DSI++: UPDATING TRANSFORMER MEMORY WITH NEW DOCUMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

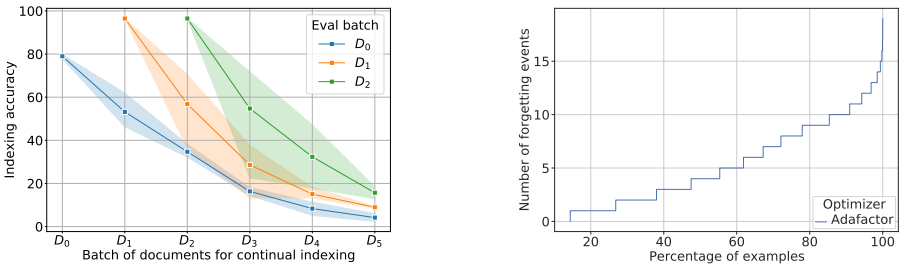
Differentiable Search Indices (DSIs) encode a corpus of documents in the parameters of a model and use the same model to map queries directly to relevant document identifiers. Despite the solid performance of DSI models, successfully deploying them in scenarios where document corpora change with time is an open problem. In this work, we introduce DSI++, a continual learning challenge for DSI with the goal of continuously indexing new documents while being able to answer queries related to both previously and newly indexed documents. Across different model scales and document identifier representations, we show that continual indexing of new documents leads to considerable forgetting of previously indexed documents. We also hypothesize and verify that the model experiences forgetting events during training, leading to unstable learning. To mitigate these issues, we investigate two approaches. The first focuses on modifying the training dynamics. Flatter minima implicitly alleviates forgetting, so we explicitly optimize for flatter loss basins and show that the model stably memorizes more documents (+12%). Next, we introduce a parametric memory to generate pseudo-queries for documents and supplement them during incremental indexing to prevent forgetting for the retrieval task. Extensive experiments on a novel continual indexing benchmarks based on Natural Questions (NQ) and MS MARCO demonstrate that our proposed solution mitigates the forgetting in DSI++ by a significant margin and improves the average Hits@10 by $+21.1\%$ over competitive baselines for NQ.

1 INTRODUCTION

Differentiable Search Indices (DSIs; Tay et al. (2022)) represent a new modeling paradigm for information retrieval tasks using sequence-to-sequence learning. Specifically, DSIs leverage Transformer memory (Vaswani et al., 2017) to encode all of the information in a corpus of documents and then use that memory to answer user queries directly, thereby simplifying the retrieval process. DSIs achieve this functionality by jointly optimizing for indexing (or memorization) and retrieval tasks. The indexing task requires learning a mapping from document content to its identifier, typically represented by integers or short strings (document identifiers, abbreviated *docids*). Then, the retrieval task necessitates mapping user queries to relevant docids. Apart from its simplicity and end-to-end differentiable nature, DSI significantly outperforms state-of-the-art "retrieve-and-rank" methods, based on dual-encoders (Ni et al., 2022).

Despite the remarkable performance of DSI models, there remain open questions about their applicability in the practical setting of dynamic corpora. Consider the realistic scenario wherein new documents are continually added to the indexed corpus. Updating the index in dual-encoder based methods requires computing embeddings for new documents, followed by re-indexing all document embeddings (Karpukhin et al., 2020). In contrast, index construction using a DSI involves training a Transformer model. Therefore, the model must be re-trained from scratch every time the underlying corpus is updated, thus incurring prohibitively high computational cost in comparison to dual-encoders. In this work, we aim to address this issue by devising methods for effective incremental indexing using Transformer memory, without re-training the DSI model from scratch.

Continual (or incremental) learning (Thrun, 1995; Parisi et al., 2019) is a biologically-inspired machine learning paradigm that deals with continuous learning of new tasks by preserving past knowledge and using it to efficiently learn new concepts. Based on this paradigm, we propose *DSI++ (DSI*



(a) **Explicit** forgetting during continual indexing (b) **Implicit** forgetting during memorization

Figure 1: (a) Indexing accuracy of D_0, D_1, and D_2 document corpora visualized as we continuously index new documents (averaged over 3 runs). We observe that continual indexing of new documents leads to severe forgetting of the previously memorized documents. (b) Cumulative histogram of forgetting events, i.e., events when individual documents go from being classified correctly to incorrectly, over the course of memorization of the initial D_0 corpus (T5-Base w/ Adafactor optimizer).

+ *new documents)*, a continual learning challenge for DSI to incrementally index new documents while maintaining the ability to answer user queries related to both previously and newly indexed documents. To enable DSI++, we introduce a novel benchmark constructed from the existing Natural Questions (Kwiatkowski et al., 2019) dataset, simulating continual addition of documents to the system. To the best of our knowledge, there is no prior work studying incremental learning for DSI.

A naive solution for DSI++ is to continuously fine-tune the model with an indexing objective over new documents. However, Figure 1a shows that continual indexing of new documents leads to catastrophic forgetting of the previously memorized documents (more details in §2.1), a common phenomenon in connectionist networks wherein learning of the new concepts interferes with the previously acquired knowledge (McCloskey & Cohen, 1989; French, 1999). Furthermore, when we investigate the learning dynamics of the DSI model during memorization (Figure 1b, we observe a significant number of documents (approx. 88%) experience forgetting events after they have been memorized. Concretely, a *forgetting event* (Toneva et al., 2019) is defined as when an individual document goes from being classified correctly to incorrectly over the course of learning. Therefore, *implicit forgetting* during memorization and *explicit forgetting* from continual indexing of new documents are two key challenges that must be overcome to successfully implement a DSI++ system.

Mirzadeh et al. (2020) show that geometrical properties of the minima play a role in forgetting. In particular, models in flatter minima tend to undergo less forgetting. Further, Mehta et al. (2021) showed that explicitly optimizing for flatter loss basins using Sharpness-Aware Minimization (SAM; Foret et al. (2021)) reduces forgetting. Building on these works, we show that flatter minima induced by SAM reduce implicit forgetting during memorization, thereby leading to more stable memorization. Next, to alleviate explicit forgetting from continual indexing of new documents, we investigate how to effectively index both old and new documents. We propose leveraging a parametric memory to generate pseudo-queries for old and new documents and use them during continual indexing to mitigate retrieval task forgetting. Our main contributions can be summarized as follows:

- We introduce DSI++, a continual learning challenge for Differentiable Search Indexes. To enable DSI++ evaluations, we create a benchmark based on the existing Natural Questions (Kwiatkowski et al., 2019) dataset. To understand the severity of the forgetting phenomenon across multiple scenarios, we analyze a suite of pre-trained models (T5-Base, T5-Large, T5-XL) and different document identifier representations.
- We hypothesize and verify that the DSI model experiences forgetting events throughout memorization. To alleviate these, we propose modifying training dynamics to promote flatter minima using SAM and show that the model stably memorizes $+12\%$ documents.
- We propose generative memory-based experience rehearsal approach to alleviate explicit forgetting during continual indexing and improve the average Hits@10 by $+21.1\%$ over considered baselines.

2 DSI++: CONTINUAL LEARNING CHALLENGE FOR DSI

2.1 PROBLEM SETUP

We focus on a setup where we receive an initial corpus of documents, $D_0 = \{d_1, \cdots, d_n\}$, and user queries corresponding to a subset of them, $R_0 = \{< q_j, j >, \forall j \in \mathcal{Y}_D\}$, where $D \subset D_0$. DSI paradigm involves two tasks: (i) *memorization task* where the goal is to learn an indexer $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$, a text-to-text model like T5 (Raffel et al., 2020) parameterized by $\theta \in \mathbb{R}^P$, that takes document tokens ($x \in \mathcal{X}$) as input and maps it to a document identifier (docid) $j \in \mathcal{Y}$, and (ii) *retrieval task* where the goal is to use the same indexer f_θ to directly map a user query q to a relevant docid $j \in \mathcal{Y}$. Following Tay et al. (2022), two different prompts are used to differentiate between these tasks.

Tay et al. (2022) discuss several variants for representing docids – *unstructured atomic* and *structured string* docids, where each document is assigned a unique token and tokenized string, respectively. Under the unified text-to-text format, both of the above tasks are cast as generation tasks, i.e., decoding one unique token (unstructured atomic) or decoding a tokenized string sequentially, one token at a time (naively/ semantically structured).

For DSI evaluation, we report *indexing accuracy* for memorization task and *Hits@N* ($N \in \{1, 10\}$) metric for retrieval task. Indexing accuracy and Hits@N are defined as the proportion of documents correctly memorized and the proportion of correct documents ranked in the top N predictions, respectively. To simulate a dynamic corpus scenario, we sequentially update the D_0 corpus with batches of documents $D_1, D_2, ...$ where D_1 arrives first followed by D_2, and so on. Moreover, with the new D_i corpus, we do not assume any queries corresponding to documents.

Goal: Learn a DSI++ system that incrementally indexes $D_1, D_2, \cdots$ in f_θ while being able to answer queries related to previously as well as additionally indexed documents.

2.2 BENCHMARK FOR DSI++.

To enable research on DSI++, we introduce a new benchmark constructed from the Natural Questions dataset (NQ; Kwiatkowski et al. (2019)). The NQ dataset consists of Wikipedia articles and corresponding natural language questions. Similar to Tay et al. (2022), we consider Wikipedia articles for memorization and the retrieval task as identifying the Wikipedia article that answers the given question. We use the original NQ train split to construct train(80%)/ validation(20%) splits and use NQ validation as a test split for our setup. We randomly sample $50K$ unique articles to constitute the initial D_0 corpus. Next, we construct five corpora ($D_1, \cdots, D_5$), each containing $10K$ unique articles, to sequentially add them to the DSI model. Corresponding to articles in each of these corpora, we filter queries from original NQ train/ validation splits to construct $R_i^{train}, R_i^{val}, R_i^{test}$ ($\forall i \in \{0, \cdots, 5\}$) splits. Except for R_0, one only requires R_i^{test} to evaluate for retrieval on previously and newly indexed articles. Table 2 (in Appendix A.1) reports exact dataset statistics. Similarly, we construct another benchmark from the MS MARCO dataset (Nguyen et al., 2016).

2.3 EVALUATION METRICS

In this subsection, we formally define metrics to summarize the model performance as we incrementally index new documents. Let $P_{n,o}$ denote the performance (e.g., indexing accuracy) on corpus o after training on corpus n. Following prior works (Lopez-Paz & Ranzato, 2017; Riemer et al., 2019), we compute the **average performance** (A_n), **forgetting** (F_n) and **learning performance** (LA_n) metrics after indexing the corpus n.

F_n (or *backward transfer*) measures the influence of continual indexing the corpus n on the performance of all previously indexed documents $o, (0 \leq o < n)$. LA_n measures the learning capability when the model sees a new corpus n (or *forward transfer*) and is defined to be average over new corpora $(1, \cdots, n)$. Say we incrementally index n^{th} corpus, then A_n, F_n and LA_n are defined as follows:

$$A_n = \frac{1}{n+1} \sum_{o=0}^{n} P_{n,o}; \quad F_n = \frac{1}{n} \sum_{o=0}^{n-1} \max_{o' \in \{0, \cdots, n-1\}} (P_{o',o} - P_{n,o}); \quad LA_n = \frac{1}{n} \sum_{o=1}^{n} P_{o,o} \tag{1}$$

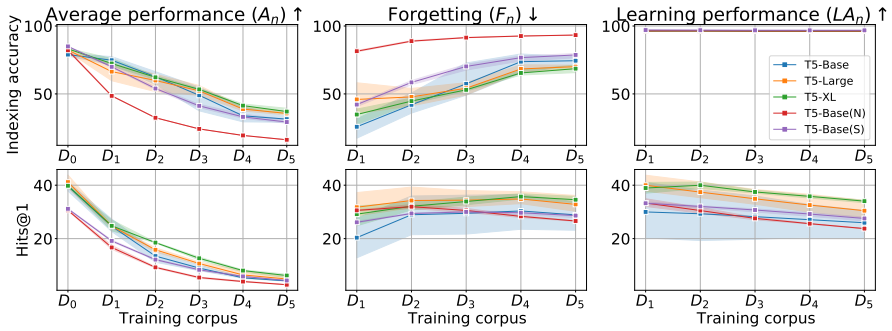


Figure 2: Systematic study about forgetting and forward transfer when incrementally indexing new corpus of documents across different model sizes (T5-Base, T5-Large, T5-XL) and docid representations. We use atomic docids by default and denote (N)/(S) for naively/ semantically structured docids. ↑ indicates higher is better, ↓ indicates lower is better. We observe that by increasing the model scale, the average A_n and learning LA_n performance improves. However, forgetting F_n is severe across all model scales. Moreover, we observe that naively structured docids, T5-Base(N), underperform unstructured atomic docids, T5-Base, across all metrics - indexing accuracy, Hits@1, (see Figure 5 in Appendix A.3 for Hits@10 results). Imbuing the docid space with semantic (S) structure alleviates the forgetting compared to an arbitrary/ naive (N) structure.

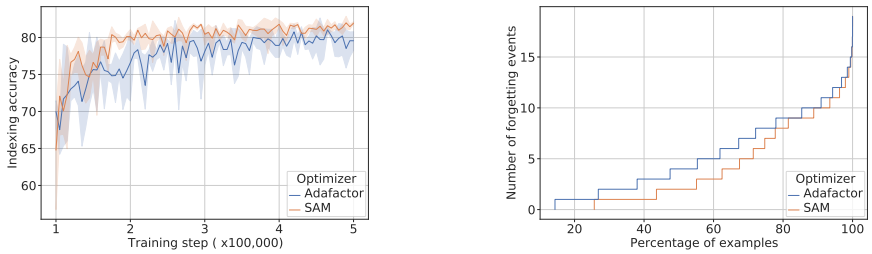
2.4 Case study: Catastrophic Forgetting and Forward Transfer

After introducing the DSI++ problem setup, benchmark, and evaluation metrics, we study the behavior of the DSI model as new documents are continuously added to the system. Concretely, we are interested in investigating the following for continual training of the DSI model with indexing objective on new documents – (Q1) How severe is the forgetting for the originally indexed documents?, (Q2) How does continual updating of the DSI model over a sequence of corpora affect the forgetting?, (Q3) How does the updated DSI model perform on newly indexed documents, especially the retrieval task?, (Q4) How do different docid representation strategies affect forgetting?, and (Q5) How does the DSI model scale affect forgetting? Figure 2 visualizes results (averaged over 3 runs) on the validation split of DSI++ and help us convincingly answer these questions.

Forgetting (or negative backward transfer). From Figure 2, we see that T5-Base model with atomic docid representation (blue line plots), undergo significant forgetting and this trend holds across all DSI evaluation metrics - indexing accuracy, Hits@1, and Hits@10 (see 5 in Appendix A.3). For the originally indexed D_0 corpus, indexing accuracy and Hits@1 drop by approx. 25 and 20 points, respectively. Further, as we continue indexing the sequence of $D_1, \cdots, D_5$ corpora, we see that forgetting becomes even more severe. For example after continual indexing the D_5 corpus, F_5 (forgetting) for indexing accuracy increases to 75. These results provide evidence to answer (Q1) & (Q2) that the DSI model undergoes severe forgetting under continual indexing of new documents.

Forward transfer. To answer (Q3), we visualize the learning performance (LA_n) for all DSI metrics for sequential indexing. From Figure 2, we see LA_n increases for indexing accuracy, suggesting that the DSI model is plastic enough to index new documents. However, from Figure 2, we see a declining trend for Hits@1. Due to the continuous indexing updates, the underlying DSI model drifts and becomes less effective for the retrieval task. These findings hint at an approach that replays indexing and retrieval tasks during continual learning (hence our proposed method in Section 4).

Docid representations. For studying (Q4), we consider unstructured atomic, naively(N) structured, and semantically(S) structured docid representations. From Figure 2, we see that T5-Base(N) underperforms T5-Base by a significant margin. For example, average performance A_0 for Hits@1 metric is approx. 30 and 39 for naive and atomic docids, respectively. Furthermore, as the naively structured approach treats unstructured docids as tokenizable strings as opposed to dedicated unique tokens in the case of atomic docids, they are relatively more prone to interference from new docids (see F_n subplot for indexing accuracy). Imbuing semantic structure to the naive docid space helps to reduce forgetting however still underperforms unstructured atomic docids.



(a) Indexing accuracy during memorization (b) Cumulative histogram of forgetting events

Figure 3: Investigating the effectiveness of SAM for alleviating implicit forgetting in the T5-Base model. (a) We observe serious fluctuations in the indexing accuracy in the case of the Adafactor optimizer, thereby, suggesting unstable memorization. SAM leads to relatively stable memorization of the documents. (b) A forgetting event (Toneva et al., 2019) is defined when an individual document goes from being classified correctly to incorrectly over the course of memorization. SAM increases the percentage of examples experiencing zero forgetting events by absolute 12% over Adafactor.

Model scale. As atomic docids are superior to naive docids, we only consider atomic docids for answering (Q5). From Figure 2, we observe that larger models outperform their smaller counterparts in terms of the average performance A_n and the learning performance LA_n (T5-XL $>$ T5-Large $>$ T5-Base). However, empirically we report that forgetting F_n is severe across all model scales, without any clear best performer, and therefore, we focus on T5-Base for the rest of our experimentation.

3 Implicit Forgetting during Memorization: SAM

Memorization (or indexing) is a primary task in the DSI paradigm (Tay et al., 2022) where the goal is to learn a neural corpus indexer that takes document content as input and maps it to a document identifier (docid). Under unstructured atomic docid representation strategy, each docid is assigned a unique token / class label. Now given the large number of documents in the corpus (even more than a million), memorization constitutes an instance of challenging extreme classification setting (Bengio et al., 2019). Furthermore, for every class, we have only one labeled example (i.e., document and its identifier), making this task setup rare. Motivated by this largely unexplored setup, we investigate the learning dynamics for the memorization task over the course of training.

Forgetting events. In Figure 3a, we visualize the indexing accuracy for T5-Base model, optimized with Adafactor (Shazeer & Stern, 2018), and note that the model performance fluctuates a lot over the course of training, thereby suggesting unstable memorization. We hypothesize that the model continuously undergoes the forgetting phenomenon wherein subsequent mini-batch updates interfere with the previously memorized documents. To differentiate this phenomenon from forgetting due to adding new documents, we refer to earlier one as *implicit forgetting* and the latter as *explicit forgetting*. To quantify instability during memorization, we compute forgetting event (Toneva et al., 2019) statistics. *Forgetting event* is defined when an individual document goes from being classified correctly to incorrectly over the course of memorization. In Figure 3b, we plot the cumulative histogram of forgetting events and see that almost 88% of the documents undergo forgetting at least once, thus, validating our hypothesis about implicit forgetting.

Flatter minima and forgetting. Mirzadeh et al. (2020) shows that during sequential learning of tasks, geometric properties of the minima for each task affect forgetting. Specifically, Mirzadeh et al. (2020) derives an upper bound for forgetting in terms of the maximum eigenvalue λ_1^{max} of the Hessian for task loss at minima. Therefore, by lowering λ_1^{max} (or alternatively promoting flatter minima), one can mitigate forgetting of previous tasks. Further, Mehta et al. (2021) shows that pre-trained initialization implicitly alleviate forgetting as they prefer flatter minima and explicitly optimizing for the flatter loss basins using Sharpness-Aware Minimization (Foret et al., 2021) (SAM) further lessens forgetting. Based upon these observations, we hypothesize that modifying the training dynamics of memorization task using SAM should alleviate implicit forgetting.

Sharpness-Aware Minimization. For the loss function f, SAM seeks to find the parameters w that lie in the neighborhood with uniformly low loss regions by optimizing the following minimax objective: $\min_w \max_{||\epsilon||_2 \leq \rho} f(w+\epsilon)$, where the maximization region is defined to be a ℓ^p ball with radius ρ for $p = 2$. Foret et al. (2021) estimates the gradient of the inner maximization by employing first-order approximation as follows: $\nabla_w \max_{||\epsilon||_2 \leq \rho} f(w+\epsilon) \approx \nabla_w f(w)\big|_{w+\hat{\epsilon}(\mathbf{w})}$, where $\hat{\epsilon}(\mathbf{w}) = \rho \nabla_w f(w) / ||\nabla_w f(w)||_2$. For complete details about this derivation, we defer readers to (Foret et al., 2021). Building on this work, Bahri et al. (2022) demonstrates successful applicability of SAM for language model generalization, especially in pre-trained T5 models. We mainly follow (Bahri et al., 2022) to set our hyper-parameters: $\rho = 0.15$, batch size=32 for inner maximization step.

SAM alleviates Implicit forgetting. We investigate the applicability of SAM for alleviating implicit forgetting phenomenon. Concretely, we use pre-trained T5-Base model to memorize D_0 corpus, containing 50K unique documents. We compare the performance of the SAM optimizer with the vanilla Adafactor optimizer. In Figure 3a, we see that SAM outperforms Adafactor in terms of the overall indexing accuracy. Further, we note that SAM undergoes less severe fluctuations during the course of training, thus, hinting at lesser forgetting. To bolster this claim, in Figure 3b, we see that SAM has significant more percentage of documents corresponding to lower cumulative number of forgetting events. For example, SAM stably (with zero forgetting events) memorizes $+12\%$ more documents compared to Adafactor. We also note that SAM (35.9 ± 2.2) outperforms Adafactor (32.5 ± 6.4) when evaluated on the retrieval task (Hits@1) corresponding to D_0. Therefore, we set SAM to be our default optimizer for rest of the experiments.

4 Explicit Forgetting: Generative Memory

DSI paradigm consists of two tasks – memorization and retrieval. In the previous section, we showcase that SAM alleviates implicit forgetting by stably memorizing documents. In this section, we focus on the forgetting phenomenon that arises from the continual indexing of new documents, specifically in the context of the retrieval task. Through our systematic study (in Section 2.4), we show that irrespective of the model scale and docid representations, DSI models undergo severe forgetting. Moreover, we observe that the learning performance LA_n keeps declining for the retrieval task (see Figures 2 and 5 for Hits@1 and Hits@10, respectively). This observation suggests that as we continuously update the DSI model with the indexing objective, the model starts to forget the retrieval task. In DSI, both memorization and retrieval tasks return docid for the given input, so by setup, we can assume access to the contents of the previous documents and continue indexing both old and new documents with the hope to reduce forgetting of the retrieval task. However, in Figure 4, we see that the model still undergoes forgetting (more discussion in Section 5.3).

Episodic memory. According to the Complementary Learning Systems (CLS) (McClelland et al., 1995) theory, humans rely on an *episodic memory* – a module that stores past experiences to rehearse/ revisit them and thereby retain previously learned knowledge. Based on this motivation, memory-based approaches (Sodhani et al., 2022) for continual learning use a subset of previous task data to regularize the future task learning while minimizing forgetting. Experience Replay (ER) (Chaudhry et al., 2019) is one such approach that samples previous task data (from episodic memory) to co-train with the current task. Based upon this line of work, one approach for DSI++ is to retain ground-truth queries for the retrieval task in an episodic memory and use them to co-train with incremental indexing task. However, in DSI++, we do not have access to ground-truth queries for an incoming stream of new documents. Even if one retains queries for the initial D_0 corpus, we show in Table 1 that such a method suffers from forward transfer to newly indexed documents.

Generative memory. Recent years have seen significant progress in the capabilities of the generative language models (Raffel et al., 2020; Brown et al., 2020). Motivated by the success of these models and in-applicability of the episodic memory for DSI++, we pose a questions – *instead of retaining the ground-truth queries, can we learn a parametric model to generate such queries given a document?* Concretely, we propose to train a query generator model to sample queries for previously seen documents, and supplement them during incremental indexing. Since we use the generator model to sample queries for sparse experience replay, hence our proposed method – *generative memory*. Moreover, generative memory is also used to generate pseudo-queries for incoming batch of new documents, thus, enabling semi-supervised learning of the retrieval task.

Added corpus	Method	Eval corpus = D_0 (Catastrophic forgetting)			Eval corpus = D_1 (Forward transfer)		
		Index acc.	Hits@1	Hits@10	Index acc.	Hits@1	Hits@10
D_0	-	$81.8_{1.2}$	$35.9_{2.2}$	$66.9_{0.9}$	-	-	-
D_1	cl(D_1)	$52.4_{3.5}$	$19.2_{3.9}$	$43.6_{5.7}$	$96.5_{0.0}$	$31.7_{6.4}$	$55.6_{4.9}$
	cl($U_1 = D_0 \cup D_1$)	$78.2_{0.5}$	$28.9_{8.9}$	$59.0_{7.9}$	$91.8_{0.4}$	$34.0_{2.4}$	$60.2_{1.9}$
D_1	cl(U_1)+epsmem(D_0)	$77.8_{0.5}$	$22.9_{1.5}$	$51.4_{0.5}$	$93.1_{0.0}$	$13.1_{2.1}$	$39.6_{3.1}$
	cl(U_1)+genmem(D_0)	$77.8_{0.3}$	$26.0_{6.9}$	$54.9_{8.3}$	$93.0_{0.5}$	$8.6_{4.8}$	$31.6_{11.8}$
D_1	cl(U_1)+epsmem(D_1)	$53.2_{3.1}$	$7.7_{2.1}$	$26.0_{2.0}$	$96.5_{0.0}$	$48.3_{2.3}$	$70.7_{1.9}$
	cl(U_1)+genmem(D_1)	$50.1_{0.8}$	$7.0_{1.2}$	$23.1_{2.2}$	$96.5_{0.0}$	$57.7_{1.5}$	$76.7_{0.9}$
D_1	cl(U_1)+genmem(U_1)	$78.2_{0.3}$	$18.4_{2.8}$	$47.5_{3.9}$	$92.1_{0.3}$	$48.5_{6.1}$	$73.8_{2.9}$

Table 1: Comparing performance on incremental indexing of D_1 corpus across different methods - cl(D_1): continue fine-tuning with indexing task on D_1, cl(U_1): continue fine-tuning on the updated corpus U_1, cl(U_1)+epsmem(D): continual indexing of U_1 along with ER of queries for D, cl(U_1)+genmem(D): continual indexing of U_1 along with ER of pseudo-queries for D. We observe that continual indexing on the updated corpus cl(U_1) reduces forgetting in comparison with just indexing new corpus cl(D_1). Next ER with either D_0 or D_1 hurts forward transfer or forgetting, respectively. Our proposed approach of augmenting pseudo-queries for all documents along with continual indexing, cl(U_1)+genmem(U_1), alleviates forgetting and improves forward transfer.

5 Experimentation

5.1 Implementation details

We utilize the pre-trained T5-Base (Raffel et al., 2020) model to initialize all models and randomly initialize the additional parameters for atomic docid tokens. While indexing D_0 corpus, we train all the models for a maximum of 1M steps with a warmup of 100K steps. During continual indexing of other corpora, we train for a maximum of 100K steps with a warmup of 100 steps. For the rest of hyper-parameters, we follow Tay et al. (2022) – set a learning rate to 0.001, batch size to 128, and input sequence length to 32. We evaluate models after every 5K steps and retain the checkpoint yielding the best performance. For the initial training with D_0 corpus, we co-train on indexing and retrieval tasks, therefore we use the average of all DSI metrics (indexing accuracy, Hits@1, and Hits@10) for model selection. For the continual learning experiments, we have access to only indexing accuracy for all involved corpora and so we use it for model selection. For training generative memory, we use retrieval dataset R_0 (corresponding to D_0 corpus). We set the maximum sequence length for document contents to 1024, target length for generated queries to 32, batch size to 128, train for a maximum of 100K steps, and use BLUE for model selection. We use beam decoding to generate pseudo-queries. We tune the learning rate amongst $\{0.001, 0.0005\}$ and linear warmup amongst $\{1K, 10K\}$. For all our experiments, we use T5X (Roberts et al., 2022) framework along with 4-8 TPUv4 chips for training the models.

5.2 Methods

We compare our proposed generative memory with following methods:

• **continual indexing, cl($\mathbf{D_n}$).** The DSI model is sequentially fine-tuned with the indexing objective on the incoming corpus of documents D_n.

• **continual indexing with all seen documents, cl($\mathbf{U_n}$).** The DSI model is continuously fine-tuned with the indexing objective on the updated corpus U_n ($= \bigcup_{i=0}^{n} D_i$). Also, we sample documents from old ($\bigcup_{i=0}^{n-1} D_i$) and new (D_n) corpora in equal proportion.

• **experience replay using generative memory, genmem($\mathbf{D_n}$).** In this method, the proposed generative memory model is used to generate pseudo-queries corresponding to the corpus D_n. Next, these pseudo-queries are used for experience replay of the retrieval task samples.

• **experience replay using episodic memory, epsmem($\mathbf{D_n}$).** In this method, ground-truth queries corresponding to the D_n^{th} corpus are used for experience replay of the retrieval task.

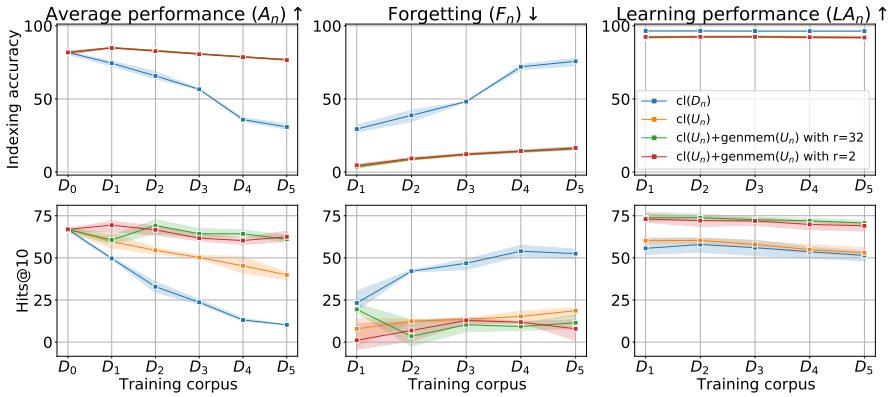


Figure 4: Investigating the effectiveness of generative memory in mitigating forgetting when continuously indexing new corpus D_n (T5-Base model and atomic docids representation). ↑ indicates higher is better, ↓ indicates lower is better. We observe that continual indexing of old and new documents cl(U_n) help to alleviate forgetting of older documents when evaluated on retrieval tasks. However, average Hits@10 (A_n) still undergo 23 points drop after sequential updates ($D_0 \rightarrow D_1 \cdots \rightarrow D_5$). Generative memory enables sparse replaying of old and new documents pseudo-queries. We observe that by augmenting generative memory during continual indexing not only reduces the forgetting (F_n) but also improves average Hits@10 (A_n) by $+21.1\%$ over considered baselines (see Figure 6 in Appendix A.3 for Hits@1 results).

5.3 Results

In this section, we revisit some of the questions, (Q1)-(Q3), raised in our case study (see Section 2.4) to investigate effectiveness of our proposed generative memory-based approach. Towards answer these questions, in Table 1, we report the performance of the DSI model on D_0 (to study forgetting phenomenon) and D_1 corpora (to answer forward transfer question) after continual indexing on D_1. In Figure 4, we report overall performance across DSI metrics as we continuously update the model with the sequence of five corpora ($D_1, \cdots, D_5$). See Table 3 for results on MS MARCO dataset.

Does generative memory alleviate forgetting of old documents? In Table 1, we report Hits@1 to be 35.9 for the model after training on D_0. We see that continually indexing both D_0 and D_1 corpora (cl(U_1) - 28.9), significantly reduce forgetting the retrieval task (Hits@1) over just indexing the new corpora D_1 (cl(D_1) - 19.2). Next, we look at the performance of the ER approaches when augmented with the continual indexing of all documents. We see that both episodic memory (cl(U_1)+epsmem(D_0) - 22.9), and generative memory (cl(U_1)+genmem(D_0) - 26.0) reduce forgetting compared to cl(D_1) when we replay (pseudo-)queries corresponding to D_0 corpus. Moreover, generative memory outperforms episodic memory, without retaining original queries. Although from Table 1, we see generative memory underperforms cl(U_1), from Figures 4 and 6, we see that generative memory outperforms cl(U_5) both in terms of average performance A_n and forgetting F_n over five sequential updates. *These results convincingly show that the ER with generative memory significantly alleviate forgetting the retrieval task compared to considered baselines.*

Does generative memory enable forward transfer to new documents? One of the goals of DSI++ is to enable answering queries related to newly indexed documents. Towards this goal, in Table 1, we look at the retrieval task performance (Hits@1 metric) for D_1 after incrementally indexing D_1. To compare different methods, we consider a baseline in the form of ER with ground-truth queries for D_1 (cl(U_1)+epsmem(D_1) - 48.3). We see that without any fine-tuning on the retrieval task for D_1, incremental learning with indexing objective shows impressive forward transfer (or zero-shot gains, cl(D_1) - 31.7 and cl(U_1) - 34.0). Moreover, ER with generative memory outperforms supervised baseline (cl(U_1)+genmem(D_1) - 57.7). However, we notice that replaying queries corresponding to either D_0 or D_1 hurt forward transfer to D_1 (cl(U_1)+genmem(D_0) - 8.6) or amplify forgetting of D_0 (cl(U_1)+genmem(D_1) - 7.0), respectively. These results suggest that memory module should include (pseudo-)queries corresponding to both old and new documents. From Figure 4, we see that

continual indexing method cl(U_n) has a downward trend for LA_n (Hits@10), therefore, eventually forgetting the retrieval task. On the other hand, ER with generative memory is relatively constant, providing evidence against forgetting. *In summary, we show that ER with the generative memory improves the overall performance for the retrieval task, reducing forgetting of previously indexed documents and enabling forward transfer to newly indexed documents.*

Investigating sparsity of experience replay (ER) on forgetting. ER with generative memory co-trains the indexing and pseudo-labeled retrieval tasks. Tay et al. (2022) introduces a mixing ratio *r* to define the ratio of indexing to retrieval samples. The mixing ratio is inversely related to the sparsity of ER, i.e., higher r (more indexing samples) corresponds to sparse updates from pseudo-labeled retrieval samples. Following (Tay et al., 2022), we consider $r = \{2, 32\}$ for our analysis. From Figure 4, we see that $r = 32$ (sparse replay) slightly outperforms $r = 2$ in terms of average performance, forgetting and learning accuracy. These results suggest that even sparse regularization updates from ER positively influence the backward and forward transfer in DSI++.

Analyzing index construction time for DSI++. Index construction using DSI involves training a Transformer model. DSI++ setup enables incremental updating of the indexer. We require 350K training steps to index D_0 corpus of 50K documents. If we index an additional D_1 to D_5 corpora (10K each) by re-training the DSI model every time, the total number of steps would be around 1.75M. On the other hand, our proposed approach requires just above 300K additional updates to incrementally index all five corpora, almost 6 times fewer updates.

6 Related Work

We review relevant prior works along two dimensions – application setups related to DSI++, and approaches to alleviate forgetting during continual learning. Petroni et al. (2019) views pre-trained language models as knowledge bases. Based on this Zhu et al. (2020) introduces an experimentation setup where task is to update facts stored within the pre-trained models. Next, De Cao et al. (2021) introduces, KnowledgeEditor, a hyper-network based method to efficiently update the facts. Although interesting line of works around fact updation, it is challenging to know whether pre-trained models have actually learned the fact or our probing mechanism is unable to extract the relevant facts. On the other hand, we explicitly focus on the memorization task in DSI++. This helps us to answer questions related to catastrophic forgetting more convincingly rather than bounded by the mechanism how we probe these models.

For a fixed capacity model – (1) Regularization-based approaches constrain the model updates by adding a penalty term to the original objective. However, these approaches are too stable and prevent learning new tasks, (2) Optimization-based approaches encode the necessary inductive biases required to enable continual learning by modifying the training dynamics. Flatter minima are shown to alleviate forgetting (Mirzadeh et al., 2020; Mehta et al., 2021). Our work extends on this line of research and showcases applicability of SAM for minimizing implicit forgetting during memorization (see Section 3). (3) Memory-based (aka data-based regularization) approaches constrain the parameter updates based upon the previous task examples sampled from memory. Sparse experience replay using episodic memory (Chaudhry et al., 2019) is a prominent approach and in Section 4, we discuss limitations of it for DSI++. Shin et al. (2017); Sun et al. (2020) learns a parametric model to reconstruct the examples from previous tasks. However, (Sun et al., 2020) shows that such approach still underperforms episodic memory. In our work, we do not generate example pairs (x, y) but rather generate pseudo-queries (y), similar to concurrent works (Zhuang et al., 2022; Bonifacio et al., 2022). We show that our approach outperforms episodic memory.

7 Conclusion

DSI++ presents a novel direction for exploration of DSI models to solve one of the critical requirements for them to be usable in most production setups where new documents are added to the corpus continuously. Though DSI models are prone to catastrophic forgetting while we index new documents, we have shown through extensive experiments that our two proposed solutions to optimize for the flatter loss basins using SAM and using generative memory alleviate forgetting to a significant extent. With this work, we lay down foundations for further research in this space, so that the strategies here do not just benefit DSI, but the continual learning community in general.

ETHICS STATEMENT

Training large models is expensive, and has a detrimental impact on the environment. Continual learning on top of existing models is cheaper and better compared to retraining from scratch since it requires a much smaller number of steps. With DSI++, we aim to reduce the need to retrain DSI models from scratch whenever a new set of documents is added to the corpus thereby making it cheaper and better for the environment.

REPRODUCIBILITY STATEMENT

We introduce a novel benchmark constructed from the existing and publicly available Natural Questions (NQ) (Kwiatkowski et al., 2019) and MS MARCO (Nguyen et al., 2016) datasets. In our benchmark, we split the NQ and MS MARCO datasets into different subsets to enable development of continual learning systems. We plan to release these splits so that they can act as a standard benchmark for future works in this space. Further, in Section 3 and Section 5.1, we detail all hyperparameters to enable reproducibility of all our experiments. We also plan to release our code on publication.

REFERENCES

Dara Bahri, Hossein Mobahi, and Yi Tay. Sharpness-aware minimization improves language model generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7360–7371, 2022.

Samy Bengio, Krzysztof Dembczynski, Thorsten Joachims, Marius Kloft, and Manik Varma. Extreme classification (dagstuhl seminar 18291). In *Dagstuhl Reports*, volume 8. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. Inpars: Data augmentation for information retrieval using large language models. *arXiv preprint arXiv:2202.05144*, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, 2021.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. *arXiv preprint arXiv:2112.09153*, 2021.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320, 2020.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPs*, 2016.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, 2022.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2463–2473, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019.

Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. Scaling up models and data with `t5x` and `seqio`. *arXiv preprint arXiv:2203.17189*, 2022.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Shagun Sodhani, Mojtaba Faramarzi, Sanket Vaibhav Mehta, Pranshu Malviya, Mohamed Abdelsalam, Janarthanan Janarthanan, and Sarath Chandar. An introduction to lifelong supervised learning. *arXiv preprint arXiv:2207.04354*, 2022.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. {LAMAL}: {LA}nguage modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*, 2020.

Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *arXiv preprint arXiv:2202.06991*, 2022.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, 8, 1995.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. Bridging the gap between indexing and retrieval for differentiable search index with query generation. *arXiv preprint arXiv:2206.10128*, 2022.

A Appendix

A.1 Dataset

Dataset	#D	#Train	#Validation	#Test
R_0	50K	53.8K	13.5K	3.9K
R_1	10K	10.7K	2.7K	809
R_2	10K	10.6K	2.7K	787
R_3	10K	10.7K	2.7K	727
R_4	10K	10.9K	2.7K	772
R_5	10K	10.7K	2.7K	847

Table 2: Dataset statistics for DSI++: memorization and retrieval tasks.

A.2 MS MARCO

In Table 3, we report Hits@1 to be 78.0 for the DSI model after training on D_0 (constructed from MS MARCO dataset). We see that continually indexing both D_0 and D_1 corpora (cl(U_1) - 75.9 and cl(U_1)+genmem(U_1) - 73.0), significantly reduce forgetting the retrieval task (Hits@1) over just indexing the new corpora D_1 (cl(D_1) - 65.5). Next, we look at the retrieval task performance (Hits@1 metric) for D_1 after incrementally indexing D_1. We see that without any fine-tuning on the retrieval task for D_1, incremental learning with indexing objective shows impressive forward transfer (or zero-shot gains, cl(D_1) - 47.6 and cl(U_1) - 43.6). Moreover, ER with generative memory, cl(U_1)+genmem(U_1) - 80.6, performs far superior to just incremental indexing objective.

In summary, we show that ER with the generative memory improves the overall performance for the retrieval task, reducing forgetting of previously indexed documents and enabling forward transfer to newly indexed documents. We show that our results hold across two datasets – Natural Questions and MS MARCO, thus, showcasing the generalizability of our approach.

Added corpus	Method	Eval corpus = D_0 (Catastrophic forgetting) Index acc.	Hits@1	Hits@10	Eval corpus = D_1 (Forward transfer) Index acc.	Hits@1	Hits@10
D_0	-	99.6	78.0	95.4	-	-	-
D_1	cl(D_1)	46.2	65.5	86.8	99.8	47.6	75.1
	cl($U_1 = D_0 \cup D_1$)	99.4	75.9	94.2	99.8	43.6	70.9
D_1	cl(U_1)+genmem(U_1)	99.4	73.0	94.6	99.8	80.6	94.6

Table 3: **MS MARCO:** Comparing performance on incremental indexing of D_1 corpus across different methods - cl(D_1): continue fine-tuning with indexing task on D_1, cl(U_1): continue fine-tuning on the updated corpus U_1, cl(U_1)+genmem(D): continual indexing of U_1 along with ER of pseudo-queries for D. We observe that continual indexing on the updated corpus cl(U_1) reduces forgetting in comparison with just indexing new corpus cl(D_1). Our proposed generative memory-based approach i.e., augmenting pseudo-queries for all documents along with continual indexing, cl(U_1)+genmem(U_1), alleviates forgetting (on D_0 corpus) and improves forward transfer (on D_1 corpus).

A.3 Additional Results

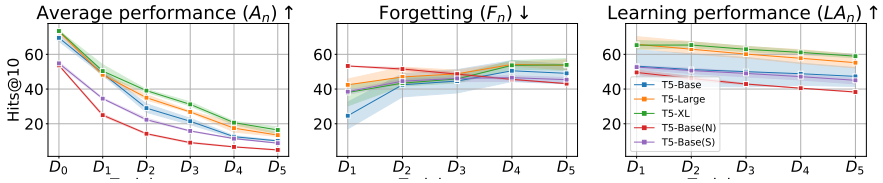


Figure 5: Systematic study about forgetting and forward transfer when incrementally indexing new corpus of documents across different model sizes (T5-Base, T5-Large, T5-XL) and docid representations. We use atomic docids by default and denote (N)/(S) for naively/semantically structured string docids. ↑ indicates higher is better, ↓ indicates lower is better. We observe that by increasing the model scale, the average A_n and learning LA_n performance improves. However, forgetting F_n is severe across all model scales. Moreover, we observe that naive string docids (N) underperforms atomic docids across Hits@10 metric. Similar to Figure 2, imbuing the docid space with semantic (S) structure alleviates the forgetting compared to an arbitrary/ naive (N) structure.

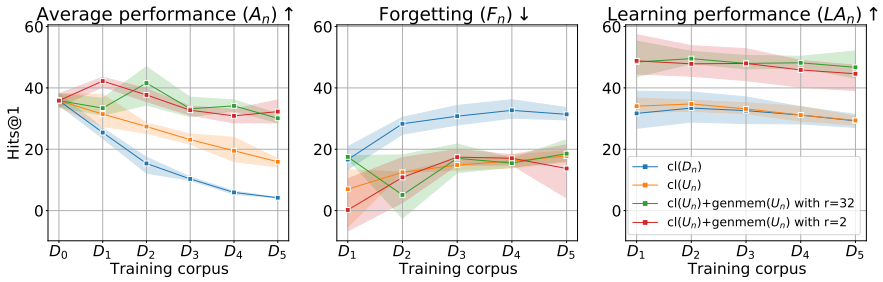


Figure 6: Investigating the effectiveness of generative memory in mitigating forgetting when continuously indexing new corpus D_n (T5-Base model and atomic docids representation). ↑ indicates higher is better, ↓ indicates lower is better. We observe that continual indexing of old and new documents cl(U_n) help to alleviate forgetting of older documents when evaluated on retrieval tasks. However, average Hits@1 (A_n) still undergo 19 points drop after sequential updates ($D_0 \rightarrow D_1 \cdots \rightarrow D_5$). Generative memory enables sparse replaying of old and new documents pseudo-queries. We observe that by augmenting generative memory during continual indexing not only reduces the forgetting (F_n) but also improves average Hits@1 (A_n) by $+17.3\%$ over considered baselines