updating approach is quick and straightforward. We evaluate the retriever on $Q_{initial}$ and Q_{new} with pre-indexed $C_{initial}$ and C_{new}.

Training-based Update. In this advanced setup for update, we take the model pretrained on $C_{initial}$ and continually pretrain it on C_{new}. Subsequently, we finetune it using a combination of datasets, $R_{initial}$ and R'_{new}. Like indexing-based updates, we evaluate the updated retrieval model on $Q_{initial}$ and Q_{new} with pre-indexed $C_{initial}$ and C_{new}.

3.2 Benchmark

To evaluate the performance of retrieval models in a dynamic scenario, we employ STREAMINGQA (Liška et al., 2022) designed for temporal knowledge updates. Unlike other benchmarks on temporal retrieval (Deveaud et al., 2023), StreamingQA is the only benchmark that includes both the timestamps of question-asked time and document publication dates, which is critical for considering the temporal dynamics. The temporal information is prepended to the text in the format of '*Today is Wednesday, May 6, 2020.* [question]' for question, and '*Thursday, February 7, 2019.* [document text]' for documents (Liška et al., 2022). The dataset covers 14 years and includes over 50 million passages, more than double the content size of Wikipedia used in DPR (21M).

Temporal Information. StreamingQA includes a corpus spanning from 2007 to 2020, along with a supervised dataset of question-document pairs covering the years 2007 to 2019. In our work, $C_{initial}$ comprises articles from 2007 to 2019 and C_{new} consists of articles from 2020. Regarding the supervised dataset, the questions in $R_{initial}$ are asked in the time range of 2007 to 2019 to query articles from this period, and the questions in R'_{new} are asked in 2020 to query articles from 2020. Notably, all questions in the evaluation dataset $Q_{initial}$ and Q_{new} are asked in 2020, beginning with the prefix 'Today is [Day], [Month Date] , 2020', although they query articles from 2007 to 2019 ($C_{initial}$) and 2020 (C_{new}), respectively.

Pseudo-Queries for R'_{new}. The original StreamingQA dataset lacks query-document pairs from C_{new}, making it challenging to explore training-based updates including supervised learning. To address this, we generate additional 90,000 queries from C_{new}. To make this, we employ a trained model similar to the one used in docT5query[1] for query generation. The size of this additional dataset R'_{new} is similar to that of $R_{initial}$. Examples of generated dataset are in Table 11 and details of the query construction are explained in Appendix A.5.

4 Experimental setup

4.1 Retrieval Models

Generative Retrieval (GR). We select SEAL (Bevilacqua et al., 2022) that employs the substrings in a passage as document identifiers and MINDER (Li et al., 2023b) and LTRGR (Li et al., 2023a) that uses a combination of the titles, substrings, and pseudo-queries as identifiers. We choose these three models as baselines since they can more effectively update individual pieces of knowledge by autoregressively generating tokens in context using the FM-index. The FM-index for constrained decoding provides complete information about the documents with compression, allowing for the generation of a specific n-gram in the documents for each decoding step (Bevilacqua et al., 2022). Conversely, other GR models that use document IDs as identifiers (Tay et al., 2022; Wang et al., 2022) store contexts as sequential vectors mapped to document IDs, which introduces an additional mapping step between contexts and identifiers and may potentially limit the distinctive advantages of GR. Implementation details are in Appendix A.2.2.

Dual-Encoder (DE). We select Spider (Ram et al., 2022) and Contriever (Izacard et al., 2022) as representative models for DE. Since our experiments include a pretraining phase to store the corpus itself, we use Spider and Contriever as baselines that focus on the *self-supervised methods*. Since these models do not include a supervised method, we use DPR (Karpukhin et al., 2020a) during a finetuning phase, and adhere to its original training scheme such as utilizing in-batch negative training. Our DE baselines, Spider (+DPR) and Contriever (+DPR), include both phases. Implementation details are in Appendix A.2.1.

Sparse Retrieval. Although our main focus is on transformer-based semantic search models to explore corpora adaptation, we also evaluate BM25 (*Lucene*)[2], a traditional lexical matching retriever.

4.2 Effective Continual Pretraining of GR

When continually pretraining GR on C_{new}, we employ LoRA (Hu et al., 2021) widely recognized for its training efficiency. To better target key parameters for incorporating new knowledge, we analyze which parameters undergo the most significant change during the acquisition of new knowledge. This analysis is inspired by works on model merging (Ansell et al., 2023; Wortsman et al., 2022). We refer to these crucial parameters in learning new knowledge as *Dynamic Parameters* (DPs).

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

[2] https://github.com/castorini/pyserini

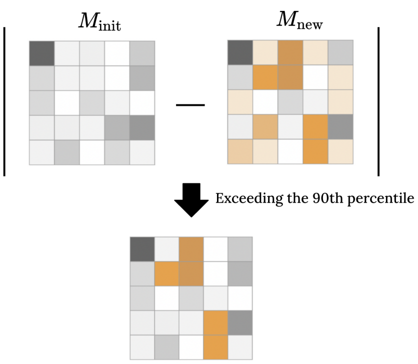

Figure 2: Analysis on key parameters in acquiring new knowledge. Through this analysis, we identify the locations of the top 10% most activated parameters.

Layer	Projection	Avg num of DPs
FFN	FC1	1.1M
	FC2	77K
	Total	**1.87M**
ATTN	Query	41K
	Key	35K
	Total	**76K**

Table 2: Average number of Dynamic Parameters (DPs), the parameters that have a large impact on acquiring new knowledge per block. It reveals that DPs are significantly more prevalent in the fully connected layer, exceeding those in the attention layer.

To identify DPs, as illustrated in Figure 2, we follow these steps: (1) Pretrain the model on $C_{initial}$ as M_{init}, and continually pretrain the model on C_{new} with full parameters as M_{new}. (2) Calculate the absolute differences in parameters between M_{init} and M_{new}. (3) Identify parameters that exceed the 90th percentile of these absolute differences.

DPs are twice as prevalent in the feed-forward networks (FFN) compared to the attention layer, as shown in Table 2. This result aligns well with previous studies on the memorization of factual knowledge (Geva et al., 2021; Dai et al., 2022). Consequently, based on this analysis, we apply LoRA on FFN in addition to the attention layer during the continual pretraining phase. The performance is noticeably improved compared to full-parameters and conventional LoRA. In contrast to GR, DE experiences significant degradation when this approach is applied (See Appendix A.6); thus, we pretrain DE with full parameters.

4.3 Evaluation

We assess retrieval performance with three evaluation dataset, $Q_{initial}$, Q_{new}, and Q_{total}. First, we evaluate the retention of initial knowledge by 2,000 questions that should be answered from the $C_{initial}$. Second, we assess the acquisition of new knowledge by 3,000 questions that should be answered from C_{new}. Both sets of 5,000 questions are randomly extracted from the entire evaluation data of StreamingQA, maintaining the ratio (16.60%) of each question type for initial knowledge and new knowledge. Finally, we assess total performance by calculating the unweighted average of the above two performances. Furthermore, we measure computational and memory efficiency in Section 6.

4.4 Metric

We assess the retrieval performance and efficiency. For retrieval performance, we use $Hits@5$, which measures whether the gold-standard passages is included in the top 5 retrieved passages. Most document search systems do not limit results to one or provide too many; we consider 5 to be a reasonable number for assessment. Additionally, we report full results of $Hits@k$ and $AnswerRecall@k$ ($k \in \{5, 10, 50, 100\}$) in Appendix A.8. Answer Recall measures whether the retrieved passage contains an exact lexical match for the gold-standard answer. For retrieval efficiency, we measure inference FLOPs, indexing time, inference latency, and storage footprint in Section 6.

5 Results and Analysis

In this section, we showcase the adaptability and temporal robustness of GR and DE and provide an analysis on the effective continual pretraining approach and utilizing R'_{new} during the training-based updates. We also discuss the performance of BM25 in dynamic environments.

GR has greater adaptability in both update scenarios. We define *adaptability* as the ability of retrieval models to maintain the performance after the updates. To evaluate the adaptability, we examine the performance on Q_{total} in each update scenario and compare it with that on $Q_{initial}$ in StaticIR (See Table 3).

First, *in indexing-based updates, GR exhibits 4% greater adaptability to new corpora compared to DE*. Specifically, when we look from $Q_{initial}$ of StaticIR to Q_{total}, GR maintains average performance, while DE demonstrates a 4% degradation on average. Second, *in training-based updates, GR*

Evaluation	Performance ($hit@5$) Q_{total} ($Q_{total}^{\text{w/o bias}}$)	$Q_{initial}$	Q_{new}	$Q_{new}^{\text{w/o bias}}$	Inference Efficiency FLOPs	Indexing Time	Latency (T_{online} / $T_{offline}$)	Storage Footprint
StaticIR								
Spider $_{\text{DE}}$	-	19.65%	-	-	9.0e+10	18.9h	**24.48ms** / 26m	173.8G
Contriever $_{\text{DE}}$	-	16.10%	-	-	9.0e+10	18.9h	212.4ms † / 9.8m	88.8G
SEAL $_{\text{GR}}$	-	34.95%	-	-	**4.3e+10**	**2.7h**	545.9ms / **1m 5s**	**34.5G**
MINDER $_{\text{GR}}$	-	**37.90%**	-	-	**4.3e+10**	**2.7h**	424.6ms / **1m 5s**	**34.5G**
LTRGR $_{\text{GR}}$	-	37.85%	-	-	**4.3e+10**	**2.7h**	424.6ms / **1m 5s**	**34.5G**
Indexing-based Update								
Spider $_{\text{DE}}$	24.82% (16.5%)	15.60%	34.03%	17.40%	1.0e+11	20.4h	**24.84ms** / 28m	196.8G
Contriever $_{\text{DE}}$	19.66% (11.01%)	13.75%	28.53%	8.27%	1.0e+11	20.4h	228.8ms† / 10.5m	99.8G
SEAL $_{\text{GR}}$	33.05% (35.13%)	32.75%	33.50%	37.50%	**4.3e+10**	**3.1h**	612.2ms / **1m 26s**	**37.5G**
MINDER $_{\text{GR}}$	**38.63%** (38.56%)	**37.65%**	**39.70%**	**39.47%**	**4.3e+10**	**3.1h**	485.4ms / **1m 26s**	**37.5G**
LTRGR $_{\text{GR}}$	38.30% (37.47%)	37.30%	39.30%	37.63%	**4.3e+10**	**3.1h**	485.4ms / **1m 26s**	**37.5G**
Training-based Update								
Spider $_{\text{DE}}$	36.99% (19.58%)	21.75%	**52.23%**	17.40%	1.0e+11	20.4h	**24.84ms** / 28m	196.8G
Contriever $_{\text{DE}}$	23.85% (9.82%)	8.20%	39.50%	11.43%	1.0e+11	20.4h	228.8m† / 10.5m	99.8G
SEAL $_{\text{GR}}$	41.01% (38.89%)	38.25%	43.77%	39.53%	**4.3e+10**	**3.1h**	612.2ms / **1m 26s**	**37.5G**
MINDER $_{\text{GR}}$	**41.54%** (39.31%)	**38.85%**	44.23%	**39.77%**	**4.3e+10**	**3.1h**	485.4ms / **1m 26s**	**37.5G**
LTRGR $_{\text{GR}}$	41.02% (39.14%)	38.50%	43.53%	39.77%	**4.3e+10**	**3.1h**	485.4ms / **1m 26s**	**37.5G**

† For Contriever, T_{online} is measured using faiss-cpu.
Spider and Contriever are further supervised using DPR.

Table 3: Results of DynamicIR. Our experiments are divided into 3 setups, (1) StaticIR, (2) Indexing-based updates, and (3) Training-based updates. For each setups, we assess the performance on Q_{total}, $Q_{total}^{\text{w/o bias}}$, $Q_{initial}$, Q_{new}, and $Q_{new}^{\text{w/o bias}}$ where the bias-inducing timestamps are removed. $Q_{new}^{\text{w/o bias}}$ is the average of $Q_{initial}$ and $Q_{new}^{\text{w/o bias}}$. Efficiency is evaluated using 4 metrics on the right side. For Inference Latency, T_{online} indicates the time required for query embedding and search, and $T_{offline}$ represents the time for loading the indexed corpus. We highlight the best scores in **bold** for each setup. Additionally, the zero-shot performance for all models is provided in Appendix 7.

shows 11% greater adaptability to new corpora compared to DE. Notably, GR shows a 5% average gain in performance. On the other hand, DE demonstrates a 6% degradation on average.

For DE, we extract the update score from $Q_{total}^{\text{w/o bias}}$ instead of Q_{total}. Because DE exhibits a significant inherent bias towards the lexical overlap of timestamps when evaluating Q_{new}. We delve deeper into this phenomenon below.

DE shows significant bias towards temporal data. *We observe a bias in DE towards the lexical overlap of timestamps* from the unusually high performance on Q_{new} not only in training-based updates (31% higher than $Q_{initial}$) but also in *indexing-based updates* (19% higher than $Q_{initial}$) where the models never encounter new corpora during training. This phenomenon stems from the temporal information, where all timestamps in the queries and in the documents of the evaluation dataset to be retrieved are set to *the year 2020*, introducing bias towards lexical overlap. In Table 3, $Q_{new}^{\text{w/o bias}}$ shows

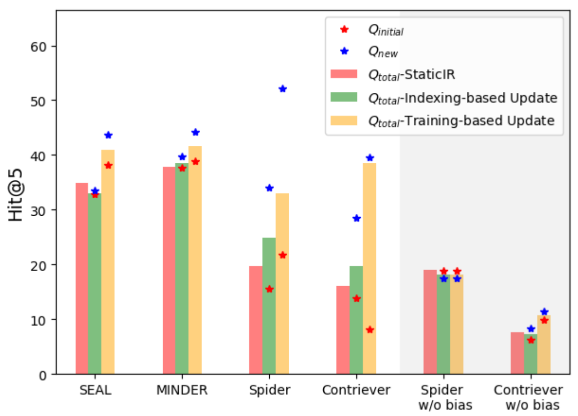

Figure 3: Visualization of total performance in DynamicIR. The star marks highlight the change in the gap between $Q_{initial}$ and Q_{new} of DE before and after the elimination of the bias-inducing factor (timestamp).

that removing bias-inducing timestamps significantly reduces DE's performance on Q_{new}, bringing it to a level similar to $Q_{initial}$. See the change in

Model	Continual Pretraining	Q_{total}	$Q_{initial}$	Q_{new}
SEAL $_{GR}$	ours (attn+ffn)	**41.01%**	**38.25%**	**43.77%**
	convent. LoRA	31.69%	32.00%	31.37%
	full params	29.92%	28.50%	31.33%
MINDER $_{GR}$	ours (attn+ffn)	**41.54%**	**38.85%**	**44.23%**
	convent. LoRA	38.83%	37.60%	40.07%
	full params	38.83%	35.30%	42.37%

Table 4: Analysis of the effectiveness of our continual pretraining approach targeting the key parameters. The results indicate hit@5 scores for training-based updates on activating full parameters (557M params), convent. LoRA (2.4M params), and our approach (3.1M params).

Model	R'_{new}	Q_{total}	$Q_{initial}$	Q_{new}
Spider $_{DE}$	with	**36.99%**	21.75%	**52.23%**
	w/o	35.77%	**29.90%**	41.63%
Contriever $_{DE}$	with	**23.85%**	8.20%	**39.50%**
	w/o	19.12%	**13.90%**	24.33%
SEAL $_{GR}$	with	**41.01%**	**38.25%**	**43.77%**
	w/o	37.91%	37.25%	38.90%
MINDER $_{GR}$	with	**41.54%**	**38.85%**	**44.23%**
	w/o	37.80%	38.15%	40.03%

Table 5: Analysis the effectiveness of R'_{new} with pseudo-queries in training-based updates. In this table, w/o refers to only using $R_{initial}$ during finetuning. The results in hit@5 show that it is effective to include the R'_{new}.

the gap between $Q_{initial}$ (red star) and Q_{new} (blue star) before and after removing timestamps in Figure 3. For more detailed explanations, refer to Appendix A.4.

GR better acquires new corpora even without parameter updates. We assess the ability to acquire new knowledge through Q_{new} in both update scenarios. For DE, we consider the performance of $Q_{new}^{\text{w/o bias}}$ instead of Q_{new}, reflecting the impact of bias as described above.

In indexing-based updates, Table 3 demonstrates that *GR excels in retrieving new knowledge even without parameter updates*. GR achieves a 2% higher score in Q_{new} compared to $Q_{initial}$ of StaticIR. Conversely, DE shows a 2% average degradation in $Q_{new}^{\text{w/o bias}}$. Similarly, for training-based updates, while DE decreases by 2 – 5% in $Q_{new}^{\text{w/o bias}}$, GR gains 6 – 9% in Q_{new} and 2 – 5% in $Q_{new}^{\text{w/o bias}}$.

GR better preserves initial knowledge. To assess the ability to retain initial knowledge, we analyze the performance on $Q_{initial}$ in both update setups, comparing it with $Q_{initial}$ in StaticIR.

For GR, Table 3 shows no signs of forgetting; rather, training on new corpora enhances performance on $Q_{initial}$ of training-based updates. This phenomenon can be explained by our results that selectively activating the well-targeted parameters for updates mitigates forgetting. Detailed explanations are described in the following analysis section. Additionally, unlike DE, which stores documents into single vectors, GR learns to generate tokens within documents directly, leveraging the power of autoregressive language models. This approach enables GR to access and update specific knowledge, thereby helping retain more unchanged knowledge.

On the other hand, DE shows a 3 – 4% degradation in indexing-based updates and a 0 – 8% decrease in training-based updates. Consequently, our observation indicates that *DE tends to forget initial knowledge more during updates compared to GR.*

Applying LoRA to FFN benefits GR in both preservation and acquisition of knowledge. Based on the analysis described in Section 4.2, our continual pretraining approach significantly improves the adaptability of GR. As shown in Table 4, activating FFN modules, which include many key parameters for adapting to new knowledge, helps not only Q_{new} but also $Q_{initial}$, compared to using conventional LoRA (convent. LoRA) and full parameters (full params). Specifically, targeting the key parameters helps mitigate the forgetting issue by updating sparsely, which surprisingly is more effective than the conventional approach of updating fewer parameters in LoRA.

Additionally, it maximizes the acquisition of new knowledge even more than training with full parameters in SEAL. Since our evaluation set consists mostly of *unseen data* during training and generalization ability is crucially assessed rather than memorization, well-targeted sparse updates outperform full parameter updates by reducing overfitting and improving generalizability.

R'_{new} enhances the overall performance of GR. We analyze the effectiveness of utilizing R'_{new}, query-document pairs where the queries are pseudo-queries generated from C_{new} using docT5query. In addition to the results of related works (Mehta et al., 2022; Zhuang et al., 2023; Lin and Ma, 2021; Mallia et al., 2021; Nogueira and Lin, 2019; Wang et al., 2022; Pradeep et al., 2023), our findings on dynamic corpora demonstrate that employing R'_{new} generated from new corpora is

Evaluation	Performance (*hit@5*)				
	Q_{total}	$Q_{total}^{\text{w/o bias}}$	$Q_{initial}$	Q_{new}	$Q_{new}^{\text{w/o bias}}$
[Static] BM25	-	-	43.35%	-	-
[Update] BM25	43.54%	41.19%	37.25%	49.83%	45.13%

Table 6: Performance of BM25 in StaticIR and Indexing-based updates. Through these results, we see the bias of temporal information via difference between Q_{new} and $Q_{new}^{w/o\ bias}$ and adaptability through a comparison between **[Static]** $Q_{initial}$ and **[Update]** $Q_{total}^{w/o\ bias}$ which is unweighted average of $Q_{initial}$ and $Q_{new}^{w/o\ bias}$.

beneficial for retrieving not only new knowledge but also initial knowledge for GR (See Table 5).

We believe experiencing benefits on $Q_{initial}$ despite training with R'_{new} is also attributed to the utilization of language model attributes for learning language distributions. Conversely, in the case of DE, we observe a 5 – 8% degradation in $Q_{initial}$, indicating forgetting. Moreover, since DE has a bias towards timestamps, if we explore $Q_{new}^{w/o\ bias}$ instead of Q_{new}, R'_{new} would not help DE at all.

BM25 shows temporal bias and limited adaptability. We investigate how BM25 performs in dynamic corpora with temporal information. Notably, BM25 surpasses transformer-based retrieval models on the StreamingQA benchmark, in contrast to its performance (Chen et al., 2022; Wang et al., 2023; Li et al., 2023b,a; Bevilacqua et al., 2022; Zeng et al., 2023) on other benchmarks such as KILT (Petroni et al., 2021), MSMARCO (Bajaj et al., 2018) and NaturalQuestions (Kwiatkowski et al., 2019) (See Table 6). However, BM25 exhibits limitations in handling temporal data, with a 4.7% degradation observed when timestamps are removed ($Q_{new} \rightarrow Q_{new}^{w/o\ bias}$ in **[Update]**), indicating a bias towards lexical matching of data from 2020. Furthermore, it struggles to maintain its initial performance, experiencing a 2.16% degradation when integrated with a new 6M corpus ($Q_{initial}$ in **[Static]** $\rightarrow$ $Q_{total}^{w/o\ bias}$ in **[Update]**). These findings underscore the need for retrieval models to move beyond textual matching, focusing not only on semantic searching (Magesh et al., 2024) but also on adapting to evolving corpora and maintaining robustness across diverse data characteristics.

6 Computation & Memory Efficiency

In this section, we provide the results of computational and memory efficiency. We use an 80G A100 GPU.

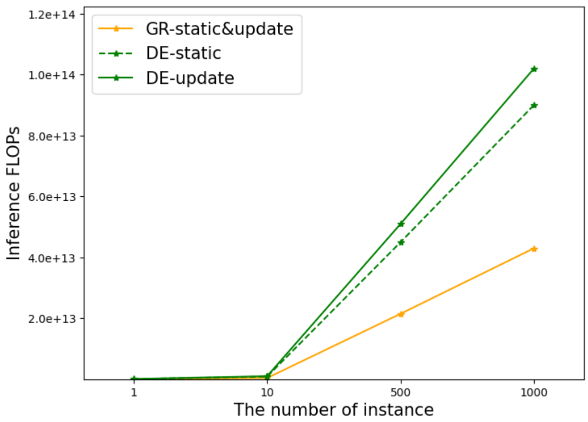

Figure 4: Inference FLOPs according to the number of instances. The flops for GR on both the static and updated corpus are identical, as it maintains consistent flops regardless of the corpus size unlike DE.

Inference FLOPs. We analyze the inference FLOPs [‡] of GR and DE to assess their computational efficiency. The FLOPs function and detailed calculations are in Appendix A.7. As shown in Table 3, our results reveal that *GR requires 2 times fewer computations per instance over DE*. Figure 4 illustrates that GR offers superior efficiency as the number of instances increases. Moreover, unlike DE, which exhibits $\mathcal{O}(N)$ complexity, where N represents the corpus size, GR maintains a constant $\mathcal{O}(1)$ complexity.

Indexing Time. There is a difference in the concept of indexing between GR and DE. For DE, this involves embedding, which converts the corpus into representations using an encoder. In GR, indexing refers the data processing of document identifiers to constrain beam search decoding, ensuring the generation of valid identifiers. Note that we process data without applying sharding.

As shown in Table 3, our results exhibit that *GR (3.1h) requires 6 times less time than DE (20.4h) for indexing $C_{initial}$ and C_{new}*. The crucial aspect of indexing is that unlike GR, DE necessitates re-indexing the entire corpus each time whenever the model is updated, irrespective of the corpus update.

Inference Latency. Inference process can be divided into two stages: (1) loading a pre-indexed corpus and (2) retrieving, which includes query

[‡]FLOPs (Floating Point Operations) is the number of floating-point arithmetic calculations.

embedding and search. We classify the former as *offline latency* ($T_{offline}$) and the latter is referred to as *online latency* (T_{online}). We measure both. T_{online} in Table 3 is reported for a single instance.

Table 3 shows *GR is 10 times faster than DE when retrieving from updated corpora for $T_{offline}$*. Unlike DE, which stores each passage representation in vector form, GR does not need much time to load the index since it stores knowledge within its parameters. *For T_{online}, however, GR is 20 times slower than DE using faiss-gpu*. Although DE requires 2 times more inference flops, it seems that the FAISS (Johnson et al., 2019) module contributes significantly to the inference speed of DE. For GR, there is potential for further speedup using techniques such as FlashAttention (Dao et al., 2022; Dao, 2023), speculative decoding (Leviathan et al., 2023), and a Mixture of Experts (Shazeer et al., 2017; Lepikhin et al., 2020; Kudugunta et al., 2021).

Storage Footprint. We measure the storage footprint of the retrieval model and the pre-indexed corpus, which are required for performing retrieval.

Table 3 indicates that *GR has 4 times less storage requirements over DE for updated corpora*. Notably, the memory requirements for DE are directly affected by the corpus size, as they store representations of all documents in vector form outside the retrieval model. In contrast, GR has minimal dependence on the corpus size by storing knowledge in its internal parameters.

7 Conclusion

In this work, we explore the practicality of GR in terms of adaptability to new corpora, temporal robustness, and inference efficiency, compared to DE. By establishing a DynamicIR setup, we showcase how retrieval models perform in dynamic environments where knowledge evolves over time, with and without parameter updates. Our findings indicate that GR better adapts to changing knowledge and less forgets by leveraging the power of language models. In terms of temporal robustness, GR retrieves information effectively regardless of data characteristics, while DE struggles with temporal information. Additionally, GR has fewer inference FLOPs, reduced indexing time, and lower memory requirements. By comprehensively exploring GR across these three aspects, we underscore its potential as an IR system.

8 Limitations

Our study has certain limitations. First, in the StreamingQA benchmark, all timestamps in the queries and in the documents to be retrieved from Q_{new} are set to the year 2020. This matching may introduce a bias towards the lexical overlap of temporal information when evaluating the acquisition of new knowledge. For a more dynamic evaluation, it is better to consider diverse query timestamps. Second, due to the scarcity of datasets that reflect temporal updates, we rely only on StreamingQA. While this dataset comprises 50 million articles spanning 14 years, a more comprehensive assessment across various datasets is needed to generalize our findings. Third, our findings cover large-scale corpus updates, yet they raise the question of how these results apply across multiple update frequencies. Lastly, while our results highlight the numerous advantages of GR in terms of adaptability to new corpora, inference flops, and memory, our evaluation of online inference latency demonstrates that DE has a faster speed compared to GR, primarily due to the FAISS module.

Acknowledgements

This work was partly supported by the Samsung Electronics grant (General-purpose generative retrieval, 2022, 50%) and the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2024-00397966, Development of a Cybersecurity Specialized RAG-based sLLM Model for Suppressing Gen-AI Malfunctions and Construction of a Publicly Demonstration Platform, 50%). An earlier version of this work was presented at the Gen-IR workshop@SIGIR 2024.

References

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. 2023. Composable sparse fine-tuning for cross-lingual transfer.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Wen tau Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. In *arXiv pre-print 2204.10628*.

Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, Yixing Fan, and Xueqi Cheng. 2023. Continual learning for generative retrieval over dynamic corpora. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23. ACM.

Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yiqun Liu, Yixing Fan, and Xueqi Cheng. 2022. Corpusbrain: Pre-train a generative retrieval model for knowledge-intensive language tasks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22. ACM.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2020. Autoregressive entity retrieval.

Petra Galuščáková Romain Deveaud, Gabriela Gonzalez-Saez, Philippe Mulhem, Lorraine Goeuriot, Florina Piroi, and Martin Popel. 2023. Longeval-retrieval: French-english dynamic test collection for continuous web search evaluation.

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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2022. Simcse: Simple contrastive learning of sentence embeddings.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. 2018. End-to-end retrieval in continuous space.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning.

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

Nattiya Kanhabua and Avishek Anand. 2016. Temporal information retrieval. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 1235–1238, New York, NY, USA. Association for Computing Machinery.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

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.

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

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur 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. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Hyunji Lee, Jaeyoung Kim, Hoyeon Chang, Hanseok Oh, Sohee Yang, Vlad Karpukhin, Yi Lu, and Minjoon Seo. 2022a. Contextualized generative retrieval. *arXiv preprint arXiv:2210.02068*.

Hyunji Lee, Sohee Yang, Hanseok Oh, and Minjoon Seo. 2022b. Generative multi-hop retrieval. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1417–1436.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023a. Learning to rank in generative retrieval.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. 2023b. Multiview identifiers enhanced generative retrieval.

Jimmy Lin and Xueguang Ma. 2021. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models.

Varun Magesh, Faiz Surani, Matthew Dahl, Mirac Suzgun, Christopher D. Manning, and Daniel E. Ho. 2024. Hallucination-free? assessing the reliability of leading ai legal research tools.

Antonio Mallia, Omar Khattab, Nicola Tonellotto, and Torsten Suel. 2021. Learning passage impacts for inverted indexes.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C):109–165. Funding Information: The research reported in this chapter was supported by NIH grant NS21047 to Michael McCloskey, and by a grant from the Sloan Foundation to Neal Cohen. We thank Sean Purcell and Andrew Olson for assistance in generating the figures, and Alfonso Caramazza, Walter Harley, Paul Macaruso, Jay McClelland, Andrew Olson, Brenda Rapp, Roger Rat-cliff, David Rumelhart, and Terry Sejnowski for helpful discussions.

Sanket Vaibhav Mehta, Jai 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.

Sanket Vaibhav Mehta, Jai Gupta, Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Jinfeng Rao, Marc Najork, Emma Strubell, and Donald Metzler. 2023. Dsi++: Updating transformer memory with new documents.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models.

Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docttttquery.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. Kilt: a benchmark for knowledge intensive language tasks.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases?

Ronak Pradeep, Kai Hui, Jai Gupta, Adam D. Lelkes, Honglei Zhuang, Jimmy Lin, Donald Metzler, and Vinh Q. Tran. 2023. How does generative retrieval scale to millions of passages?

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision.

Devendra Singh Sachan, Mike Lewis, Dani Yogatama, Luke Zettlemoyer, Joelle Pineau, and Manzil Zaheer. 2023. Questions are all you need to train a dense passage retriever.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

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.

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

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

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Hao Sun, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Allen Sun, Weiwei Deng, Qi Zhang, and Mao Yang. 2023. A neural corpus indexer for document retrieval.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval.

Hansi Zeng, Chen Luo, Bowen Jin, Sheikh Muhammad Sarwar, Tianxin Wei, and Hamed Zamani. 2023. Scalable and effective generative information retrieval.

Hansi Zeng, Chen Luo, and Hamed Zamani. 2024. Planning ahead in generative retrieval: Guiding autoregressive generation through simultaneous decoding.

Yujia Zhou, Jing Yao, Zhicheng Dou, Ledell Wu, and Ji-Rong Wen. 2022. Dynamicretriever: A pre-training model-based ir system with neither sparse nor dense index.

Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. 2022. Bridging the gap between indexing and retrieval for differentiable search index with query generation.

Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. 2023. Bridging the gap between indexing and retrieval for differentiable search index with query generation.

Model	Q_{total}	$Q_{initial}$	Q_{new}
Spider DE	13.28%	8.95%	17.60%
Contriever DE	18.74%	7.15%	30.33%
SEAL GR	20.60%	19.80%	21.40%
MINDER GR	25.87%	25.00%	26.73%

Table 7: Zero-shot performance on updated corpora. It demonstrates the zero-shot performance in hit@5 achieved without further training from released checkpoints. Overall, it exhibits a similar trend to our models trained on StreamingQA dataset.

A Appendix

A.1 Zero-shot performance of DE and GR

We conduct zero-shot experiments to assess the base performance of retrieval models on StreamingQA, utilizing Spider trained on NQ, Contriever trained on CCNet and Wikipedia, SEAL trained on KILT, and MINDER trained NQ. The results of the zero-shot experiments are presented in in Table 7.

A.2 Implementation Details

A.2.1 Dual Encoder

Spider. Spider experiments are conducted using 8× A100 80GB GPUs, and our implementation setup is primarily based on Spider. SPIDER (Ram et al., 2022)[§] code. We employ the bert-large-uncased pretrained model (336M) from HuggingFace, with fp16 enabled and weight sharing, configuring a batch size of 512 and a maximum sequence length of 240. For the pretraining stage, we run a full epoch with a learning rate of 2e-05 and a warm-up of 2,000 steps. The pretraining data is made by running the spider code on the provided documents from StreamingQA. This yields 95,199,412 pretraining data from base corpus and 21,698,933 from new corpus, which are used for StaticIR and DynamicIR, respectively. It takes about 5 days for pretraining the base model and 25 hours for continual pretraining the updated model. For the finetuning stage, we run for maximum 10 epochs with learning rate of 1e-05 and warm-up of 1,000 steps with batch size of 512. We select the best checkpoint with lowest validation loss.

Contriever. Contriever experiments are done on 4× A100 40GB GPUs. We employ bert-large-uncased pretrained model (336M) and follow the paper (Izacard et al., 2022) and their official codebase[¶]for the implementation and hyperparameter setup. We adjust the per_gpu batch size from 256 to 64 to fit in our gpu resource. Total step size is 110,000 for base (warmup 4,000 steps) and 16,000 (warmup 1,000 steps) for continual pretraining on $Cnew$, which is equivalent to one epoch. Learning rate is set to 1e-04. For the finetuning stage, we run contriever for maximum 10 epochs (about 8000 steps, warmup for 100 steps) with eval frequency of 200 steps and select the checkpoint with lowest eval loss. The per_gpu batch size is set to 32. All the hyperparemeters are the same with the pretraining setup, except the ones mentioned above.

A.2.2 Generative Retrieval

SEAL. We employ the bart-large pre-trained model (400M) for GR and train the model in Fairseq framework for using SEAL.(Bevilacqua et al., 2022)[‖]. Due to this context, when we utilize LoRA method, we implement the method within the Fairseq framework. For the pretraining stage of the base retrieval model in StaticIR, we generate 2 random spans and 1 full passage with the publication timestamp as input for each instance using the past corpus, resulting in 130,897,221 (130M) unsupervised data. We train the initial model on 16× A100 40GB GPUs with a batch size of 7,400 tokens and a learning rate of 6e-5. Subsequently, for the finetuning stage in StaticIR using $R_{initial}$, we use 10 random spans as document identifiers per question, resulting in 994,020 (994K). We train this model using 4× A100 80GB GPUs with batch size of 11,000tokens and a learning rate of 6e-5. In the continual pretraining stage for the updated model in training-based updates of DynamicIR, we use 3 random spans and 1 full passage with the publication timestamp as input for each instance, utilizing the updated corpus, which results in 24,471,541 (24M) unsupervised data. We train this updated model using 4× A100 80GB GPUs with a batch size of 11,000 tokens and a learning rate of 1e-4. Subsequently for finetuning stage in training-based update of DynimicIR using $R_{initial}$ and R'_{new}, we generate 10 random spans as passage identifiers per question, respectively, resulting in

[§]https://github.com/oriram/spider

[¶]https://github.com/facebookresearch/contriever

[‖]https://github.com/facebookresearch/SEAL

MINDER $_{GR}$	Q_{total}	$Q_{initial}$	Q_{new}
w/o title	41.54%	38.85%	44.23%
with pseudo-title	40.86%	38.15%	43.57%

Table 8: MINDER with and without Titles as Identifiers. The results in hit@5 indicate that there is little difference between the use of identifiers with and without the title.

1,894,020(1.8M) data. During inference, we set the beam size to 10.

MINDER. We use 2× A100 80GB GPUs for MINDER experiments. We use the pretrained model which is used for SEAL experiments, since MINDER has identical pretraining process to that of SEAL. For retrieval model of StaticIR, we create MINDER-specific data comprising of 10 spans and 5 pseudo-queries as passage identifiers per question, resulting in 1,491,030 (1.4M). For retrieval model of training-based updates in DynamicIR, we generate 10 spans and 5 pseudo-queries, resulting in 2,841,030 (2.8M) data. We run all MINDER models for maximum 10 epochs using with max token of 18,000 and a learning rate of 6e-5. During inference, we set the beam size to 10.

LTRGR. We use 4× A100 80GB GPUs for the learning-to-rank phase. We employ MINDER to create base models and then follow the configuration of LTRGR when learning to rank, except for setting the number of epochs and hits to 5 and 150, respectively, and omitting the title. During inference, we set the beam size to 10. When generating the training dataset for learning to rank, due to computational memory issues in processing 150 hits of our large finetuning dataset, we randomly sample 25% from $R_{initial}$ when training initial models used for StaticIR and Indexing-based update setups. For updated models, we randomly sample 25% from the combination of $R_{initial}$ and R'_{new}, maintaining a 1:1 ratio between them.

A.3 Difference in the presence of Titles as Identifiers for MINDER

The original MINDER model employs three components, titles, substrings, and pseudo-queries, as its identifiers. However, as the StreamingQA dataset lacks title information, we exclude document titles when constructing the MINDER model. To investigate the impact of this omission on performance, we conduct an analysis within training-based updates by finetuning utilizing pseudo-titles generated by GPT-3.5. Our results demonstrate that the omission of titles, in comparison to the utilization of pseudo-titles, has a negligible impact on performance as shown in Table 8.

A.4 Exploration of DE's bias towards lexical overlap of timestamps

All timestamps in the queries and in the documents to be retrieved are set to the year 2020. In this context, to clarify the bias of DE towards temporal information, we finetune the models using a dataset where query dates are removed. Subsequently, we evaluate the models using an evaluation dataset where query dates are eliminated. This experiment is viable because, out of a total of 5,000 evaluation instances, only 7 cases require different documents for the same question but with different query timestamps. Through the results $Q_{new}^{\text{w/o bias}}$ in Table 9 compared to Q_{new} in Table 3, we identify that the unexpectedly high performance of DE models stems from the lexical overlap with the timestamp. On the other hand, GR conducts retrievals more stably with fewer constraints on the lexical characteristics. See the change in the gap between $Q_{initial}$ and Q_{new} before and after removing timestamps in Figure 3.

A.5 Constructing the query-document pairs from new corpus

Reflecting the original evaluation dataset's distribution which balanced similar proportions of new (2020) and base (2007 – 2019) data, we replicate this distribution in our query generation based on new corpus. We randomly selected 90,000 passages from the 6 million 2020 passages. Subsequently, we finetuned a T5-base model on the query-document pairs from StreamingQA's base corpus, applying a hyperparameter configuration similar to docT5 query generation, feeding date-prefixed passages as input and producing date-prefixed queries as output. The training process comprises three epochs, with each taking roughly 45 minutes on an NVIDIA A6000 GPU. We then use the trained T5 model to generate one pseudo-query for each of the 90,000 selected passages, a process lasting approximately 90 minutes. Ensuring alignment with our study's temporal focus, we verify that the date information in the generated queries corresponded to 2020. Following a manual adjustment to ensure the queries are asked in 2020, we assemble the queries and corresponding

w/o timestamp	Indexing-based updates $Q^{\text{w/o bias}}_{initial}$	Indexing-based updates $Q^{\text{w/o bias}}_{new}$	Training-based updates $Q^{\text{w/o bias}}_{initial}$	Training-based updates $Q^{\text{w/o bias}}_{new}$
Spider $_{\text{DE}}$	18.90%	**17.40%**	18.90%	**17.40%**
Contriever $_{\text{DE}}$	6.25%	**8.27%**	9.85%	**11.43%**
SEAL $_{\text{GR}}$	35.35%	37.50%	35.30%	39.53%
MINDER $_{\text{GR}}$	36.85%	39.47%	38.45%	43.57%

Table 9: Ablation Study on the bias towards temporal information. DE shows a lexical bias toward timestamps on Q_{new} where all queries are asked in 2020 and the gold documents are published also in 2020. When removing the timestamp of query, the performance drastically drops, while GR does not exhibit noticeable changes.

Spider $_{\text{DE}}$	Q_{total}	$Q_{initial}$	Q_{new}
Full parameters	36.99%	21.75%	52.23%
LoRA	26.44%	10.05%	42.83%

Table 10: Spider with and without LoRA when pretraining on C_{new}. The results in hit@5 show that DE achieves higher performance when pretraining with full parameters not to apply LoRA

documents into an additional finetuning dataset for the retrieval models, a process that takes about four hours in total. Examples of the finetuning dataset are in Table 11.

A.6 Application of LoRA on DE

Unlike GR, LoRA does not improve the retrieval performance of DE. As shown in Table 10, it is evident that DE achieves higher performance when pretraining on C_{new} with the full parameters rather than using LoRA. The degradation in hit@5 is noticeable not only in Q_{new} but also in $Q_{initial}$, indicating that the application of LoRA is not beneficial for both retaining initial knowledge and acquiring new knowledge.

A.7 Calculation Details of Inference FLOPs

We approximately measure FLOPs per instance using DE_{flops} for DE and GR_{flops} for GR defined as below. We use the notation IP for inner product, FW for a forward pass, and $Beam$ for beam search.

$$DE_{flops} = FW^{enc}_{flops} + C\frac{1}{n_{\text{cluster}}}n_{\text{nearest}}IP_{flops}$$
$$GR_{flops} = FW^{enc}_{flops} + LBeam_{flops}$$
$$IP_{flops} = d_{\text{model}} + (d_{\text{model}} - 1)$$
$$FW_{flops} = 2N + 2n_{\text{layer}}n_{\text{ctx}}d_{\text{attn}}$$
$$Beam_{flops} = (FW^{dec}_{flops} + IP_{flops}|V|\log|V|)B$$

where C is the corpus size, L is the sequence length of output, d_{model} is dimension of hidden vector, N is the model size, n_{layer} is the number of layers, n_{ctx} is the length of input context, d_{attn} is the dimension of attention, V is the vocab size, and B is the beam size. n_{cluster} is the total number of centeroids (clusters), n_{nearest} is the number of clusters to search, $|V|\log|V|$ is the complexity of obtaining possible token successors with FM-index (Bevilacqua et al., 2022). We calculate FW_{flops} for the transformer based on Table 1 in (Kaplan et al., 2020) and apply it to the encoder and decoder.

We provide an approximate calculation of inference flops for DE and GR on updated corpora. For DE using the bert-large-uncased, its configurations are N=336M, d_{model}=1,024, n_{layer}=24, n_{ctx}=512, n_{nearest}=1, n_{cluster}=1, and C=50M. For query embedding, FW_{flops} is 697M, and for searching, $C \times IP_{flops}$ is 102B. The total inference flops (DE_{flops}) amount to approximately 102B + 697M $\approx$ 102.7B. For GR using the bart-large, its configurations are N=400M, d_{model}=1,024, n_{layer}=12, n_{ctx}=1,024, V=50,265, L=10, and B=10. For the encoding process, FW_{flops} is 425M, and for the decoding process, FW_{flops} is 42.5B. The total inference flops (GR_{flops}) amount to approximately 425M + 42.5B $\approx$ 43B.

Note that for DE, we employ the exhaustive (brute-force) search method adopted by our baselines. Some models can employ approximate search techniques, such as clustering, introducing a trade-off between speed and accuracy as they conduct exhaustive searches within nearby clusters.

A.8 Full performance on Hit and Answer Recall

We present the full results of evaluating the performance of DE and GR in both StaticIR and DynamicIR (indexing-based updates and training-based updates). We employ Hit@N and Answer Recall@N metrics, where N is set to 5, 10, 50, and 100, to assess retrieval performance. The results are in Table 12 and Table 13 for Hit and Answer Recall, respectively.

Pseudo-Query	Gold Passage
Today is Sunday, October 25, 2020. When did the pay gap between Pakistani employees and white employees decrease to 2%?	Monday, October 12, 2020. In 2019 median hourly earnings for white Irish employees were 40. 5% higher than those for other white employees at 17.55, while Chinese workers earned 23.1% more at 15.38 an hour and Indian workers earned 14.43 an hour - a negative pay gap of 15.5%. Annual pay gap Breaking down the data by gender, the ONS said ethnic minority men earned 6.1% less than white men while ethnic minority women earned 2.1% more than white women. The ONS added that ethnicity pay gaps differed by age group. Ämong those aged 30 years and over, those in ethnic minority tend to earn less than those of white ethnicities,ït said. In contrast, those in the ethnic minority group aged 16 to 29 years tend to earn more than those of white ethnicities of the same age. Gender pay gap The ONS found that the pay gap of 16% for Pakistani employees aged more than 30 shrank to 2% for those aged 16-29.
Today is Sunday, May 2, 2020. What was the top level of the FTSE 100?	Tuesday, April 28, 2020. But the big weekly shop has made a comeback, with the amount families spend on an average shopping trip hitting a record high. The new tracking data comes after Tesco boss Dave Lewis said the pandemic had changed people's shopping habits, which he said have r̈everted to how they were 10 or 15 years ago.M̈eanwhile, is this the end of loo roll wars? Spaghetti hoops have overtaken lavatory paper as the most out-of-stock item in Britain's stores. Follow our guide to minimising your risk of catching Covid-19 while shopping. The oil giant said there would continue to be an ëxceptional level of uncertaintyïn the sector. Meanwhile, the FTSE 100 soared to a seven-week high. Follow live updates in our markets blog.
Today is Tuesday, March 24, 2020. Why did President Trump sign an executive order banning hoarding?	Tuesday, March 24, 2020. President Donald Trump signs executive order banning hoarding March 23 (UPI) – President Donald Trump on Monday signed an executive order to prevent hoarding and price gouging for supplies needed to combat the COVID-19 pandemic. During a briefing by the White House Coronavirus Task Force, Trump and Attorney General William Barr outlined the order which bans the hoarding of vital medical equipment and supplies including hand sanitizer, face masks and personal protection equipment. Ẅe want to prevent price gouging and critical health and medical resources are going to be protected in every form,T̈rump said. The order will allow Health and Human Services Secretary Alex Azar to designate certain essential supplies a s scarce, which will make it a crime to stockpile those items in excessive quantities. Barr said the limits prohibit stockpiling in amounts greater than r̈easonable personal or business needsör for the purpose of selling them in ëxcess of prevailing market pricesädding that the order is not aimed at consumers or businesses stockpiling supplies for their own operation. Ẅe're talking about people hoarding these goods and materials on an industrial scale for the purpose of manipulating the market and ultimately deriving windfall profits,ḧe said.
Today is Tuesday, November 27, 2020. What is the name of the radio channel Joe Biden was on?	Monday, November 16, 2020. 'Heal the damage': Activists urge Joe Biden to move beyond b̈order securityÄs Joe Biden prepares to take office, activists say the president-elect must not only take mean ingful action to stabilize the US-Mexico border, but also reckon with his own history of militarizing the border landscape and communities. Biden has promised to end many of the Trump administration's border policies, but has yet to unveil the kind of bold immigration plan that would suggest a true departure from Obama-era priorities. Cecilia Muoz, Obama's top immigration adviser who memorably defended the administration's decision to deport hundreds of thousands of immigrants, was recently added to Biden's transition team. Biden has stated that he will cease construction of the border wall, telling National Public Radio in August that there will be n̈ot another foot of wall,änd that his administration will close lawsuits aimed at confiscating land to make way for construction. His immigration plan will also rescind Trump's declaration of a n̈ational emergencyön the southern border, which the Trump administration has used to siphon funds from the Department of Defense to finance construction, circumventing Congress in an action recently declared illegal by an appeals court. Some lawmakers along the border find these developments heartening, after Trump's border wall construction has devastated sensitive ecosystems, tribal spaces, and communities, a nd has been continuously challenged in court.

Table 11: Examples of Finetuning dataset R'_{new} created by docT5.

Model	Method	hit@5			hit@10			hit@50			hit@100		
		Total	initial	New	Total	initial	New	Total	initial	New	Total	initial	New
Spider	StaticIR	19.65	19.65	–	25.40	25.40	–	38.20	38.20	–	44.50	44.50	–
	Index-based Update	24.82	15.60	34.03	30.67	20.20	41.13	44.92	32.80	57.03	51.28	38.45	64.10
	Train-based Update	36.99	21.75	52.23	43.74	26.95	60.53	58.75	40.40	77.10	64.84	46.95	82.73
Contriever	StaticIR	16.10	16.10	–	20.25	20.25	–	33.80	33.80	–	40.90	40.90	–
	Index-based Update	21.14	13.75	28.53	25.17	17.35	36.90	39.44	29.45	54.43	46.26	35.65	62.17
	Train-based Update	23.85	8.20	39.50	29.26	10.55	47.97	43.66	20.35	66.97	49.64	25.35	73.93
SEAL	StaticIR	34.95	34.95	–	41.80	41.80	–	57.25	57.25	–	63.10	63.10	–
	Index-based Update	33.13	32.75	33.50	39.64	38.90	40.37	54.14	54.50	53.77	59.71	60.55	58.87
	Train-based Update	41.01	38.25	43.77	47.99	45.30	50.67	62.90	60.20	65.60	67.79	65.00	70.57
MINDER	StaticIR	37.90	37.90	–	45.00	45.00	–	59.60	59.60	–	64.00	64.00	–
	Index-based Update	38.68	37.65	39.70	45.27	44.40	46.13	60.87	60.60	61.13	66.13	66.35	65.90
	Train-based Update	41.54	38.85	44.23	48.29	45.60	50.97	63.12	60.80	65.43	68.43	66.25	70.60

Table 12: Full results on the Hit of DE and GR.

Model	Method	answer recall @5			answer recall @10			answer recall @50			answer recall @100		
		Total	initial	New	Total	initial	New	Total	initial	New	Total	initial	New
Spider	StaticIR	37.55	37.55	–	47.45	47.45	–	67.65	67.65	–	74.80	74.80	–
	Index-based Update	44.24	33.45	55.03	52.93	41.50	64.37	70.77	61.70	79.83	76.68	69.20	84.17
	Train-based Update	55.79	41.05	70.53	64.32	49.90	78.73	79.25	68.90	89.60	83.63	75.50	91.77
Contriever	StaticIR	28.90	28.90	–	37.60	37.60	–	60.20	60.20	–	68.25	68.25	–
	Index-based Update	31.34	25.15	40.63	41.84	34.80	52.40	63.05	55.15	74.90	70.98	64.30	81.00
	Train-based Update	37.14	20.15	54.13	46.54	28.15	64.93	66.21	48.65	83.77	72.33	56.85	87.80
SEAL	StaticIR	58.25	58.25	–	66.30	66.30	–	80.45	80.45	–	83.60	83.60	–
	Index-based Update	55.85	56.80	54.90	63.68	64.45	62.90	77.58	78.95	76.20	81.49	82.75	80.23
	Train-based Update	62.44	59.95	64.93	70.25	68.10	72.40	81.65	80.30	83.00	85.02	84.10	85.93
MINDER	StaticIR	59.50	59.50	–	68.10	68.10	–	80.35	80.35	–	83.75	83.75	–
	Index-based Update	54.23	54.45	54.00	62.96	63.75	62.17	76.54	78.00	75.07	79.79	81.20	78.37
	Train-based Update	56.74	55.35	58.13	64.45	63.70	65.20	77.19	77.40	76.97	80.34	80.50	80.17

Table 13: Full results on the Answer Recall of DE and GR.