PromptDSI: Prompt-based Rehearsal-free Continual Learning for Document Retrieval

Tuan-Luc Huynh[1] (✉), Thuy-Trang Vu[1], Weiqing Wang[1], Yinwei Wei[2], Trung Le[1], Dragan Gasevic[2], Yuan-Fang Li[1], and Thanh-Toan Do[1]

[1] Department of Data Science & AI, Monash University, Australia
tuan.huynh1@monash.edu
[2] Department of Human-Centred Computing, Monash University, Australia

Abstract. Differentiable Search Index (DSI) utilizes pre-trained language models to perform indexing and document retrieval via end-to-end learning without relying on external indexes. However, DSI requires full re-training to index new documents, causing significant computational inefficiencies. Continual learning (CL) offers a solution by enabling the model to incrementally update without full re-training. Existing CL solutions in document retrieval rely on memory buffers or generative models *for rehearsal*, which is infeasible when accessing previous training data is restricted due to privacy concerns. To this end, we introduce PromptDSI, a prompt-based, *rehearsal-free* continual learning approach for document retrieval. PromptDSI follows the Prompt-based Continual Learning (PCL) framework, using learnable prompts to efficiently index new documents without accessing previous documents or queries. To improve retrieval latency, we remove the initial forward pass of PCL, which otherwise greatly increases training and inference time, with a negligible trade-off in performance. Additionally, we introduce a novel topic-aware prompt pool that employs neural topic embeddings as fixed keys, eliminating the instability of prompt key optimization while maintaining competitive performance with existing PCL prompt pools. In a challenging rehearsal-free continual learning setup, we demonstrate that PromptDSI variants outperform rehearsal-based baselines, match the strong cache-based baseline in mitigating forgetting, and significantly improving retrieval performance on new corpora[3].

Keywords: Continual Learning · Document Retrieval

1 Introduction

Differentiable Search Index (DSI) [33] leverages a Transformer model [34] to encode corpus information directly into the model parameters through end-to-end optimization. This end-to-end retrieval paradigm eliminates the need to construct traditional inverted indices or vector databases used in sparse [29] or dense retrieval [12]. However, in real-world scenarios where new documents

[3] Code is available at: https://github.com/LouisDo2108/PromptDSI.

are continually added, re-training DSI from scratch for each update is computationally prohibitive, necessitating continual learning (CL) methods [22, 8]. This challenging setting is known as dynamic corpora [23], or lifelong information retrieval [9]. DSI++ [23] employs generative replay, while IncDSI [15] uses constrained optimization in an instance-wise continual learning setup. However, DSI++ requires an additional query generation model, leading to substantial computational overhead during training and introducing the challenge of maintaining this model for continual indexing. Meanwhile, IncDSI relies on caching queries for all previously indexed documents, resulting in high memory demands and potential data privacy concerns [6].

Rehearsal-free prompt-based continual learning (PCL) methods [40, 39, 37, 30, 35] offer a promising approach to alleviate the need to access previous documents or queries. These methods have shown competitive performance compared to rehearsal-based techniques in class-continual learning settings within the vision domain [35]. However, adapting PCL methods to document retrieval remains unexplored due to the extreme classification problem caused by the instance-wise nature, in which each document is a unique class and there are usually at least 100k documents. Moreover, existing PCL methods require two forward passes through the Transformer model, which is not suitable for retrieval systems with low latency requirements.

In this work, we propose PromptDSI, a prompt-based *rehearsal-free* continual learning DSI for document retrieval. PromptDSI uses learnable prompts to index new documents while keeping the the DSI backbone frozen, while not accessing previous documents or queries. To overcome the inefficiencies of PCL methods and tailor them for retrieval, we introduce several model-agnostic modifications. We eliminate the inefficient initial forward pass in PCL methods by using intermediate layer representations instead of the pre-trained language model's final layer `[CLS]` token for query-key matching. This reduces computational overhead with a minimal performance trade-off. Furthermore, due to the lack of distinct semantics across new corpora, PromptDSI often collapses to using a limited set of prompts, leading to underutilized parameters. Inspired by neural topic modeling [11, 9], we propose using neural topic embeddings mined from the initial corpus as fixed keys in the prompt pool of PCL methods. This strategy eliminates the training instability of prompt keys and improves the interpretability of prompt selection. Finally, while existing PCL methods follow multi-layer prompting [39, 30, 35], it is unclear if this is optimal for document retrieval. To explore the stability-plasticity trade-off, we conduct a comprehensive layer-wise prompting study to identify the most effective prompting layers. Overall, our work makes the following contributions:

- We introduce PromptDSI, the first PCL method for classification-based, end-to-end document retrieval.
- We propose two novel approaches to adapt existing PCL methods to document retrieval: (i) an efficient single-pass PCL approach for low-latency retrieval and (ii) using neural topic embeddings as fixed prompt keys to

stabilize query-key matching in PCL's prompt pool, addressing prompt underutilization and enhancing interpretability.
– We conduct a thorough analysis which verifies that single-layer prompting is sufficient for optimal performance when adapting existing PCL methods to continual learning for document retrieval.
– Experimental results on the NQ320k and MSMARCO 300k datasets under the challenging rehearsal-free setup show that PromptDSI performs on par with IncDSI, a strong baseline that requires caching previous training data.

2 Related Work

2.1 End-to-end Retrieval

End-to-end retrieval is an emerging paradigm that aims to replace the conventional "retrieve-then-rank" pipeline [29, 12] with a single Transformer model [34], pioneered by Differentiable Search Index (DSI) [33]. By integrating corpus information into the model parameters, DSI eliminates specialized search procedures, and enables end-to-end optimization. DSI can be divided into two subcategories: classification-based [15] and generative retrieval [23, 4], with the former being more resilient to catastrophic forgetting during continual indexing [23]. Therefore, we focus on the classification-based DSI approach in this work. Improving document identifier representation remains the main research focus of the community [45, 46], while the challenging continual learning (CL) setting is receiving increasing attention [23, 15, 4]. However, existing solutions either rely on caching previous queries or a generative model for rehearsal. PromptDSI builds upon IncDSI [15], which handles CL by caching all previous queries, and pushes DSI towards rehearsal-free CL.

2.2 Continual Learning (CL)

Addressing catastrophic forgetting in continual learning (CL) has been an active research area [36]. Two popular approaches are regularization-based methods, which constrain model updates via regularization terms or knowledge distillation [14, 20], and replay-based methods, which use memory buffers or generative models for rehearsal [3, 2]. These approaches dominate lifelong information retrieval studies [23, 4]. Recently, Prompt-based Continual Learning (PCL) has emerged as a solution for scenarios where historical data access is restricted due to privacy regulations [6]. Various studies [40, 39, 30, 37] have explored the use of prompts to guide frozen pre-trained models in learning new tasks without memory buffers. Concurrent to our work, [13] introduces one-stage prompt-based continual learning; however, their method is applied only to CODA-Prompt [30] and evaluated exclusively in the vision domain. In contrast, PromptDSI adapts vision-domain PCL methods to a more challenging instance-wise continual learning setting, where each document is a unique class, with at least 100k documents.

We propose a model-agnostic modification: efficient single-pass PCL, and validate its performance across three popular PCL methods. Additionally, we introduce fixed neural topic embeddings as prompt keys to mitigate prompt pool underutilization observed in certain PCL methods and to enhance explainability.

3 Background

3.1 Differentiable Search Index (DSI)

Indexing Stage. A Transformer model [34] f_Θ parameterized by Θ is trained to learn a mapping function from a document $d \in \mathcal{D}$ to its document identifier (docid) $id \in \mathcal{I}$ during the indexing stage: $f_\Theta \colon \mathcal{D} \to \mathcal{I}$. **Document representation** and **docid representation** are two crucial aspects of DSI that need to be predefined.

Document Representation. The original DSI is trained to map a document's texts to its corresponding docid. However, the model may need to process queries that are distributionally different from training documents at retrieval time, leading to a mismatch between indexing and retrieval. To bridge this gap, the current standard approach is to generate pseudo-queries that serve as document representations during indexing [47, 26]. Following this, we leverage `docT5query` [25] to generate pseudo-queries that supplement annotated queries, which are collectively used as inputs during the indexing stage: $f_\Theta \colon \mathcal{Q} \to \mathcal{I}$.

Docid Representation. We adopt atomic docids, where each document is assigned a unique integer identifier. Denote the classification-based DSI model as f_Θ, parameterized by $\Theta = \{\theta_e, \theta_l\}$, where θ_e represents the encoder weights, and $\theta_l \in \mathbb{R}^{\text{dim} \times |\mathcal{D}|}$ denotes the linear classifier weights, with $|\mathcal{D}|$ being the total number of documents. Each docid corresponds to a unique column vector $V \in \mathbb{R}^{\text{dim}}$ in θ_l, where the docid (i.e., the unique integer) specifies the position of the column vector in θ_l. Unlike dense retrieval, which uses dual encoders and an external index with contrastive learning [12, 43], classification-based DSI uses a single encoder and stores document embeddings as classifier weights, training with a seq2seq objective [32] to map queries to docids.

Retrieval Stage. Given a query $q \in \mathcal{Q}$, DSI returns a ranked list of documents by sorting the inner products of the query embedding $h_q = f_{\theta_e}(q)[0] \in \mathbb{R}^{\text{dim}}$ (i.e., the `[CLS]` token) and the linear classifier weight in descending order:

$$f_\Theta(q) = \left[\arg\overset{(1)}{\max}_{id \in \mathcal{I}}(\theta_l^\top \cdot h_q), \arg\overset{(2)}{\max}_{id \in \mathcal{I}}(\theta_l^\top \cdot h_q), \dots \right],$$

where $\arg\overset{(i)}{\max}_{id \in \mathcal{I}}(\theta_l^\top \cdot h_q)$ denotes the i^{th} ranked docid.

3.2 Continual Learning in DSI

The continual learning setup in DSI assumes that there are T+1 corpora: $\{D_0, \ldots, D_T\}$ with corresponding query sets $\{Q_0, \ldots, Q_T\}$ and corresponding docid sets $\{I_0, \ldots, I_T\}$, where D_t=$\{d_1^t, \ldots, d_{|D_t|}^t\}$. D_0 is often a large-scale corpus with annotated query-document pairs. $D_{>0}$=$\{D_1, \ldots, D_T\}$ are new corpora with completely new documents and no annotated query-document pairs arriving sequentially. Denote the parameters of the DSI model at timestep t (i.e., after indexing $D_{\leq t}$=$\{D_0, \ldots, D_t\}$) as Θ_t. At every timestep t>0, the previous DSI model $f_{\Theta_{t-1}}$ trained upto $D_{\leq t-1}$ has to train on corpus D_t and updates its parameters to Θ_t. The evaluation at timestep t will use f_{Θ_t} on $\{Q_0, \ldots, Q_t\}$ to predict $\{I_0, \ldots, I_t\}$. To index new documents, at timestep t, DSI's linear classifier θ_l is expanded to $\theta_l = \{\theta_l; \theta_{l,t}\}$ where $\theta_{l,t} \in \mathbb{R}^{\text{dim} \times |D_t|}$ denotes the expanded portion and $\{\cdot;\cdot\}$ is concatenation.

3.3 Prompt Tuning

Prompt tuning [18, 19] introduces a small set of learnable soft prompts to instruct frozen pre-trained language models on downstream tasks. In this work, we adopt the prefix tuning [19] variant. Given an input sequence $x \in \mathbb{R}^{\text{dim} \times n}$ with length n and embedding dimension of dim, prompt token $p \in \mathbb{R}^{\text{dim} \times m}$ with length m is prepended to the input of self-attention [34] as follows:

$$\text{Attention}(xW_q, [p_k, x]W_k, [p_v, x]W_v),$$

where W_q, W_k, W_v are projection matrices and $p_k, p_v \in \mathbb{R}^{\text{dim} \times \frac{m}{2}}$ are equally split from p.

4 PromptDSI: Single-pass Rehearsal-free Prompt-based Continual Learning for Document Retrieval

PCL methods [40, 37, 35, 30] can be seamlessly integrated into classification-based DSI. However, such an approach is not efficient for a document retrieval system. Moreover, simple adaptation also faces the issue of prompt underutilization. In this section, we will further elaborate on these issues and propose corresponding solutions.

4.1 Prompt-based Continual Learning (PCL)

PCL methods introduce a prompt pool $\mathbf{P} = \{p_1, \ldots, p_M\}$ with M prompts and a set of corresponding prompt keys $\mathbf{K} = \{k_1, \ldots, k_M\}$, where each prompt $p_i \in \mathbb{R}^{\text{dim} \times m}$ with length m is paired with a key $k_i \in \mathbb{R}^{\text{dim}}$ in a key-value manner. $\mathbf{P}$ acts as an external memory for the pre-trained language model (PLM), enabling storage of new information without disrupting its inherent knowledge or explicitly retaining previous training data.

As depicted in Figure 1 (left), given a query q, the **first pass** extracts the `[CLS]` token h_q of the input query q from a frozen PLM θ_e. A set of top-N prompt

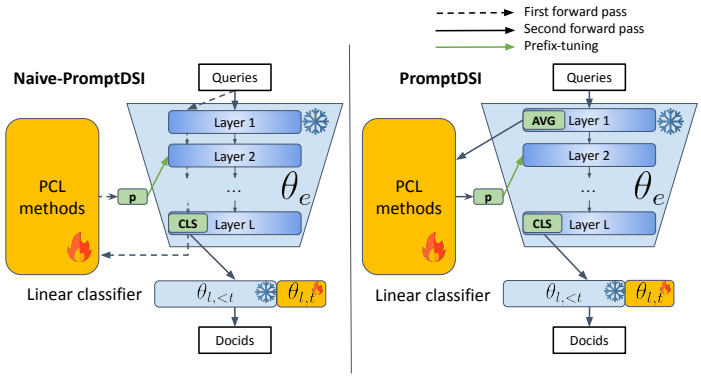

Fig. 1. Naive-PromptDSI (left) integrates PCL methods into DSI using two forward passes, causing increased training/inference time. PromptDSI (right) enhances efficiency by using intermediate layer representations (i.e. average of token embeddings `[AVG]`) for prompt selection, effectively removing an additional forward pass. In this example, prompts p are prefix tuning to layer 2 of the DSI's encoder θ_e. At timestep $t > 0$, $\theta_{l,<t}$ refers to the frozen portion of the linear classifier, while $\theta_{l,t}$ represents the expanded portion used for training on corpus D_t.

keys from the prompt pool are optimized with cosine distance to align them with the `[CLS]` token h_q using the following query-key matching mechanism:

$$\mathcal{L}_{\text{match}} = \sum_{i \in S_q} \gamma(h_q, k_i), \quad \text{s.t.} \quad S_q = \underset{\{s_i\}_{i=1}^{N} \subseteq [1,M]}{\arg\min} \sum_{i=1}^{N} \gamma(h_q, k_{s_i}), \tag{1}$$

where S_q denotes a set of top-N prompt key ids and $\gamma(\cdot,\cdot) = 1 - \cos(\cdot,\cdot)$ is the cosine distance. In the **second pass**, the same query q is reprocessed by θ_e; however, this time the previously selected top-N corresponding prompts $p \subset \mathbf{P}$ are prefix-tuned to the PLM, resulting in an enhanced `[CLS]` token: $\hat{h}_q = f_{p,\theta_e}(q)[0] \in \mathbb{R}^{\text{dim}}$, which has been instructed by the knowledge from prompts p.

4.2 Single-pass PCL

As previously discussed, existing PCL methods involve two forward passes during training and inference. Given that a forward pass in Transformer models is already computationally expensive, this two-pass design is unsuitable for retrieval systems, as it exacerbates latency during both training and inference. We refer to this naive adaptation of two-pass PCL methods to DSI as Naive-PromptDSI.

To mitigate the inefficiency of Naive-PromptDSI, we introduce PromptDSI, a streamlined variant that eliminates the first pass typically required by PCL methods. As illustrated on the right side of Figure 1, instead of using the `[CLS]` token from the first pass for prompt selection, PromptDSI leverages the average token embeddings `[AVG]` from the intermediate layer immediately preceding

the prompting layer: `[AVG]` $= \frac{1}{|q|}\sum_{i=1}^{|q|} f_{\theta_e^{l-1}}(q)_i \in \mathbb{R}^{\text{dim}}$, where $|q|$ denotes the sequence length of query q, and $f_{\theta_e^{l-1}}(q)_i$ represents the output of the $(l-1)^{\text{th}}$ layer of the encoder θ_e at position i. If $l = 1$, we use the average of the embeddings from the pre-trained language model's embedding layer. This adjustment approximates the semantic richness typically provided by the `[CLS]` token. In section 5.4, we show that this design incurs only minor task performance degradation while speeding up both training and inference. We hypothesize that this is thanks to queries in document retrieval are semantically simple compared to the images in vision domain, leading to shallow query embeddings are sufficient for prompt selection.

4.3 Topic-aware Prompt Keys

Given a prompt pool $\mathbf{P}$, L2P [40] shares $\mathbf{P}$ for all incoming tasks, while S-Prompt++ [37, 35] allocates a single key-prompt pair for each incoming tasks and freezes previous pairs. Since the learnable prompt keys in the prompt pool are optimized to represent new corpora, the lack of distinct semantic boundaries among corpora in document retrieval (i.e, corpora both consist of documents of similar topics) causes these keys to become highly similar across corpora. This leads to a collapse in prompt selection (i.e., the query-key matching mechanism in Eq. (1)) to a small subset of prompts, resulting in underutilization of parameters, which is visualized and elaborated in Section 5.5.

Instabilities in training prompt pools for PCL methods have been reported in the literature [24, 44]. We observe that in PCL methods, the query embeddings used for prompt selection are deterministic [40], suggesting that optimizing prompt keys $\mathbf{K}$ might be unnecessary. Inspired by neural topic modeling techniques [11, 9], we propose using neural topic embeddings derived from D_0 as fixed prompt keys. We employ BERTopic [10] to cluster document embeddings and generate neural topic embeddings via a class-based TF-IDF procedure (we refer readers to [10] for details of BERTopic). By assuming each document semantically belongs to a neural topic, we employ these topic-aware fixed prompt keys to stabilize the query-key matching mechanism, addressing underutilization of prompts. Furthermore, this approach facilitates knowledge transfer between documents within the same topic and allows better interpretability compared to general-purpose or corpus-specific prompts in existing PCL methods. We refer to this PromptDSI variant as $\mathbf{PromptDSI_{Topic}}$.

4.4 Optimization Objective

Denote PromptDSI's encoder and linear classifier as θ_e and θ_l, respectively. At timestep $t = 0$, PromptDSI is optimized using cross-entropy loss $\mathcal{L}_{\text{CE}}$ on D_0:

$$\mathcal{L}_0 = \sum_{q \in Q_0} \mathcal{L}_{\text{CE}}(f_{\theta_l}(f_{\theta_e}(q)), \text{id}_q), \tag{2}$$

where $\text{id}_q \in I_0$ is the one-hot encoded ground truth docid of query q.

We study three popular PCL methods: L2P [40], S-Prompt++ (S++) [37, 35], and CODA-Prompt (CODA) [30]. During continual indexing, the encoder θ_e is frozen while prompt pool $\mathbf{P}$, prompt keys $\mathbf{K}$, and a portion of the linear classifier θ_l are optimized. We refer to $f_{\mathbf{extra},\theta_e}$ as PromptDSI's encoder parameterized by θ_e and a set of learnable components **extra**. Denote $\mathbf{P}_t$ and $\mathbf{K}_t$ as prompts and prompt keys allocated for indexing D_t, at timestep $t > 0$, the general optimization objective for PromptDSI with L2P or S++ is:

$$\mathcal{L}_t^{\text{L2P}} = \sum_{q \in Q_t} \min_{\mathbf{P}_t, \mathbf{K}_t, \theta_{l,t}} \mathcal{L}_{\text{CE}}(f_{\theta_l}(f_{\mathbf{P}_t, \mathbf{K}_t, \theta_e}(q)), \text{id}_q) + \mathcal{L}_{\text{match}}, \tag{3}$$

where $\mathcal{L}_{\text{match}}$ is defined in Eq. (1) and $\text{id}_q \in I_t$. Unlike $\mathbf{PromptDSI_{L2P}}$, which optimizes all key-prompt pairs, $\mathbf{PromptDSI_{S++}}$ optimizes only one key-prompt pair per timestep. $\mathbf{PromptDSI_{CODA}}$ introduces learnable attention vectors $A \in \mathbb{R}^{\text{dim}}$ for each prompt. Instead of using $\mathcal{L}_{\text{match}}$, it computes a weighted sum of prompts: $P = \sum_{i=1}^{M} \alpha_i p_i$, where $\alpha_i = \cos(h_q \odot A_i, k_i)$ and $\odot$ denotes the Hadamard product. It is trained end-to-end with the following objective:

$$\mathcal{L}_t^{\text{CODA}} = \sum_{q \in Q_t} \min_{\mathbf{P}_t, \mathbf{K}_t, A_t, \theta_{l,t}} \mathcal{L}_{\text{CE}}(f_{\theta_l}(f_{\mathbf{P}_t, \mathbf{K}_t, A_t, \theta_e}(q)), \text{id}_q). \tag{4}$$

Using precomputed neural topic embeddings as fixed prompt keys, $\mathbf{PromptDSI_{Topic}}$ omits optimizing prompt keys $\mathbf{K}_t$ in Equation 3 and also removes $\mathcal{L}_{\text{match}}$:

$$\mathcal{L}_t^{\text{Topic}} = \sum_{q \in Q_t} \min_{\mathbf{P}_t, \theta_{l,t}} \mathcal{L}_{\text{CE}}(f_{\theta_l}(f_{\mathbf{P}_t, \theta_e}(q)), \text{id}_q). \tag{5}$$

5 Experiments

5.1 Experimental Setting

Datasets We evaluate PromptDSI on two well-known binary relevance document retrieval datasets: Natural Questions (NQ320k) [16] and a modified version of MSMARCO (MSMARCO 300k) [1, 5] (Table 1). NQ320k refers to the title-de-duplicated version of the Natural Questions dataset, containing 320k query-document pairs from approximately 108k documents [38, 31, 15]. MSMARCO 300k is a modified subset of the MS MARCO Document Ranking dataset [1, 5] with 300k documents, which is established in previous works [23, 15, 31]. For each corpus, queries from the official training set are split 80%/20% for training and validation, while the official development set is used for testing. To mimic the continual learning (CL) setup, each dataset is split into an initial corpus (D_0) containing 90% of the total documents, and five new corpora (D_1-D_5), each with 2% of the total documents. Each document in the train set is supplied with up to 15 additional pseudo-queries generated using `docT5query` [25] along with annotated natural queries. Each query corresponds to one relevant document.

Table 1. The NQ320k and MSMARCO 300k dataset statistics used in our study.

NQ320k					
Corpus	Document	Train Queries	Validation Queries	Test Queries	Generated Queries
D_0	98743	221194	55295	6998	1480538
D_1	2000	4484	1091	152	29997
D_2	2000	4417	1085	153	29992
D_3	2000	4800	1298	177	29991
D_4	2000	4346	1107	116	29992
D_5	1874	4131	964	140	28105
MSMARCO 300k					
D_0	289424	262008	65502	4678	4312150
D_1	2000	1768	480	40	29787
D_2	2000	1799	457	35	29805
D_3	2000	1800	450	30	29774
D_4	2000	1772	475	29	29821
D_5	2000	1851	430	30	29779

Evaluation metrics. We adopt Hits@{1, 10} and Mean Reciprocal Rank (MRR)@10 as document retrieval metrics. To evaluate CL performance, after training on corpus D_t, we report average performance (A_t), forgetting (F_t), and learning performance (LA_t), following previous works [23]. In the main result tables, We report the results of initial corpus D_0 and new corpora D_1-D_5 separately. We emphasize on A_t as it reflects both stability and plasticity [30]. Let $P_{t,i}$ be the performance of the model on corpus D_i in some metrics, after training on corpus D_t, where $i \leq t$. Forgetting of D_0 is calculated as: $\max(P_{5,0} - P_{0,0}, 0)$. With $t > 0$, the CL metrics are defined as follows:

$$A_t = \frac{1}{t}\sum_{i=1}^{t} P_{t,i} \quad LA_t = \frac{1}{t}\sum_{i=1}^{t} P_{i,i} \quad F_t = \frac{1}{t}\sum_{i=0}^{t-1} \max_{i' \in \{0,\ldots,t-1\}} (P_{i',i} - P_{t,i})$$

Baselines. We adopt the well-established BERT [7] and SBERT [28][4] as backbones and compare PromptDSI with both CL and non-CL baselines.[5] While recent dense encoders such as BGE [42], and NV-Embed [17] can also serve as backbones and may yield improved retrieval performance, our goal is not to benchmark classification-based DSI frameworks with state-of-the-art backbones, but to analyze their continual learning behavior. Due to resource and space constraints, we leave such comparisons to future work.

- **Sequential Fine-tuning** sequentially optimizes on new corpora without accessing previous ones, serving as the performance lower bound in CL.
- **DSI++ [23]** involves sequential fine-tuning on new corpora, using a query generation model to generate pseudo queries for sparse experience replay.
- **IncDSI [15]** indexes new documents by sequentially caching previous query embeddings and expanding θ_l. The new θ_l's embeddings are determined by

[4] HuggingFace model identifiers: `google-bert/bert-base-uncased` and `sentence-transformers/all-mpnet-v2`

[5] We omit comparisons with regularization-based methods, as they underperform compared to replay-based approaches [41].

solving constrained optimization problems. Since IncDSI is designed for online learning, we include "IncDSI*", a variant that solves the optimization for several epochs. We regard IncDSI as a strong baseline on D_0.
- **Multi-corpora Fine-tuning (Multi)** only fine-tunes on new corpora D_1-D_5, which are merged as one corpus. It is equivalent to the multi-task learning baseline in CL, serving as the performance upper bound on new corpora.
- **Joint Supervised (Joint)** trains on both initial and all new corpora, similar to the conventional supervised learning setup.
- **Dense Passage Retrieval (DPR) [12]** is a popular BERT-based dual-encoder trained with BM25 [29] hard negatives and in-batch negatives. DPR serves as a strong dense retrieval baseline and performs zero-shot retrieval on new corpora.

Implementation Details. We employ AdamW [21] and use a single NVIDIA A100 80GB GPU for all experiments. Following [15], we train BERT/SBERT on D_0 for 20 epochs, using batch size 128 and 1024, learning rate $1e^{-4}$ and $5e^{-5}$ for NQ320k and MSMARCO 300k, respectively. All subsequent methods are trained with batch size 128 and initialized from the same BERT/SBERT checkpoint trained on D_0. We randomly sample pseudo-queries from previous corpora to substitute for the query generation model in DSI++[6]. We reproduce IncDSI [15] using its official implementation. For PCL methods in PromptDSI, we leverage open-source implementations. PromptDSI$_{\text{L2P/S++}}$ use a prompt pool of size 5, prompt length 20 and top-1 prompt selection. PromptDSI$_{\text{CODA}}$ uses a prompt length 10 and 2 prompts per task. For BERT-based PromptDSI, we use learning rate $1e^{-4}$ and $5e^{-4}$; for SBERT-based PromptDSI, we use $1.5e^{-4}$ and $1e^{-3}$ for NQ320k and MSMARCO 300k, respectively. For PromptDSI$_{\text{Topic}}$, we use BERTopic [10] to mine neural topics from D_0, adhering closely to the author's best practices guide [7]. PromptDSI variants are trained for 10 epochs since prefix-tuning [19] requires longer training to converge. Other full-model fine-tuning baselines are trained for 5 epochs. Results for CL methods are averaged over three runs, with standard deviations reported accordingly. The layer-wise prompting study in Section 5.6 is conducted using a fixed random seed.

5.2 Main Results

We present our results of BERT/SBERT-based methods in Table 2 and 3. Performance of the latter is often better thanks to the better representation.

Among non-CL methods, across backbones and datasets, **Multi-corpora Fine-tuning** completely forgets D_0 and heavily overfits D_1-D_5. Both **Joint** and **DPR** achieve a balance between initial and new corpora; however, they generally underperform IncDSI and PromptDSI. Among CL methods, **Sequential Fine-tuning** suffer sfrom severe catastrophic forgetting and significantly

[6] Since the code for DSI++ has not been released.
[7] https://maartengr.github.io/BERTopic/getting_started/best_practices/best_practices.html

Table 2. BERT-based methods performance after indexing D_5. H@10 and M@10 denote Hits@10 and MRR@10. $\mathbf{D_0}$ denotes $P_{5,0}$ on D_0's test queries. † denotes results from [15]. Params. denotes number of trainable parameters. **Bold** and underline highlight the top and second best CL methods. The underscript numbers denote the standard deviations.

	NQ320k					MSMARCO 300k					
BERT-based	$\mathbf{D_0}\uparrow$ H@10	$\mathbf{D_0}\uparrow$ M@10	$A_5\uparrow$ H@10	$A_5\uparrow$ M@10	Params.↓	$\mathbf{D_0}\uparrow$ H@10	$\mathbf{D_0}\uparrow$ M@10	$A_5\uparrow$ H@10	$A_5\uparrow$ M@10	Params.↓	Rehearsal free
Non CL Methods (For Reference)											
Multi	0.0	0.0	91.4	83.9	193 M	0.0	0.0	92.2	83.9	340 M	-
Joint	85.9	70.1	84.7	68.4	193 M	76.8	55.2	78.3	53.8	340 M	-
DPR	70.3	51.9	70.1	49.8	220 M	68.8†	-	62.8†	-	220 M	-
CL Methods											
Sequential	$0.0_{0.0}$	$0.0_{0.0}$	$27.4_{2.8}$	$22.2_{1.2}$	193 M	$0.0_{0.0}$	$0.0_{0.0}$	$31.9_{0.8}$	$27.2_{0.6}$	340 M	✓
DSI++	$2.6_{0.0}$	$2.6_{2.6}$	$28.6_{1.9}$	$22.1_{28.6}$	193 M	$2.6_{0.1}$	$2.4_{0.0}$	$28.6_{7.3}$	$24.2_{5.1}$	340 M	✗
IncDSI	$\underline{86.4}_{0.1}$	$\mathbf{72.2}_{0.3}$	$85.8_{0.6}$	$69.5_{0.3}$	**7.6 M**	$79.5_{0.9}$	$57.5_{1.3}$	$81.8_{2.2}$	$61.4_{3.2}$	**7.7 M**	✗
IncDSI*	$\mathbf{86.5}_{0.2}$	$\underline{72.0}_{0.1}$	$85.2_{1.0}$	$68.6_{0.9}$	**7.6 M**	$\mathbf{80.6}_{0.0}$	$\mathbf{59.0}_{0.0}$	$84.8_{0.1}$	$65.1_{0.0}$	**7.7 M**	✗
PromptDSI (Ours) with											
L2P	$86.1_{0.1}$	$71.2_{0.2}$	$\mathbf{90.8}_{0.4}$	$73.2_{1.8}$	**7.6 M**	$\underline{80.5}_{0.0}$	$58.7_{0.0}$	$86.7_{1.4}$	$\underline{67.4}_{1.1}$	<u>7.8 M</u>	✓
S++	$86.1_{0.2}$	$71.4_{0.1}$	$\underline{90.5}_{0.4}$	$\underline{73.8}_{0.7}$	**7.6 M**	$80.4_{0.1}$	$58.6_{0.1}$	$\underline{86.8}_{1.5}$	$67.4_{0.8}$	<u>7.8 M</u>	✓
CODA	$86.0_{0.0}$	$71.2_{0.1}$	$90.3_{0.3}$	$\mathbf{74.2}_{0.8}$	<u>7.7 M</u>	$\mathbf{80.6}_{0.1}$	$\underline{58.9}_{0.2}$	$\mathbf{87.9}_{0.6}$	$66.8_{0.3}$	<u>7.8 M</u>	✓
Topic	$86.0_{0.1}$	$71.3_{0.0}$	$\underline{90.5}_{0.5}$	$\mathbf{74.2}_{1.4}$	8.8 M	$80.4_{0.0}$	$58.6_{0.1}$	$86.7_{0.5}$	$\mathbf{67.5}_{0.5}$	10.4 M	✓

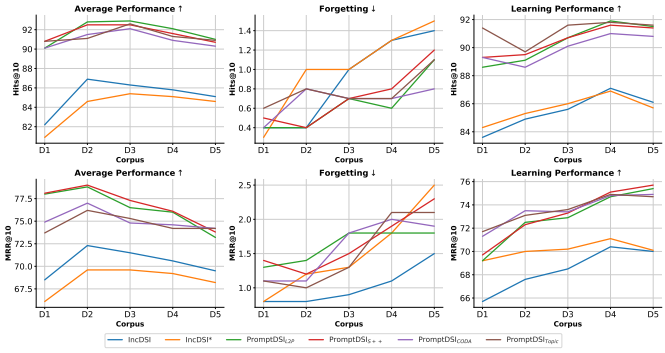

Fig. 2. Continual indexing performance of BERT-based methods on NQ320k

overfitting the most recent corpora. **DSI++**, even with sparse experience replay, provides only slight improvements over Sequential Fine-tuning, highlighting that maintaining a memory buffer or a generative model for rehearsal is non-trivial. Overall, Sequential Fine-tuning and DSI++ employ full-model fine-tuning CL methods (i.e., huge trainable parameters), but suffers from catastrophic forgetting. In contrast, IncDSI and PromptDSI are significantly more parameter-efficient, as a result of freezing the backbone.

IncDSI maintains strong performance across all corpora, with IncDSI* (10 epochs) further improving overall metrics. However, it still falls short of optimal performance on new corpora. All **PromptDSI** variants outperform IncDSI and IncDSI* in terms of A_5 by large margins across metrics, datasets, and backbones while maintaining D_0 performance close to that of IncDSI. $\mathbf{PromptDSI_{Topic}}$ removes prompt-key optimization to stabilize query-key matching, achieving comparable or superior performance among PromptDSI variants.

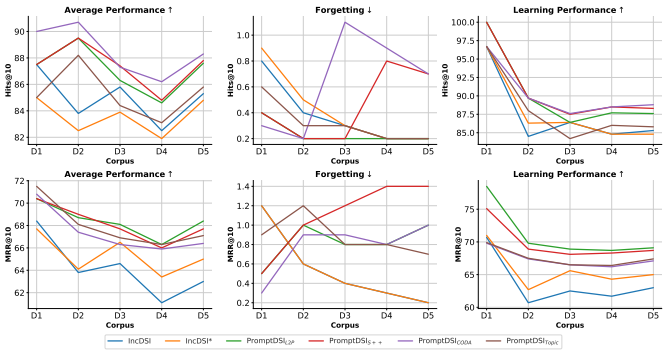

Fig. 3. Continual indexing performance of BERT-based methods on MSMARCO 300k

Table 3. SBERT-based methods performance after indexing D_5. $\mathbf{D_0}$ denote $P_{5,0}$ on D_0's test queries. H@10 and M@10 denotes Hits@10 and MRR@10. Params. denotes number of trainable parameters. **Bold** and underline highlight the top and second best CL methods. The underscript numbers denote the standard deviations.

	NQ320k					MSMARCO 300k					
SBERT-based	$\mathbf{D_0}$ ↑ H@10	$\mathbf{D_0}$ ↑ M@10	A_5 ↑ H@10	A_5 ↑ M@10	Params.↓	$\mathbf{D_0}$ ↑ H@10	$\mathbf{D_0}$ ↑ M@10	A_5 ↑ H@10	A_5 ↑ M@10	Params.↓	Rehearsal free
Non CL methods (For reference)											
Multi	0.0	0.0	92.0	83.6	193 M	0.0	0.0	94.7	87.2	340 M	-
Joint	86.8	71.5	86.4	69.7	193 M	76.3	55.5	84.1	61.2	340 M	-
CL methods											
Sequential	$0.0_{0.0}$	$0.0_{0.0}$	$25.2_{2.0}$	$20.3_{0.8}$	193 M	$0.0_{0.0}$	$0.0_{0.0}$	$27.1_{2.1}$	$23.3_{1.2}$	340 M	✓
DSI++	$2.7_{0.0}$	$2.4_{0.1}$	$28.5_{4.8}$	$22.3_{2.7}$	193 M	$2.8_{0.1}$	$2.5_{0.1}$	$29.0_{4.7}$	$23.3_{0.5}$	340 M	✗
IncDSI	$\mathbf{87.1}_{0.0}$	$72.3_{1.1}$	$86.6_{0.8}$	$70.5_{2.4}$	**7.6 M**	$80.9_{1.0}$	$58.6_{1.2}$	$82.8_{1.6}$	$64.0_{3.3}$	**7.7 M**	✗
IncDSI*	$87.0_{0.0}$	$\mathbf{72.6}_{0.0}$	$87.3_{0.1}$	$73.2_{0.0}$	**7.6 M**	$\mathbf{82.0}_{0.0}$	$\mathbf{60.0}_{0.0}$	$84.1_{0.1}$	$68.0_{0.2}$	**7.7 M**	✗
PromptDSI (Ours) with											
L2P	$86.9_{0.1}$	$\mathbf{72.6}_{0.1}$	$\mathbf{91.1}_{0.1}$	$74.0_{3.8}$	**7.6 M**	$\underline{81.6}_{0.1}$	$\underline{59.1}_{0.0}$	$87.4_{0.4}$	$70.1_{0.4}$	$\underline{7.8\,\text{M}}$	✓
S++	$86.8_{0.0}$	$\underline{72.5}_{0.1}$	$\underline{91.0}_{0.5}$	$\underline{74.8}_{2.7}$	**7.6 M**	$81.4_{0.1}$	$58.9_{0.1}$	$\underline{87.5}_{0.9}$	$\mathbf{71.4}_{0.3}$	$\underline{7.8\,\text{M}}$	✓
CODA	$\underline{87.0}_{0.1}$	$72.1_{0.9}$	$\mathbf{91.1}_{0.5}$	$74.2_{4.1}$	$\underline{7.7\,\text{M}}$	$81.5_{0.0}$	$59.0_{0.2}$	$\underline{87.5}_{0.7}$	$70.0_{1.0}$	$\underline{7.8\,\text{M}}$	✓
Topic	$86.8_{0.0}$	$72.1_{0.2}$	$\mathbf{91.1}_{0.3}$	$\mathbf{75.1}_{1.9}$	9.0 M	$81.3_{0.1}$	$\underline{59.1}_{0.2}$	$\mathbf{88.1}_{1.9}$	$\underline{70.5}_{1.8}$	10.6 M	✓

We further analyze the continual indexing of BERT-based PromptDSI and IncDSI in Figures 2 and 3. PromptDSI consistently outperforms IncDSI in Average and Learning Performance across datasets, with minimal forgetting, often matching IncDSI or surpassing IncDSI*. These results highlight the superior stability-plasticity trade-off of PromptDSI, despite being rehearsal-free.

5.3 Memory Complexity Analysis

DSI++ requires storing an additional T5 [27] model for query generation and IncDSI requires caching a matrix $\mathbf{Z} \in \mathbb{R}^{\text{dim} \times \sum_{t=0}^{T} |D_t|}$, where each column of $\mathbf{Z}$ represents an average query embedding. Consequently, the memory usage and indexing time of IncDSI increase linearly with the number of documents. For our experiments, caching requires approximately 318 MiB for NQ320k and 977 MiB for MSMARCO 300k. Extending to the full MS MARCO dataset (8.8M passages) requires about 25 GiB of memory. In many cases, loading such a large memory

Table 4. Performance comparison between Naive-PromptDSI and PromptDSI on NQ320k.

Methods	Metric	Naive-PromptDSI Hits@10	Naive-PromptDSI MRR@10	PromptDSI Hits@10	PromptDSI MRR@10	Single-pass speedup
$\text{PromptDSI}_{\text{L2P}}$	$\mathbf{D_0}\uparrow$	$86.1_{0.1}$	$71.3_{0.1}$	$86.1_{0.1}$	$71.2_{0.2}$	4.3x
	$A_5\uparrow$	$90.3_{0.5}$	$74.1_{0.3}$	$90.8_{0.4}$	$73.2_{1.8}$	
$\text{PromptDSI}_{\text{S++}}$	$\mathbf{D_0}\uparrow$	$86.1_{0.1}$	$71.3_{0.2}$	$86.1_{0.2}$	$71.4_{0.1}$	4.2x
	$A_5\uparrow$	$90.5_{0.2}$	$74.0_{0.9}$	$90.5_{0.4}$	$73.8_{0.7}$	
$\text{PromptDSI}_{\text{CODA}}$	$\mathbf{D_0}\uparrow$	$86.0_{0.1}$	$71.2_{0.1}$	$86.0_{0.0}$	$71.2_{0.1}$	1.7x
	$A_5\uparrow$	$90.6_{0.2}$	$74.2_{0.6}$	$90.3_{0.3}$	$74.2_{0.8}$	

footprint onto conventional GPUs may not be possible. In contrast, PromptDSI is rehearsal-free, eliminating the need to store $\mathbf{Z}$. It employs a prompt pool and a set of prompt keys, i.e., $\mathbf{P} \cup \mathbf{K} \in \mathbb{R}^{\text{dim}\times(M(m+1))}$, which consists of M prompts $p \in \mathbb{R}^{\text{dim}\times m}$ and M prompt keys $k \in \mathbb{R}^{\text{dim}}$. The L2P/S++ variants require approximately 315 KiB, which is about three orders of magnitude smaller than IncDSI's overhead on NQ320k, while the largest "Topic" variant requires approximately 11.9 MiB—roughly one to two orders of magnitude smaller than IncDSI or the additional T5 model used in DSI++.

5.4 Single-pass PromptDSI Analysis

Table 4 presents the comparison between Naive-PromptDSI and PromptDSI. Using intermediate layer representations as query embeddings for prompt selection (i.e., bypassing the first forward pass), we observe only a minimal trade-off in performance across all PromptDSI variants. This is primarily reflected in MRR@10, with a maximum drop of just 0.9%. Notably, with our implementations, the single-pass L2P and S++ variants can be up to 4 times faster by omitting the first forward pass. Although the speedup is less significant for the CODA variant, which employs a weighted sum of prompts strategy, it still achieves a notable 1.7x speedup. Overall, these results confirm that the proposed single-pass PCL methods in PromptDSI meet the low-latency requirements of typical information retrieval systems, with only a negligible impact on performance.

5.5 Prompt Pool Utilization Analysis

Figure 4 (left) illustrates the prompt pool utilization of PromptDSI variants.[8]. We observe that L2P frequently selects a small subset of prompts across all corpora. While S++ benefits from SBERT's better similarity-based embeddings to achieve more diverse task-specific prompt selection, this effect does not extend to L2P. These findings suggest that optimizing the prompt pool and ensuring diverse selection remain challenging due to instability. With a topic-aware prompt pool, $\text{PromptDSI}_{\text{Topic}}$ achieves better prompt utilization, leading to improved performance in some cases (Table 3) while mitigating training instability. Notably, in Figure 4 (bottom-right), prompt pool utilization exceeds 60%, with over

[8] $\text{PromptDSI}_{\text{CODA}}$ applies a weighted sum of prompts instead of prompt selection.

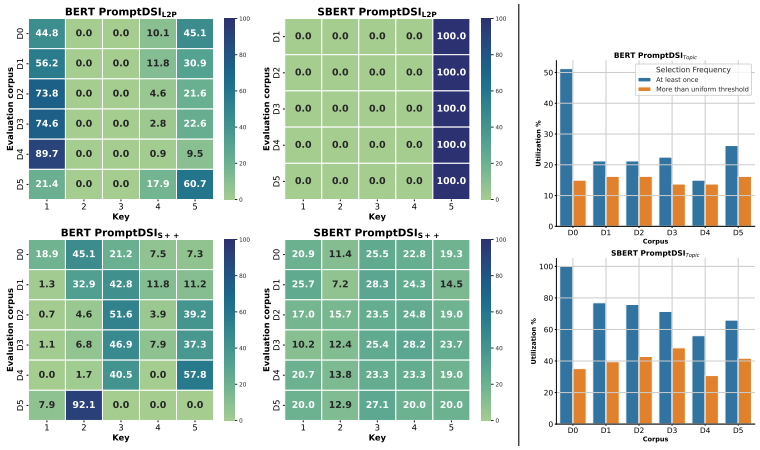

Fig. 4. Prompt pool utilization:PromptDSI$_{\mathrm{L2P/S++}}$ (left); PromptDSI$_{\mathrm{Topic}}$ (right).

30% of the prompts frequently selected (i.e., above the uniform threshold). Beyond performance improvements, topic-aware prompts enhance interpretability, unlike general-purpose (L2P) or corpus-specific (S++) prompts.

Since we uses two backbones BERT and SBERT, we adopt separate BERTopic models for each, resulting in slightly different topics for the same dataset (Figures 5 and 6). While effective, topic modeling may benefit further from LLMs due to their superior representational and contextual capabilities. Table 5 presents queries associated with the top 8 topics identified by SBERT-based BERTopic on the NQ320k validation set, which PromptDSI$_{\mathrm{Topic}}$ selects during inference. These neural topics reflect frequently queried subjects, with queries often containing recurring or semantically similar terms.

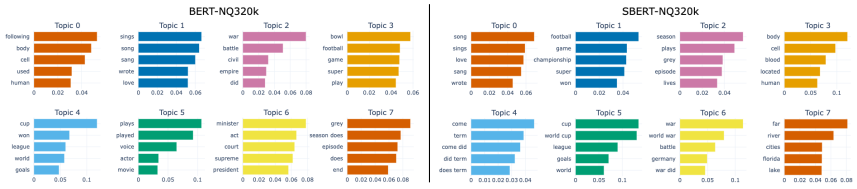

Fig. 5. Top 8 NQ320k topics mined using BERTopic with BERT/SBERT and their corresponding most frequent terms. A total of 80 topics were identified with BERT and 91 with SBERT from D_0.

Table 5. Examples annotated queries from the NQ320k validation set that are matched correctly to the corresponding topic-aware prompts during inference. Topic embeddings are mined using SBERT-based BERTopic. Topic-specific terms are highlighted.

Topic	Queries
0 (Music)	who **sang** the **song** no other love have i who **sings** the **song** it feels like rain who **sings** whenever you want me i'll be there
1 (Football)	when was the last time **cleveland browns** were in the playoffs has any **nfl** team gone 16-0 and won the **superbowl** what **nfl** team never went to the **super bowl**
2 (Movies)	who is the **actor** that **plays** spencer reid who died in vampire diaries **season** 1 **episode** 17 who **played** the mother on father knows best how many **season** of the waltons are there
3 (Human biology)	where does the **amino acid** attach to **trna** when a person is at rest approximately how much **blood** is being held within the **veins** where are **lipids** synthesized outside of the endomembrane system
4 (Origin)	where does the last name broome **come from** where does spring water found in mountains **come from** where does the last name de leon **come from**
5 (Soccer)	who is going to the 2018 **world cup** when did **fifa** first begun and which country was it played who came second in the **world cup** 2018
6 (War)	who fought on the western front during **ww1** when did the allies start to win **ww2** when did the united states declare **war** on **germany** what were the most effective weapons in **ww1**
7 (Geography)	how wide is the **mississippi** river at davenport **iowa** bob dylan musical girl from the north **country** what are the four **deserts** in **north america**

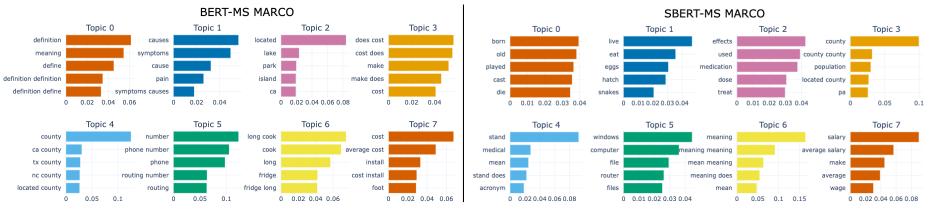

Fig. 6. Top 8 MSMARCO 300k topics mined using BERTopic with BERT/SBERT and their corresponding most frequent terms. A total of 182 topics were identified with BERT and 193 with SBERT from D_0.

5.6 Layer-wise Prompting Study

To determine the optimal layers for prompting (i.e., prefix-tuning), we follow the protocol of [39]. We assume prompting layers are contiguous and limit the search space to three layers. Using SBERT-based PromptDSI, we attach prompts to each layer of θ_e to identify effective layers for multi-layer prompting. Given that PromptDSI variants improve A_5 at the cost of a slight degradation in D_0, we define our main criteria as follows: (1) *Lower forgetting on* D_0, (2) *Moderate performance on* A_5, and (3) *Fewer prompting layers are preferable*. For single-layer prompting, Figure 7 (left) shows that CODA retains more knowledge but performs worse than L2P and S++. Overall, PromptDSI outperforms IncDSI across prompting layers. However, L2P and S++ exhibit severe forgetting in the top layers (10-12), leading us to exclude them from further analysis. Notably, layer 2 achieves comparable or lower forgetting than IncDSI on D_0 while maintaining strong A_5, making it a promising choice. We select S++ for

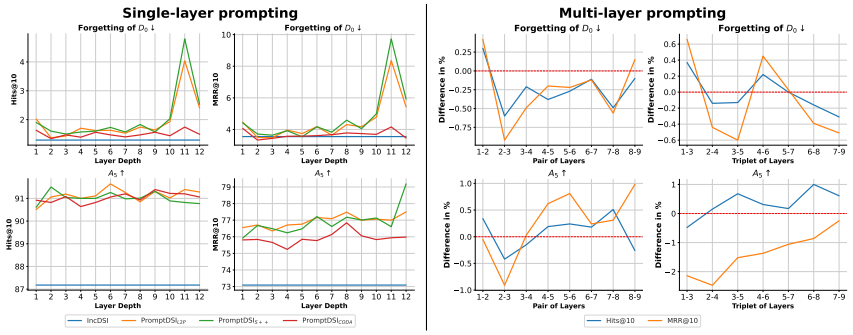

Fig. 7. Layer-wise prompting analysis of SBERT-based PromptDSI on NQ320k. **Left half:** Single-layer prompting analysis of different PromptDSI variants. **Right half:** Multi-layer prompting analysis of SBERT-based PromptDSI$_{S++}$, illustrating the performance changes (i.e., percentage gains and drops) when using 2-layer prompting (first column) and 3-layer prompting (second column) compared to single-layer prompting at layer 1, indicated by the red dashed line.

multi-layer prompting analysis due to its high forgetting on D_0. As shown in Figure 7 (right), two-layer prompting (first column) slightly reduces forgetting, but the gains in A_5 are marginal ($< 1\%$ for MRR@10, $< 0.5\%$ for Hits@10). Three-layer prompting significantly degrades MRR@10 (up to 2.5%) despite increased computation, offering minimal reduction in forgetting. Since multi-layer prompting incurs higher computational costs, these findings suggest that single-layer prompting, particularly in the lower layers (1-5), achieves the best trade-off between efficiency and performance. Consequently, we primarily perform single-layer prefix-tuning on layer 2 of DSI's encoder in our experiments, which satisfies all desirable criteria.

6 Conclusion

We present PromptDSI, a novel prompt-based approach for rehearsal-free continual learning in document retrieval. We adapt prompt-based continual learning (PCL) methods and introduce single-pass PCL. We validate our approach on three standard PCL methods. Our topic-aware prompt pool addresses parameter underutilization, ensuring diverse and efficient prompt usage, while also enhancing interpretability. Results show that PromptDSI outperforms rehearsal-based DSI++ and matches IncDSI in mitigating forgetting, while achieving superior performance on new corpora.

Acknowledgments. This material is based on research sponsored by Defense Advanced Research Projects Agency (DARPA) under agreement number HR0011-22-2-0047. The U.S. Government is authorised to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as

necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. This work is supported by the Australian Research Council Discovery Early Career Researcher Award DE250100032 and Monash eResearch capabilities, including M3.

References

1. Bajaj, P., Campos, D., Craswell, N., et al.: Ms marco: A human generated machine reading comprehension dataset. arXiv preprint arXiv:1611.09268 (2016)
2. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. NeurIPS (2020)
3. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. In: ICLR (2019)
4. Chen, J., Zhang, R., Guo, J., de Rijke, M., Chen, W., Fan, Y., Cheng, X.: Continual learning for generative retrieval over dynamic corpora. In: CIKM (2023)
5. Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Lin, J.: Overview of the trec 2021 deep learning track. In: TREC (2021)
6. Custers, B., Sears, A.M., Dechesne, F., Georgieva, I., Tani, T., Van der Hof, S.: EU personal data protection in policy and practice. Springer (2019)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
8. French, R.M.: Catastrophic forgetting in connectionist networks. Trends in cognitive sciences (1999)
9. Gerald, T., Soulier, L.: Continual learning of long topic sequences in neural information retrieval. In: ECIR (2022)
10. Grootendorst, M.: Bertopic: Neural topic modeling with a class-based tf-idf procedure. arXiv preprint arXiv:2203.05794 (2022)
11. Gupta, P., Chaudhary, Y., Runkler, T., Schuetze, H.: Neural topic modeling with continual lifelong learning. In: ICML (2020)
12. Karpukhin, V., Oguz, B., Min, S., et al.: Dense passage retrieval for open-domain question answering. In: EMNLP (2020)
13. Kim, Y., Li, Y., Panda, P.: One-stage prompt-based continual learning. In: ECCV (2024)
14. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al.: Overcoming catastrophic forgetting in neural networks. PNAS (2017)
15. Kishore, V., Wan, C., Lovelace, J., Artzi, Y., Weinberger, K.Q.: Incdsi: Incrementally updatable document retrieval. In: ICML (2023)
16. Kwiatkowski, T., Palomaki, J., Redfield, O., et al.: Natural questions: A benchmark for question answering research. TACL (2019)
17. Lee, C., Roy, R., Xu, M., et al.: Nv-embed: Improved techniques for training llms as generalist embedding models. In: ICLR (2025)
18. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691 (2021)
19. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190 (2021)
20. Li, Z., Hoiem, D.: Learning without forgetting. TPAMI (2017)
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
22. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. In: Psych. Learn. Motiv. Elsevier (1989)

23. Mehta, S., Gupta, J., Tay, Y., et al.: DSI++: Updating transformer memory with new documents. In: EMNLP (2023)
24. Moon, J.Y., Park, K.H., et al.: Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning. In: ICCV (2023)
25. Nogueira, R., Lin, J., Epistemic, A.: From doc2query to doctttttquery. Online preprint (2019)
26. Pradeep, R., Hui, K., Gupta, J., Lelkes, A., Zhuang, H., Lin, J., Metzler, D., Tran, V.: How does generative retrieval scale to millions of passages? In: EMNLP (2023)
27. Raffel, C., Shazeer, N., Roberts, A., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR (2020)
28. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: EMNLP-IJCNLP (2019)
29. Robertson, S., Zaragoza, H., et al.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval (2009)
30. Smith, J.S., Karlinsky, L., Gutta, V., et al.: Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In: CVPR (2023)
31. Sun, W., Yan, L., Chen, Z., Wang, S., Zhu, H., Ren, P., Chen, Z., Yin, D., Rijke, M., Ren, Z.: Learning to tokenize for generative retrieval. NeurIPS (2023)
32. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. NeurIPS (2014)
33. Tay, Y., Tran, V., Dehghani, M., et al.: Transformer memory as a differentiable search index. NeurIPS (2022)
34. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. NeurIPS (2017)
35. Wang, L., Xie, J., Zhang, X., et al.: Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. NeurIPS (2023)
36. Wang, L., Zhang, X., Su, H., Zhu, J.: A comprehensive survey of continual learning: Theory, method and application. TPAMI (2024)
37. Wang, Y., Huang, Z., Hong, X.: S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. NeurIPS (2022)
38. Wang, Y., Hou, Y., Wang, H., Miao, Z., Wu, S., Chen, Q., Xia, Y., Chi, C., Zhao, G., Liu, Z., et al.: A neural corpus indexer for document retrieval. NeurIPS (2022)
39. Wang, Z., Zhang, Z., Ebrahimi, S., et al.: Dualprompt: Complementary prompting for rehearsal-free continual learning. In: ECCV (2022)
40. Wang, Z., Zhang, Z., Lee, C.Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., Pfister, T.: Learning to prompt for continual learning. In: CVPR (2022)
41. Wu, T., Caccia, M., Li, Z., Li, Y.F., Qi, G., Haffari, G.: Pretrained language model in continual learning: A comparative study. In: ICLR (2021)
42. Xiao, S., Liu, Z., Zhang, P., et al.: C-pack: Packed resources for general chinese embeddings. In: SIGIR (2024)
43. Xiong, L., Xiong, C., Li, Y., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020)
44. Yadav, P., Sun, Q., Ding, H., et al.: Exploring continual learning for code generation models. In: ACL (2023)
45. Zeng, H., Luo, C., Jin, B., Sarwar, S.M., Wei, T., Zamani, H.: Scalable and effective generative information retrieval. In: TheWebConf (2024)
46. Zhang, P., Liu, Z., Zhou, Y., Dou, Z., Liu, F., Cao, Z.: Generative retrieval via term set generation. In: SIGIR (2024)
47. Zhuang, S., Ren, H., Shou, L., et al.: Bridging the gap between indexing and retrieval for dsi with query generation. arXiv preprint arXiv:2206.10128 (2022)