AdapterSwap: Continuous Training of LLMs with Data Removal and Access-Control Guarantees

William Fleshman*, Benjamin Van Durme

Johns Hopkins University, 3101 Wyman Park Dr, Baltimore, MD 21218

Abstract

Large language models (LLMs) are increasingly capable of completing knowledge intensive tasks by recalling information from a static pretraining corpus. Here we are concerned with LLMs in the context of evolving data requirements. For instance: batches of new data that are introduced periodically; subsets of data with user-based access controls; or requirements on dynamic removal of documents with guarantees that associated knowledge cannot be recalled. We wish to satisfy these requirements while at the same time ensuring a model does not forget old information when new data becomes available. To address these issues, we introduce AdapterSwap, a training and inference scheme that organizes knowledge from a data collection into a set of dynamically composed low-rank adapters. Our experiments demonstrate AdapterSwap's ability to support efficient continual learning, while also enabling organizations to have fine-grained control over data access and deletion.

Keywords

LLM, adapter, access-control, data removal, forgetting

1. Introduction

Generative Large Language Models (LLMs) continue to improve in handling a broad range of Natural Language Understanding (NLU) and Natural Language Generation (NLG) tasks. To achieve these improvements, models have grown in size such that training or fine-tuning a full model for a custom task or data distribution is not possible on commodity hardware [1]. Under such constraints, organizations may look to other solutions for leveraging LLMs with their own data.

Parameter Efficient Fine-Tuning (PEFT) is a collection of approaches for adapting a model to new tasks or data domains. One method of PEFT is to train a small number of new parameters (adapters) which enable the model to perform well in the current setting [1, 2, 3]. These approaches allow organizations to update models with their data in a computationally efficient manner.

Beyond compute, organizations may face additional challenges such as incorporating knowledge from a continuous stream of data or ensuring a model adheres to user-based access-controls applied to the data. Additionally, data protection policies or legal outcomes could lead to organizations losing access to a subset of data, preventing the use of models which used that subset during training [4, 5].

Accordingly, we introduce AdapterSwap, a parameter efficient approach for continual learning which addresses data access-control and removal while retaining the ability to acquire new knowledge. We achieve these results through a thoughtful consideration of Low-Rank Adaptation (LoRA) [3]. First, data is segmented into appropriately sized groups based on access-control levels and available computing resources. A general purpose LM is used as a base model, with a LoRA adapter fine-tuned on each group. During inference, a retriever model is used to select *adapters* relevant to the query, which are optionally filtered according to salient access-controls. A weighted combination of these adapters are then applied to the base model to produce an appropriate response. Crucially, if a document must later be removed from the corpus, only the impacted adapter will need to be retrained. From the user perspective, removing data is then a relatively low latency, computationally efficient process.

CAMLIS'24: Conference on Applied Machine Learning for Information Security, October 24–25, 2024, Arlington, VA

*Corresponding author.

will.fleshman@jhu.edu (W. Fleshman); vandurme@jhu.edu (B. Van Durme)
https://fleshman.dev (W. Fleshman); https://www.cs.jhu.edu/~vandurme/ (B. Van Durme)

[1] For example, the Falcon-7B model used in our experiments was pretrained with 384 40GB A100 GPUs.

A motivating application of AdapterSwap is displayed in Figure 1. In this fictitious example, AdapterSwap has been fit to hospital data with subsets of the data subject to various access-controls. Both a cardiologist and member of the finance office submit the query '*How much does the average cardiac visit cost?*' The retriever model selects the most relevant adapters to the query for which each user has access. The cardiologist's unique access to appointment notes and patient records enables the model to access specific payments from cardiac patients and respond accordingly. In contrast, the finance office's access to payroll and supply expenditures results in a response from the hospital's perspective without leaking the private patient information to the unauthorized employee.

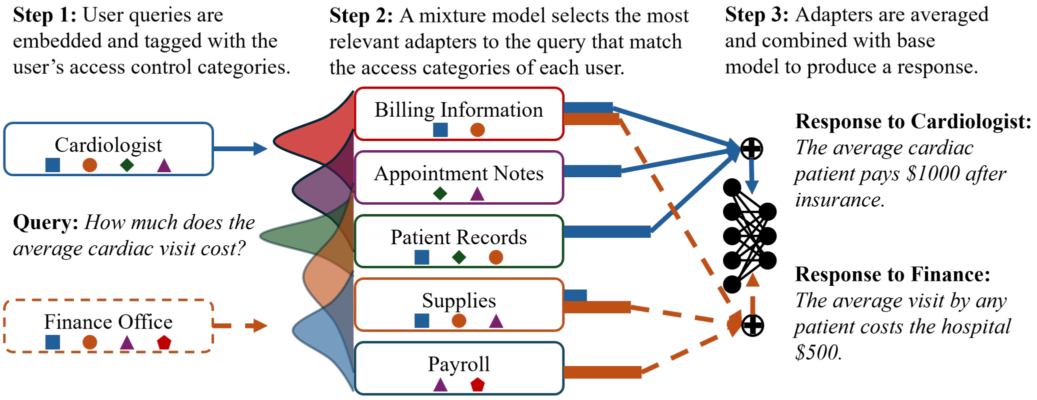
Step 1: User queries are embedded and tagged with the user's access control categories.
Step 2: A mixture model selects the most relevant adapters to the query that match the access categories of each user.
Step 3: Adapters are averaged and combined with base model to produce a response.
Cardiologist
Query: *How much does the average cardiac visit cost?*
Finance Office
Billing Information
Appointment Notes
Patient Records
Supplies
Payroll
Response to Cardiologist: *The average cardiac patient pays $1000 after insurance.*
Response to Finance: *The average visit by any patient costs the hospital $500.*

Figure 1: Motivating application of AdapterSwap. A mixture model selects the most relevant adapters to each users' query with the appropriate access-controls (indicated by shapes). Selected adapters are then combined and applied to a base model to produce personalized responses for each user.

The rest of this paper is structured as follows. In Section 2 we discuss challenges which motivate our research. In Section 3 we provide context with prior works from which we build upon. Section 4 details AdapterSwap, our proposed approach. We demonstrate and quantify benefits of AdapterSwap through careful experimentation in Section 5. We discuss additional related works in Section 6. Finally, we conclude and suggest future research in Section 7.

Specifically, in this work we:

- Develop an efficient approach for continuous knowledge acquisition through the training and dynamic composition of multiple LoRA adapters;
- Demonstrate our method's ability to guarantee data access-control and handle data removal in an efficient manner;
- Quantify our performance via a document completion task across a diverse set of LLMs: Falcon-7B, Gemma-7B, Llama-2-7B, and Mistral-7B; and
- Show that our approach mitigates forgetting better than iterative fine-tuning and retraining.

2. Motivation

Our work is primarily motivated by three issues faced when fine-tuning and deploying LLMs in real-world organizations. Namely, how to utilize data with access-controls, how to remove knowledge from a model retroactively, and how to update knowledge over time as new data becomes available.

2.1. Data Access-control

It is common for organizations to apply access-control to their sensitive data [6]. For example, only certain employees at a hospital have access to patient records to protect privacy. Similarly, employees at a law firm are concerned with attorney-client privilege. Access-control can also be an important aspect of business models such as a news aggregator giving access to users and their personalized chatbots based on their paid subscriptions [7]. We would like a language model trained on these data to inherit and guarantee these access restrictions.

2.2. Data Protection and Removal

Organizations can unexpectedly lose the rights to maintain certain data. For example, data protection policies such as the General Data Protection Regulation (GDPR) [4] allow users to recall their data from corporations. *The Stack*, a popular dataset comprised of source code repositories, allows users to opt-out of having their code in future versions of the dataset but offers no solution for models trained on previous versions [8]. Similarly, the removal of training data later found to be copyright protected or unlicensed might be mandated through legal action, a growing concern for LM producers [9, 5]. Existing models provide no mechanism to remove all knowledge from individual training examples, so to comply with these mandates would require the entire model to be retrained. Therefore, we would like a more efficient approach to guarantee the removal of data from models.

2.3. Catastrophic Forgetting

Organizations often have access to continuous or evolving streams of data. *Catastrophic forgetting* is an issue that arises when a machine learning model forgets previously seen information as it learns from new data [10]. LLMs have been shown to suffer from forgetting during the fine-tuning process [11]. We would like a method that addresses this issue and guarantees the ability to recall old information as new knowledge is continuously acquired.

3. Background

3.1. Parameter Efficient Fine-Tuning

As language models have become more specialized, growing in size and capabilities, fine-tuning an entire model has become unreasonable on commodity hardware. To address these challenges, several methods have been developed for performing *parameter efficient fine-tuning*.

References [12] and [13] present techniques for training a sparse subset of parameters in a multi-task setting. Alternatively, *prompt tuning* and *prefix tuning* concatenate learned task-specific embeddings to the sequence of inputs or activations being processed by a model [14, 15].

In the wider context of transfer learning, adapter layers provide a straight forward mechanism to efficiently and effectively generalize a base model to a target task or domain by fine-tuning a new set of parameters (an adapter) on the target data [1, 2]. Low-rank adapters (LoRA) have emerged as a parameter efficient approach to fine-tuning large language models with reasonable amounts of compute [3]. In this work, we leverage LoRAs to continuously update and control a language model's knowledge.

3.2. Model Averaging and Segmentation

Several approaches have been suggested for combining model weights or outputs with demonstrated increases in performance or efficiency in certain scenarios. Reference [16] proposed *Model Soups* which average the weights of multiple models trained on the same data with different hyper-parameters. While they show that the soups increase performance and robustness of language models, they do not address combinations of models trained on separate data.

AdapterFusion was introduced by [17] as an approach to multi-task learning that segments task-specific knowledge into separate adapters that are then combined via an attention mechanism. While segmenting tasks is similar to segmenting data based on access controls, the attention mechanism adds additional complexity and the model dependency prevents the efficient removal of data.

Reference [18] extended ideas from AdapterFusion with their approach *AdapterDrop.* AdapterDrop prunes adapters for increased efficiency but still lacks the ability to address efficient data removal or continual training.

AdaMix was proposed by [19] and uses a mixture-of-experts approach to combining adapters at each layer for the purpose of parameter sharing but not for specific knowledge segmentation.

Hierarchical Adapters by [20] and *AdapterSoup* by [21] are the most similar to our approach as they train individual adapters on segmented domains in the training data, but their focus is on combining the adapters to perform well with out-of-domain queries, while our focus is on in-domain access-control and efficient knowledge deletion.

4. AdapterSwap

4.1. Data Segmentation and Adapter Training

The first stage of our approach is choosing a data partitioning scheme. Data is segmented into separate access-control categories and further sharded based on desired *per-shard* compute requirements[2]. A pretrained LLM is then used as a base model for fine-tuning a separate LoRA adapter per data partition. As the information from each partition is isolated to a single adapter, the adapter inherits the access-control categories of the data. This partitioning and fine-tuning stage can be done once for a static dataset, or on a continual basis as data arrives.

4.2. Retrieval Model

Similar to AdapterSoup, we use a Gaussian Mixture Model (GMM) to retrieve the subset of adapters relevant to a given query during inference. To fit the GMM we use a pretrained SBERT [22] model to embed randomly held-out samples from each partition into vectors of dimension 768[3]. We further reduce the dimension of these vectors by applying linear discriminant analysis (LDA). While previous approaches such as [21] and [23] use principal components analysis (PCA), in this work we compare PCA to LDA. LDA has the theoretical benefit of maximizing the linear separability of the clusters in the lower dimensional space by using their labels in a supervised manner. We found that using LDA over PCA significantly improved our downstream retrieval accuracy which we discuss in Section 5.3. Finally, the GMM is fit on the lower dimensional vectors with the number of components equal to the number of adapters. The efficiency of LDA and GMMs allows for cheap retraining of the retriever if new adapters are added over time.

4.3. Inference

During inference, queries are embedded using the same approach, and the GMM is used to rank potential adapters. The user's access control categories are applied so that restricted adapters are prevented from being selected. We explore several retrieval modes where either the top-1, top-2, or top-3 adapters with the highest GMM density are selected and combined with the base model. We also attempted averaging all non-restricted adapters with equal weight or by weighting according to GMM density, but the results for those scenarios were poor and are therefore omitted. In practice, the choice of retrieval mode could vary by context. For example, our experiments show that top-1 results are likely best conditioned on knowledge that the information being retrieved was isolated to a single adapter. However, combinations of multiple adapters have been shown to perform better for out of domain queries [21].

4.4. Data Removal

Finally, if circumstances arise that require permanently removing data from our training set, only the weights associated with the LoRA adapter trained on the removed data need to be retrained; less than 0.1% of the base model's parameters in our experiments. This contrasts with emerging approaches such as [24] for efficiently fine-tuning over the entire base model; which offers no savings as all fine-tuned parameters would require discarding if data is removed.

[2] See Section 5.2 for a discussion on this topic.

[3] Specifically, we use the all-mpnet-base-v2 model from https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

In Section 5 we demonstrate AdapterSwap using several models under both access-control and data removal scenarios. We also compare AdapterSwap's ability to prevent *forgetting* with alternative methods for continuous learning. An overview of AdapterSwap training is illustrated in Figure 2.

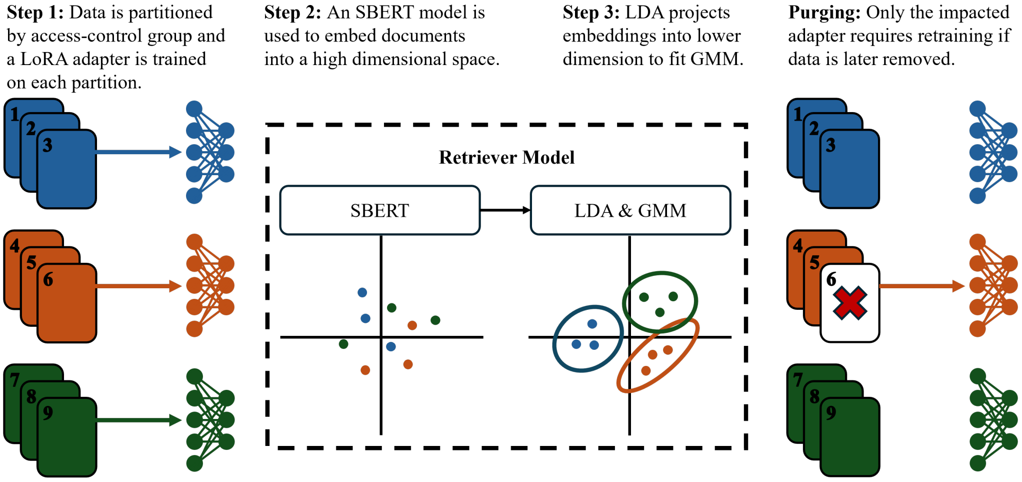
Step 1: Data is partitioned by access-control group and a LoRA adapter is trained on each partition.
Step 2: An SBERT model is used to embed documents into a high dimensional space.
Step 3: LDA projects embeddings into lower dimension to fit GMM.
Purging: Only the impacted adapter requires retraining if data is later removed.
Retriever Model
SBERT
LDA & GMM
1
2
3
4
5
6
7
8
9

Figure 2: AdapterSwap overview. Individual adapters are trained on partioned access-control groups. A retriever model is fit using LDA and a GMM over SBERT representations. If data is removed only the impacted adapter requires retraining.

5. Experiments

We demonstrate the effectiveness by AdapterSwap through multiple experiments. First, we describe the datasets and models used for our experiments in Section 5.1. In Section 5.2 we establish the training times and baseline performance when fitting multiple adapters over a sharded dataset.

Next, in Section 5.3 we quantify our ability to retrieve and compose mixtures of adapters during inference. We demonstrate AdapterSwap's ability to conform to data access-controls in Section 5.4 and effectiveness when removing data in Section 5.5. Finally, we compare AdapterSwap's resilience to forgetting against two alternative approaches for continuous learning in Section 5.6

For all of our experiments, we evaluate AdapterSwap by segmenting documents into equal halves and measuring the perplexity of the model while force decoding the second half given the first. This document completion task is suited for determining if a model has trained on and remembered particular samples from the datasets. We also report training times in GPU Hours using a single 80GB A100 GPU.

5.1. Data and Models

We use two datasets for our experimentation. First, we use the subset of C4 [25] utilized by [21] which contains 21 website domains where unique pages from the domain represent separate documents. We treat each domain as a separate LoRA training group for our experiments involving access-control and data purging. The list of training domains and their corresponding document counts are shown in Table 1.

We also use an English subset of the WMT News Crawl Dataset [26]. Passages from articles published in the year 2020 were extracted and deduplicated following [27]. These passages were then segmented into LoRA training groups based on their month of publication. The chronological nature of this dataset makes it suitable for measuring a model's ability to recall previous training data as subsequent months are trained.

We leverage a diverse set of LMs as base models to ensure our approach generalizes. We replicate experiments across Falcon-7B [28], Gemma-7B [29], Llama-2-7B [30] and Mistral-7B-v0.1 [31].

Table 1
Subset of domains from C4 used as AdapterSwap training groups. Size is represented by the number of documents from each domain.

Domain	Size
androidheadlines.com	41894
booking.com	57218
csmonitor.com	47625
dailymail.co.uk	77150
entrepreneur.com	39373
eventbrite.com	85647
express.co.uk	72435
forums.macrumors.com	68513
frontiersin.org	12053
glassdoor.com	40227
ign.com	41275
insiderpages.com	48072
instructables.com	72154
journals.plos.org	10630
librarything.com	41617
link.springer.com	82720
lonelyplanet.com	39284
medium.com	48522
npr.org	55632
pcworld.com	40202
wired.com	42061

We utilize the HuggingFace [32] library to access the pretrained models and fit LoRA adapters using Parameter Efficient Fine-Tuning [33]. All adapters were trained on a single 80GB A100 GPU to standardize timing comparisons. In practice, AdapterSwap can be trained in parallel across several devices. Each adapter was trained for 10 epochs with a batch size of 20. Rank 32 LoRA adapters were applied to all attention layers for the News Crawl data and rank 64 adapters on all linear layers for C4. All adapters were initialized with the same random seed, which was identified by [21] as being necessary for adapter mixing. A detailed list of hyperparameters are included in appendix.

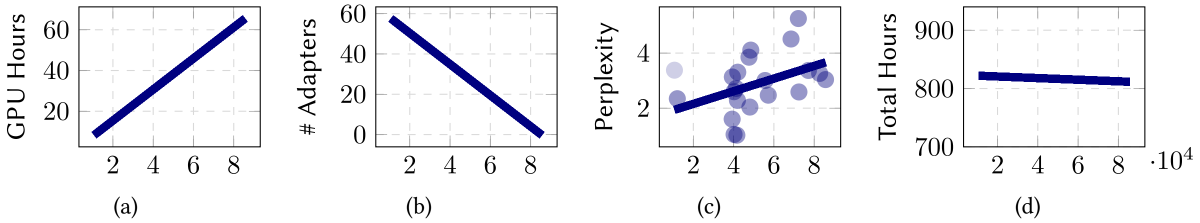

Figure 3: (a) Average training time for a single adapter given the data partition size. (b) Total number of adapters needed per data partition size for 1,064,304 documents divided equally among partitions. (c) Observed perplexity per partition size when partitioning dataset by domain. (d) Total GPU hours required to train all adapters using an equal partition size per adapter.

5.2. Shard Size, Time, and Performance Trade-offs

For each model, we trained a separate adapter on all training groups. Because our groups naturally differ in size we are able to measure average training time and performance for different sharding strategies. Figure 3 displays differences in training time and document completion performance as a function of shard size. Partitioning the dataset into smaller groups results in the need for more adapters but enables faster training of each. The individual training time is an important characteristic when faced with the need to retrain adapters if data is later purged. We observe that smaller partition sizes tend to result in better perplexities, likely due to adapters having a higher ratio of parameters to training tokens. Overall, the total GPU hours is roughly equivalent across partitioning schemes, with a slight overhead resulting from adding each additional adapter.

5.3. Retrieval

An optimal retriever should return the adapters which have knowledge related to a given query. In our case we know each sample was seen by only a single adapter, which we refer to as the *oracle* adapter, but multiple adapters might be necessary in general information retrieval scenarios. Table 2 displays the document completion performance across all models when using the oracle adapter, as well as an average of the top-1, top-2, and top-3 adapters as ranked by the retriever model. We also show the top-1 adapter with PCA as the projection method to compare with previous works [21, 23]. For all models, using the top-1 adapter with LDA resulted in the best completions and retrieved the oracle adapter with accuracy varying from 69% to 81%. With the exception of the Gemma-7B model, results for the top-2 and top-3 schemes are reasonable and [21] show that higher mixtures work well when the query is out of domain. The accuracy for retrieving the oracle adapter in the top-3 ranged from 93% to 95%, suggesting a potential inference scheme like [34]'s, where output is selected from the mixture and individual adapters via a separate content-selector.

Table 2
Perplexity and accuracy retrieving oracle adapter (the adapter trained on the data being queried) when applying the top-3, top-2, and top-1 adapters from the retriever model. The oracle column corresponds to just using the single correct adapter for inference. We also compare top-1 performance when using PCA instead of LDA. A lower perplexity, and higher accuracy indicates better performance.

Model	↓ (↑)Top-3	↓ (↑)Top-2	↓ (↑)Top-1	↓ (↑)Top-1 PCA	↓Oracle
Falcon	14.4 (93%)	14.4 (86%)	**4.2** (77%)	5.4 (52%)	2.9
Gemma	1456 (93%)	588 (82%)	**3.1** (69%)	4.8 (46%)	1.7
Llama-2	25.8 (95%)	19.5 (86%)	**2.4** (81%)	3.3 (53%)	1.8
Mistral	43.0 (93%)	34.8 (88%)	**1.9** (75%)	6.7 (45%)	1.4

5.4. Access-Control

AdapterSwap can be directly applied to scenarios where data is organized into access-control categories. We simulate this with the C4 dataset by assigning each domain to a different category. For each document completion, we use our retrieval model to return the top-1 adapter under two scenarios: with access to all adapters and with access to all but the adapter trained on the restricted domain.

The results are summarized in Table 3. As expected, the best completion was achieved using the adapter with access to the relevant data, and performance dropped significantly when the retriever did not have access to the domain. This demonstrates AdapterSwap's ability to enforce access-control at the adapter level, providing the best results to users while simultaneously preventing unauthorized access to restricted training data.

Table 3
Perplexity across access-control scenarios. *No Access*: Using the top adapter retrieved for the query excluding the oracle adapter. *With Access*: Using the top adapter retrieved among all adapters.

Model	↓No Access	↓With Access
Falcon	16.1	**4.2**
Gemma	68.0	**3.1**
Llama-2	15.2	**2.4**
Mistral	28.2	**1.9**

5.5. Data Removal

We also measure our ability to efficiently purge documents from our dataset after training. When a piece of data is removed from the corpus, only the adapter fine-tuned on that data requires retraining. We show this by attempting to complete documents for each adapter before and after removing them and retraining the adapter.

Table 4 summarizes the results for this experiment. We see a significant performance drop when trying to complete documents that have been purged from their adapter as the model is now guaranteed to have lost access to that data. Referring back to Figure 3, retraining a single adapter is up to 80x more efficient than if you had to fine-tune over the entire dataset. Purging with AdapterSwap provides the guarantee that removed documents will not contribute to later inferences without the huge cost of full retraining.

Table 4
Perplexity completing documents using an adapter with (*Before Purge*) and without (*Purged*) the purged data.

Model	**↓Before Purge**	**↓Purged**
Falcon	**2.9**	13.4
Gemma	**1.7**	25.8
Llama-2	**1.8**	11.9
Mistral	**1.4**	15.4

5.6. Catastrophic Forgetting

Finally, we compare AdapterSwap's ability to recall past information with two alternative strategies. A naive approach to handling new streams of data is to iteratively fine-tune a LM as new data becomes available. This workflow can cause older information to be 'overwritten' within the architecture's parameters. Alternatively, as new data is added, the model can be fine-tuned from scratch on all available data at once. While this mitigates some 'forgetting' it requires longer training times as models are discarded and retrained. We use News Crawl to evaluate both methods and compare to AdapterSwap. We iteratively measure performance on the first month of our dataset as we add subsequent months of data to our models. For this experiment we only use Falcon-7B as the baseline due to the increased computational demand required by the alternative approaches.

Figure 4 displays the performance of AdapterSwap compared to chronological fine-tuning and full retraining. Chronological fine-tuning quickly degrades as more data is presented to the model. Retraining performs better, but suffers from the fixed capacity of a single adapter. AdapterSwap maintains a static performance when recalling data from the first month as that adapter remains unchanged over time.

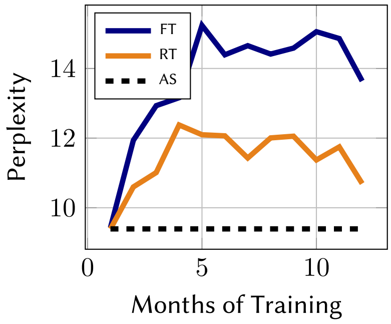

Figure 4: Perplexity of the first month of data measured after month by month training. FT indicates chronological fine-tuning as data becomes available. RT indicates retraining with new data and all preceding data. AS indicates AdapterSwap performance using the first month adapter.

6. Additional Related Work

6.1. Knowledge Editing

Recent work in knowledge editing of LLMs has considered approaches which either add additional parameters to a model, or directly edit existing parameters to update information [35]. Directly updating the existing parameters is attractive as it does not require any additional parameters, and updates can be applied whenever new knowledge is available [36, 37]. Knowledge vectors, in combination with hidden representations of specific entities, have also been proposed as a tool to update or remove knowledge [38]. In all cases, these approaches lack the ability to guarantee that any specific training example can be removed entirely from the model.

6.2. Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) has become a popular approach to incorporating new data into a pre-trained LLM without having to rely on retraining or fine-tuning [39, 40]. While it offers some advantages, there remain challenges that can make deploying an effective RAG-based solution difficult. For example, RAG is limited by how much retrieved context can be employed based on the underlying LLM's context window size. This contrasts with AdapterSwap where each adapter in our experiments represented more than 50 million tokens on average. Recent work has shown that all evidence in the context window is not treated equally, with models favoring evidence at the start and end of each window [41]. Further, RAG is fully dependent on the ability of a retriever model to locate all relevant documents without introducing too much noise into the context [42, 40, 43]. In addition, as transformers are quadratic in the number of tokens considered, then requiring forced decoding over retrieved content can add significant latency at inference time. While our solution avoids these issues, we note that AdapterSwap does not preclude the use of RAG, and a hybrid approach could be useful in some circumstances.

6.3. Federated Learning

Federated learning [44] also deals with learning from siloed data, typically aggregating gradients on local data before averaging into a global model. However, federated learning is intended to produce a single centralized model without mixing data silos and thus does not provide any mechanism for access control or deletion. Some work has combined federated learning and PEFT methods [45, 46, 47], but these do not address adapter mixing, deletion, or data silos as distinct knowledge sources. Furthermore, the privacy benefits of federated learning are unclear when applied to LLMs with large capacity for memorization [48, 49, 50, 51].

7. Conclusion

In this paper, we introduced AdapterSwap, a parameter efficient approach to continuous learning with access-control and data removal guarantees. We fine-tuned adapters using four modern pretrained language models on separate domains. We showed that knowledge from specific domains can be masked via access-control by preventing a retriever from accessing the controlled adapter at inference. The multiple-adapter scheme also enables efficient knowledge removal via data deletion and adapter retraining. The non-parametric behavior of AdapterSwap enables knowledge from the past to be retained, and we showed that AdapterSwap outperforms both chronological fine-tuning and retraining.

AdapterSwap enables a rich set of future research opportunities. We would like to directly improve the approach by exploring better retrieval and adapter mixing methods. Additionally, AdapterSwap could be further scaled to specific down-stream tasks such as question answering and directly compared or combined with alternative data management schemes such as a RAG framework.

References

[1] A. Bapna, O. Firat, Simple, scalable adaptation for neural machine translation, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 1538–1548. URL: https://aclanthology.org/D19-1165. doi:10.18653/v1/D19-1165.

[2] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for nlp, 2019. arXiv:1902.00751.

[3] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, W. Chen, Lora: Low-rank adaptation of large language models, CoRR abs/2106.09685 (2021). URL: https://arxiv.org/abs/2106.09685. arXiv:2106.09685.

[4] European Parliament, Council of the European Union, Regulation (EU) 2016/679 of the European Parliament and of the Council, 2016. URL: https://data.europa.eu/eli/reg/2016/679/oj.

[5] Y. Lu, Digital media outlets sue openai for copyright infringement, The New York Times (2024). URL: https://www.nytimes.com/2024/02/28/technology/openai-copyright-suit-media.html.

[6] V. C. Hu, D. Ferraiolo, D. R. Kuhn, et al., Assessment of access control systems, US Department of Commerce, National Institute of Standards and Technology . . . , 2006.

[7] Tars, News over a chatbot, 2024. URL: https://hellotars.com/chatbot-templates/media-publication/r1FvBF/news-over-a-chatbot.

[8] D. Kocetkov, R. Li, L. Ben Allal, J. Li, C. Mou, C. Muñoz Ferrandis, Y. Jernite, M. Mitchell, S. Hughes, T. Wolf, D. Bahdanau, L. von Werra, H. de Vries, The stack: 3 tb of permissively licensed source code, Preprint (2022).

[9] M. M. Grynbaum, R. Mac, The times sues openai and microsoft over a.i. use of copyrighted work, The New York Times (2023). URL: https://www.nytimes.com/2024/02/28/technology/openai-copyright-suit-media.html.

[10] M. McCloskey, N. J. Cohen, Catastrophic interference in connectionist networks: The sequential learning problem, Psychology of Learning and Motivation 24 (1989) 109–165. URL: https://www.sciencedirect.com/science/article/pii/S0079742108605368. doi:https://doi.org/10.1016/S0079-7421(08)60536-8.

[11] Y. Luo, Z. Yang, F. Meng, Y. Li, J. Zhou, Y. Zhang, An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2023. arXiv:2308.08747.

[12] D. Guo, A. Rush, Y. Kim, Parameter-efficient transfer learning with diff pruning, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Online, 2021, pp. 4884–4896. URL: https://aclanthology.org/2021.acl-long.378. doi:10.18653/v1/2021.acl-long.378.

[13] Y.-L. Sung, V. Nair, C. A. Raffel, Training neural networks with fixed sparse masks, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan (Eds.), Advances in Neural Information Processing Systems, volume 34, Curran Associates, Inc., 2021, pp. 24193–24205. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/cb2653f548f8709598e8b5156738cc51-Paper.pdf.

[14] B. Lester, R. Al-Rfou, N. Constant, The power of scale for parameter-efficient prompt tuning, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 3045–3059. URL: https://aclanthology.org/2021.emnlp-main.243. doi:10.18653/v1/2021.emnlp-main.243.

[15] X. L. Li, P. Liang, Prefix-tuning: Optimizing continuous prompts for generation, in: C. Zong, F. Xia, W. Li, R. Navigli (Eds.), Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Online, 2021, pp. 4582–4597. URL: https://aclanthology.org/2021.acl-long.353. doi:10.18653/v1/2021.acl-long.

353.

[16] M. Wortsman, G. Ilharco, S. Y. Gadre, R. Roelofs, R. Gontijo-Lopes, A. S. Morcos, H. Namkoong, A. Farhadi, Y. Carmon, S. Kornblith, L. Schmidt, Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, in: K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, S. Sabato (Eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 23965–23998. URL: https://proceedings.mlr.press/v162/wortsman22a.html.

[17] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, I. Gurevych, AdapterFusion: Non-destructive task composition for transfer learning, in: P. Merlo, J. Tiedemann, R. Tsarfaty (Eds.), Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Association for Computational Linguistics, Online, 2021, pp. 487–503. URL: https://aclanthology.org/2021.eacl-main.39. doi:10.18653/v1/2021.eacl-main.39.

[18] A. Rücklé, G. Geigle, M. Glockner, T. Beck, J. Pfeiffer, N. Reimers, I. Gurevych, AdapterDrop: On the efficiency of adapters in transformers, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 7930–7946. URL: https://aclanthology.org/2021.emnlp-main.626. doi:10.18653/v1/2021.emnlp-main.626.

[19] Y. Wang, S. Agarwal, S. Mukherjee, X. Liu, J. Gao, A. H. Awadallah, J. Gao, AdaMix: Mixture-of-adaptations for parameter-efficient model tuning, in: Y. Goldberg, Z. Kozareva, Y. Zhang (Eds.), Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 5744–5760. URL: https://aclanthology.org/2022.emnlp-main.388. doi:10.18653/v1/2022.emnlp-main.388.

[20] A. Chronopoulou, M. Peters, J. Dodge, Efficient hierarchical domain adaptation for pretrained language models, in: M. Carpuat, M.-C. de Marneffe, I. V. Meza Ruiz (Eds.), Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Seattle, United States, 2022, pp. 1336–1351. URL: https://aclanthology.org/2022.naacl-main.96. doi:10.18653/v1/2022.naacl-main.96.

[21] A. Chronopoulou, M. Peters, A. Fraser, J. Dodge, AdapterSoup: Weight averaging to improve generalization of pretrained language models, in: A. Vlachos, I. Augenstein (Eds.), Findings of the Association for Computational Linguistics: EACL 2023, Association for Computational Linguistics, Dubrovnik, Croatia, 2023, pp. 2054–2063. URL: https://aclanthology.org/2023.findings-eacl.153. doi:10.18653/v1/2023.findings-eacl.153.

[22] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using Siamese BERT-networks, in: K. Inui, J. Jiang, V. Ng, X. Wan (Eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3982–3992. URL: https://aclanthology.org/D19-1410. doi:10.18653/v1/D19-1410.

[23] R. Aharoni, Y. Goldberg, Unsupervised domain clusters in pretrained language models, in: D. Jurafsky, J. Chai, N. Schluter, J. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 7747–7763. URL: https://aclanthology.org/2020.acl-main.692. doi:10.18653/v1/2020.acl-main.692.

[24] R. Pan, X. Liu, S. Diao, R. Pi, J. Zhang, C. Han, T. Zhang, Lisa: Layerwise importance sampling for memory-efficient large language model fine-tuning, 2024. arXiv:2403.17919.

[25] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[26] T. Kocmi, R. Bawden, O. Bojar, A. Dvorkovich, C. Federmann, M. Fishel, T. Gowda, Y. Graham, R. Grundkiewicz, B. Haddow, R. Knowles, P. Koehn, C. Monz, M. Morishita, M. Nagata, T. Nakazawa, M. Novák, M. Popel, M. Popović, Findings of the 2022 conference on machine translation (WMT22),

in: P. Koehn, L. Barrault, O. Bojar, F. Bougares, R. Chatterjee, M. R. Costa-jussà, C. Federmann, M. Fishel, A. Fraser, M. Freitag, Y. Graham, R. Grundkiewicz, P. Guzman, B. Haddow, M. Huck, A. Jimeno Yepes, T. Kocmi, A. Martins, M. Morishita, C. Monz, M. Nagata, T. Nakazawa, M. Negri, A. Névéol, M. Neves, M. Popel, M. Turchi, M. Zampieri (Eds.), Proceedings of the Seventh Conference on Machine Translation (WMT), Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 1–45. URL: https://aclanthology.org/2022.wmt-1.1.

[27] A. Liška, T. Kočiský, E. Gribovskaya, T. Terzi, E. Sezener, D. Agrawal, C. de Masson d'Autume, T. Scholtes, M. Zaheer, S. Young, E. G.-M. S. Austin, P. Blunsom, A. Lazaridou, Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models, arXiv preprint arXiv:2205.11388 (2022).

[28] G. Penedo, Q. Malartic, D. Hesslow, R. Cojocaru, A. Cappelli, H. Alobeidli, B. Pannier, E. Almazrouei, J. Launay, The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023. arXiv:2306.01116.

[29] J. Banks, T. Warkentin, Gemma: Introducing new state-of-the-art open models, Google (2024). URL: https://blog.google/technology/developers/gemma-open-models/.

[30] H. Touvron, L. Martin, K. R. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. M. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. S. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. M. Kloumann, A. V. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, T. Scialom, Llama 2: Open foundation and fine-tuned chat models, ArXiv abs/2307.09288 (2023). URL: https://api.semanticscholar.org/CorpusID:259950998.

[31] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed, Mistral 7b, 2023. arXiv:2310.06825.

[32] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Huggingface's transformers: State-of-the-art natural language processing, 2020. arXiv:1910.03771.

[33] S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, B. Bossan, Peft: State-of-the-art parameter-efficient fine-tuning methods, https://github.com/huggingface/peft, 2022.

[34] S. Feng, W. Shi, Y. Bai, V. Balachandran, T. He, Y. Tsvetkov, Cook: Empowering general-purpose language models with modular and collaborative knowledge, arXiv preprint arXiv:2305.09955 (2023).

[35] Y. Yao, P. Wang, B. Tian, S. Cheng, Z. Li, S. Deng, H. Chen, N. Zhang, Editing large language models: Problems, methods, and opportunities, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 10222–10240. URL: https://aclanthology.org/2023.emnlp-main.632. doi:10.18653/v1/2023.emnlp-main.632.

[36] N. De Cao, W. Aziz, I. Titov, Editing factual knowledge in language models, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 6491–6506. URL: https://aclanthology.org/2021.emnlp-main.522. doi:10.18653/v1/2021.emnlp-main.522.

[37] K. Meng, D. Bau, A. Andonian, Y. Belinkov, Locating and editing factual associations in GPT, Advances in Neural Information Processing Systems 36 (2022). ArXiv:2202.05262.

[38] E. Hernandez, B. Z. Li, J. Andreas, Inspecting and editing knowledge representations in language models, 2023. arXiv:2304.00740.

[39] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih,

T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. arXiv:2005.11401.

[40] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, Q. Guo, M. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey, 2024. arXiv:2312.10997.

[41] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, Transactions of the Association for Computational Linguistics 12 (2023) 157–173. URL: https://api.semanticscholar.org/CorpusID:259360665.

[42] S. Barnett, S. Kurniawan, S. Thudumu, Z. Brannelly, M. Abdelrazek, Seven failure points when engineering a retrieval augmented generation system, 2024. arXiv:2401.05856.

[43] D. P. Reyes, Navigating retrieval augmented generation (rag) challenges and opportunities, Flybridge (2024). URL: https://www.flybridge.com/ideas/navigating-retrieval-augmented-generation-rag-challenges-and-opportunities.

[44] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.

[45] Y. Kim, J. Kim, W.-L. Mok, J.-H. Park, S. Lee, Client-customized adaptation for parameter-efficient federated learning, in: A. Rogers, J. Boyd-Graber, N. Okazaki (Eds.), Findings of the Association for Computational Linguistics: ACL 2023, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 1159–1172. URL: https://aclanthology.org/2023.findings-acl.75. doi:10.18653/v1/2023.findings-acl.75.

[46] S. Babakniya, A. R. Elkordy, Y. H. Ezzeldin, Q. Liu, K.-B. Song, M. El-Khamy, S. Avestimehr, Slora: Federated parameter efficient fine-tuning of language models, 2023. arXiv:2308.06522.

[47] Z. Zhang, Y. Yang, Y. Dai, Q. Wang, Y. Yu, L. Qu, Z. Xu, FedPETuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models, in: A. Rogers, J. Boyd-Graber, N. Okazaki (Eds.), Findings of the Association for Computational Linguistics: ACL 2023, Association for Computational Linguistics, Toronto, Canada, 2023, pp. 9963–9977. URL: https://aclanthology.org/2023.findings-acl.632. doi:10.18653/v1/2023.findings-acl.632.

[48] S. Gupta, Y. Huang, Z. Zhong, T. Gao, K. Li, D. Chen, Recovering private text in federated learning of language models, in: A. H. Oh, A. Agarwal, D. Belgrave, K. Cho (Eds.), Advances in Neural Information Processing Systems, 2022. URL: https://openreview.net/forum?id=dqgzfhHd2-.

[49] W. Shi, A. Ajith, M. Xia, Y. Huang, D. Liu, T. Blevins, D. Chen, L. Zettlemoyer, Detecting pretraining data from large language models, 2023. arXiv:2310.16789.

[50] K. Tirumala, A. H. Markosyan, L. Zettlemoyer, A. Aghajanyan, Memorization without overfitting: Analyzing the training dynamics of large language models, 2022. arXiv:2205.10770.

[51] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, C. Zhang, Quantifying memorization across neural language models, 2023. arXiv:2202.07646.

A. Training Details

- We trained each adapter for our experiments on a single 80GB A100 GPU for 10 epochs with a batch size of 4 and 5 gradient accumulation steps. We utilized the AdamW optimizer with default settings.
- For adapters trained on C4 domains we used rank 64 LoRAs with $\alpha = 128$ applied to all linear layers.
- For the News Crawl experiment we used LoRA adapters with rank 32 and $\alpha = 64$ applied to just the attention layers.
- For all experiments we initialized adapters using a random seed of 42. We confirm [21]'s finding that using the same initialization is critical if mixing adapters.