TOWARDS ADAPTIVE AND EVOLVING LARGE LANGUAGE MODELS: CONTINUAL LEARNING AND KNOWLEDGE EDITING

FENG YUJIE

PhD

The Hong Kong Polytechnic University

2026

The Hong Kong Polytechnic University
Department of Data Science and Artificial Intelligence

Towards Adaptive and Evolving Large Language Models: Continual Learning and Knowledge Editing

FENG Yujie

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
August 2025

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: FENG Yujie

Abstract

Large language models (LLMs) have become central to a wide range of applications, but their ability to adapt to ever-evolving environments, including changing tasks, data, and user needs, remains a significant challenge. Knowledge adaptation in LLMs is crucial for maintaining their effectiveness in dynamic settings where continuous learning and knowledge updates are essential. Traditional methods, such as multi-task learning, attempt to integrate new knowledge with previously learned information by jointly training on both new and historical datasets. While these strategies can balance the integration of new and historical knowledge, they are computationally expensive, memory-intensive, and often inefficient.

This limitation constrains the adaptability of models in real-world, continually changing scenarios. In such settings, models must continuously acquire new knowledge while retaining previously learned information to maintain stability and performance. Moreover, it is crucial for models to update outdated knowledge to ensure consistency and reliability. To address these demands, this thesis examines strategies for optimizing knowledge adaptation in continual learning (CL) and model editing, with a focus on improving efficiency, reliability, and practical applicability.

Continual learning (CL) enables models to acquire new knowledge from a sequence of tasks while retaining previously learned information, addressing the challenge of catastrophic forgetting. We first propose Knowledge Identification and Fusion (KIF), which localizes the importance of model parameters for different tasks and consoli-

dates both task-specific and task-shared knowledge by leveraging a model merging mechanism. We then introduce Recurrent KIF, which dynamically estimates parameter importance and iteratively integrates knowledge, overcoming the limitations of static importance estimation in earlier approaches. Finally, we propose Adaptive Iterative Model Merging (AIMMerging), which determines the optimal timing for model merging by leveraging learning and forgetting signals extracted from the training trajectory. Together, these progressive methods advance the continual learning field, providing scalable and effective solutions for continuous knowledge adaptation in large language models.

Model editing focuses on updating outdated or incorrect facts encoded in LLMs, ensuring that the new knowledge is incorporated without disrupting unrelated information. To address this, we propose a geometric knowledge editing method that distinguishes neurons associated with new knowledge updates from those linked to general knowledge perturbations by analyzing the geometric relationships of parameter changes induced by fine-tuning. By masking updates to general-knowledge-related neurons, we prevent negative impacts on generalization ability, while optimizing updates to new-knowledge-related neurons, thereby improving the precision, stability, and effectiveness of model editing.

To demonstrate the practical impact of our approaches, we evaluate them on dialogue state tracking (DST), a core task in task-oriented dialogue systems. The proposed continual learning techniques enhance the robustness and adaptability of LLM-driven DST models in cross-domain and evolving-task scenarios, highlighting the real-world value of knowledge adaptation.

In summary, this thesis develops and evaluates a set of methodologies for knowledge adaptation in large language models under practical constraints. By addressing key challenges such as catastrophic forgetting, efficient knowledge integration, and reliable model updates, the proposed approaches strengthen the robustness and applicability of LLMs in dynamic environments. These methods—spanning continual

learning techniques that mitigate forgetting and editing strategies that ensure precise and consistent updates—offer targeted solutions to common limitations in real-world adaptation. Extensive experiments validate their effectiveness, showing consistent improvements in stability, generalization, and adaptability. Beyond empirical results, this thesis provides methodological insights that can guide future research and practical deployment of scalable and reliable LLM adaptation. Related contributions have been published or submitted to top NLP conferences.

Publications Arising from the Thesis

1. Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, Xiao-Ming Wu. "Towards LLM-driven Dialogue State Tracking", in *Conference on Empirical Methods in Natural Language Processing (EMNLP) (2023).*

2. Yujie Feng, Xu Chu, Yongxin Xu, Guangyuan Shi, Bo Liu, Xiao-Ming Wu. "TaSL: Continual Dialog State Tracking via Task Skill Localization and Consolidation", in *The 62nd Annual Meeting of the Association for Computational Linguistics (ACL) (2024).*

3. Yujie Feng, Bo Liu, Xiaoyu Dong, Zexin Lu, Li-Ming Zhan, Albert Lam, Xiao-Ming Wu. "Continual Dialogue State Tracking via Reason-of-Select Distillation", in *Findings of the Association for Computational Linguistics (ACL-Findings) (2024).*

4. Yujie Feng, Xujia Wang, Zexin Lu, Shenghong Fu, Guangyuan Shi, Yongxin Xu, Yasha Wang, Philip S Yu, Xu Chu, Xiao-Ming Wu. "Recurrent Knowledge Identification and Fusion for Language Model Continual Learning", in *The 63rd Annual Meeting of the Association for Computational Linguistics (ACL) (2025).*

5. Yujie Feng, Liming Zhan, Zexin Lu, Yongxin Xu, Xu Chu, Yasha Wang, Jiannong Cao, Philip S Yu, Xiao-Ming Wu. "Geometric Knowledge Editing for

Large Language Models", in *Conference on Empirical Methods in Natural Language Processing (EMNLP) (2025).*

6. Yujie Feng, Jian Li, Xiaoyu Dong, Pengfei Xu, Xiaohui Zhou, Yujia Zhang, Zexin Lu, Yasha Wang, Alan Zhao, Xu Chu, Xiao-Ming Wu. "AIMMerging: Leveraging Training Trajectories for Adaptive Iterative Model Merging in Language Model Continual Learning", in *Conference on Empirical Methods in Natural Language Processing (EMNLP) (2025).*

7. Yujie Feng, Jian Li, Zhihan Zhou, Pengfei Xu, Yujia Zhang, xiaoyu li, Xiaohui Zhou, Alan Zhao, Xi Chen, Xiao-Ming Wu. "Micro-Macro Retrieval: Reducing Long-Form Hallucination in Large Language Models", in *The Fourteenth International Conference on Learning Representations (ICLR) (2026).*

8. Yujie Feng, Xu Chu, Yongxin Xu, Zexin Lu, Bo Liu, Philip S Yu, Xiao-Ming Wu. "KIF: Knowledge Identification and Fusion for Language Model Continual Learning", submitted to TPAMI.

9. Yujie Feng, Hao Wang, Jian Li, Xu Chu, Zhaolu Kang, Yiran Liu, Yasha Wang, Philip S. Yu, Xiao-Ming Wu. "FOREVER: Forgetting Curve-Inspired Memory Replay for Language Model Continual Learning", submitted to ACL 2026.

Acknowledgments

Time flies, and in the blink of an eye I have reached the end of my doctoral journey. Looking back on these three years, I not only gained valuable growth in research ability, but also harvested the precious mentorship of my advisor and the friendship of my colleagues and friends, which I will cherish for life.

First and foremost, I would like to express my deepest gratitude to my supervisor, Professor Xiao-Ming Wu, for her continuous support and guidance throughout my Ph.D. study. Under her supervision, I have not only learned numerous research skills, but more importantly, I have also been inspired by her rigorous attitude toward scientific research and her passion for academic exploration. Whenever I encountered challenges or doubts in my research, she provided insightful advice and constant encouragement, which helped me push forward with confidence. It has truly been my great fortune and honor to pursue my Ph.D. under her mentorship.

I am also sincerely grateful to my lab mates and colleagues. Their support, companionship, and encouragement have made my doctoral life colorful and joyful. The moments we shared—whether it was having meals together, chatting, traveling, hiking, or playing board games—turned the otherwise monotonous research life into unforgettable memories. I would like to especially thank Bo Liu, Zexin Lu, Zhihao Ding, Liming Zhan, Xiaoyu Dong, Guangyuan Shi, Yuankai Luo, and Zaoyu Chen for their invaluable help and friendship during these years.

Finally, I owe my deepest gratitude to my family and my girlfriend. Their uncondi-

tional love, patience, and support have been my strongest source of strength. They have always been there for me, encouraging me through moments of doubt and cheering me on through every challenge. Without their understanding and unwavering support, I would not have been able to complete this doctoral journey.

To all those who have accompanied and supported me along the way, I extend my heartfelt thanks. It is your presence that has made my Ph.D. life both meaningful and unforgettable.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Illustration of Knowledge Adaptation. Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing (NLP) tasks, from text generation and reasoning to dialogue understanding and knowledge-intensive applications [77, 20, 54, 168, 8]. Central to their success is the vast amount of knowledge encoded during large-scale pretraining. However, as real-world environments evolve, new information emerges, outdated knowledge must be corrected, and task requirements continually shift. This necessitates knowledge adaptation—the process by which LLMs acquire, update, and refine knowledge to remain effective in dynamic contexts. Unlike static pretraining, knowledge adaptation emphasizes continual refinement, enabling models to remain aligned with current data distributions and user needs [17, 123, 157].

Importance of Knowledge Adaptation. Knowledge adaptation [169] is critical for ensuring the long-term usability and reliability of LLMs. In practice, users expect models to seamlessly incorporate novel knowledge while preserving prior competencies, ensuring both stability and adaptability. Without effective adaptation, models risk catastrophic forgetting, inconsistencies in predictions, or the perpetuation of outdated knowledge. Moreover, the ability to adapt efficiently is increasingly important

given the rising costs of large-scale retraining. As such, knowledge adaptation forms a foundational requirement for deploying LLMs in real-world, continually evolving environments [164, 137].

Applications of Knowledge Adaptation. The importance of knowledge adaptation is reflected in numerous applications. In dialogue systems, models must incorporate emerging intents and domain-specific terminology while retaining general conversational abilities [49, 187]. In knowledge-intensive tasks such as question answering and fact verification, LLMs must stay consistent with up-to-date world knowledge while avoiding factual conflicts [99]. Similarly, in task-oriented dialogue systems, continual adaptation ensures robustness across domains and evolving task configurations [47]. Beyond NLP, adaptation is crucial in broader AI systems where long-term interaction with dynamic environments requires ongoing learning, efficient updating, and reliable knowledge retention [14].

Challenges in Knowledge Adaptation. Despite its necessity, knowledge adaptation in large language models faces three major challenges:

(1) Catastrophic Forgetting. When adapting to new tasks or data, LLMs often overwrite previously acquired knowledge, leading to severe performance degradation on earlier tasks. This phenomenon, known as catastrophic forgetting [95], undermines the reliability of models in dynamic environments. Continual Learning (CL) seeks to mitigate this issue by enabling sequential acquisition of knowledge while retaining past competencies, but effective solutions must balance stability and plasticity across evolving tasks.

(2) Precision of Knowledge Updates. In many scenarios, adaptation requires modifying specific facts or correcting outdated knowledge without altering unrelated capabilities [40]. However, parameter updates in LLMs can propagate broadly, unintentionally distorting general reasoning or unrelated facts. This makes reliable and fine-grained model editing particularly challenging. Ensuring precision requires dis-

tinguishing between neurons or parameters associated with the new knowledge and those tied to general knowledge, while minimizing unintended side effects.

(3) Scalability and Efficiency. Given the size of modern LLMs, adaptation methods must remain computationally and memory efficient [127, 72]. Full-model retraining or multi-task learning is often impractical due to resource demands. Similarly, repeated fine-tuning can be prohibitively expensive in real-world applications where knowledge evolves rapidly. This highlights the need for scalable approaches—such as parameter-efficient continual learning or lightweight editing—that preserve effectiveness while reducing computational burden.

Motivation and Thesis Scope. While large language models have demonstrated remarkable capabilities, their practical deployment is still constrained by challenges in **retaining prior knowledge**, **ensuring reliable updates**, and **scaling adaptation efficiently**. These challenges highlight the need for methodologies that can support dynamic, fine-grained, and resource-aware adaptation. This thesis addresses these needs by developing and evaluating novel approaches in **continual learning**—to mitigate catastrophic forgetting and enable sequential knowledge acquisition—and **model editing**—to achieve precise and reliable updates of specific knowledge without harming general capabilities. Beyond methodological contributions, we investigate their applications in dialogue state tracking (DST), a representative task in task-oriented dialogue systems, to demonstrate practical effectiveness. Overall, this work aims to advance both the theoretical foundations and the applied value of knowledge adaptation in large language models, contributing toward more robust, efficient, and generalizable AI systems.

1.1 Continual Learning

Continual Learning (CL) has emerged as a key paradigm within knowledge adaptation, particularly in scenarios where models must operate under dynamic and evolving conditions. Unlike conventional approaches that assume static data distributions or fixed task boundaries, CL enables models to acquire new knowledge sequentially while retaining previously learned information [77, 20, 54]. This property makes CL especially relevant for large language models (LLMs), which are increasingly deployed in real-world settings where domains, tasks, and user demands change continuously.

The most critical obstacle in CL is catastrophic forgetting [95], where the acquisition of new tasks severely degrades performance on previously learned ones. The root cause lies in the sequential nature of training: as models are optimized for new objectives, their parameters may drift away from configurations that encoded earlier knowledge. Existing solutions to this issue fall into three main categories: (1) Replay-based methods, which store and rehearse samples or representations from past tasks[168, 8]; (2) Regularization-based methods, which constrain parameter updates to preserve important weights from earlier tasks [12, 148, 114]; (3) Architecture-based methods, which allocate new parameters or modules for each task to avoid interference [116, 188, 132]. While these strategies provide partial relief, they often require trade-offs between memory efficiency, adaptability, and scalability.

To address these limitations, this thesis proposes a line of research advancing continual learning in LLMs:

Knowledge Identification and Fusion (KIF): This framework identifies task-specific and task-shared parameters, and integrates them through a model merging mechanism, thereby improving knowledge consolidation across tasks.

Recurrent KIF: Building upon KIF, this method dynamically re-estimates parame-

ter importance and iteratively fuses knowledge, mitigating the shortcomings of static importance estimation.

Adaptive Iterative Model Merging (AIMMerging): This approach further enhances flexibility by determining the optimal timing for model merging through monitoring learning and forgetting signals extracted from the training trajectory.

Together, these methods form a coherent set of solutions that significantly improve the robustness and adaptability of LLMs under continual learning. By reducing catastrophic forgetting, enabling dynamic integration, and scaling efficiently across tasks, our work provides a principled foundation for continual knowledge adaptation in large-scale models.

1.2 Model Editing

Model editing is an essential component of knowledge adaptation in large language models (LLMs). Unlike continual learning, which focuses on sequentially acquiring new knowledge across tasks, model editing aims to directly and efficiently update specific knowledge—such as facts, relations, or domain-specific rules—while leaving unrelated knowledge intact [97, 21]. This capability is especially valuable in real-world scenarios where models must quickly adapt to evolving facts (e.g., changes in world events or domain regulations) without retraining on large datasets.

Despite its potential, model editing presents several key challenges: (1) Precision. Updates must target specific pieces of knowledge without unintentionally modifying unrelated facts. Failure to achieve this can compromise the reliability of the model's outputs. (2) Consistency. Newly edited knowledge must remain coherent across diverse contexts, ensuring that updates generalize beyond the exact prompt used for editing. (3) Stability. Edits should not degrade the model's generalization ability or cause performance regressions in unrelated tasks.

Prior work has proposed multiple strategies to address these challenges. Retraining- or fine-tuning-based methods apply targeted updates using small datasets, but risk overfitting and require high computational cost [180, 136, 80]. Memory-based approaches store external knowledge bases and dynamically retrieve updates, but often struggle with integration and scalability [100]. Locate-then-edit-based methods, such as MEMIT [96], identify neurons or subspaces associated with specific knowledge and apply targeted modifications. While promising, these approaches still face difficulties in distinguishing new-knowledge updates from general-knowledge perturbations, limiting their precision and stability.

To overcome these limitations, this thesis introduces a geometric knowledge editing (GeoEdit) framework. The key idea is to analyze the geometric relationships of parameter updates induced by fine-tuning, thereby separating: General-knowledge-related neurons, which contribute broadly to the model's generalization; New-knowledge-related neurons, which encode specific factual updates. By masking updates to the former while optimizing updates for the latter, our approach ensures that editing enhances the accuracy of new knowledge while minimizing interference with existing knowledge. This leads to more reliable and precise model adaptation.

In summary, model editing complements continual learning by providing a fine-grained mechanism for updating specific knowledge in LLMs. Whereas CL addresses long-term adaptation across evolving tasks, model editing enables rapid and localized updates, ensuring that LLMs remain both up-to-date and reliable. Together, these two paradigms form a comprehensive strategy for robust knowledge adaptation in dynamic environments.

1.3 Dialogue State Tracking

Dialogue State Tracking (DST) is a fundamental component of task-oriented dialogue systems (ToDS), responsible for accurately tracking the user's goals, intents, and preferences throughout multi-turn interactions [39, 26]. By maintaining a structured representation of the dialogue state, DST enables dialogue systems to generate contextually relevant responses and perform task-specific actions such as booking or recommendation. With the growing demand for intelligent assistants that operate across multiple domains (e.g., travel, healthcare, finance), robust and adaptable DST has become a critical research focus.

Despite significant progress, DST continues to face several challenges: (1) Multi-domain adaptation and generalization. Real-world systems must generalize across diverse domains with heterogeneous slot-value spaces. A plethora of models have been proposed to address these challenges [23, 7], but effective adaptation across domains remains difficult. (2) Co-reference resolution. Users frequently employ pronouns or implicit references to previously mentioned entities. This co-reference issue complicates the model's ability to maintain accurate states [126]. (3) Error propagation. Errors in earlier turns may accumulate and propagate through subsequent turns, degrading the overall dialogue quality [183, 162]. These challenges collectively hinder the reliability and scalability of DST in real-world conversational agents.

Existing approaches have explored a range of strategies to address these issues: Schema-guided methods explicitly leverage domain ontologies to improve generalization by aligning dialogue states with predefined structures [49, 105, 184, 25]. Contextual modeling approaches enhance DST performance by handling co-reference and long-range dependencies with advanced attention mechanisms or memory architectures [36, 161, 87, 151]. While effective, these approaches often require substantial labeled data, struggle to handle evolving domains, and remain vulnerable to cumulative errors.

Building on our research in knowledge adaptation, this thesis makes the following contributions to dialogue state tracking (DST): (1) LLM-driven DST. We first explore the use of large language models (LLMs) as the backbone for DST, demonstrating their strong capability and adaptability across different experimental settings. (2) Continual DST. To address the challenges of domain shifts and evolving tasks in dialogue systems, we propose two continual learning frameworks: Reason-of-Select (RoS), which enhances the model's ability to identify and retain task-relevant knowledge while reducing interference from irrelevant information. Task Skill Localization and Consolidation (TaSL), which further localizes task-specific skills within the model and consolidates them via a model merging mechanism, thereby mitigating catastrophic forgetting and supporting stable continual adaptation. Together, these contributions illustrate how knowledge adaptation techniques can be applied to DST, improving robustness, adaptability, and scalability in dynamic conversational environments.

In summary, DST serves as a representative and practical application of knowledge adaptation. By integrating different continual learning techniques, we demonstrate how LLM-based DST models can achieve improved robustness, adaptability, and reliability in complex, evolving dialogue environments. This case study validates the practical utility of knowledge adaptation, bridging methodological innovations with real-world applications.

1.4 Contributions

This thesis investigates **knowledge adaptation in large language models (LLMs)** from two complementary perspectives: *continual task-level adaptation* and *precise knowledge-level updating*. Rather than treating these as isolated problems, this work develops a progressively evolving methodological framework that enables LLMs to adapt at different granularities while maintaining stability and reliability.

Contribution 1: A Progressive Framework for Continual Knowledge Adaptation.

A central challenge in continual learning (CL) is catastrophic forgetting, where acquiring new knowledge leads to degradation of previously learned capabilities. To systematically address this issue, this thesis proposes a sequence of increasingly refined methods:

- *Knowledge Identification and Fusion (KIF)* introduces importance-aware parameter fusion by identifying task-specific and task-shared knowledge, enabling selective retention during model updates.
- *Recurrent KIF* extends this idea by dynamically re-estimating parameter importance across training stages, allowing the model to adaptively adjust to evolving task distributions.
- *Adaptive Iterative Model Merging (AIMMerging)* further advances the framework by incorporating learning and forgetting signals to determine merging timing, thereby aligning adaptation with the model's internal learning dynamics.

These methods form a coherent progression: from static importance estimation, to dynamic re-estimation, and finally to signal-driven adaptive merging. Together, they establish a robust framework for continual knowledge adaptation in LLMs.

Contribution 2: Geometric Knowledge Editing for Fine-Grained and Reliable Updates.

While continual learning operates at the task level, many real-world applications require fine-grained factual updates without disturbing unrelated knowledge. To address this complementary challenge, this thesis proposes *GeoEdit*, a geometric knowl-

edge editing framework that analyzes the structure of fine-tuning trajectories in parameter space.

By distinguishing neurons associated with new knowledge from those responsible for general representations, GeoEdit selectively masks general-knowledge-related neurons and optimizes updates on new-knowledge-related ones. This neuron-level geometric perspective enables more precise and consistent updates, reducing unintended side effects and improving editing reliability.

Importantly, this contribution extends the adaptation principle from task-level parameter fusion to neuron-level geometric modulation, thereby bridging coarse-grained continual learning and fine-grained knowledge editing.

Contribution 3: Practical Validation through Continual Dialogue State Tracking (DST).

To demonstrate the practical utility of the proposed knowledge adaptation framework, this thesis applies the developed methodologies to dialogue state tracking (DST), a challenging setting characterized by domain heterogeneity, evolving user requirements, and error propagation.

Building on the continual adaptation principles developed in earlier chapters, we design the *Reason-of-Select (RoS)* framework for continual DST. This system improves cross-domain generalization, mitigates forgetting in evolving-task scenarios, and validates the effectiveness of knowledge adaptation techniques in real-world systems.

Together, these contributions advance knowledge adaptation in LLMs across multiple granularities—ranging from task-level continual learning to neuron-level knowledge editing—and demonstrate their applicability in dynamic real-world scenarios. By establishing both theoretical foundations and practical validation, this thesis provides a unified perspective on how LLMs can adapt reliably and evolve continuously.

Chapter 2

Background and Related Works

In this chapter, we provide a comprehensive overview of the foundational concepts and prior research relevant to knowledge adaptation in large language models (LLMs). We first introduce the notion of knowledge adaptation and its importance in enabling models to remain effective in dynamic and evolving environments. Next, we review continual learning, discussing its role in mitigating catastrophic forgetting and supporting incremental knowledge acquisition. We then examine model editing, which focuses on precise and reliable updates to existing knowledge while preserving unrelated information. Finally, we provide background on dialogue state tracking (DST), a core application domain that highlights the practical value of knowledge adaptation techniques. Through this chapter, we aim to establish a solid conceptual foundation for the methods and contributions presented in subsequent chapters.

2.1 Knowledge Adaptation

Knowledge adaptation [169] aims to transfer knowledge from a source domain to a target domain and generalizes the concept of knowledge transfer to dynamic and evolving environments, particularly in large language models (LLMs). Rather than

solely leveraging information from a static source task to improve a target task, knowledge adaptation emphasizes the continuous integration of new knowledge while maintaining consistency with previously acquired information. Before formally defining knowledge adaptation, we first introduce the concepts of *Domain* and *Task*, which provide the foundation for describing knowledge in machine learning models.

Definition 2.1 (Domain)**.** *A domain $\mathcal{D}$ is defined as a pair:*

$$\mathcal{D} = \{\mathcal{X}, P(X)\}, \tag{2.1}$$

where:

- *$\mathcal{X}$ is the feature space, representing all possible input instances (e.g., text, vectors, images).*
- *$P(X)$ is the probability distribution over the feature space $\mathcal{X}$, and $X = \{x_i \in \mathcal{X}\}_{i=1}^{n}$ denotes a set of input samples. A domain characterizes the environment or source from which data is drawn.*

Definition 2.2 (Task)**.** *A task $\mathcal{T}$ is defined as a tuple:*

$$\mathcal{T} = \{\mathcal{Y}, P(Y|X), f(\cdot)\}, \tag{2.2}$$

where:

- *$\mathcal{Y}$ is the label space, representing all possible outputs or target values.*
- *$P(Y|X)$ is the conditional distribution mapping inputs to outputs.*
- *$f : \mathcal{X} \to \mathcal{Y}$ is the predictive function, learned from labeled data $\{(x_i, y_i)\}_{i=1}^{n}$.*

In summary, a domain $\mathcal{D}$ specifies the type of data (through the feature space $\mathcal{X}$) and its underlying statistical properties (captured by $P(X)$) relevant to a machine learning problem. A task $\mathcal{T}$ defines both the objective to be predicted (through the

label space $\mathcal{Y}$) and the predictive function $f(\cdot)$ that maps inputs from $\mathcal{X}$ to outputs in $\mathcal{Y}$, based on the associated domain $\mathcal{D}$. Importantly, a task is not merely a set of labels or examples—it represents the function $f(\cdot)$ that the model seeks to learn.

Building on these definitions, knowledge adaptation extends the concept of knowledge transfer by enabling models to continuously incorporate new information from evolving domains while maintaining consistency with previously acquired knowledge. Formally, we define knowledge adaptation as follows.

Definition 2.3 (Knowledge Adaptation [169])**.** *Given a source domain $\mathcal{D}_S$ with task $\mathcal{T}_S = \{\mathcal{Y}_S, f_S(\cdot)\}$ and a target domain $\mathcal{D}_T$ with task $\mathcal{T}_T = \{\mathcal{Y}_T, f_T(\cdot)\}$, knowledge adaptation aims to improve the predictive function $f_T(\cdot)$ by leveraging prior knowledge $\mathcal{K} = \{\mathcal{D}_S, \mathcal{T}_S, f_S(\cdot)\}$, while continuously integrating new knowledge and updating outdated information. Formally, the objective is to minimize the expected loss on the target task:*

$$\min_{f_T} \mathbb{E}_{(x_T, y_T) \sim P_T(X_T, Y_T)} \left[\mathcal{L}\Big(f_T(x_T), y_T \Big) \right], \tag{2.3}$$

subject to the constraint that new updates do not degrade previously learned capabilities. Where $\mathcal{L}$ denotes the loss function associated with the task $\mathcal{T}_T$, and $P_T(X_T, Y_T)$ represents the joint distribution of the input-output pairs in the target domain.

Knowledge adaptation extends traditional transfer learning by explicitly considering dynamic environments and continual updates. In practice, LLMs must balance three key aspects: **retention** of previously acquired knowledge, **integration** of new knowledge from evolving data or tasks, and **consistency** to ensure updates do not introduce conflicts or reduce generalization ability. This thesis builds upon existing approaches—including continual learning and model editing—to develop a unified framework for scalable, robust, and efficient knowledge adaptation in LLMs.

2.2 Continual Learning

Continual learning (CL) [157] focuses on developing algorithms that accumulate knowledge from non-stationary data. The formal definition is as follows:

Definition 2.4 (Continual Learning)**.** *Continual learning aims to progressively accumulate knowledge from a sequence of tasks* $\{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$*. Each task* $\mathcal{T}_k$ *includes a distinct dataset* $\mathcal{D}_k = \left\{\left(x_i^k, y_i^k\right)\right\}_{i=1}^{N_k}$ *of size* N_k*, where* $x_i^k \in \mathcal{X}_k$ *and* $y_i^k \in \mathcal{Y}_k$*.*

The model, parameterized by Θ*, is trained sequentially on these tasks to minimize the following objective:*

$$\mathcal{L} = \mathbb{E}_{(x,y)\sim\bigcup_{k=1}^{K}\mathcal{D}_k}\left[-\log p_\Theta(y \mid x)\right] \tag{2.4}$$

Memory-free scenario: *The model cannot access historical data during training. Each task* $\mathcal{T}_t$ *is observed only once, and the model must retain knowledge from previous tasks while learning from the current task.*

Memory-replay scenario: *A practical approach where a small subset of data from previous tasks is stored in a memory buffer to facilitate continual learning. Specifically,* $|\mathcal{M}|$ *samples from each previous task* $\mathcal{T}_i$ *are randomly stored in memory* $\mathcal{M}_i$*. During training on the current task* $\mathcal{T}_k$*, the model is optimized jointly on the new task data* $\mathcal{D}k$ *and the memory buffer* $\mathcal{M}< k$*.*

A continual learning algorithm must mitigate catastrophic forgetting of previous tasks while facilitating effective knowledge transfer across tasks.

Conventional continual learning methods can be categorized into three main types: (i) Regularization-based methods impose explicit constraints to preserve the knowledge of previous tasks by penalizing the variation of each parameter based on its contribution to past tasks [57, 68], such as in the EWC [61] method. However, these methods typically restrict the learning of task-shared knowledge by only considering parameter importance from the perspective of past tasks. This limitation often

leads to a suboptimal balance between retaining previous knowledge and excelling at new tasks, as it does not account for the relevance of parameters to both past and current tasks. (ii) Rehearsal-based methods mitigate catastrophic forgetting by either storing old training samples [141, 102] or training generative models to provide pseudo-samples of previous tasks [48, 75], enabling data replay during the learning of new tasks. (iii) Architecture-based methods aim to prevent task interference by dynamically expanding model capacity or assigning dedicated parameter subspaces to each task [114, 152]. While effective at minimizing forgetting, these methods often hinder knowledge transfer (KT) between tasks.

Continual Learning for LLMs: In the era of LLMs, model mixture-based methods that employ parameter-efficient fine-tuning (PEFT) have become the dominant approach [48, 117, 182], typically falling into two categories: model ensemble and model merging techniques.

Model ensemble methods allocate independent PEFT blocks to each task, effectively isolating task-specific parameters [26, 102, 60, 116, 69, 38, 129]. For instance, O-LoRA [138] enforces orthogonality among LoRA adapters, while SAPT [179] uses a selection module to combine task-specific blocks via task correlations. Though effective for knowledge preservation, they hinder inter-task transfer and scale poorly due to growing memory overhead [171].

In contrast, model merging techniques combine multiple models into a single unified model [13, 2, 112], addressing memory constraints. For example, global model merging approaches [146, 52] perform a weighted fusion of models before and after training, often assuming that all model parameters contribute equally to each task. Fine-grained approaches like Feng et al. [27] leverage parameter importance masks to enable neuron- or matrix-level fusion. Recently, Feng et al. [29] introduced a multi-round merging paradigm for CL, demonstrating that integrating merges during model iterations can significantly enhance model performance.

2.3 Model Editing

Model editing, also referred to as knowledge updating, involves modifying the behavior of an initial target model on specific edit examples without compromising its performance on unrelated examples. Given an initial model f_θ and a set of input-output knowledge pairs $D_{old} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$, the task is to update the model parameters to obtain a new model f_{θ_e} and a corresponding set of new input-output pairs $D_{new} = \{(x_1, y_1^{\text{new}}), (x_2, y_2^{\text{new}}), \ldots, (x_k, y_k^{\text{new}})\}$, where k denotes the number of knowledge pairs to be updated. To illustrate this process, consider an edit example (x_i, y_i^{new}) with the edit label $y_i^{\text{new}} \neq f_\theta(x_i)$, the post-edit model f_{θ_e} should generate the expected output such that $f_{\theta_e}(x_e) = y_e$. The objective of the post-edit model f_{θ_e} is to meet three essential properties: reliability, generality, and locality [128].

Definition 2.5 (Reliability)**.** *Reliability measures the accuracy of the updated model on the new knowledge. Specifically, the output for "Who is the President of the US?" should be updated from "Joe Biden" to "Donald Trump." This can be formalized as follows:*

$$\mathbb{E}_{x_e, y_e \sim D_{new}} \mathbb{1} \left\{ argmax_y f_{\theta_e}(y|x_e) = y_e \right\}. \tag{2.5}$$

Definition 2.6 (Generality)**.** *Generality means that the new model f_{θ_e} should also update rephrased in-scope examples $I(x_e, y_e)$. Such as the answer to "Who holds the position of the President of the US?" should also be changed from "Joe Biden" to "Donald Trump". This is evaluated by the average accuracy of f_{θ^*} on examples from the equivalence neighborhood, as expressed by:*

$$\mathbb{E}_{x'_e, y'_e \sim I(x_e, y_e)} \mathbb{1} \left\{ argmax_y f_{\theta_e}(y|x'_e) = y'_e \right\}. \tag{2.6}$$

Definition 2.7 (Locality)**.** *A good edit should modify relevant knowledge without affecting other irrelevant out-of-scope examples $O(x_e, y_e)$. For example, the question, "Who said: this is a battle for the soul of the nation?" should remain unchanged as*

"Joe Biden". Locality (or specificity) is defined as:

$$\mathbb{E}_{x'_e, y'_e \sim O(x_e, y_e)} \mathbb{1} \left\{ argmax_y f_{\theta_e}(y|x'_e) = f_\theta(y|x'_e) \right\}. \tag{2.7}$$

Existing model editing methods can be classified as follows:

(1) *Fine-tuning* is an intuitive and straightforward way to update the model's knowledge [26, 33, 181]. Recent parameter-efficient fine-tuning methods like Prefix-Tuning [73] and LoRA [45] have made fine-tuning more suitable for knowledge editing. Ni et al. [100] proposes the "Forgetting before Learning" (F-Learning) paradigm, which employs parametric arithmetic to facilitate the forgetting of old knowledge and the learning of new knowledge. However, this often leads to unintended changes in unrelated knowledge, negatively affecting generalization [128, 35].

(2) *Meta-learning* ("learning to learn") methods use a hypernetwork to generate task-specific updates, allowing LLMs to quickly adapt to new data without retraining from scratch [15, 117, 133].

(3) *Locate-and-edit* methods usually locate influential parameters and then edit them by introducing a perturbation [170, 56, 155]. Classic methods like ROME [96] and MEMIT [97] use causal reasoning to identify key neuron activations and adjust specific weights. More recent approaches, such as AlphaEdit [21], improve the method by projecting perturbations onto the null space of preserved knowledge.

2.4 Dialogue State Tracking

In task-oriented dialogue systems, a dialogue with T turns of conversations between the system and the user can be represented as $\{(A_1, U_1), (A_2, U_2) \ldots, (A_T, U_T)\}$, where A represents system response and U represents user input. A predefined slot set $\mathcal{S} = \{S_1, \ldots, S_J\}$ is given, where J is the total number of slots. The dialogue context at turn t includes previous turns of interactions, denoted as $\mathcal{X}_t =$

$\{(A_1, U_1), (A_2, U_2) \ldots, (A_t, U_t)\}$. The dialogue state at turn t is represented as a set of (slot, value) pairs, denoted as $\mathcal{B}_t = \{(S_1, V_1^t), \ldots, (S_J, V_J^t)\}$, where V_J^t is the value of slot S_J. For multi-domain DST, following previous works [66], a slot is defined as the concatenation of the specific domain and the slot, e.g., "restaurant-area". If no information is provided in the dialogue about a specific slot, the value associated with that slot is set to "NONE". Essentially, the DST problem is defined as learning a dialogue state tracker $\mathcal{F} : \mathcal{X}_t \rightarrow \mathcal{B}_t$.

In continual DST, we aim to train a model $f : \mathcal{X} \times \mathcal{T} \rightarrow \mathcal{Y}$ across a sequence of dialogue domains $\mathcal{T}_1, \ldots, \mathcal{T}_K$. Each dialogue domain has its dataset $\mathcal{D}_k$ for $\mathcal{T}_k$. This model predicts the target y based on input x and task $\mathcal{T}_k \in \mathcal{T}$. The notation f_k refers to the model after training on task $\mathcal{T}_k$, while $\hat{f}_k$ denotes the model after averaging for $\hat{f}_{k-1}$ and f_k. Within a given task $\mathcal{T}_k$, a dialogue with M turns of interaction between the system and the user is denoted as $\mathcal{X}_M = \{(A_1, U_1), (A_2, U_2) \ldots, (A_M, U_M)\}$, where A and U represent the system's response and the user's input, respectively.

Multi-Domain Dialogue State Tracking Recent studies in multi-domain DST have extensively utilized the pre-trained language models to achieve high-performance results [108, 167, 119, 24, 140, 31]. For example, Xie et al. [150] proposed a multi-stage correctable dialogue state tracking method to mitigate the error propagation phenomenon, while Wang and Xin [126] proposed a jointly decision making and a dialogue update technique to prevent error accumulation. In addition, Wang et al. [135] and Manotumruksa et al. [92] solve the challenge of co-referencing by learning correlations between slots, for example, by using a combination of slot prompts or hard copy mechanism.

LLMs for Dialogue State Tracking While large language models such as GPT-3 [6] and T5 [106] have gained significant popularity, the efficient utilization of these models remains a significant challenge. Recent advancements in parameter-efficient fine-tuning (PEFT) techniques have effectively alleviated this problem, such as LoRA[45] and Prefix Tuning[82]. For instance, both Lee et al. [65] and Yang et al.

[163] proposed a prompt-tuning method that leverages domain-specific prompts and context information to improve the performance of DST task. Meanwhile, Ma et al. [87] and Chen et al. [10] introduced the prefix tuning approach, which involves modifying the input prompt by adding specific tokens at the beginning of the dialogue, aiming to enhance the efficiency of model fine-tuning.

Recently, Heck et al. [39] exclusively tested ChatGPT's performance on the Multiwoz 2.1 dataset. With the emergence of open-source large language models, such as LLaMa [122], it provides a series of higher-quality backbone models with different parameters. Leveraging LLaMa and the technique of instruction tuning has proven to achieve better results [121], opening new avenues for our research.

Continual Dialogue State Tracking In the realm of Continual DST, pioneering efforts by Madotto et al. [90] and Liu et al. [81] have leveraged these CL strategies to set benchmark performance using PLMs. The DST-EGQA approach by Cho et al. [14] reformulates the DST task to an example-guided question-answering task, aiming to align distribution shifts across different domains to mitigate forgetting. However, these methods overlook DST task correlations that could enhance knowledge transfer. The recent Continual Prompt Tuning (CPT) method by Zhu et al. [187] attempts knowledge transfer via domain-specific soft prompts but depends on inefficient memory replay and extensive retraining. This dataset-driven approach is inefficient and lacks robustness.

2.5 Low-Rank Adaptation

Low-Rank Adaptation (LoRA) [45] is a parameter-efficient fine-tuning method to adapt LLMs to novel tasks. It operates under the assumption that parameter changes during full fine-tuning on a downstream task reside within a low-rank space. This approach allows for not having to fine-tune the entire model, thus significantly im-

proving computational efficiency and resource utilization. Inspired by the low-rank internal dimensionality [1], LoRA hypothesizes the updates to the weights also has a low "intrinsic rank" during adaptation. For instance, consider a pre-trained weight matrix $W^{(0)} \in \mathbb{R}^{out \times in}$ that takes x as input, LoRA modifies the output h of $W^{(0)}$ with a low-rank decomposition:

$$h = W^{(0)}x + \Delta W x = W^{(0)}x + BAx, \tag{2.8}$$

while $B \in \mathbb{R}^{out \times r}$, $A \in \mathbb{R}^{r \times in}$, and the rank $r \ll \min(in, out)$. A is initialized from a random Gaussian distribution and B is initialized with all zeros. During training, $W^{(0)}$ remains fixed, and only BA is updated. Due to its effectiveness, LoRA has emerged as one of the most favored PEFT methods within the NLP community [104].

Chapter 3

Language Model Continual Learning

3.1 Introduction

Why CL for LLMs. Continual learning (CL) is vital for the effective deployment of large language models (LLMs) in evolving environments, allowing them to sequentially acquire new knowledge and circumventing the necessity of costly retraining [77, 20, 54, 168, 8]. CL is particularly critical for LLMs in dynamic environments such as dialogue systems, personalized assistants, and domain-specific applications.

Challenges. The core challenge in CL lies in effectively balancing the retention of previously learned knowledge (mitigating catastrophic forgetting, CF [95]) with the acquisition of new knowledge (facilitating knowledge transfer, KT [59]). Successfully managing this inherent trade-off is vital for practical deployment.

Current Research. Existing approaches span replay-based methods, regularization on important parameters, and parameter-isolation/expansion techniques. While replay can curb forgetting, it raises storage/privacy concerns and scales poorly with

task count. Regularization methods depend on accurate importance estimates that are typically computed once and then fixed, limiting adaptability in long task sequences. Isolation/expansion improves stability but restricts cross-task sharing and inflates model size. Recently, model-merging ideas have emerged, but most adopt *fixed* merging schedules or global heuristics, lacking mechanisms to adapt *when* and *how* to merge as learning progresses.

Our Solution. We introduce a suite of model-merging based CL methods tailored for LLMs that (i) localize where knowledge resides, (ii) update importance estimates *recurrently* as tasks arrive, and (iii) adaptively determine *when* to merge using learning/forgetting signals:

Knowledge Identification and Fusion (KIF). We localize the importance of model parameters for each task and explicitly separate *task-specific* and *task-shared* components. KIF then consolidates knowledge via a targeted *model merging* mechanism, preserving shared competence while reducing interference among task-specific parameters. This yields stronger retention and more principled cross-task integration than naive fine-tuning or static merges.

Experiments show that KIF significantly mitigates forgetting compared to fine-tuning and EWC-like methods, while maintaining competitive adaptability across diverse CL benchmarks.

Recurrent KIF. To overcome the limitations of *static* importance estimation, Recurrent KIF *dynamically* re-estimates parameter importance as new tasks arrive and *iteratively* integrates knowledge. By updating importance signals and re-fusing parameters recurrently, it accommodates representation drift and maintains a better stability–plasticity balance over long task sequences.

Our evaluations demonstrate that Recurrent KIF achieves higher retention on earlier tasks and superior average accuracy in longer task sequences, outperforming both KIF and prior state-of-the-art consolidation methods.

Adaptive Iterative Model Merging (AIMMerging). AIMMerging determines the *optimal timing and intensity* of merging by monitoring training trajectories, leveraging learning and forgetting signals. This adaptive scheduler avoids premature or delayed merges, triggering consolidation precisely when it best preserves prior knowledge while enabling effective acquisition of new skills.

Empirical results highlight that AIMMerging achieves the best trade-off between stability and adaptability, consistently outperforming both static merging and recurrent consolidation baselines in different tasks.

3.2 Related Work

Although continual learning (CL) has been extensively studied, our work is most closely related to two complementary lines of research: *parameter importance identification* and *model merging*. These techniques form the foundation of our proposed KIF, Recurrent KIF, and AIMMerging methods.

Parameter Importance Identification. Identifying important parameters or knowledge regions within LLMs has gained significant attention in the NLP community [178, 79, 27, 155, 117, 175]. This research improves our understanding of LLMs and enhances their performance across a variety of tasks, including model editing [139, 30], compression [174, 55]. In the context of CL, Du et al. [18] use the gradient magnitudes to selectively update parameters. Feng et al. [28] employ gradient-based metrics to compare the parameter importance distributions of current and historical tasks, merging task-shared regions to promote KT and retaining task-specific regions to prevent CF.

Model Merging. Another relevant direction is model merging, where multiple models trained on different tasks or domains are combined into a single unified model. Early work investigated weight interpolation [13, 2, 112], functional regu-

larization [146, 52], and task-specific adapters [48, 117, 182]. More recently, methods such as TIES-Merging [156] and RegMean [52] have shown that carefully designed merging strategies can preserve task performance without requiring joint retraining. These techniques highlight the potential of merging as a practical alternative to replay or parameter isolation.

3.3 Methodology – Knowledge Identification and Fusion (KIF)

3.3.1 Introduction

To address these limitations, we introduce **K**nowledge **I**dentification and **F**usion (KIF) [28], a novel CL framework designed to improve KT between tasks without relying on memory replay. Our approach is motivated by recent findings that model parameters contribute unevenly to performance [101]. For instance, the authors in [177] uncovered a core region in LLMs that is crucial for all languages, suggesting that preserving these vital regions can mitigate forgetting. Additionally, findings from [186] indicate that fine-tuning with LoRA often preserves many unnecessary parameter changes, pointing to substantial redundancies within parameter-efficient fine-tuning (PEFT) blocks.

Based on these insights, KIF facilitates KT by pinpointing and consolidating the importance distribution of model parameters across tasks. Initially, KIF utilizes a group-wise importance-aware ***knowledge identification*** technique that employs gradient trajectories to identify tiny regions within the parameter space that store crucial knowledge for the current task. By comparing the importance distribution with those of previous tasks, we can then differentiate between task-specific and task-shared regions, as illustrated in Figure 3.1. The subsequent ***knowledge fusion***

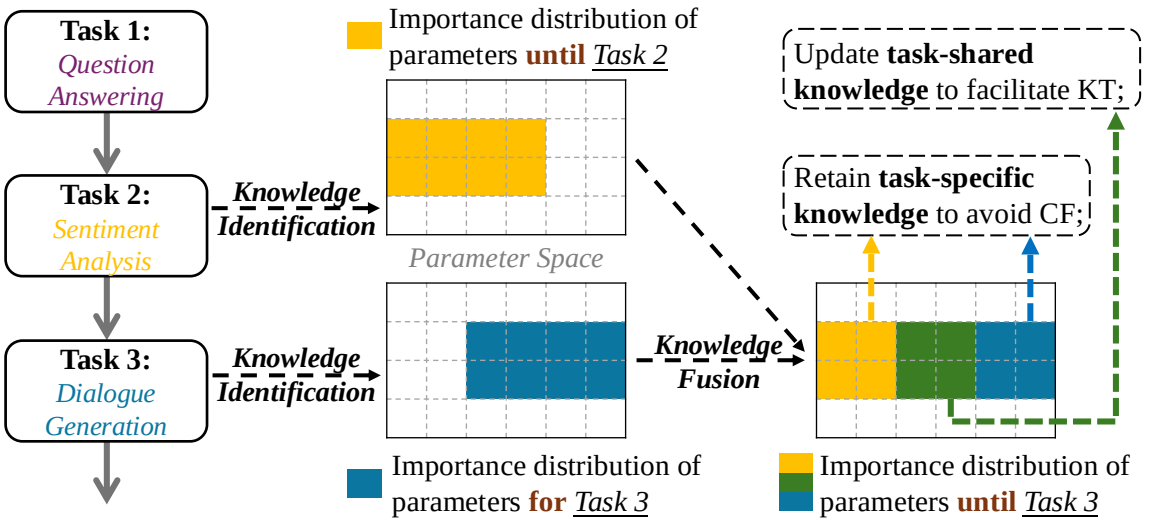


Figure 3.1: Conceptual illustration of KIF. By identifying task-relevant areas across both previously accumulated and current tasks, we can consolidate the task-shared and task-specific parameters to facilitate efficient knowledge transfer (KT) and mitigate catastrophic forgetting (CF).

phase then categorically integrates weights from previous tasks with the current one, efficiently mitigating CF.

In detail, KIF first reconstructs the model or PEFT block into fine-grained "***skill units***". A skill unit refers to a distinct subset of model parameters that encapsulates specific functional capabilities or knowledge relevant to a particular task, such as the Query matrix within the self-attention layer. By operating at this finer granularity, we can localize and consolidate task-specific and shared knowledge within a single PEFT block, rather than adapting a separate PEFT block for each task as in previous works.

The importance-aware knowledge identification method employs a new group-wise metric to compute importance scores, effectively quantifying the significance of each skill unit for the current task. Our knowledge fusion phase, then based on a fine-grained model averaging strategy, effectively manages different types of knowledge. This stage facilitates forward KT by using previously fine-tuned weights as the start-

ing point for new tasks. For backward KT, we merge knowledge from both current and past tasks into localized task-shared skill units, boosting their effectiveness. To prevent CF, we ensure the integrity of skill units containing prior task-specific knowledge is maintained, safeguarding them from being altered by the learning of new tasks.

Given the widespread adoption and success of LoRA in fine-tuning LLMs, there is a compelling opportunity to optimize the KIF framework specifically for LoRA. While the initial KIF framework has shown promising results, its validation has been limited to a specific sub-task, and its broader applicability on general CL benchmarks remains explored. Additionally, the reliance on first-order gradients for knowledge identification in KIF may not yield precise importance localization. The knowledge fusion phase also involves many hyperparameters, which could complicate the averaging process.

To address these issues, we propose KIFLoRA, a variant of KIF optimized for LoRA. KIFLoRA redesigns the LoRA adapter into new skill units based on parameter dependencies, streamlining knowledge management throughout sequential task learning. It incorporates an orthogonal loss during fine-tuning to ensure distinct latent semantic features are captured within the same LoRA adapter. For knowledge identification, KIFLoRA utilizes a new second-order gradient approximate group-wise metric, enhancing the precision of importance scores.

In knowledge fusion, it shifts from hard-mask to soft-mask averaging, employing an adaptive technique that flexibly integrates task-specific and shared parameters according to the importance of the skill unit. Additionally, we combine KIF with memory replay to broaden its application scope, introducing the KIF-M model. KIF framework and its variants excel in mitigating CF and showcase remarkable capabilities for KT, outperforming SOTA methods.

The main contributions are summarized as follows:

- We introduce the Knowledge Identification and Fusion (KIF) **framework** for language model continual learning. By pinpointing and amalgamating task-specific and task-shared knowledge at a granular skill unit level, KIF ensures effective knowledge transfer and significantly reduces catastrophic forgetting, addressing the shortcomings of previous methodologies.
- We devise various group-wise knowledge identification and fine-grained knowledge fusion **techniques**. Notably, our KIF utilizes either first-order gradients or an innovative second-order gradient approximation for assessing parameter importance. For knowledge fusion, we implement strategies ranging from categorical hard-masking to adaptive soft-masking, enhancing the flexibility and precision of knowledge integration.
- The KIF framework is distinguished by its robust **generalizability** and **extensibility**. Its skill units are designed flexibly, allowing for seamless tailoring to PEFT methods, such as LoRA, optimizing performance across different model architectures. Moreover, integrating KIF with memory replay enables further performance enhancements and broadens its applicability to diverse scenarios.
- Extensive **evaluation** on two CL benchmarks demonstrates the superiority of our KIF framework and its variants in facilitating knowledge transfer and mitigating catastrophic forgetting, especially in memory-free scenarios. Furthermore, KIF consistently excels across diverse model sizes (from 220M to 7B), various model architectures (T5 and LLaMA-2), and unseen tasks.

3.3.2 Proposed Method: KIF

Overview KIF includes two key components: (i) ***Knowledge Identification***, utilizing a group-wise importance metric to precisely identify the importance distribution of parameters across tasks, and (ii) ***Knowledge Fusion***, which employs an innovative fine-grained model averaging strategy to integrate model weights from

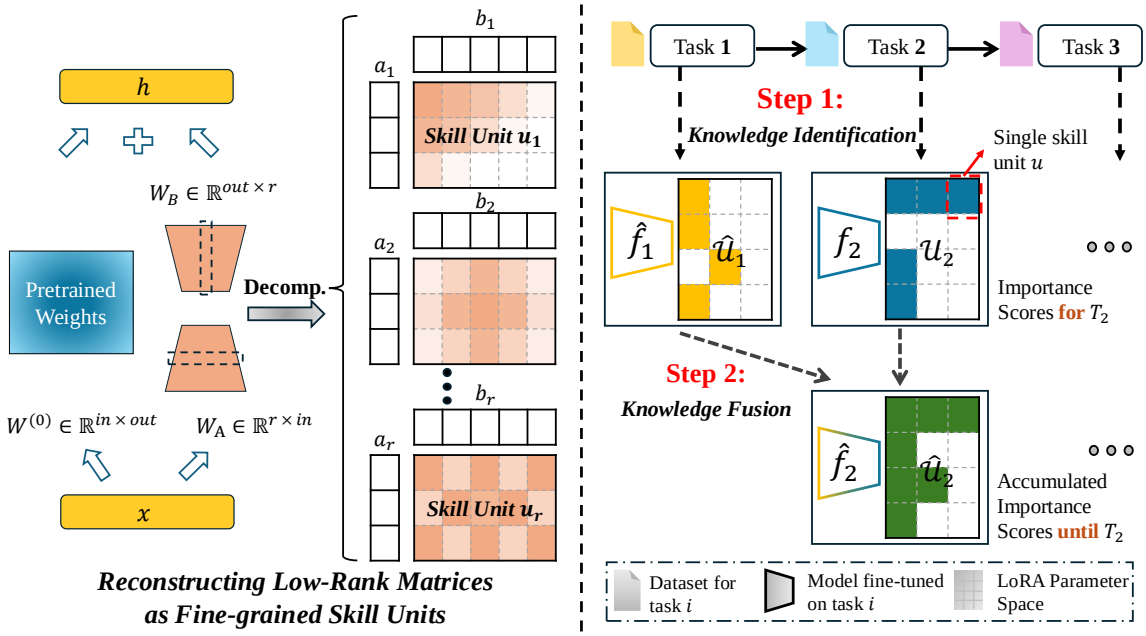


Figure 3.2: *Left:* Depiction of reconstructing LoRA as fine-grained skill units. *Right:* **Overview of KIF. Step 1**: We compute the importance scores of skill units for the current task k using our importance-aware knowledge identification method during fine-tuning. **Step 2**: Based on a categorical model averaging strategy, the knowledge fusion mechanism merges the model $\hat{f}_{k-1}$, which accumulates knowledge of all previous tasks, with the current task's model f_k. This integration is strategically guided by the importance distributions of skill units across different tasks. This iterative process continues with the addition of each new task.

both current and previous tasks for effective knowledge transfer. Figure 3.2 presents a detailed overview of our proposed KIF framework, with the following subsections elaborating on each component and the corresponding extensions.

Fine-Grained Skill Unit

Before localizing the importance of parameters, it is essential to define a structure that facilitates better organization and understanding of the knowledge stored within the model. This structure aids in addressing CF and enhancing KT. We define this

basic structural element of stored skills or knowledge in the model as a "skill unit." Depending on different usage scenarios, we propose two structures for skill units:

Matrix-Level Skill Unit In this division strategy, we define skill units as individual matrices in the model, such as the query or key matrices in the self-attention layer or the A matrix in LoRA. This approach aligns with the original KIF framework and offers the advantage of being model-agnostic. It is suitable for both traditional full-parameter fine-tuning and modern parameter-efficient fine-tuning methods like LoRA, ensuring broad generalizability.

LoRA-Tailored Skill Unit While the matrix-level division is straightforward and intuitive, it does not adequately address intra-matrix redundancy or inter-matrix dependencies.

To address these limitations and the inefficiencies noted in recent work [138], which treats each LoRA adapter as a container for task-specific knowledge, we propose a more refined decomposition of model matrices into new, finer-grained "skill units" based on parameter dependencies specifically tailored for LoRA. The construction process for these LoRA-tailored skill units unfolds as follows.

As shown in Eq (**??**), A and B can be viewed as a combination of a set of vectors: $A = [a_1, a_2, \cdots, a_r]$, $B = [b_1, b_2, \cdots, b_r]$, where $a_i \in \mathbb{R}^{in}$, $b_i \in \mathbb{R}^{out}$. Thus BA can be further disassembled as:

$$\begin{aligned} W &= W^{(0)} + b_1 a_1 + b_2 a_2 + \cdots + b_r a_r \\ &= W^{(0)} + u_1 + u_2 + \cdots + u_r, \end{aligned} \tag{3.1}$$

where each skill unit u_i is a matrix formed by the product of the vectors b_i, a_i. Thus, LoRA can be conceptualized as an aggregation of knowledge across multiple skill units:

$$W = W^{(0)} + \sum_{i=1}^{r} b_i a_i = W^{(0)} + \sum_{i=1}^{r} u_i \tag{3.2}$$

This division strategy significantly reduces the redundancy inherent in the traditional approach of utilizing separate LoRA adapters for each task [138]. To ensure that

different skill units within the same layer learn distinct latent semantic features, we incorporate a regularization term as introduced in [173]:

$$R(A, B) = \|A^T A - I\|_F^2 + \|B^T B - I\|_F^2, \tag{3.3}$$

where I is the identity matrix, this forces A and B to be orthogonal after training.

Importance-aware Knowledge Identification

To calculate the importance of each skill unit u, we introduce a group-wise metric that mitigates the significant computational and storage burdens associated with previous parameter-level importance calculation methods [63]:

$$\mathcal{I}(u) = \frac{1}{in \times out} \sum_{i=1}^{in} \sum_{j=1}^{out} s(w_{ij}) \tag{3.4}$$

where w_{ij} denotes the trainable parameters, and $in \times out$ represents the total parameter count in a skill unit u. The function $\mathcal{I}(u)$ measures the collective importance of all parameters within each skill unit, with higher values indicating greater importance. The importance function $s(\cdot)$ for individual parameters, inspired by the pruning community [64], is quantified by measuring the impact of its removal on the loss. Specifically, to estimate the importance of W_i, we measure the change in loss when the parameter is zeroed out using:

$$\begin{aligned} I_{W_i} &= |\Delta\mathcal{L}(\mathcal{D})| = |\mathcal{L}_{W_i}(\mathcal{D}) - \mathcal{L}_{W_i=0}(\mathcal{D})| \\ &= \left| \frac{\partial \mathcal{L}(\mathcal{D})}{\partial W_i} W_i - \frac{1}{2} W_i H_{ii} W_i + \mathcal{O}(\|W_i\|^3) \right| \end{aligned} \tag{3.5}$$

where H is the Hessian matrix, W_i is the i-th parameter in W, and $\mathcal{L}$ is the next-token prediction loss. If removing a parameter has a significant influence, then the model is sensitive to it, and we should retain it. Based on Eq. (3.5), we can derive the following two metrics for calculating importance:

First-Order Gradient-Based Metric In the original KIF framework, we defined the importance function $s(\cdot)$ as the magnitude of the gradient-weight product:

$$I_{W_i} = |W_i \nabla_{W_i} \mathcal{L}| \tag{3.6}$$

This metric primarily considers the first-order gradient term in determining parameter importance. However, the significance of a parameter is also influenced by second-order gradients, represented by the Hessian matrix. Overlooking this second-order gradient information can introduce biases in the identification process, potentially degrading model performance. Therefore, we propose the following new metric.

Second-Order Gradient Approximation Metric The direct computation of the Hessian matrix for LLMs is impractical due to its $\mathcal{O}(N^2)$ complexity. To address this, we propose a second-order gradient approximation metric that balances the efficiency of first-order methods with the precision of second-order gradients. To reduce computational demands, we approximate the diagonal of the Hessian, H_{ii}, using the Fisher information matrix [113]. The importance is then defined as:

$$I_{W_i} \approx \left| \frac{\partial \mathcal{L}(\mathcal{D})}{\partial W_i} W_i - \frac{1}{2} \sum_{j=1}^{N} \left(\frac{\partial \mathcal{L}(\mathcal{D}_j)}{\partial W_i} W_i \right)^2 \right| \tag{3.7}$$

However, calculating the importance as specified in Eq. (3.7) across the entire training set introduces significant challenges, as model training typically accesses only mini-batch data. This limitation subjects the metric to variability due to stochastic sampling and training dynamics, introducing substantial uncertainty in estimating parameter sensitivity. To mitigate this, we propose a more reliable importance metric based on sensitivity smoothing and uncertainty quantification [173]:

$$\bar{I}^{(t)}\left(w_{ij}\right) = \alpha_1 \bar{I}^{(t-1)}\left(w_{ij}\right) + \left(1-\alpha_1\right) I^{(t)}\left(w_{ij}\right) \tag{3.8}$$

$$\begin{aligned} \bar{U}^{(t)}\left(w_{ij}\right) = \alpha_2 \bar{U}^{(t-1)}\left(w_{ij}\right) + \\ \left(1-\alpha_2\right) \left| I^{(t)}\left(w_{ij}\right) - \bar{I}^{(t)}\left(w_{ij}\right) \right| \end{aligned} \tag{3.9}$$

where α_1 and α_2 are smoothing factors, and t is the iteration number. $\bar{I}^{(t)}$ represents smoothed sensitivity and $\bar{U}^{(t)}$ is the uncertainty term quantified by the local variation between $I^{(t)}$ and $\bar{I}^{(t)}$. Using the exponential moving average of the importance metric, we can retain and examine the trajectory gradient for a longer time, yielding a more

Algorithm 1 Importance-aware Knowledge Identification

Training dataset $\mathcal{D}_k$ for task $\mathcal{T}_k$; total training iterations T; hyperparameters α_1, α_2. $t = 1, \ldots, T$ Sample a mini-batch from $\mathcal{D}_k$ and compute the gradient $\nabla\mathcal{L}$; Compute the sensitivity $I\left(w_{ij}\right)$ via Eq. (3.6) or Eq. (3.7); Update $\bar{I}^{(t)}$ via Eq. (3.8) and $\bar{U}^{(t)}$ via Eq. (3.9); Compute the importance score $\mathcal{I}(u_i^k)$ for each skill unit u_i^k by Eq. (3.4), for $i = 1, \ldots, n$. f_k and importance scores $\mathcal{I}(\mathcal{U}_k)$ for $\mathcal{U}_k$.

precise importance assessment. Importance is ultimately determined by:

$$s^{(t)}\left(w_{ij}\right) = \bar{I}^{(t)}\left(w_{ij}\right) \cdot \bar{U}^{(t)}\left(w_{ij}\right) \tag{3.10}$$

To calculate the importance score of each skill unit for the current task $\mathcal{T}_k$, we use Eq. (3.4) during fine-tuning. The model f with n skill units is denoted as $\mathcal{U} = \{u_1, \ldots, u_n\}$, with their importance scores for task $\mathcal{T}_k$ denoted by $\mathcal{I}(\mathcal{U}_k) \in \mathbb{R}^n$. The detailed computation process is provided in Algorithm 1.

After computing the importance scores for each skill unit at the current task $\mathcal{T}_k$, it is crucial to compare these with scores from all previously learned tasks to distinguish between task-specific and task-shared skill units. To streamline this process and avoid the inefficiencies associated with storing scores for each past task, we aggregate importance scores from all prior tasks into a cumulative score for tasks up to $\mathcal{T}_{k-1}$. This approach allows for iterative refinement of accumulated scores without the need to separately store past task scores. The skill units with these cumulative scores up to $\mathcal{T}_{k-1}$ are denoted as $\hat{\mathcal{U}}_{k-1}$, calculated using:

$$\mathcal{I}(\hat{\mathcal{U}}_{k-1}) = \beta\text{Norm}(\mathcal{I}(\hat{\mathcal{U}}_{k-2})) + (1-\beta)\text{Norm}(\mathcal{I}(\mathcal{U}_{k-1})) \tag{3.11}$$

where $\beta \in [0, 1]$ and $\text{Norm}(\cdot)$ normalizes the importance scores to the [0, 1] range, ensuring consistency across models. The initial scores, $\mathcal{I}(\hat{\mathcal{U}}_1)$, are set equal to $\mathcal{I}(\mathcal{U}_1)$. Following this, the importance distribution for skill units up to task $\mathcal{T}_{k-1}$ is combined with that of the current task, $\mathcal{T}_k$, to facilitate the knowledge fusion process.

Knowledge Fusion

Following knowledge identification, the next critical phase is knowledge fusion, where knowledge within each skill unit is integrated into a cohesive framework. This process utilizes a sophisticated model averaging approach that accounts for various factors to optimize task performance.

Traditional coarse-grained model averaging operates under the assumption that all model weights hold equal importance for the training task [61, 19], typically implemented through the following iterative computation:

$$\hat{f}_k = \lambda \hat{f}_{k-1} + (1 - \lambda) f_k \tag{3.12}$$

However, coarse-grained methods may overemphasize weights that are irrelevant to the current task, contaminating previously acquired task-specific knowledge and leading to forgetting. To overcome these limitations, we introduce two fine-grained averaging strategies that focus on skill units instead of the entire model. Our method distinguishes between task-shared and task-specific skill units, applying a weighted averaging technique to the parameters within each unit. This refined approach offers a tailored solution, addressing the unique requirements of each task more effectively.

Static Weighted Fusion In the original KIF framework, this merging strategy initiates by establishing importance thresholds δ through quantiles, selecting the top 20% of skill units based on importance scores. A skill unit u_i^k is considered important (denoted as $(u_i^k)^+$) if its score $\mathcal{I}(u_i^k)$ exceeds δ_k, and unimportant ($(u_i^k)^-$) otherwise.

The static weighted fusion strategy tailors parameter integration for each skill unit,

according to its importance across different tasks. The method is defined as follows:

$$\hat{u}_i^k = \begin{cases} \gamma\hat{u}_i^{k-1} + (1-\gamma)u_i^k, & \text{if } (\hat{u}_i^{k-1})^+, (u_i^k)^+ \\ \hat{u}_i^{k-1}, & \text{if } (\hat{u}_i^{k-1})^+, (u_i^k)^- \\ u_i^k, & \text{if } (\hat{u}_i^{k-1})^-, (u_i^k)^+ \\ \frac{1}{2}(\hat{u}_i^{k-1} + u_i^k), & \text{if } (\hat{u}_i^{k-1})^-, (u_i^k)^- \end{cases} \tag{3.13}$$

This approach performs the element-wise adjustment of parameters within each skill unit based on its relevance to previous and current tasks. The hyperparameter γ is employed to control their influences.

Adaptive Weighted Fusion As a sophisticated extension to the static method, which necessitated the manual setting of multiple hyperparameters, this adaptive strategy simplifies the process by automatically adjusting to various scenarios. This enhancement increases the efficiency and flexibility of the averaging process. Specifically, for a specific skill unit u_i, the weighting coefficients are determined by:

$$\hat{\lambda}_i^{k-1} = \exp\left(\mathcal{I}(\hat{u}_i^{k-1})/\tau\right), \lambda_i^k = \exp\left(\mathcal{I}(u_i^k)/\tau\right) \tag{3.14}$$

where both *exp* and temperature coefficient τ are scaled to the raw importance score. Then the updated model parameters are:

$$\hat{u}_i^k = \left(\frac{\hat{\lambda}_i^{k-1}}{\hat{\lambda}_i^{k-1} + \lambda_i^k}\right) \cdot \hat{u}_i^{k-1} + \left(\frac{\lambda_i^k}{\hat{\lambda}_i^{k-1} + \lambda_i^k}\right) \cdot u_i^k \tag{3.15}$$

By applying Eq. (3.13) or Eq. (3.15) in our knowledge fusion phrase, it effectively addresses the challenges in CL:

- **Mitigating CF:** When $\mathcal{I}(\hat{u}_i^{k-1})$ is high and $\mathcal{I}(u_i^k)$ is low, this indicates that the skill unit u_i is more critical for previous tasks. According to Eq. (3.13) or (3.15), we maintain the previous task-specific knowledge in $\hat{u}_i^{k-1}$ untouched, preventing contamination from the less relevant u_i^k.

Algorithm 2 KIF

Dataset $\mathcal{D}_k$ for task $k = 1, \ldots, K$; initial pre-trained model f_0; hyperparameters $\alpha_1, \alpha_2, \beta, \tau$. *# sequential tasks.* task $k = 1, \ldots, K$ *# knowledge identification.* Get f_k and calculate $\mathcal{U}_k$ by Algorithm (1); $k = 1$ *# initialization at beginning task.* $\hat{f}_1 \leftarrow f_1, \hat{\mathcal{U}}_1 \leftarrow \mathcal{U}_1$; *# knowledge fusion.* skill unit $i = 1, \ldots, n$ Calculate $\hat{u}_i^k$ by Eq. (3.15); Get the averaged model $\hat{f}_k$ based on $\hat{\mathcal{U}}_k$; Calculate accumulated importance score $\mathcal{I}(\hat{\mathcal{U}}_k)$ according to Eq. (3.11);

- **Facilitating Forward KT:** Conversely, if $\mathcal{I}(\hat{u}_i^{k-1})$ is low and $\mathcal{I}(u_i^k)$ is high, it suggests that u_i holds greater importance for the current task. As training initiates from the endpoint of the previous task, maintaining the integrity of parameters in u_i^k leverages historically learned knowledge to enhance performance on the current task.
- **Enabling Backward KT:** When both $\mathcal{I}(\hat{u}_i^{k-1})$ and $\mathcal{I}(u_i^k)$ are high, u_i is vital for both previous and current tasks. Here, we integrate newly acquired knowledge into this task-shared skill unit to support backward KT.
- **Handling Low Relevance:** If both $\mathcal{I}(\hat{u}^{k-1}i)$ and $\mathcal{I}(u^k i)$ are low, indicating minimal relevance to any task, a simple averaging of parameters within this unit is adequate.

Knowledge fusion is conducted before starting a new task in CL, utilizing the averaged model for subsequent task initialization. Only the importance scores of $\hat{\mathcal{U}}_{k-1}$ and $\mathcal{U}_k$ are preserved for use between tasks, starting with $\hat{\mathcal{U}}_1 = \mathcal{U}_1$ estimated from f_1 on D_1. Detailed implementation of entire KIF algorithm is provided in Algorithm 2.

3.3.3 KIF with Memory Replay: KIF-M

To adapt KIF for scenarios where historical data are accessible, we introduce a memory replay-enhanced version, KIF-M, designed to further improve model performance. The implementation process is shown in Figure 3.3.

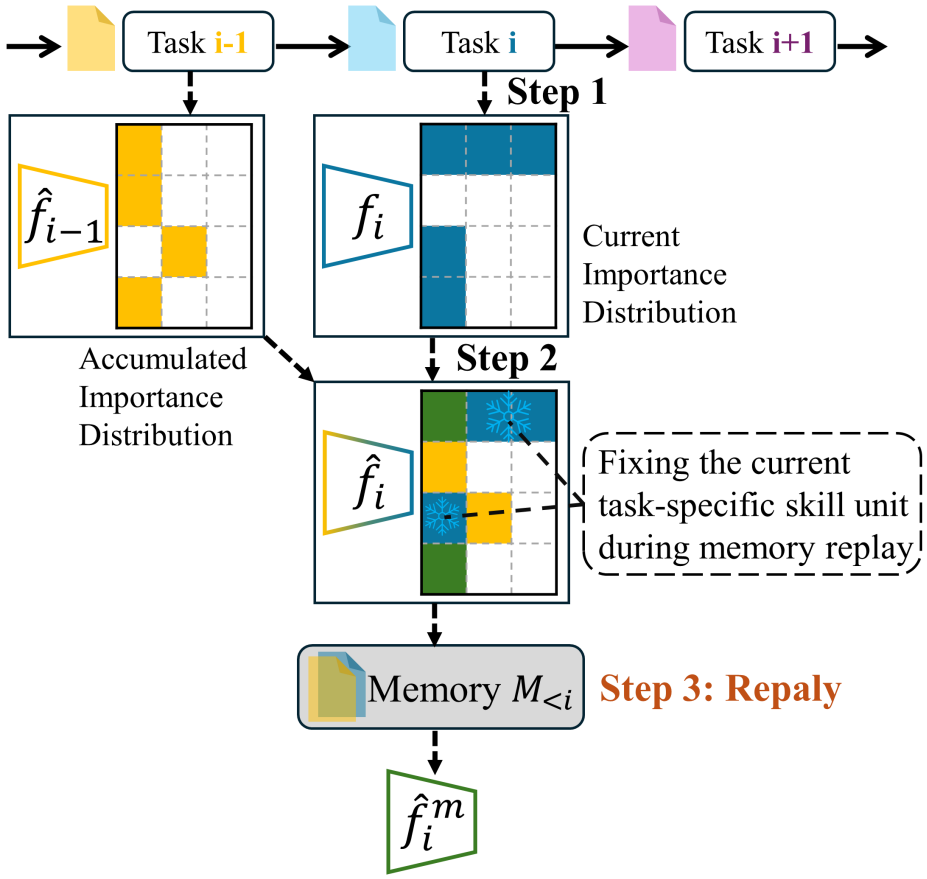


Figure 3.3: Overview of KIF-M. By fixing the current task-specific skill unit during the replay phase, we can further enhance the performance of KIF.

Following the training on task $\mathcal{T}_k$ using the knowledge identification and fusion, we obtain the model $\hat{f}_k$. Before the next task arrives, a replay stage is introduced, where the model $\hat{f}_k$ is fine-tuned using historical data stored in the memory buffer $\mathcal{M}_{<k}$. This process results in a new model $\hat{f}_k^m$, which helps recover forgotten knowledge.

Specifically, during the replay stage, we avoid fine-tuning the parameters of all skill units with historical data, as this could potentially degrade the model's performance on the current task. Instead, we maintain the integrity of the current task-specific skill units and selectively fine-tune the remaining parameters. This approach achieves a better balance between retaining previous knowledge and excelling in new tasks.

3.3.4 Experiments

Experimental Setup

Datasets We utilize the SuperNI Benchmark [143], a comprehensive benchmark designed to evaluate diverse NLP tasks using expert-written instructions. This benchmark facilitates thorough evaluation in more practical settings for the CL of LLMs. It includes tasks in dialogue generation, information extraction, question answering, summarization, and sentiment analysis. Following the established CL setup [179], three tasks are chosen from each type, creating a sequence of 15 tasks in total for evaluation. For each task, 1,000 instances from the dataset are randomly selected for training, and 100 instances are used for validation and testing.

Additionally, we utilize the Long Sequence Benchmark [109], which consists of 15 classification datasets tailored for CL. Consistent with previous work [138], we randomly select 1,000 samples per task for training and reserve 500 samples per class for validation and testing. The experiments include two different task orders for each benchmark.

Evaluation Metrics We denote $a_{j,i}$ as the testing performance (Accuracy for classification task and Rouge-L [78] for others) on the i-th test set of task right after training on j-th task. The performance of CL is assessed using three metrics:

$$\mathbf{AP} = \frac{1}{K}\sum_{i=1}^{K} a_{K,i} \tag{3.16}$$

$$\mathbf{FWT} = \frac{1}{K-1}\sum_{i=2}^{K} a_{i-1,i} \tag{3.17}$$

$$\mathbf{BWT} = \frac{1}{K-1}\sum_{i=1}^{K-1} a_{K,i} - a_{i,i} \tag{3.18}$$

Average Performance (AP) [16] calculates the average performance across all tasks after training on the final task. **Forward Transfer (FWT)** [83] evaluates a model's generalization ability by measuring the averaged zero-shot performance.

Table 3.1: The overall results on two continual learning benchmarks with T5-large model. All results are averaged over two different orders of each benchmark.

Method	Memory-Replay	SuperNI Benchmark			Long Sequence Benchmark		
		AP↑	FWT↑	BWT↑	AP↑	FWT↑	BWT↑
SeqLoRA	✗	6.43	1.58	-30.94	9.72	6.81	-73.37
IncLoRA	✗	19.12	2.03	- 31.24	62.50	2.62	-15.34
ProgPrompt [109]	✗	3.34	5.29	-33.18	7.98	6.63	-66.71
EWC [61]	✗	17.46	4.20	-28.61	45.45	3.73	-25.93
L2P [145]	✗	12.73	1.14	-7.95	57.98	8.36	-16.63
LFPT5 [103]	✗	24.76	7.46	-24.41	67.01	9.48	-12.80
O-LoRA [138]	✗	25.89	8.14	-24.59	69.24	10.15	-4.05
KIF (ours)	✗	26.41	**11.78**	-18.55	70.71	11.80	-3.67
KIFLoRA (ours)	✗	**27.43**	11.02	**-16.91**	**72.29**	**12.89**	**-3.04**
Replay	✓	35.37	2.35	-15.79	70.98	3.28	-15.42
SAPT [179]	✓	51.54	8.88	**-0.57**	82.02	9.86	-1.25
KIF-M (ours)	✓	52.12	11.13	-1.31	84.21	**12.32**	-1.08
KIFLoRA-M (ours)	✓	**53.01**	**11.21**	-0.81	**84.36**	11.71	**-0.98**

Backward Transfer (BWT) [58] assesses the impact of new learning on previous tasks. Negative BWT indicates the model lost some previously acquired knowledge.

Baselines We evaluate KIF against the following PEFT-based CL baseline methods: ***SeqLoRA:*** sequentially trains the LoRA on the task orders. ***IncLoRA:*** incremental learning of new LoRA parameters on a sequential series of tasks. ***Replay:*** replays real samples from old tasks when learning new tasks to avoid forgetting. ***EWC [61]:*** finetune the model with a regularization loss designed to minimize updates to parameters critical to previously learned tasks. ***L2P [145]:*** uses the input to dynamically select and update prompts from a fixed prompt pool. ***LFPT5 [103]:*** continuously trains a soft prompt for each task with generative replay. ***Prog-Prompt [109]:*** sequentially concatenates previous learned prompts to the current one during the training and testing time. ***O-LoRA [138]:*** learns tasks in distinct LoRA subspaces, ensuring orthogonality, and aggregates all LoRA weights for testing. ***SAPT [179]:*** leverages pseudo samples and a shared attention framework to align PEFT block

Table 3.2: The comparison between KIF and other CL methods. Specifically, **RF** indicates whether the method is rehearsal-free. **PE** indicates whether the method is parameter efficient. **UT** indicates whether the method can be applied to solve unseen tasks. **KT** indicates whether the method enables knowledge transfer.

Method	RF	PE	UT	KT
EWC [61]	✓			✓
OGD [22]	✓		✓	
LFPT5 [103]		✓		✓
L2P [145]	✓	✓		
EIP [144]	✓	✓		
O-LoRA [138]	✓	✓	✓	
MoCL [132]	✓		✓	✓
SAPT [179]		✓	✓	✓
KIF	✓	✓	✓	✓

learning and selection.

Table 3.2 compares our KIF with other baselines, highlighting three key advantages: data privacy-friendliness, model parameter efficiency, and improved generalization capabilities. To clarify the specific identification and fusion techniques used in KIF and its variants: ***KIF:*** our foundational framework incorporates matrix-level skill units, utilizes a first-order gradient-based metric for knowledge identification (Eq. 3.6), and employs static weighted fusion (Eq. 3.13). ***KIFLoRA:*** an extension of KIF that introduces LoRA-tailored skill units, adopts a second-order gradient approximation metric (Eq. 3.7), and features adaptive weighted fusion (Eq. 3.15). ***KIF-M and KIFLoRA-M:*** these enhancements of KIF and KIFLoRA integrate memory replay to further boost performance.

Implementation Details We utilize two distinct language model architectures: the

encoder-decoder T5 model [106] and the decoder-only LLaMA model [122]. In KIF, the hyperparameters α_1 and α_2 in Eq. (3.8) and Eq. (3.9) are set to 0.85. We set β in Eq. (3.11) to 0.7 and τ in Eq. (3.14) to 0.15. Following [118], we allocate 2% of the original training set volume for replay samples in all replay-based baseline methods. Hyperparameters are as follows:

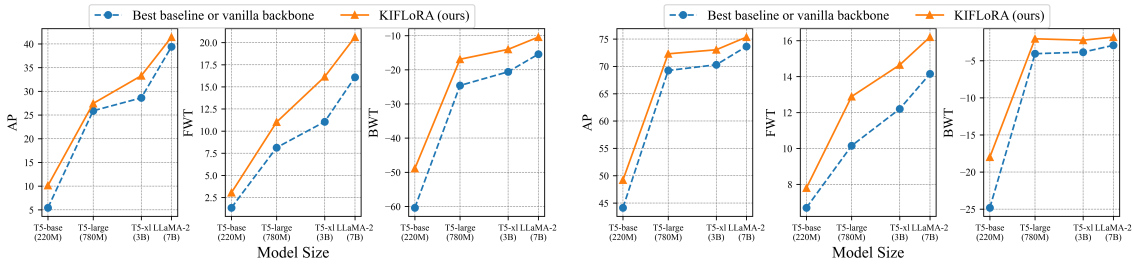
- T5-base (220M), T5-large (780M), and T5-XL (3B): Training was conducted with a learning rate of 3e-4, batch size of 8, maximum input length of 512, maximum target length of 128, a rank of 8, targeting modules [q, v] and 10 epochs.
- LLaMA (7B): With a learning rate of 3e-4, a batch size of 128, a cutoff length of 512, and 10 epochs. LoRA settings were rank = 8, alpha = 16, dropout = 0.05, targeting modules [q_proj, v_proj]. For testing, settings included temperature = 0.02, top_p = 0, top_k = 1, max new tokens = 128.

Main Results

The overall CL results of various methods using the same T5-large backbone are summarized in Table 3.1.

Firstly, KIF and its variants consistently demonstrate superior CL performance by effectively facilitating knowledge transfer. Compared to previous replay-free methods like IncLoRA and O-LoRA, our methods excel in addressing the critical challenges of CF and KT, achieving the highest AP, FWT, and BWT when learning tasks sequentially.

Furthermore, our enhanced version, KIFLoRA, significantly outperforms KIF. Specifically, KIF-M achieves average gains of 1.3% in AP and 0.3% in BWT, underscoring the effectiveness of enhancements to each framework component. Notably, even without memory replay, KIF-M surpasses the performance of vanilla repla on the Long Sequence Benchmark, particularly in BWT.



(a) Results on the SuperNI benchmark

(b) Results on the Long Sequence benchmark

Figure 3.4: Performance of KIFLoRA across various backbones.

For the memory-based methods, to ensure a fair comparison, we adopted the same data synthesis-based generative replay approach used in SAPT, ensuring that the amount of data used in memory replay is consistent across methods. The results show that both KIF-M and KIFLoRA-M surpass the best SAPT models, further proving the versatility and robustness of our framework. For subsequent experimental analyses and comparisons, we will focus on KIFLoRA, the most effective memory-free version, for further evaluations.

Secondly, KIFLoRA consistently demonstrates superior performance across various backbones. To further validate our framework's effectiveness, we conducted experiments using a range of parameter-level backbones. As depicted in Figure 3.4, performance improvements are evident with increasing model size. For instance, in T5-XL, KIFLoRA boosts the AP metric from 28.6% to 33.3%, and shows substantial enhancements in FWT and BWT, rising from 11.1% to 16.1% and improving from -20.6% to -14.0%, respectively. These results further validate the robust generalization ability of our method.

Thirdly, our knowledge fusion technique effectively counters catastrophic forgetting. To rigorously assess our model's ability to mitigate forgetting, we evaluated its performance on the initial task after training on subsequent tasks using the Long Sequence Benchmark. Figure 3.5 illustrates that KIFLoRA significantly reduces the rate of for-

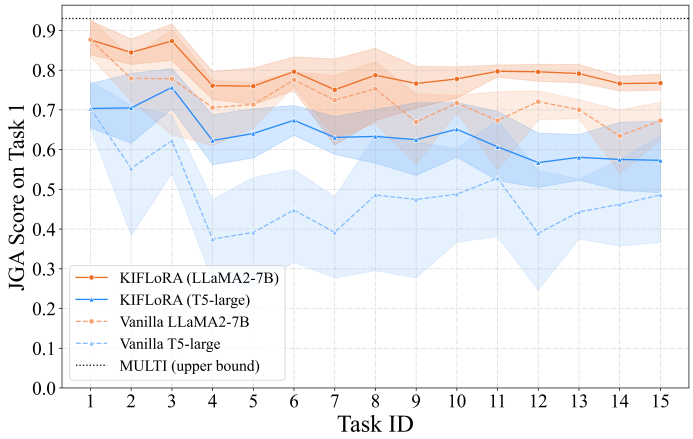


Figure 3.5: Performance trajectory of Task 1 on Long Sequence Benchmark throughout the continual learning process.

getting, with an average performance decline of only 10% after training on the final task. In contrast, vanilla backbones suffer a substantial average performance drop of 22%, highlighting our method's enhanced capacity to preserve learned knowledge.

Performance on Unseen Tasks

Building on prior research [179], we selected three tasks from each category to evaluate our method's cross-task generalization capabilities, a critical measure for CL algorithms. The results, indicated by the average Rouge-L scores in Table 3.3, highlight the effectiveness of our approach. T5-ZS refers to the zero-shot task adaptation methods employed. KIFLoRA exhibited the best performance among all methods tested, a success largely due to its proficient identification and management of task-specific and shared parameters.

Visualization of Skill Units

We visualized the distribution of importance scores for skill units across tasks on the T5-large model, as shown in Figure 3.6 and Figure 3.7, leading to several critical

Table 3.3: Results on unseen tasks using the T5-Large backbone.

	Unseen Tasks					**Avg.**
	Dialog	**IE**	**QA**	**Sum**	**SA**	
T5-ZS	7.49	6.70	4.28	12.14	4.54	7.03
O-LoRA	4.39	9.89	25.38	8.26	**50.41**	19.67
LFPT5	6.96	**35.32**	35.00	13.26	21.51	22.41
KIFLoRA	**10.32**	31.34	**37.13**	**14.20**	47.17	**28.03**

insights:

- There is noticeable variability in the importance of skill units for the same task, with important skill units comprising only a small fraction of all trainable LoRA parameters.
- The distribution of important skill units is task-dependent, illustrating the presence of both task-shared and task-specific parameters. This observation substantiates the feasibility of the design motivations behind the KIF framework.
- For classification tasks, such as those in the Long Sequence Benchmark (Figure 3.6), the skill units in the encoder, particularly the lower layers closer to the model input, play a more pivotal role. Conversely, for generative tasks, like dialogue generation and summarization in the SuperNI benchmark (Figure 3.7), both encoder and decoder units are crucial, impacting both lower and upper layers of the network.
- Within each layer, the importance of the Query (Q) matrices in the attention mechanism consistently exceeds that of the Value (V) matrices.

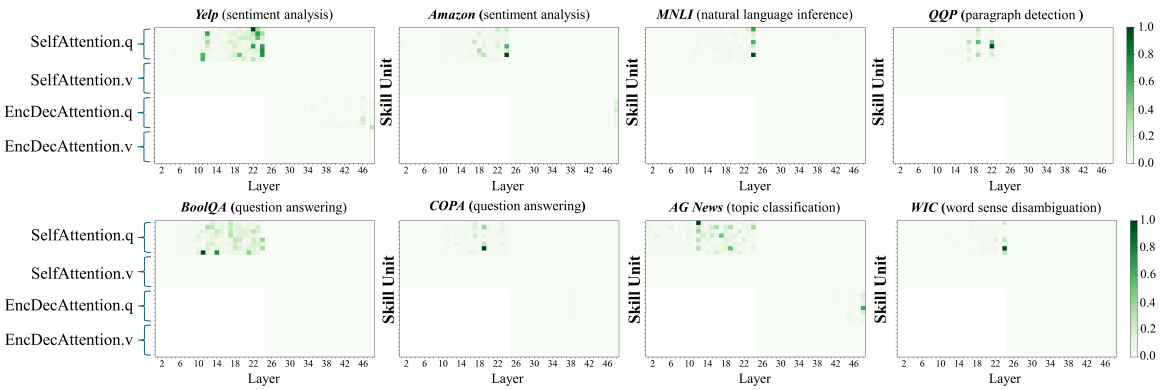


Figure 3.6: Visualization of importance scores for LoRA-tailored skill units across different tasks on T5-large within the Long Sequence Benchmark.

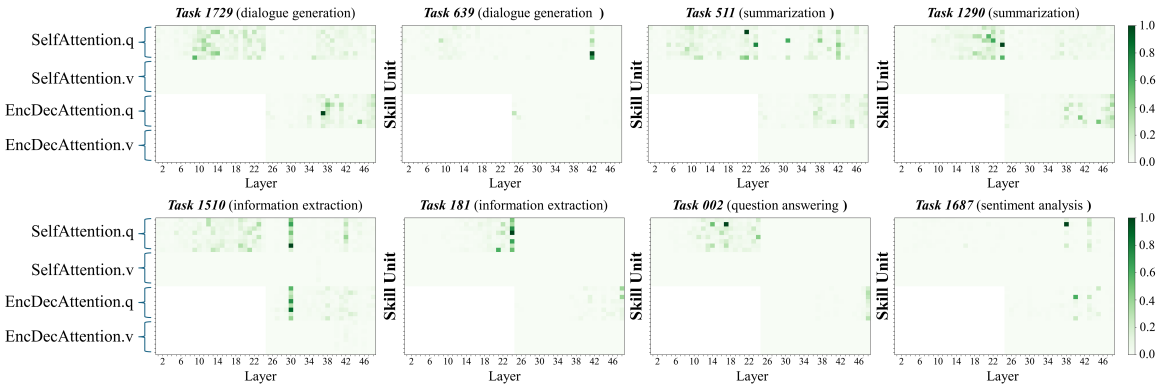


Figure 3.7: Visualization of importance scores for LoRA-tailored skill units across different tasks on T5-large within the SuperNI Benchmark.

Ablation Study

In this section, we assess the impact of importance-aware knowledge identification, fine-grained knowledge fusion, and memory size on model performance.

Effect of the Proposed Importance Metric in Knowledge Identification We explore alternative importance scoring methods to validate the effectiveness of our approach: (i) To evaluate the impact of using moving averages on trajectory gradients, we modify $s(\cdot)$ in Eq. (3.4) to focus solely on sensitivity, as outlined in Eq. (3.7); (ii) To verify the validity of our proposed approximate second-order gradient

Table 3.4: Ablation Study for Importance-Aware Knowledge Identification.

Method	AP	FWT	BWT
vanilla T5-large	54.10	3.32	-29.63
$s(\cdot) = I(\cdot)$	70.48	10.39	-3.81
$s(\cdot) = \left\vert\nabla_{w_{ij}}\mathcal{L}\right\vert$	68.82	10.80	-6.22
KIFLoRA (ours)	**72.29**	**12.89**	**-2.04**

importance metric in Eq. (3.4), we implement a first-order Taylor expansion as described in Eq. (3.6). The comparative results, presented in Table 3.4, show that using moving averages for importance scoring substantially enhances model performance. Additionally, compared to the first-order gradient approximation, our novel second-order gradient method results in performance improvements of up to 3.4%, 2.1%, and 4.2% across the evaluated metrics. These findings underscore the significant role of accurate knowledge identification.

Effect of Adaptive Knowledge Fusion We evaluate our fine-grained, skill unit-level adaptive weighted fusion against two coarse-grained strategies: (i) Weight-Ensemble, which uniformly averages the LoRA parameters using a global weight as per Eq. (3.12), and (ii) Exponential Moving Average (EMA) [120], which computes a running average of parameters at each fine-tuning iteration.

According to Table 3.5, Weight-Ensemble significantly enhances the performance of the vanilla model, demonstrating the advantages of coarse-grained averaging in CL. While EMA generally outperforms Weight-Ensemble, it does not match the effectiveness of our fine-grained approach. EMA's frequent parameter adjustments within the same task can lead to suboptimal outcomes. In contrast, our method, which strategically averages weights only after the completion of each task, boosts computational efficiency and model performance.

Table 3.5: Ablation Study for Fine-Grained Knowledge Fusion.

Method	AP	FWT	BWT
vanilla T5-large	54.10	3.32	-29.63
Weight-Ens.	63.28	7.71	-11.82
EMA	62.76	8.23	-9.80
KIFLoRA (ours)	**72.29**	**12.89**	**-3.04**

Table 3.6: Ablation Study on Memory Size.

	Memory Size			
	2%	5%	10%	50%
Replay	71.2	72.4	73.8	76.1
KIF-M	74.2	75.1	75.7	79.2
KIFLoRA-M	**74.3**	**75.4**	**76.2**	**79.8**

The Effect of Memory Size We examine how varying the memory size affects the performance of Replay, KIF-M, and KIFLoRA-M. By adjusting the memory size per task $|M|$ to 2%, 5%, 10%, 50%, we summarized the results in Table 3.6. As expected, increasing the memory size generally enhances the performance of all methods, with Replay showing the most significant improvement. This is predictable as Replay heavily depends on memory to mitigate catastrophic forgetting. When the memory size is substantially increased, Replay approaches the functionality of multi-task learning, which, while effective, incurs considerable storage and computational costs.

Conversely, KIF-M and KIFLoRA-M utilize knowledge fusion strategies to effectively maintain parameters that store historical knowledge. Consequently, while increasing memory size does boost their performance, the improvements are less dramatic as the memory size grows.

3.3.5 Conclusion

We introduced the novel Knowledge Identification and Fusion (KIF) framework for language model continual learning. KIFLoRA employs importance-aware knowledge identification and fine-grained knowledge fusion to effectively differentiate between task-specific and shared knowledge within each skill unit, mitigating forgetting while enhancing knowledge transfer. The framework is highly generalizable and extensible, allowing integration with memory replay methods to boost performance. Extensive testing shows that KIFLoRA excels in preserving past knowledge and performing well in new tasks, surpassing previous state-of-the-art approaches. Comprehensive experiments demonstrate demonstrate our framework's exceptional ability.

Despite significant improvements in efficiency and performance in large language model continual learning, certain limitations persist. For instance, the choice of importance thresholds following the determination of importance distribution can impact the effectiveness of knowledge fusion. Dynamically selecting these thresholds

based on data distribution may yield a more precise classification of task-shared and task-specific parameters, further enhancing performance. Additionally, merging the knowledge identification and fusion phases could allow for ongoing parameter consolidation based on their importance during model training. This integrated approach would facilitate more flexible adaptations and better address scenarios in online continual learning.

3.4 Methodology – Recurrent Knowledge Identification and Fusion (Recurrent-KIF)

3.4.1 Introduction

The success of parameter importance-based approaches is contingent on the accurate estimation of parameter importance. A key limitation lies in their reliance on ***static importance estimations***, where the parameter importance scores for previous tasks remain unchanged and are not updated during subsequent training. Over time, as model parameters gradually diverge from the state at which the Hessian was originally computed, these unadjusted importance estimates become increasingly inaccurate due to the growing truncation error in the Taylor expansion.

The human brain demonstrates remarkable CL ability through two alternating systems: the hippocampus, which quickly acquires representations for specific experiences, and the neocortex, which selectively consolidates useful memories into long-term storage. This process is known as the Complementary Learning Systems (CLS) theory [94] in neuroscience.

Drawing inspiration from the CLS theory, we propose Recurrent Knowledge Identification and Fusion (Recurrent-KIF), a novel CL framework that dynamically estimates parameter importance and iteratively fuses knowledge. Recurrent-KIF integrates an

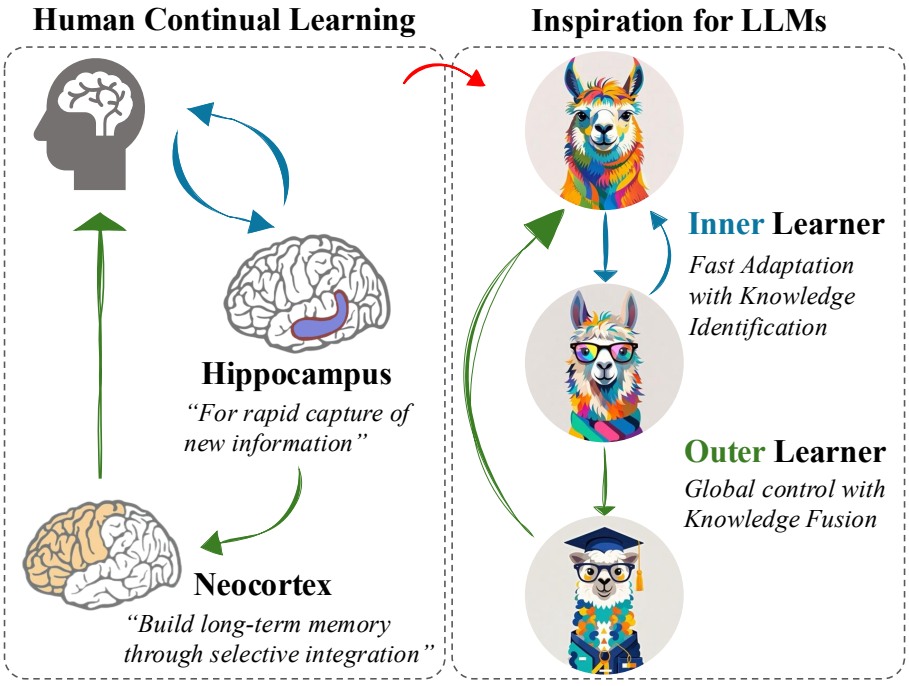


Figure 3.8: Conceptual illustration of Recurrent-KIF. Inspired by the CLS theory, Recurrent-KIF iteratively employs an inner learner to localize new knowledge and an outer learner to manage the global fusion of knowledge.

inner learner, which rapidly adapts to new task-specific knowledge, and an ***outer learner***, which manages the global fusion of new and historical knowledge (see Figure 3.8).

In detail, the inner learner adapts to new knowledge while utilizing the proposed knowledge identification method to identify important parameters. The outer learner then retrieves historical task information from a memory buffer based on the latest model state, enabling dynamic updates of the importance distributions for previous tasks. Subsequently, a knowledge fusion mechanism is employed to integrate new and historical knowledge by pruning redundant information to mitigate CF and merging key knowledge to enhance KT. Through iterative cycles of multiple rounds of fusion, Recurrent-KIF effectively captures valuable information throughout the model training process, distinguishing it from traditional post-training fusion methods. Each

knowledge fusion step adaptively updates fusion weights according to the most recent importance distributions, resulting in smoother and more controlled optimization.

We conduct extensive experiments to assess the effectiveness of Recurrent-KIF on two CL benchmarks for LLMs. The results consistently highlight the superiority of Recurrent-KIF in mitigating CF while exhibiting exceptional KT capabilities, outperforming state-of-the-art methods. Furthermore, Recurrent-KIF exhibits robust scalability across various model architectures and sizes (from 770M to 13B), underscoring its generalization ability.

Our main contributions are summarized as:

- We propose Recurrent-KIF, a novel CL **framework** for recurrent knowledge identification and fusion that dynamically estimates parameter importance and iteratively integrates knowledge.
- We introduce a new **learning paradigm** for Recurrent-KIF, featuring an inner learner that rapidly captures and localizes new information, and an outer learner that globally controls the fusion of new and historical knowledge.
- Extensive **evaluation** validates the effectiveness of Recurrent-KIF in addressing CL challenges.

3.4.2 Proposed Method: Recurrent-KIF

Notation We consider a pre-trained model $\theta \in \mathbb{R}^n$ with n parameters. After training on task $\mathcal{T}_{k-1}$, the model are denoted as θ^{k-1}. Fine-tuning on a new task $\mathcal{T}_k$ produces updated parameters θ^k. The difference $\tau^k = \theta^k - \theta^{k-1}$, referred to as the *task vector* or *training residual* [52], represents task-specific parameter updates. In the Recurrent-KIF framework, we obtain transient training residuals through each iteration of the inner and outer loops. Specifically, two task vectors are employed to capture and quantify the new knowledge learned in the inner loop and the historical

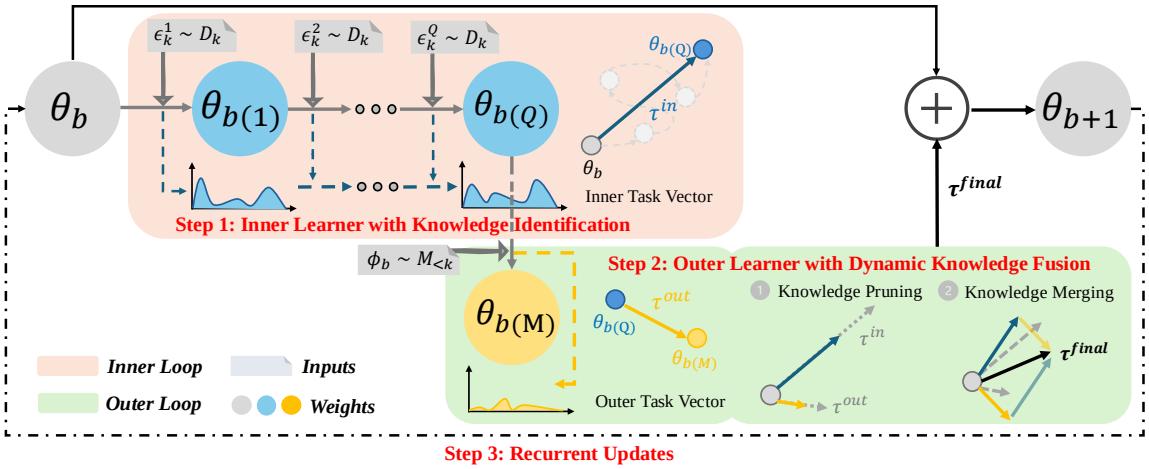


Figure 3.9: **Iterative update process of Recurrent-KIF for the b-th iteration.** The notation ϵ_k^q represents training samples drawn from $\mathcal{D}_k$, while ϕ_b refers to samples drawn from $\mathcal{M}_{<k}$. **Inner Learner (Step 1):** Performs Q iterations to rapidly adapt to the new task while identifying the parameter importance distribution. **Outer Learner (Step 2):** Retrieves historical task information using memory data and performs knowledge fusion, guided by the importance distributions of both current and historical tasks. **Recurrent Updates (Step 3):** This inner-outer loop cycle is repeated, ensuring that each fusion knowledge step is based on up-to-date importance distributions.

knowledge retrieved in the outer loop.

Overview Recurrent-KIF restructures the training process into multiple iterative learning cycles, each comprising two key components as illustrated in Figure 3.9: (i) ***Inner Learner with Knowledge Identification:*** rapidly acquires new task knowledge while estimating the corresponding parameter importance, and (ii) ***Outer Learner with Knowledge Fusion:*** utilizes a memory buffer to retrieve historical task information. By leveraging the importance distributions of both current and historical tasks, it provides global control for effective knowledge transfer through redundant knowledge pruning and key knowledge merging.

Inner Learner with Knowledge Identification

Assume the current task is $\mathcal{T}_k$, and the iterative update for the model parameters θ^{k-1} at the b-th iteration are denoted by θ_b^{k-1} [1]. In the inner loop, the model initializes with $\theta_{b(0)} = \theta_b$ and is rapidly updated over Q gradient steps using batch data ϵ_k^q sampled from $\mathcal{D}_k$ at the q-th step. After obtaining $\theta_{b(Q)}$ the task-specific updates are encapsulated in the task vector $\tau_b^{in} \in \mathbb{R}^n$:

$$\tau_b^{\text{in}} = \theta_{b(Q)} - \theta_{b(0)} \tag{3.19}$$

This task vector captures the knowledge acquired for the current task. However, τ^{in} often contains redundant information, and directly merging it into the model may compromise historical knowledge, leading to catastrophic forgetting. To address this, we propose a knowledge identification technique to identify the key parameters which storing critical knowledge within the task vector.

We use a commonly adopted importance metric in model pruning [63], defined as the magnitude of the gradient-weight product:

$$\bar{I}\left(w_{ij}\right) = \left| w_{ij} \nabla_{w_{ij}} \mathcal{L} \right| \tag{3.20}$$

where w_{ij} represents trainable parameters.

Due to stochastic batch sampling and training dynamics, the metric in Eq. (3.20) may be unreliable, introducing variability [172]. To mitigate this, we apply an exponential moving average [174] to smooth the trajectory gradients over Q inner loop iterations:

$$I_{b(q)} = \alpha_1 I_{b(q-1)} + (1 - \alpha_1)\, \bar{I}_{b(q)} \tag{3.21}$$

where α_1 is the smoothing factor, $q \in \{1, 2, ..., Q\}$ is the iteration number in the inner loop, and $I_{b(q)}$ represents smoothed importance. The inner task vector τ_b^{in} and its associated parameter importance I_b^{in} are then passed to the outer learner.

[1] For simplicity, we omit the superscripts $k-1$ in subsequent descriptions.

Outer Learner with Knowledge Fusion

The outer loop manages the global merging of knowledge, guided by parameter importance. To access historical knowledge, after acquiring $\theta_{b(Q)}$, the outer loop samples data ϕ_b from the memory buffer $\mathcal{M}_{<k}$. It then performs several training iterations, updating the parameters to $\theta_{b(M)}$. Then the outer task vector $\tau_b^{\text{out}} \in \mathbb{R}^n$, capturing historical task information, is defined as:

$$\tau_b^{\text{out}} = \theta_{b(M)} - \theta_{b(Q)} \tag{3.22}$$

Dynamic Update of Historical Importance Distribution. While obtaining the outer task vector, we calculate the historical task importance distribution based on the latest model state $\theta_{b(Q)}$, using Eq. (3.20). The update process is then expressed as:

$$\bar{I}_b^{\text{out}} = \mathbb{P}(\bar{I}_b^{\text{out}} \mid \theta_{b(Q)}) \tag{3.23}$$

This update, based on conditional probability, enables the computation of the historical importance distribution I_b^{out} using the current model state. This distinguishes it from traditional static importance estimation methods and ensures more accurate knowledge identification. However, the limited sample size from the memory buffer can introduce significant variance in the importance estimates. To address this, we also apply exponential smoothing to the previous outer loop distribution I_{b-1}^{out}:

$$I_b^{\text{out}} = \alpha_2 \bar{I}_b^{\text{out}} + (1 - \alpha_2) I_{b-1}^{\text{out}} \tag{3.24}$$

where α_2 is the smoothing factor, enhancing stability and robustness in importance estimation.

Knowledge Fusion via Importance-based Binary Mask. Knowledge fusion is guided by the importance distributions I_b^{in} and I_b^{out}. To binarize the importance

distributions, a quantile-based threshold δ is applied to select the top 20% of parameters from both I_b^{in} and I_b^{out}. This generates binary masks $m_b^{in} \in \mathbb{R}^n$ and $m_b^{out} \in \mathbb{R}^n$, defined as:

$$m_b^{\text{in}} = \mathbb{I}(I_b^{\text{in}} \geq \delta_b^{in}), m_b^{\text{out}} = \mathbb{I}(I_b^{\text{out}} \geq \delta_b^{out}) \tag{3.25}$$

where $\mathbb{I}(\cdot)$ is the indicator function that outputs 1 if the condition is met and 0 otherwise. Knowledge fusion is then performed as follows:

$$\theta_{b+1} = \theta_b + (m_b^{\text{in}} \odot \tau_b^{\text{in}} + m_b^{\text{out}} \odot \tau_b^{\text{out}}) \tag{3.26}$$

where $\odot$ denotes element-wise multiplication.

This knowledge fusion mechanism provides precise global control, effectively tackling key challenges in CL. First, redundant information in the task vectors τ^{in} and τ^{out} is filtered out via the mask operation. Second, task-shared knowledge is effectively merged to facilitate knowledge transfer. Lastly, task-specific knowledge is preserved to prevent catastrophic forgetting.

The inner and outer loops operate iteratively, enabling multi-round fusion of knowledge. This iterative process facilitates the capture and absorption of useful information generated during training, providing smoother optimization compared to traditional post-training fusion methods. Detailed implementation of Recurrent-KIF algorithm is provided in the Algorithm 3.

3.4.3 Experiments

Dataset We adopt the experimental setup from Du et al. [18], using two CL benchmark datasets: (i) **Standard CL Benchmark**, which consists of five text classification tasks from Zhang et al. [176]: AG News, Amazon Reviews, Yelp Reviews, DBpedia, and Yahoo Answers. (ii) **Long Sequence Benchmark**, a more challenging evaluation scenario comprising 15 tasks [109]: five from the Standard CL Benchmark, four from the GLUE benchmark [124], five from SuperGLUE [125], and the IMDB

Algorithm 3 The Algorithm of the proposed Recurrent-KIF

Current task dataset $\mathcal{D}_k$, memory buffer $\mathcal{M}_{<k}$, model weights θ, initial inner-loop learning rate β_{in}, outer-loop learning rate β_{out}, number of inner-loop steps Q, number of outer-loop steps S, hyperparameters α_1, α_2, total number of fusion steps N. *# training iterations.* $b = 1, \ldots, N$ sample training data ϵ_k^q $(q = 1, \ldots, Q)$ from $\mathcal{D}_k$ $\theta_{b(0)} = \theta_b$ *# inner loop.* $q = 1$ Q obtain batch samples ϵ_k^q $\theta_{b(q)} = \theta_{b(q-1)} - \beta_{in} \nabla \mathcal{L}\left(\theta_{b(q-1)}\right)$ *# knowledge identification.* compute the importance $\bar{I}\left(w_{ij}\right)$ via Eq. (3.20); update $I_{b(q)}$ via Eq. (3.21) calculate $\tau_b^{\text{in}} = \theta_{b(Q)} - \theta_{b(0)}$ and obtain I_b^{in} *# outer loop.* $\theta_{b(M)}^0 = \theta_{b(Q)}$ $s = 1$ **to** S sample memory data ϕ_b^s from $\mathcal{M}_{<k}$ $\theta_{b(M)}^s = \theta_{b(M)}^{s-1} - \beta_{out} \nabla \mathcal{L}\left(\theta_{b(M)}^{s-1}\right)$ calculate $\tau_b^{\text{out}} = \theta_{b(M)}^S - \theta_{b(Q)}$ calculate I_b^{out} via Eq. (3.24) *# knowledge fusion.* obtain m_b^{in}, m_b^{out} via Eq. (3.25) update $\theta_{b+1} = \theta_b + (m_b^{\text{in}} \odot \tau_b^{\text{in}} + m_b^{\text{out}} \odot \tau_b^{\text{out}})$

Movie Reviews dataset [89]. Following Wang et al. [138], we sample 1000 instances for training on each task and reserve 500 per class for validation.

Metrics Let $a_{i,j}$ denote the testing performance on task $\mathcal{T}_i$ after training on task $\mathcal{T}_j$. We evaluate the overall performance (OP) [9] and backward transfer (BWT) [83] after training on the final task:

$$\mathbf{OP} = \frac{1}{K} \sum_{i=1}^{K} a_{i,K} \tag{3.27}$$

$$\mathbf{BWT} = \frac{1}{K-1} \sum_{i=1}^{K-1} (a_{i,K} - a_{i,i}) \tag{3.28}$$

Baselines We compare Recurrent-KIF against a range of baseline methods, including traditional CL approaches, recent PEFT-based model ensemble and merging methods. (1) ***SeqLoRA:*** LoRA parameters are trained on a task sequence without regularization or sample replay. (2) ***IncLoRA:*** incremental learning of LoRA parameters without regularization or sample replay. (3) ***LoRAReplay:*** LoRA fine-tuning with a memory buffer. (4) ***EWC [61]:*** finetune LoRA with a regularization loss to

Method	Standard CL benchmarks		Long Sequence Benchmark	
	OP↑	BWT↑	OP↑	BWT↑
SeqLoRA	43.7	-50.4	11.6	-73.4
IncLoRA	66.4	-20.0	61.2	-26.7
LoRAReplay	68.8	-11.7	70.9	-15.4
EWC* [61]	50.3	-	45.1	-
L2P* [145]	60.7	-	56.1	-16.3
LFPT5* [103]	72.7	-	69.2	-12.8
MoELoRA* [86]	54.1	-	27.6	-
O-LoRA* [138]	75.8	-3.8	69.6	-4.1
TaSL [28]	76.3	-4.0	74.4	-5.3
VR-MCL [149]	76.0	-3.7	74.8	-4.9
MIGU* [18]	76.6	-	76.5	-
Recurrent-KIF (ours)	**78.4**	**-2.8**	**77.8**	**-3.6**
MTL	80.3	-	81.8	-
SAPT-LoRA [179]	-	-	82.0	-1.3

Table 3.7: Overall results on two CL benchmarks using the T5-large model. We report Overall Performance (OP) and Backward Transfer (BWT) after training on the final task. All results are averaged over three different task orders. Methods marked with $*$ are copied from previous papers. The last two rows represent upper bound performance.

prevent interference with previous tasks. (5) ***L2P [145]:*** dynamically selects and updates prompts from a pool on an instance-by-instance basis. (6) ***LFPT5 [103]:*** learns a soft prompt that solves tasks and generates training samples for replay. (7) ***O-LoRA [138]:*** extends IncLoRA to learn different LoRAs in orthogonal subspaces. (8) ***MoELoRA [86]:*** a vanilla MoE with LoRA number equals to the task number. (9) ***SAPT [179]:*** uses pseudo samples and a shared attention framework to align PEFT block learning and selection (10) ***TaSL [28]:*** selectively updates or retains skill regions based on parameter importance. (11) ***MIGU [18]:*** updates important parameters based on gradient magnitude. (12) ***VR-MCL [149]:*** dynamically

updates historical task parameter importance distributions using memory replay. Additionally, multi-task learning with LoRA, referred to as ***MTL***, serves as the upper bound.

Training Details We evaluate Recurrent-KIF using two distinct language model architectures: the encoder-decoder T5 model [106] (T5-large and T5-xl), and the decoder-only LLaMA model [122] (LLaMA2-7B and LLaMA2-13B)[2]. Hyperparameters α_1 and α_2 in Eq. (3.21) and Eq. (3.24) are set to 0.55, with the number of inner loop iterations Q set to 8, and the number of outer loop iterations set to 4. Following Zhao et al. [179], 2% of the original training set is used for replay samples. All experiments are averaged over 3 runs.

Main Results

The overall CL results using the same T5-large backbone are summarized in Table 3.7.

Our Recurrent-KIF effectively addresses the challenges of CF and KT simultaneously. Compared to both traditional CL methods (LoRAReplay, L2P) and model ensemble-based methods (MoELoRA, O-LoRA), Recurrent-KIF outperforms them in both CF (increasing average OP from 72.7% to 78.1% compared to O-LoRA) and KT (increasing average BWT from -13.6% to -3.2% compared to LoRAReplay). SAPT achieves the highest performance by leveraging generative replay-based data augmentation, surpassing MTL result. However, it relies heavily on external data synthesis, which can be costly in LLM settings.

Moreover, when compared to parameter importance-based methods like TaSL and VR-MCL, Recurrent-KIF consistently delivers the best OP and BWT scores. Notably,

[2]Due to being limited by an academic computing budget, we employed 8-bit quantization for LLaMA models.

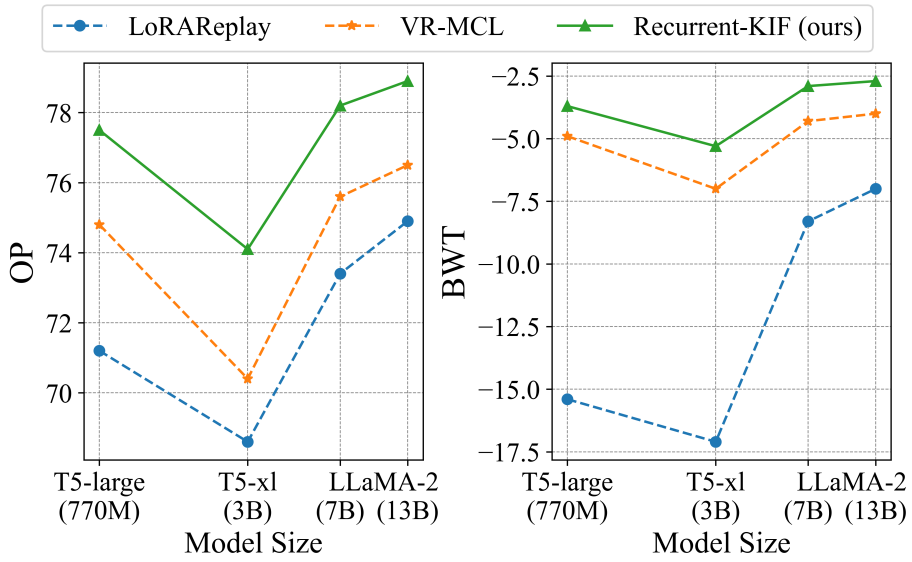
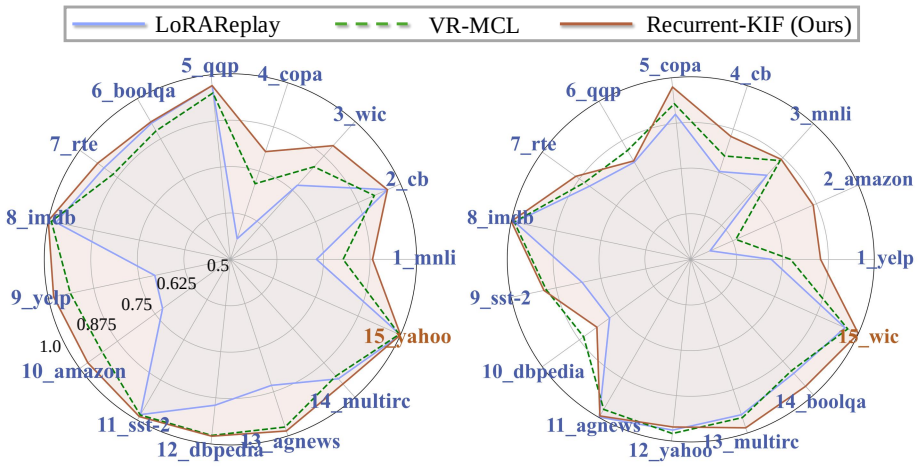


Figure 3.10: Performance of Recurrent-KIF with different backbones on the Long Sequence Benchmark.

Recurrent-KIF outperforms the state-of-the-art CL method, MIGU, increasing OP from 76.6% to 78.1%. These results underscore the effectiveness of our recurrent knowledge identification and fusion framework, validating the advantages of dynamic parameter importance estimation in mitigating forgetting and promoting knowledge transfer.

Recurrent-KIF demonstrates consistent superiority across various backbones. To further validate the robustness of Recurrent-KIF, we conduct experiments across different backbones (Figure 3.10). Across all backbone sizes, from 770M to 13B, Recurrent-KIF consistently outperforms all baseline models. For instance, using the LLaMA2-7B backbone, Recurrent-KIF boosts the OP metric from 75.6% to 78.2% compared to VR-MCL. These results emphasize the critical role of accurate parameter importance estimation and demonstrate the robust generalization capability of Recurrent-KIF across different model scales.



(a) T5-large on Task Order 1 (b) LLaMA2-7B on Task Order 3

Figure 3.11: **Impact of Catastrophic Forgetting in Continual Learning.** After fine-tuning on the final task (in orange), Recurrent-KIF demonstrates superior resistance to performance decline on previously learned tasks (in blue), outperforming baseline methods.

Dynamic importance estimation enables effective knowledge retention and transfer. Figure 3.11 illustrates the performance of various methods across all historical tasks after completing the final task. It highlights that Recurrent-KIF optimally restores the model's performance on previous tasks, with significant improvements on Amazon and Copa. Notably, on IMDB and AG News, Recurrent-KIF performs comparably to multi-task learning results. These demonstrate that Recurrent-KIF strikes an effective balance between preserving prior knowledge and excelling in new tasks.

Ablation Study

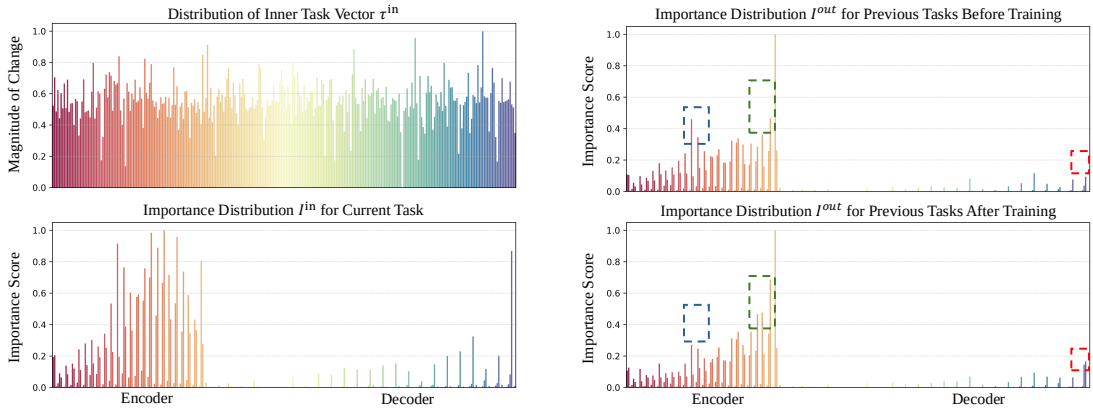
We conduct ablation studies to assess the effectiveness of the proposed techniques in Recurrent-KIF. The results for task order 1 on the Long Sequence Benchmark are shown in Table 3.8.

Method	OP	BWT
Recurrent-KIF	**77.9**	**-3.4**
- DIE	74.8	-4.8
- KI	52.3	-21.5
+ GM	72.1	-11.2
+ Adaptive	76.1	-4.1
- Share	75.8	-4.3

Table 3.8: Ablation study. "- DIE", "- KI", "- Share" refer to the removal of dynamic importance estimation, knowledge identification, and task-shared region updates, respectively. "+ GM", "+ Adaptive" represent replacing the knowledge fusion mechanism with global merging and adaptive merging strategy, respectively.

Effect of Dynamic Importance Estimations. To validate the role of dynamic importance estimation, we replace it with a static version ("- DIE"), where importance scores for historical tasks remain fixed after their initial computation. The significant performance decline (3.1% on OP and 1.4% on BWT) highlights the necessity of dynamically updating historical task importance distributions. By maintaining up-to-date importance scores, our approach improves both robustness and accuracy, thereby enhancing knowledge retention and transfer.

Effect of Importance-Based Binary Mask Strategy in Knowledge Fusion. We replace our knowledge fusion mechanism with three alternative model merging strategies: (i) Without knowledge identification ("- KI"): Directly merge τ^{in} and τ^{out} in Eq.(3.26) without applying importance-based fusion. (ii) Global importance-based merging ("+ GM"): Use a global weighted sum of importance scores I_b^{in} and I_b^{out} for fusion instead of applying element-wise masking. (iii) Adaptive Fusion [159] ("+ Adaptive"): A soft-masking method that uses raw importance scores directly for fusion instead of binary masks. Additionally, we evaluate the effectiveness of updating



(a) Normalized inner task vector and corresponding parameter importance distribution for the current task.

(b) Normalized parameter importance scores for a historical task across model states.

Figure 3.12: Visualizations of task vector and parameter importance distributions on T5-large.

the task-shared region by introducing "- Share", where m^{in} is set to 0 when both m^{in} and m^{out} are 1.

The results in Table 3.8 confirm the effectiveness of importance-based binary masking in filtering redundant information and preserving task-specific knowledge. Moreover, disabling updates to the task-shared region leads to performance drops of 2.1% and 0.9% on two evaluation metrics, demonstrating that updating task-shared parameters is critical for effective knowledge transfer between tasks.

Effect of Multi-Round Knowledge Fusion. We compare multi-round fusion with traditional single-step fusion methods and analyze the impact of the number of knowledge fusions by adjusting the inner loop iteration size Q. In Recurrent-KIF, with the total number of iterations for the inner loop fixed at N', increasing Q reduces the number of fusion steps, which is N'/Q.

Figure 3.13 presents two key findings: (i) Multi-round fusion consistently outper-

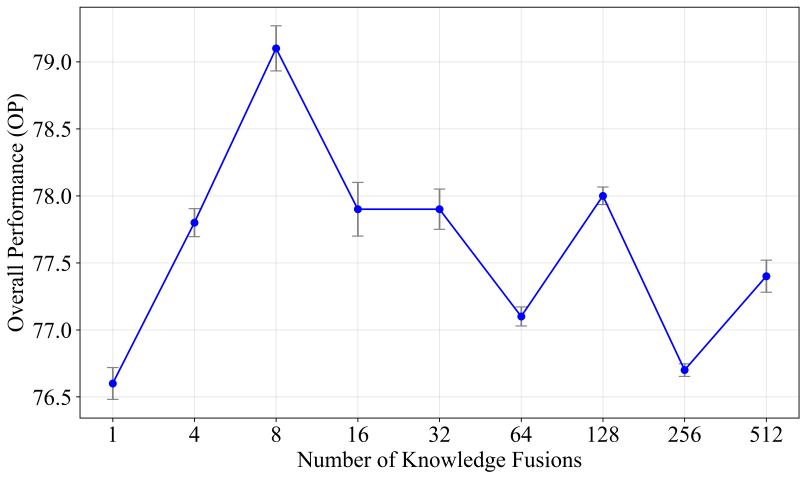


Figure 3.13: Ablation study on the number of fusions.

forms single-step fusion by leveraging intermediate training information, resulting in smoother knowledge integration, similar to the distinction between gradient descent and stochastic gradient descent; and (ii) performance improves with the number of fusion steps up to a certain point, after which diminishing returns are observed. This decline occurs because excessive fusion introduces noise and redundant updates, disrupting the balance between new and prior knowledge, and potentially causing overfitting to unconverged intermediate states.

Visualization

We present two key visualizations to analyze the effectiveness of our proposed methods:

Can the Magnitude of the Task Vector Reflect Parameter Importance? We explore the relationship between task vector magnitude and parameter importance scores. As shown in Figure 3.12(a), although the magnitude of parameter updates is generally large, only a subset of parameters, mainly in the encoder, are truly important. This indicates that a large portion of the parameters are redundant,

highlighting the need for our importance-based knowledge fusion mechanism.

Does the Importance Distribution of Historical Tasks Change with Model State? Figure 3.12(b) illustrates the shift in the importance distribution of historical tasks before and after training on a new task. While the overall distribution remains stable across model states, notable changes in specific importance scores are observed, highlighted by the dashed box. This demonstrates the value of dynamic estimation, enabling more precise identification of key parameters and enhancing knowledge fusion across tasks.

3.4.4 Conclusion

In this section, we introduce Recurrent Knowledge Identification and Fusion (Recurrent-KIF), a novel CL framework that dynamically estimates the importance of parameters for previous tasks. Recurrent-KIF iteratively employs an inner learner to localize new knowledge and an outer learner to manage the global fusion of knowledge, enabling real-time and adaptive adjustments to the fusion strategy based on evolving importance distributions. Extensive experiments demonstrate the effectiveness of Recurrent-KIF in addressing continual learning challenges.

3.5 Methodology – Adaptive Iterative Model Merging (AIMMerging)

3.5.1 Introduction

Recent model merging methods [17, 123, 157] have gained prominence for CL, largely due to their capacity for KT. Traditional approaches typically involve a single-round merge, commonly applied between pre- and post-training models, using global or

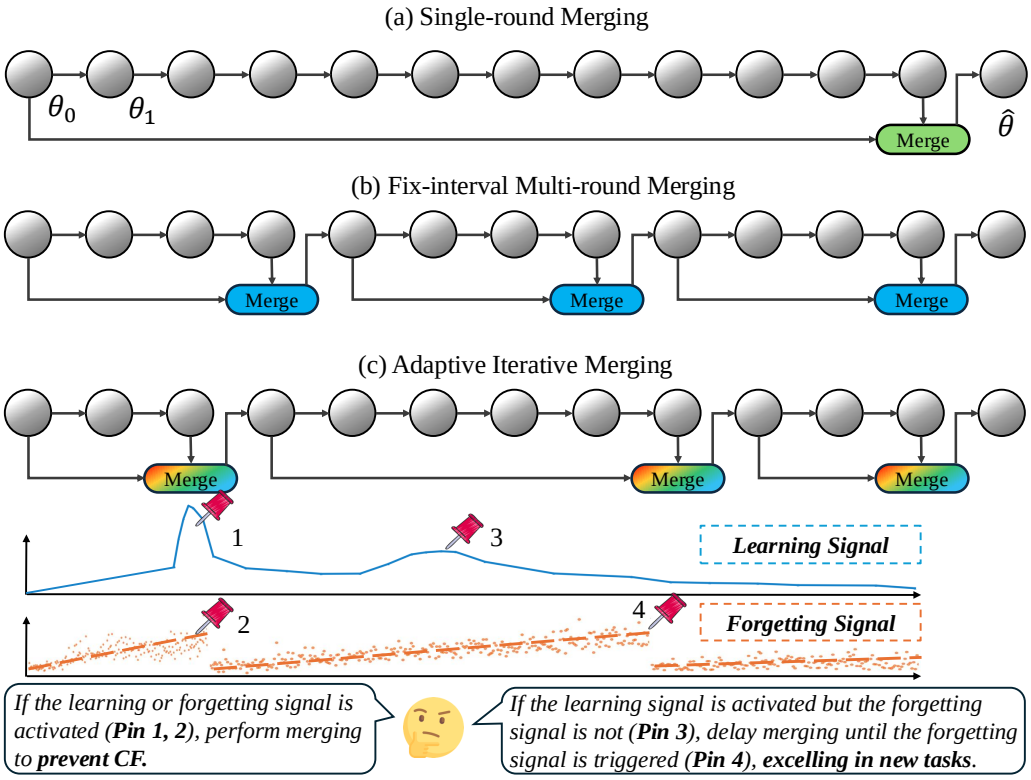


Figure 3.14: Illustration of three different model merging strategies. Guided by the learning and forgetting signals extracted from the training trajectory, our AIMMerging adaptively adjusts the merging intervals and frequency, thereby enhancing CL performance.

fine-grained strategies (Figure 3.14(a)). Departing from single-round methods, Feng et al. [29] proposed a recurrent framework that merges models iteratively after fixed training steps (Figure 3.14(b)), showing that leveraging intermediate training states through multiple merges can enhance performance.

This multi-round merging paradigm reveals significant potential and highlights the importance of optimizing the merging process. Inspired by these promising results, a critical question emerges:

How can we determine the optimal timing and frequency of merging during training to further enhance performance?

To this end, we propose a novel CL framework called **A**daptive **I**terative **M**odel **Merging** (**AIMMerging**). It achieves dynamic monitoring of the training status by innovatively employing ***learning*** and ***forgetting signals*** extracted from the training trajectory.

Based on these signals, AIMMerging consists of two key modules: the **Training Trajectory-guided Merge Controller**, responsible for adaptively scheduling the timing and frequency of model merging, and the **Rehearsal-based Knowledge Fusion Module**, which performs the global merging operation.

More specifically, the ***learning signal*** is quantified by the change in model parameters across training steps, reflecting the model's acquisition progress for new knowledge. Analysis of its trend via a sliding window identifies periods of rapid learning or slow convergence. The ***forgetting signal***, on the other hand, is derived from the loss on historical data, offering real-time insight into the extent of CF. It is triggered when the historical loss exceeds a predefined threshold or shows a notable rise, signifying potential knowledge loss.

These signals guide the merge controller with distinct functions. The learning signal, typically measured after a merge, helps determine the next merging interval, thereby influencing the overall merging frequency. In contrast, the forgetting signal is continuously monitored during training. It serves as a critical, real-time trigger, prompting an immediate merge when significant forgetting of historical knowledge occurs.

Leveraging the dynamic monitoring from the learning and forgetting signals, the **training trajectory-guided merge controller** adaptively determines the merging schedule by interpreting their interplay. Based on the findings from our preliminary study (see Section 3.5.2), the controller increases the merging frequency to proactively mitigate CF when the learning signal indicates a rapid learning phase or the forgetting signal is activated. Conversely, when the learning signal indicates a slow convergence phase or the forgetting signal remains inactive, the controller reduces the merging fre-

quency, allowing the model to focus more on learning new knowledge. Through this interaction, our method strikes a better balance between retaining previous knowledge and excelling in new tasks. The **rehearsal-based knowledge fusion module** executes the global merge operation, utilizing the relative importance weights derived from the learning and forgetting signals to merge new and historical knowledge effectively. Extensive experiments demonstrate the strong performance of our method in addressing CL challenges.

Our main contributions are summarized as:

- We propose a novel adaptive iterative model merging **framework** (AIMMerging) for CL. To the best of our knowledge, AIMMerging is the first to leverage the training trajectory by extracting learning and forgetting signals to dynamically monitor the model's state and guiding adaptive scheduling of iterative model merging.
- We introduce two novel **techniques**: the training trajectory-guided merge controller and the rehearsal-based knowledge fusion module.
- Extensive **evaluation** on three CL benchmarks utilizing four backbones (from 770M to 13B) demonstrates that AIMMerging significantly enhances knowledge transfer capabilities, achieving an average relative improvement of 80% (from -2.5% to -0.5%) in FWT and 59% (from -4.9% to -2.0%) in BWT , surpassing previous state-of-the-art methods.

3.5.2 Preliminary Study

In this section, we conduct two key analysis: (i) investigating the dynamic changes in the model's training states regarding new knowledge acquisition and historical knowledge forgetting, and (ii) examining the impact of model merging during training on both new and historical knowledge. These analyses provide valuable insights for optimizing the adaptive merging strategy.

Notation We consider a pre-trained model $\theta \in \mathbb{R}^n$ with n parameters. After training on task $\mathcal{T}_{k-1}$, the model are denoted as θ^{k-1}. Fine-tuning on a new task $\mathcal{T}_k$ produces updated parameters θ^k. The difference $\tau^k = \theta^k - \theta^{k-1}$, referred to as the *task vector* or *training residual* [52], represents task-specific parameter updates.

For traditional single-round merging methods, a merging function f_{merge} is typically used to combine the model θ^{k-1} and the fine-tuned model θ^k to obtain the final model: $\hat{\theta}^k = f_{\text{merge}}(\theta^{k-1}, \theta^k)$. In contrast, multi-round model merging methods perform merges during the training process. Specifically, assuming the total number of training iterations is J, θ_j^{k-1} represents the model's parameters at the j-th iteration. For example, if the interval between two consecutive merges is S (e.g., 100 training iterations), the merged model is represented as: $\hat{\theta}_{j+S}^{k-1} = f'_{\text{merge}}(\theta_j^{k-1}, \theta_{j+S}^{k-1})$. The model is then further trained based on $\hat{\theta}_{j+S}^{k-1}$.

Analysis of Knowledge Acquisition and Forgetting

New Knowledge Acquisition We measure the model's learning state for new knowledge by summing the absolute values of parameter changes within a fixed training interval, such as every 10 steps (Figure 3.15). In the early stages of training, the parameter changes are large and show an upward trend, indicating the rapid learning phase. As training progresses, these changes decrease, signaling a slow convergence phase. Interestingly, peaks may reappear, suggesting the model revisits unlearned or challenging knowledge.

Based on two learning scenarios, we conducted two comparative experiments: (1) increasing merging frequency during the rapid learning phase, and (2) increasing merging frequency during the slow convergence phase. As shown in Table 3.9, the results reveal that increasing merging frequency during the rapid learning phase improves performance by preventing excessive accumulation of new knowledge. However, merging during the convergence phase leads to a decline in performance, likely

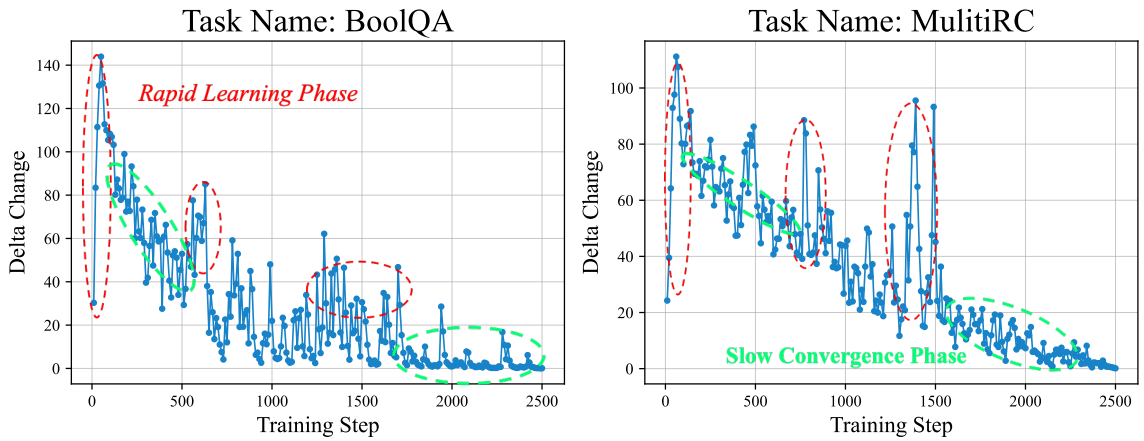


Figure 3.15: Parameter change for new knowledge acquisition during training.

Merging Strategy	OP	FWT	BWT
Fix Interval	78.3	-3.4	-2.7
Slow Convergence Phase$^+$	77.9	-3.8	-3.0
Rapid Learning Phase$^+$	**78.5**	**-2.7**	**-1.9**

Table 3.9: The impact of different merging strategies on performance. "$^+$" indicates increased merging frequency during the corresponding phases.

due to redundancy in the stable model. This insight is valuable for refining the learning signal strategy in our method.

Forgetting of Historical Knowledge During training, we sample a batch of memory data from the buffer, feeding it into the model for loss calculation without gradient updates. This allows us to monitor historical knowledge loss, as shown in Fig. 3.16. The loss for historical knowledge increases as new knowledge is learned, aligning with expectations. When the loss exceeds a predefined threshold, the forgetting signal is triggered, prompting merging to mitigate forgetting.

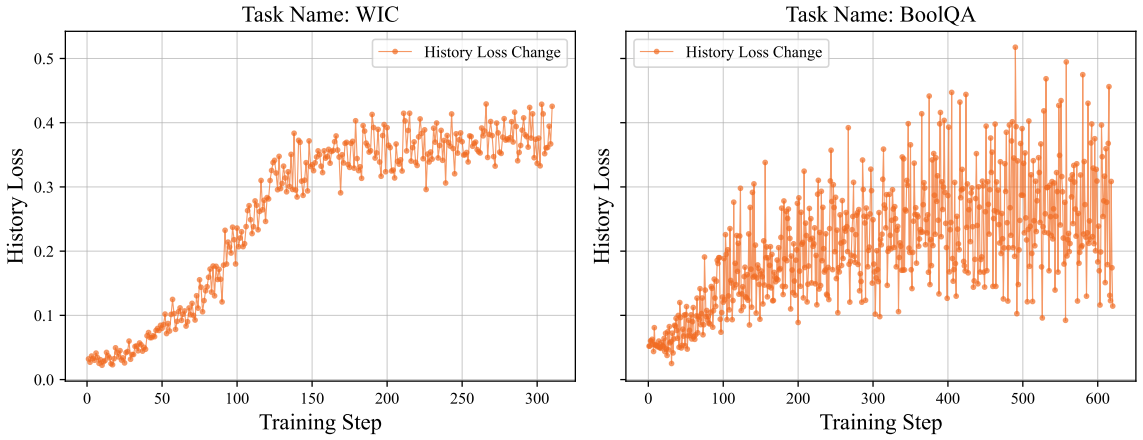


Figure 3.16: Changes in historical loss during training.

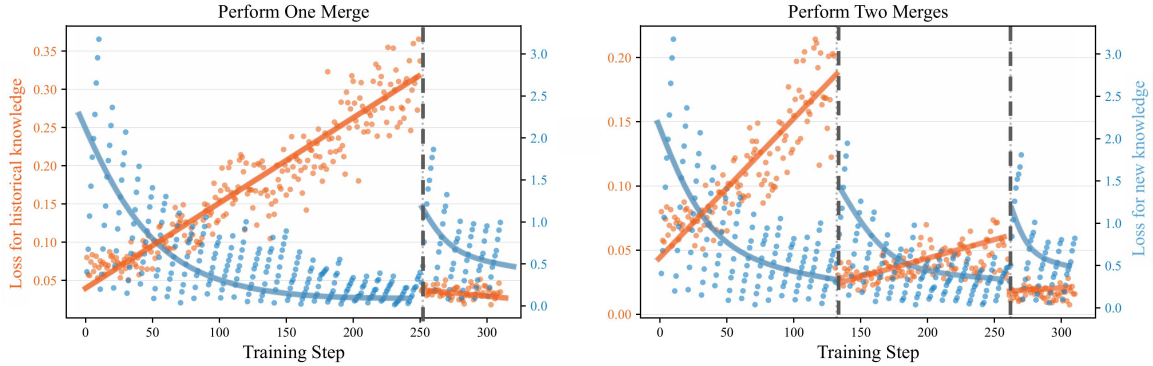


Figure 3.17: Loss change for new and historical knowledge after model merges during training.

Impact of Model Merging on New and Historical Knowledge

We perform one or two merges during training to observe the changes in loss for both new and historical knowledge (Figure 3.17). The results show that after each merge, historical knowledge loss decreases significantly, demonstrating effective mitigation of CF. However, the loss for new tasks increases, indicating that merging may interfere with learning new knowledge. Thus, selecting the appropriate merging timing is key to balancing new knowledge acquisition and historical knowledge retention.

3.5.3 Proposed Method: AIMMerging

Overview AIMMerging employs two signals, the ***learning signal*** and the ***forgetting signal***, to monitor the model's training state. Based on the real-time fluctuations of these two signals, our approach reconfigures the training process into multiple iterative model merging cycles, guided by two core components: (i) ***Training Trajectory-guided Merge Controller***: adaptively selects the timing of model merges and dynamically adjusts the intervals between subsequent merges. (ii) ***Rehearsal-based Knowledge Fusion***: responsible for generating merging weights and applying memory-replay techniques to integrate new and historical knowledge.

The Design of Merge Controller

Assume the current task is $\mathcal{T}_k$, and θ_j^{k-1} represents the model's state after j training iterations [3].

Merge Controller with Learning Signal The core function of the learning signal is to dynamically adjust the merging interval S based on the model's current state for new knowledge. Assume the b-th merge is scheduled at the j-th training iteration, and the interval since the $(b-1)$-th merge is S_b. The task vector capturing the parameter update between these two successive merges is defined as:

$$\tau_b = \theta_j - \theta_{j-S_b} \tag{3.29}$$

By summing the absolute values of the task vector elements, we obtain a measure of the model's learning state for the new task, expressed as:

$$\Lambda_b = \sum_{i=1}^{n} |\tau_b^i| \,/\, S_b \tag{3.30}$$

where τ_b^i is the i-th element of the task vector. Dividing by S_b normalizes the value, allowing for fair comparison across intervals of varying lengths.

[3]For simplicity, we omit the superscript $k-1$ in the following descriptions.

We use Λ to assess the model's learning state for new knowledge. By comparing the current value Λ_b with the previous one Λ_{b-1}, we observe the parameter change trend and adjust the merging interval from S_b to S_{b+1}.

However, considering only the trend between two consecutive values may cause the learning signal to be overly sensitive to short-term fluctuations. To address this, we adopt a sliding window approach to analyze parameter change trends across multiple historical points and capture a more reliable overall trajectory. Specifically, we maintain a list to record the historical values of Λ, denoted as $\mathcal{H} = [\Lambda_1, \Lambda_2, \ldots, \Lambda_{b-1}]$. Given a sliding window length L_w, we compare the trends between consecutive entries, i.e., between Λ_b and Λ_{b-1}, Λ_{b-1} and Λ_{b-2}, ..., up to Λ_{b-L_w+1} and Λ_{b-L_w}.

If upward trends dominate, indicating rapid learning phrase for new knowledge, we reduce the merging interval based on the magnitude of parameter changes (Case 1). If downward trends dominate, indicating slow convergence phase, we increase the merging interval (Case 3). If they are balanced, we keep the current interval (Case 2). The adjustment strategy is defined as:

$$S_{b+1} = \begin{cases} \max(S_{\min}, S_b/\gamma^-_{\text{learn}}), & \text{(Case 1)} \\ S_b, & \text{(Case 2)} \\ \min(S_{\max}, S_b \cdot \gamma^+_{\text{learn}}), & \text{(Case 3)} \end{cases} \tag{3.31}$$

where $S_{\min}$ and $S_{\max}$ denote the minimum and maximum allowed merging intervals, and γ_{learn} is a step-size adjustment factor. A cold-start phase is introduced at the beginning of training, during which no adjustments are made, lasting the length of the sliding window with an initial interval S_{init}.

Merge Controller with Both Learning and Forgetting Signals Relying only on the learning signal, the model adjusts the merging interval S based on the learning state for new knowledge, but this neglects the forgetting of historical knowledge, leading to a suboptimal merging strategy.

To address this, we further integrate the forgetting signal $\mathcal{F}$ to assist the controller adjust the merging strategy by considering both new and historical knowledge. The strategy triggers earlier or delayed merges depending on the forgetting signal, optimizing the merging interval.

We define the forgetting signal based on the loss change of historical data during the training of new tasks. In each iteration, a batch of historical data is sampled from the memory buffer and combined with the current task's batch. The historical data is used only for loss computation, excluding it from gradient updates. The forgetting signal is activated if the loss on historical tasks exceeds a predefined threshold, which is calculated using the average loss over the first $2/3 \times S_{b+1}$ steps, scaled by an adjustment factor γ_{forget} to produce the threshold δ_{b+1}. If the historical loss exceeds this threshold during subsequent training, the forgetting signal is triggered and the activation count is incremented as: $\mathcal{F}(b+1) = \mathcal{F}(b+1) + 1$.

Overall Workflow of the Merge Controller If the forgetting signal is activated multiple times (e.g., $\mathcal{F}(b+1) \geq \mathcal{F}_{max}$) before the scheduled merging interval S_{b+1}, an early merge is triggered to prevent further forgetting, i.e., the actual merging interval $S'_{b+1} < S_{b+1}$. Conversely, if the model reaches the predefined merging interval S_{b+1} without the forgetting signal being activated, the merge can be deferred to allow the model to continue focusing on learning new knowledge. The merge will be triggered either when the forgetting signal is activated ($S'_{b+1} > S_{b+1}$) or when the iteration count reaches the upper limit ($S'_{b+1} = 2 * S_{b+1}$).

In summary, our controller dynamically balances both learning and forgetting signals to optimize new knowledge acquisition while minimizing forgetting, resulting in an adaptive merging strategy.

Rehearsal-based Knowledge Fusion

When the merge controller initiates a merge, the knowledge fusion module performs the actual task knowledge fusion. Assume the b-th merge occurs at the j-th training iteration. The parameter change representing new knowledge is defined as:

$$\tau_{\text{new}_b} = \theta_j - \theta_{j-S'_b} \tag{3.32}$$

Next, we fine-tune θ_j on memory data for $S'_b/2$ steps, resulting in an updated model state $\theta_{j(M)}$. The task vector for historical knowledge is then:

$$\tau_{\text{past}_b} = \theta_{j(M)} - \theta_j \tag{3.33}$$

The final model parameters are updated by fusing both task vectors with learnable weights:

$$\hat{\theta}_j = \theta_{j-S'_b} + \alpha_1 \cdot \tau_{\text{new}_b} + \alpha_2 \cdot \tau_{\text{past}_b} \tag{3.34}$$

where α_1 and α_2 are the fusion weights, computed as follows:

- For τ_{new}, we assess the proportion of the upward trend in the learning signal's sliding window, $\mathcal{P}_{new} = L_{up}/L_w$, indicating the model's active learning of new knowledge.
- For τ_{past}, we compute the ratio of the forgetting signal's activation count $\mathcal{F}(b)$ to the maximum threshold $\mathcal{F}_{\max}$, $\mathcal{P}_{past} = \mathcal{F}(b)/\mathcal{F}_{\max}$, suggesting the extent of historical knowledge forgetting.

The fusion weights are then normalized as:

$$\alpha_1 = \frac{\mathcal{P}_{new}}{\mathcal{P}_{new} + \mathcal{P}_{past}}, \quad \alpha_2 = \frac{\mathcal{P}_{past}}{\mathcal{P}_{new} + \mathcal{P}_{past}} \tag{3.35}$$

After the fusion is completed, training continues from the updated model state $\hat{\theta}_j$.

	Standard CL			Long Sequence			SuperNI		
	OP↑	FWT↑	BWT↑	OP↑	FWT↑	BWT↑	OP↑	FWT↑	BWT↑
SeqLoRA	43.7	-9.1	-50.4	11.6	-10.8	-73.4	6.4	-13.6	-31.0
IncLoRA	66.4	-8.7	-20.0	61.2	-11.1	-26.7	8.2	-15.1	-27.4
LoRAReplay	68.8	-9.0	-11.7	70.9	-11.3	-15.4	35.4	-12.4	-15.8
EWC* [61]	50.3	-	-	45.1	-	-	35.7	-	-
L2P* [145]	60.7	-	-	56.1	1.36	-16.3	12.7	-19.1	-8.0
LFPT5* [103]	72.7	-	-	69.2	-2.5	-12.8	34.4	-0.5	-14.5
MoELoRA* [86]	54.1	-6.2	-7.7	27.6	-8.6	-13.2	21.8	-7.2	-19.0
O-LoRA* [138]	75.8	-5.9	-3.8	69.6	-8.2	-4.1	25.9	-0.1	-24.6
TaSL [28]	76.3	-5.4	-4.0	74.4	-7.9	-5.3	38.9	-1.2	-10.8
MIGU* [18]	76.6	-	-	76.5	-	-	-	-	-
VR-MCL [149]	76.0	-4.6	-3.7	74.8	-6.0	-4.9	41.1	0.2	-9.3
Recurrent-KIF* [29]	**78.4**	-3.1	-2.8	77.8	-4.6	-3.6	43.3	0.4	-8.4
AIMMerging (ours)	78.1	**-1.5**	**-0.4**	**77.9**	**-2.3**	**-1.8**	**44.3**	**2.2**	**-4.0**
MTL	80.3	-	-	81.8	-	-	50.7	-	-
SAPT-LoRA [179]	-	-	-	82.0	1.9	-1.3	51.5	1.9	-0.57

Table 3.10: Overall results on three CL benchmarks using the T5-large model. We report Overall Performance (OP), Forward Transfer (FWT), and Backward Transfer (BWT) after training on the final task. All results are averaged over different task orders. Methods marked with $*$ are copied from previous papers. The last two rows represent upper bound performance.

3.5.4 Experiments

Dataset We adopt the experimental setup from Du et al. [18], using three CL benchmark datasets: (i) **Standard CL Benchmark**, which consists of five text classification tasks from Zhang et al. [176]. (ii) **Long Sequence Benchmark**, a more challenging evaluation scenario comprising 15 tasks [109]. (iii) **SuperNI Benchmark** [143], a comprehensive benchmark for text generation, designed to evaluate 15 NLP tasks. Following Wang et al. [138], we sample 1000 instances for training on each task and reserve 500 per class for testing.

Metrics Let $a_{i,j}$ denote the testing performance on task $\mathcal{T}_i$ after training on task $\mathcal{T}_j$, and $a_{0,t}$ refers to the performance of training task t individually. We evaluate the overall performance (OP) [9], backward transfer (BWT) [58], and forward transfer (FWT) [83] after training on the final task.

Baselines We compare AIMMerging against various advanced methods, as well as both single-round and multi-round model merging methods. All methods are implemented using the LoRA framework to ensure fairness: (1) ***SeqLoRA:*** LoRA parameters are trained on a task sequence without regularization or sample replay. (2) ***IncLoRA:*** incremental learning of LoRA parameters without regularization or sample replay. (3) ***LoRAReplay:*** LoRA fine-tuning with a memory buffer. (4) ***EWC [61]:*** finetune LoRA with a regularization loss to prevent interference with previous tasks. (5) ***L2P [145]:*** dynamically selects and updates prompts from a pool on an instance-by-instance basis. (6) ***LFPT5 [103]:*** learns a soft prompt that solves tasks and generates training samples for replay. (7) ***O-LoRA [138]:*** extends IncLoRA to learn different LoRAs in orthogonal subspaces. (8) ***MoELoRA [86]:*** a vanilla MoE with LoRA number equals to the task number. (9) ***SAPT [179]:*** uses pseudo samples and a shared attention framework to align PEFT block learning and selection. (10) ***MIGU [18]:*** updates important parameters based on gradient magnitude. (11) ***TaSL [28]:*** a single-round model merging method based on parameter importance. (12) ***VR-MCL [149]:*** dynamically updates the distribution of parameter importance through memory replay. (13) ***Recurrent-KIF [29]:*** a multi-round model merging method based on fixed merging intervals. Additionally, multi-task learning, referred to as ***MTL***, serves as the upper bound.

Training Details We evaluate AIMMerging using different backbone models, including T5-large [106], Qwen3 1.7B [158], LLaMA2-7B [122], and LLaMA2-13B. For the learning signal, the initial merging interval S_{init} is set to 8, and the sliding win-

dow size L_w is set to 3. The minimum and maximum merging intervals, S_{min} and S_{max}, are set to 2 and 128, respectively. The adjustment factors γ^{+}_{learn} and γ^{-}_{learn} are selected based on the current merging interval: if $S > 64$, we set $\gamma^{+}_{\text{learn}} = 1.5$ and $\gamma^{-}_{\text{learn}} = 2$; otherwise, we set $\gamma^{+}_{\text{learn}} = 2$ and $\gamma^{-}_{\text{learn}} = 1.5$. For the forgetting signal, the threshold scaling factor γ_{forget} is set to 2, and the maximum number of allowed activations before forcing an early merge, $\mathcal{F}_{\max}$, is set to 3. Following Feng et al. [29], 2% of the original training set is used for replay samples. All experiments are averaged over 3 runs.

Main Results

The overall CL results using the same T5-large backbone are summarized in Table 3.10.

Our Training Trajectory-based AIMMerging Method Effectively Addresses Both CF and KT Challenges. Compared to traditional CL methods (LoRAReplay, EWC) and model merging approaches (O-LoRA, MoELoRA, TaSL), AIMMerging outperforms them in both CF (increasing OP from 59.5% to 66.8% compared to O-LoRA) and KT (improving BWT from -6.0% to -2.1% compared to VR-MCL). SAPT achieves the highest performance with generative replay-based data augmentation, surpassing MTL results. Notably, AIMMerging achieves a BWT of -1.8% on the long sequence benchmark, close to SAPT's -1.3%, and outperforms SAPT in FWT on SuperNI (improving from 1.9% to 2.2%).

Moreover, AIMMerging outperforms the state-of-the-art Recurrent-KIF, also based on multi-round merging, with significant improvements in both FWT (2.0%, from -2.5% to -0.5%) and BWT (2.9%, from -4.9% to -2.0%). These results show that AIMMerging effectively balances preserving prior knowledge and excelling in new tasks.

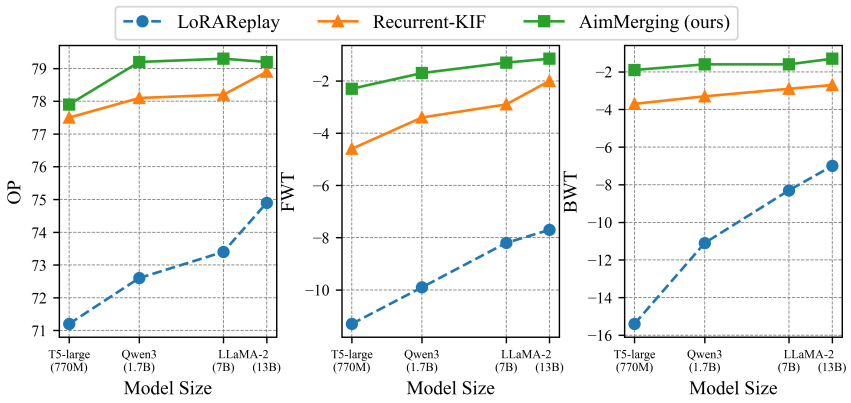


Figure 3.18: Performance of AIMMerging with different backbones on the Long Sequence Benchmark.

AIMMerging Demonstrates Consistent Superiority and Generalization Across Various Backbones. We validated the robustness of AIMMerging using backbones ranging from 770M to 13B parameters, as shown in Figure 3.18. Across all sizes, AIMMerging consistently outperforms baseline models. Notably, with the LLaMA2-7B backbone, AIMMerging improves FWT from 78.2% to 79.3% and BWT from -2.9% to -1.6% compared to Recurrent-KIF, demonstrating its strong generalization ability across different model scales.

The Adaptive Iterative Merging Framework Enables Effective Knowledge Retention. Figure 3.19 illustrates the performance of the initial task after training on subsequent tasks. AIMMerging significantly reduces catastrophic forgetting, with only a 4% performance drop after training on the final task. In contrast, vanilla replay shows a 32% drop, and Recurrent-KIF shows a 10% decline. These results underscore our model's strong backward knowledge transfer capability.

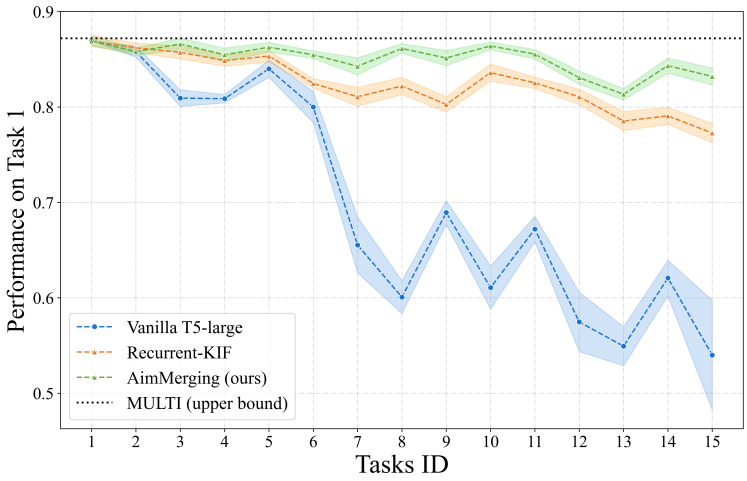


Figure 3.19: Performance trajectory of Task 1 on the longsequence benchmark during the CL process.

Ablation Study

We perform ablation studies to evaluate the effectiveness of the two key techniques in AIMMerging. Results for task order 1 on the SuperNI Benchmark are shown in Table 3.11.

Effect of Training Trajectory-guided Merge Controller. To validate the contribution of the learning signal and forgetting signal to the merge controller's decision-making, we remove the learning signal ("- LS") and forgetting signal ("- FS") individually. When only the learning signal is used, merging occurs whenever the model's iteration reaches the pre-defined interval S. When only the forgetting signal is used, merging occurs when the loss of historical tasks exceeds the threshold. The performance decline in Table 3.11 highlights the necessity of both signals. Using only one signal leads to focusing on either new knowledge learning or historical knowledge retention, while both signals allow better balance.

Method	OP	FWT	BWT
AIMMerging	**45.1**	**1.3**	-2.2
- LS	43.9	0.5	-3.4
- FS	44.3	0.8	-3.9
+ MGM	44.2	0.7	-3.7
+ IFM	44.9	1.2	**-2.1**

Table 3.11: Ablation study. "- LS", "- FS" refer to the removal of the learning signal and forgetting signal in our merge controller, respectively. "+ MGM" and "+ IFM" represent replacing the merging weights with manually set global merging weights and parameter importance-based fine-grained merging weights, respectively.

Effect of Rehearsal-based Knowledge Fusion Module. We replace the weight calculation method in our fusion mechanism with two alternatives: (i) Manually set global merging weights (via grid search). (ii) Parameter importance-based fine-grained merging weights (following Feng et al. [29]). Our results show that weights based on the learning and forgetting signals outperform manually set weights, improving three evaluation metrics by 0.9%, 0.6%, and 1.5%. Compared to parameter importance-based weights, our method shows slightly better performance, demonstrating that the learning and forgetting signals effectively capture the relevance of task vectors for knowledge updating and retention. Moreover, our approach is more efficient, avoiding the computational overhead of calculating and storing parameter importance.

Visualization

We visualize two key aspects of our method's effectiveness.

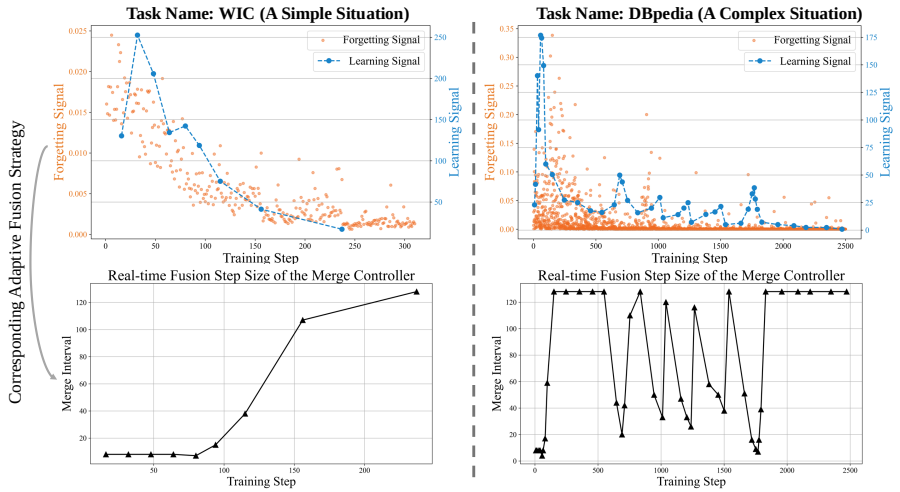


Figure 3.20: Visualizing merge controller behavior based on dynamic changes in learning and forgetting signals.

Visualizing How the Merge Controller Adjusts Merging Timing and Step Size Based on the Learning and Forgetting Signals. As shown on the left of Figure 3.20, for simpler training scenarios, the merge controller gradually increases the merging interval. In the early stages of training, when there is significant new knowledge update, increasing the merging frequency helps prevent forgetting of historical knowledge. While in later stages, reducing it avoids redundant merges. In contrast, for more complex scenarios shown on the right, our method dynamically adjusts the merging frequency based on changes in the training state. For example, the model enters multiple rapid learning phases again during the middle of training (indicated by the peaks in the figure), prompting an increase in merging frequency.

Visualizing Adaptive Iterative Merging's Effect on Catastrophic Forgetting. Figure 3.21 demonstrates the impact of our multi-round merging approach on historical knowledge forgetting. With vanilla LoRAReplay, the loss for historical tasks increases progressively, reflecting an escalating degree of forgetting. In contrast, our method effectively mitigates forgetting by selecting optimal merging points, enabling timely suppression before significant forgetting occurs. This results in a more

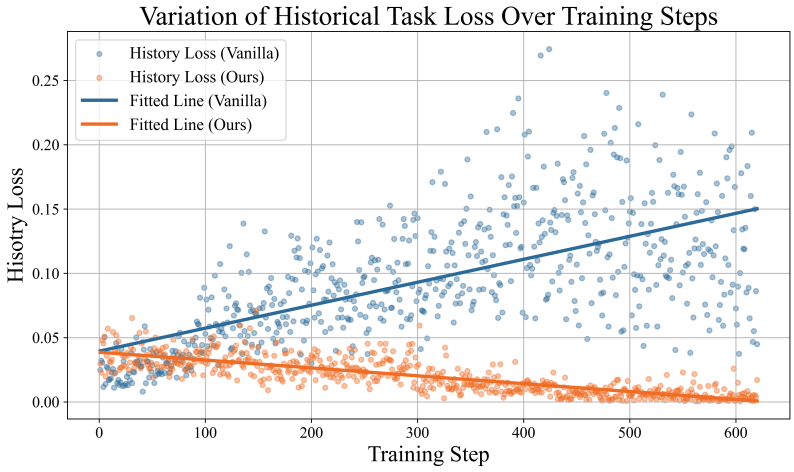


Figure 3.21: Visualization of the effect of AIMMerging in alleviating catastrophic forgetting.

stable or even decreasing overall loss trend, demonstrating the effectiveness of our approach in alleviating catastrophic forgetting.

3.5.5 Conclusion

In this section, we introduce Adaptive Iterative Model Merging (AIMMerging), a novel CL framework that enables dynamic monitoring of the training status by leveraging learning and forgetting signals extracted from the training trajectory. The framework consists of two key modules: the training trajectory-guided merge controller, which adaptively schedules the timing and frequency of model merging, and the rehearsal-based knowledge fusion module, which performs the global merging operation based on these signals. Extensive experiments validate the effectiveness of AIMMerging in addressing the key challenges of continual learning.

3.6 Chapter Review

In this chapter, we tackled the challenge of *catastrophic forgetting* in continual learning for large language models (LLMs). We proposed three progressive methods: Knowledge Identification and Fusion (KIF), Recurrent KIF, and Adaptive Iterative Model Merging (AIMMerging). KIF identifies task-specific and task-shared knowledge by estimating parameter importance and consolidating them through model merging. Recurrent KIF extends this approach by dynamically updating importance scores over time, enabling more reliable knowledge integration in long task sequences. Finally, AIMMerging introduces adaptivity into the merging process by determining the optimal timing based on learning and forgetting signals. Extensive experiments demonstrate that our methods outperform existing baselines such as EWC, MAS, and TIES-Merging in balancing stability and plasticity.

Limitations in large-scale continual learning. Although our methods scale better than traditional parameter regularization approaches, challenges remain in scenarios with extremely long task sequences or highly heterogeneous domains. The cumulative drift of representations can still degrade performance, even when dynamic importance estimation is applied. Future work may explore hybrid strategies that combine importance estimation with modular architectures or memory-replay to further enhance robustness.

Adaptive merging requires reliable signals. AIMMerging relies on the extraction of learning and forgetting signals from the training trajectory. While effective in our experiments, the quality of these signals may vary depending on task complexity and data noise. This can lead to suboptimal or delayed merging decisions. A promising direction is to incorporate meta-learning techniques to calibrate these signals or to explore uncertainty-aware merging schedules.

Trade-off between efficiency and accuracy. Our methods prioritize scalability to LLMs by avoiding heavy replay or task-specific modules. However, model merging and dynamic importance tracking still introduce additional computational overhead. While this overhead is modest compared to full retraining, further optimization—such as sparse merging strategies or low-rank importance estimation—could make these methods more efficient for deployment in resource-constrained environments.

Overall, the methods presented in this chapter provide scalable and adaptive solutions to continual learning in LLMs. By integrating parameter importance identification and model merging into a unified framework, they advance the state of the art in mitigating catastrophic forgetting while highlighting open challenges and directions for future research.

Chapter 4

Knowledge Editing for Large Language Models

4.1 Introduction

Large language models (LLMs) have demonstrated the ability to store vast amounts of knowledge during pre-training and retrieve it during inference [164, 137]. However, much of the knowledge in the real world is constantly evolving. For instance, the answer to the question "Who is the President of the United States?" was "Joe Biden" in 2024, but it is now "Donald Trump". As a result, some knowledge that was once correct in LLMs can become obsolete or inaccurate [71, 50].

Real-World Scenarios of Knowledge Editing. In practical deployment, large language models frequently require targeted knowledge updates without full retraining. Typical scenarios include correcting outdated factual information, incorporating newly emerging entities or events, refining domain-specific policies, and addressing safety-related adjustments. In such cases, retraining the entire model is often computationally prohibitive, while prompt-based solutions may not provide persistent or

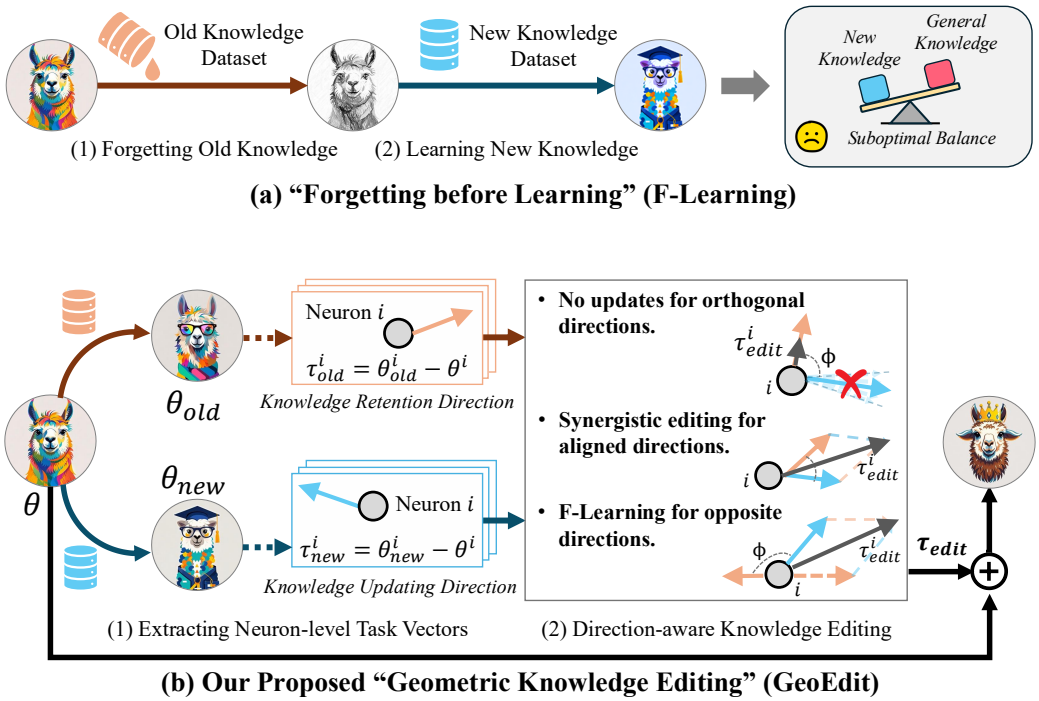


Figure 4.1: Conceptual illustration of F-Learning [100] and our proposed GeoEdit.

globally consistent updates. Knowledge editing aims to provide durable and localized parameter-level modifications that integrate new information directly into the model's internal representation.

Comparison with Context Engineering. Knowledge editing differs fundamentally from context engineering. Context engineering modifies model behavior at inference time by altering prompts, adding retrieved documents, or introducing structured instructions. It is flexible and does not alter model parameters, but its effects are transient and dependent on the provided context. In contrast, knowledge editing performs parameter-level updates that persist across inference settings, enabling consistent responses without requiring additional prompts or retrieval mechanisms. From a knowledge adaptation perspective, context engineering operates at the input level, whereas knowledge editing operates at the model level. These two approaches are complementary: context engineering provides flexible and reversible control, while

knowledge editing ensures persistent and integrated updates.

To address this issue, model editing methods have been developed to update the target new knowledge while preserving unrelated general knowledge within the model [40, 88, 139]. Specifically, current model editing methods typically follow the "locate-and-edit" paradigm [130, 127, 72]. The core idea is first to locate influential weights in LLMs and then edit them by introducing a perturbation. Although effective, these approaches incur significant computational overhead to identify the important neurons and parameters [96]. Some methods also require sampling additional data (e.g., from Wikipedia) to mitigate the impact on the general knowledge within LLMs during editing [97, 21], introducing extra costs and potential biases.

In contrast, fine-tuning with updated knowledge, as demonstrated in recent studies [180, 136, 80], offers a more straightforward solution through the use of parameter-efficient fine-tuning (PEFT) techniques. These methods employ various strategies to edit the model without the need to differentiate the importance of individual parameters [29]. Among these, the pioneering work F-Learning [100] introduces a new learning framework called "Forgetting before Learning," as illustrated in Figure 4.1(a). This approach is based on the empirical observation that new knowledge can be difficult to learn when it conflicts with existing knowledge. By first forgetting outdated knowledge, the learning of new knowledge becomes easier. However, this approach has a critical limitation: it struggles to balance the integration of new knowledge with the preservation of existing general knowledge. Specifically, F-Learning assumes that all updates between old and new knowledge are inherently conflicting, which oversimplifies the complexity of knowledge integration. Furthermore, the uncontrained forgetting process can significantly impact the model's generalization ability to out-of-scope samples, considerably reducing the performance of the Locality metric.

To address these limitations, we propose Geometric Knowledge Editing (**GeoEdit**), a novel fine-tuning-based model editing framework that enhances editing precision while strongly preserving model generalization without the need for additional unre-

lated data. The core insight of GeoEdit is to distinguish between neurons associated with new knowledge updates and those linked to general knowledge perturbations by analyzing the geometric relationships of parameter updates caused by fine-tuning. By masking the updates of ***general-knowledge-related neurons***, we prevent negative impacts on the model's generalization ability. At the same time, we optimize the update strategy for ***new-knowledge-related neurons***, further enhancing the effectiveness of model editing.

Specifically, we first fine-tune the current model separately on the old and new knowledge datasets. This allows us to derive neuron-level task vectors, τ_{old} and τ_{new}, using task arithmetic [51], which capture the directions of knowledge retention and updating w.r.t. each neuron, as shown in Figure 4.1(b). We then introduce a direction-aware knowledge identification method that computes the angle ϕ between these two directions to classify neurons, followed by customized editing strategies: (i) ***Orthogonal Knowledge Editing*** (for approximately orthogonal directions): Neurons with updates orthogonal to old knowledge are classified as general-knowledge-related neurons. These updates are considered detrimental to the model's generalization ability, so we refrain from updating these neurons. The remaining neurons are treated as new-knowledge-related neurons, which are updated with two different strategies: (ii) ***Synergistic Knowledge Editing*** (for aligned directions): When there is slight conflict between old and new knowledge, we can leverage their similarities to simultaneously integrate both. (iii) ***Conflicting Knowledge Editing*** (for opposite directions): For updates with significant conflict, we apply the F-Learning strategy, where old knowledge is first forgotten before integrating new information.

Additionally, to mitigate angular bias in high-dimensional space, GeoEdit employs a combined dimensionality reduction approach to more effectively extract angular information, ensuring the reliability of the edits. To optimize vector fusion, we introduce an importance-guided task vector fusion technique, which applies fine-grained weights to the vectors and suppresses noise from redundant parameters, further en-

hancing the effectiveness of model editing. Extensive experiments demonstrate that GeoEdit achieves the best performance among all fine-tuning-based methods and show its significant potential for complementing locate-and-edit methods, further enhancing performance.

Our main contributions are summarized as:

- We propose a novel geometric knowledge editing **framework** (GeoEdit) for updating LLMs.
- We develop new direction-aware knowledge identification and importance-guided task vector fusion **techniques**.
- Extensive **evaluation** on two widely-used datasets shows that GeoEdit overcomes the limitations of F-Learning, improving the Locality metric by 7.4% while maintaining the best performance in the Reliability and Generality metrics.

4.2 Related Work

Knowledge editing has gained significant attention due to the increasing need to update the knowledge in LLMs [116, 137, 5, 4]. Existing methods can be classified into two main approaches:

Locate and edit methods usually locate influential parameters and then edit them by introducing a perturbation [170, 56, 155]. Classic methods like ROME [96] and MEMIT [97] use causal reasoning to identify key neuron activations and adjust specific weights. Additionally, Yu et al. [166] employs gradient-based attribution to identify important weights. More recent approaches, such as AlphaEdit [21], improve the method by projecting perturbations onto the null space of preserved knowledge, demonstrating strong performance.

Fine-tuning is an intuitive and straightforward way to update the model's knowledge [26, 33, 181]. Recently, a series of PEFT methods, such as Prefix-Tuning [73] and

LoRA [46], have made knowledge editing based on fine-tuning more feasible. Zhang et al. [174] enhance update efficiency and adaptability by performing incremental parameter updates of varying magnitudes, which are determined by calculating the importance of the weight matrix.

However, both of these methods struggle to balance new knowledge updates with preserving unrelated knowledge [37, 27, 11]. For instance, locate-and-edit methods typically require large additional datasets to capture general knowledge and avoid disruption during editing [131, 44, 3, 171]. Furthermore, locate-and-edit methods primarily focus on editing the MLP layers of the model, while fine-tuning methods offer the flexibility to adjust different regions of the model.

Thus, our paper focuses on improving fine-tuning methods. By distinguishing between general knowledge and updated knowledge based on the angular divergence between the updated directions of old and new knowledge, our GeoEdit avoids updating general knowledge, ensuring model generalization while applying tailored strategies to enhance the effectiveness of knowledge updates. Additionally, our method can fine-tune parameters in regions different from those targeted by locate-and-edit methods, allowing for potential complementarity that further enhances performance.

Practical Instantiation of GeoEdit under Limited Old-Knowledge Data. GeoEdit constructs the task vector τ_{old} using an old-knowledge dataset to distinguish general-knowledge-related neurons from new-knowledge-related ones. In realistic settings, however, curated old-knowledge datasets may not always be available. In such cases, GeoEdit can be instantiated using alternative approximations. For example, the pre-trained model itself can serve as an implicit representation of old knowledge, and parameter differences before and after lightweight fine-tuning on new facts can approximate directional task vectors. Additionally, small replay sets, synthetic samples generated by the model, or Fisher-information-based approximations can be employed to estimate general-knowledge sensitivity. These strategies relax the strict

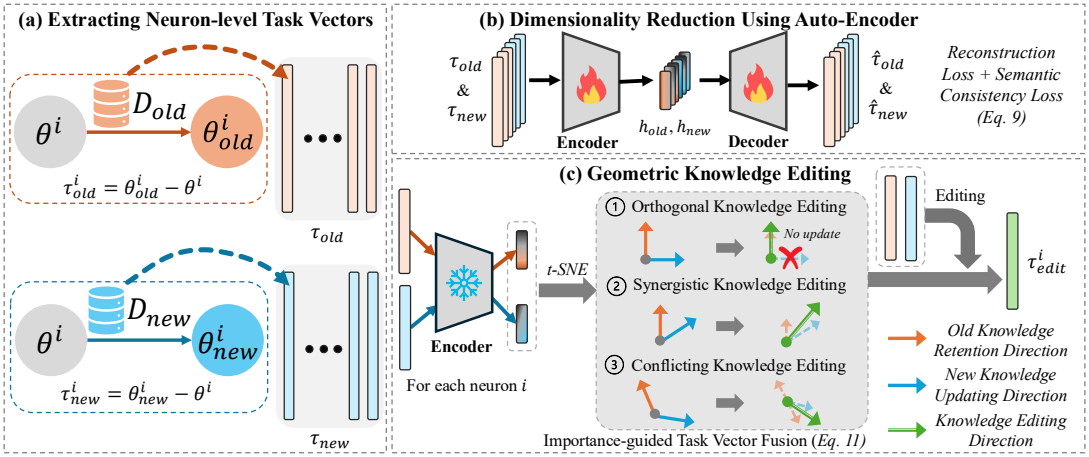


Figure 4.2: **Overview of KIF. Step (a)**: Neuron-level task vectors τ_{old} and τ_{new} are extracted for both the old and new knowledge datasets using parametric arithmetic. **Step (b)**: An auto-encoder is trained to project a low-dimensional representation of the task vectors, eliminating the angular bias issue in high-dimensional space. **Step (c)**: The latent task vectors, h_{new} and h_{old}, are reduced to two dimensions using t-SNE to compute the angular relationships, which are used to classify neurons based on the angle. Finally, after applying different editing strategies, we obtain the edited vector τ_{edit}, which is added to the initial model to generate the edited model f_{θ_e}.

requirement of large curated old datasets while preserving the geometric principle underlying GeoEdit. Therefore, although the availability of old-knowledge data improves estimation accuracy, the core mechanism of neuron-level geometric separation remains applicable under practical constraints.

4.3 Methodology

In this section, we present our method for knowledge editing in LLMs. As illustrated in Figure 4.2, GeoEdit follows a three-step process:

4.3.1 Extracting Neuron-level Task Vectors

Supervised fine-tuning (SFT) on a dataset injects new knowledge into the LLMs, reflected in model parameter changes. For an initial model f_θ with parameters θ, fine-tuning on dataset D produces updated parameters. The difference between the updated and original parameters is referred to as the *task vector* [51], calculated as:

$$\tau = \mathrm{FT}\{\theta, D\} - \theta \tag{4.1}$$

where τ is the corresponding task vector, and FT is the fine-tuning operation. Unlike F-Learning, we fine-tune the initial model f_θ separately on the old and new knowledge datasets to isolate their respective adaptations, then compute task vectors:

$$\tau_{old} = \mathrm{FT}\{\theta, D_{old}\} - \theta \tag{4.2}$$

$$\tau_{new} = \mathrm{FT}\{\theta, D_{new}\} - \theta \tag{4.3}$$

where D_{old} and D_{new} are datasets encoding outdated and updated knowledge respectively.

While prior research typically captures task vectors at the model level [51], we propose extracting them at the neuron level for finer control. Let $\theta = \{\theta^1, \theta^2, \ldots, \theta^N\}$ represent the N neurons in the LLM, where the i-th neuron is represented by $\theta^i \in \mathbb{R}^{d_n}$ with d_n dimensional parameters. The neuron-level task vectors are given by $\tau_{new} = \{\tau^1_{new}, \tau^2_{new}, \ldots, \tau^N_{new}\}$, where τ^i_{new} corresponds to the new knowledge task vector for the i-th neuron [1]. This approach enables more granular analysis of parameter changes and selective editing of knowledge-specific neurons, enhancing model editing precision.

After obtaining the task vectors for both old and new knowledge, we focus on the directional characteristics, which are more crucial than magnitudes for knowledge

[1] We define a "neuron" as the linear transformation corresponding to a single column in matrix $W \in \mathbb{R}^{in \times out}$, where W consists of *out* neurons.

editing. We define the direction of τ_{old} as the knowledge retention direction and τ_{new} as the knowledge updating direction. By analyzing the angle between these directions, we can distinguish general-knowledge-related neurons to avoid harming generalization and new-knowledge-related neurons to enhance editing effectiveness.

4.3.2 Angular Relationship Extraction through Dimensionality Reduction

Due to the tendency of high-dimensional vectors to become nearly orthogonal, it is necessary to reduce the dimensionality of the original vectors in order to better capture the underlying angular relationships. However, experiments have shown that directly applying PCA or t-SNE for dimensionality reduction on τ yields suboptimal results. Therefore, we propose an alternative approach where an auto-encoder (AE) is first used to encode the high-dimensional vectors. This effectively filters out irrelevant information and extracts meaningful features. Subsequently, applying t-SNE to the encoded vectors allows for a more accurate representation of the true angular relationships.

Thus, we train a semantic encoder and decoder, both implemented using multi-layer perceptrons (MLPs). Specifically, the semantic encoder, denoted as $\text{SemEnc}(\cdot)$, maps the high-dimensional task vectors τ_{old} and τ_{new} into the latent space as:

$$h^i = \text{SemEnc}(\tau^i) \tag{4.4}$$

where $h^i \in \mathbb{R}^{d_{latent}}$ is the latent task vector, and d_{latent} denotes its dimensionality. The decoder, $\text{Dec}(\cdot)$, then generates $\hat{\tau}^i$ from h^i as follows:

$$\hat{\tau}^i = \text{Dec}(h^i) \tag{4.5}$$

where $\hat{\tau}^i$ is the reconstructed task vector. The auto-encoder is optimized using both

a reconstruction loss and a semantic consistency loss:

$$\begin{aligned} \mathcal{L}_{AE} = \text{MSE}\left(\tau^i, \hat{\tau}^i\right) + \\ \lambda \cdot \text{KL}\left(f_{\theta+\tau^i}(x) \| f_{\theta+\hat{\tau}^i}(x)\right) \end{aligned} \tag{4.6}$$

where MSE$(\cdot)$ is mean square error loss function, and KL$(\cdot)$ is the Kullback-Leibler divergence.

4.3.3 Geometric Knowledge Editing

After training the AE, we project τ_{old} and τ_{new} into the latent space and then apply t-SNE to further project them into a 2D space to compute the angular relationships. GeoEdit then edit the original task vectors to obtain the edited task vector τ_{edit}. This vector is subsequently added to the initial model, resulting in the final edited model f_{θ_e}.

Direction-aware Knowledge Identification For neuron i, we first use the encoder to reduce the dimensionality of τ^i_{old} and τ^i_{new} yielding the latent task vectors h^i_{old} and h^i_{new}. We then apply t-SNE to obtain the 2D vectors $\hat{h}^i_{old}$ and $\hat{h}^i_{new}$. Next, we compute the angular divergence ϕ as:

$$\phi = \arccos \frac{\hat{h}^i_{old} \cdot \hat{h}^i_{new}}{\left|\hat{h}^i_{old}\right| \cdot \left|\hat{h}^i_{new}\right|} \tag{4.7}$$

Neurons with angles near orthogonality (within the range of ϕ_1 to ϕ_2) are classified as general-knowledge-related, while the remaining neurons are classified as new-knowledge-related.

Importance-guided Task Vector Fusion We then apply customized editing strategies based on the classification of neurons as follows:

$$\tau_{edit}^{i} = \begin{cases} \alpha^{i}\tau_{old}^{i} + \beta^{i}\tau_{new}^{i}, & \text{if } \phi \in (0^{\circ}, \phi_1) \\ 0, & \text{if } \phi \in [\phi_1, \phi_2] \\ -\alpha^{i}\tau_{old}^{i} + \beta^{i}\tau_{new}^{i}, & \text{if } \phi \in (\phi_2, 180^{\circ}) \end{cases} \tag{4.8}$$

where $\alpha^i, \beta^i \in [0, 1]$ are the fusion weights, automatically assigned based on the neuron's importance to both new and old knowledge, removing the need for manual adjustment.

To calculate the fusion weights, we measure the importance of each neuron by analyzing the gradient trajectory of its parameters during fine-tuning. The importance is determined by the collective contribution of its trainable parameters:

$$\mathcal{I}(\theta^i) = \frac{1}{d_n} \sum_{j=1}^{d_n} s(w_j) \tag{4.9}$$

where w_j represents the trainable parameters and d_n is the total number of parameters in neuron θ^i. The function $\mathcal{I}(\theta^i)$ reflects the importance of the neuron, with higher values indicating greater significance. The function $s(\cdot)$ computes the importance of individual parameters based on the magnitude of the gradient-weight product [174]:

$$s\left(w\right) = |w \nabla_w \mathcal{L}| \tag{4.10}$$

Due to stochastic sampling and training dynamics, the metric in Eq. (4.10) may vary, reducing reliability [28]. To address this, we apply an exponential moving average to smooth the trajectory gradients across training iterations.

We normalize the importance scores $\mathcal{I}_{\text{old}} = \{\mathcal{I}_{\text{old}}^1, \ldots, \mathcal{I}_{\text{old}}^N\}$ and $\mathcal{I}_{\text{new}} = \{\mathcal{I}_{\text{new}}^1, \ldots, \mathcal{I}_{\text{new}}^N\}$ independently to the range [0, 1]. This yields the final fusion weights $\alpha = \{\alpha^1, \ldots, \alpha^N\}$ and $\beta = \{\beta^1, \ldots, \beta^N\}$, which are then applied to the corresponding task vectors τ_{old} and τ_{new} for editing.

By applying Eq. (4.8), our GeoEdit effectively addresses the challenges in model editing:

- **Preserving general knowledge** (Case 2): We mask updates to general-knowledge-related neurons to avoid negatively impacting the model's generalization ability.
- **Improving knowledge editing** (Case 1 & 3): For acute angles, we leverage the similarity between old and new knowledge for efficient integration. For obtuse angles, significant conflict triggers a "forget-then-learn" strategy, optimizing the updates for new-knowledge-related neurons.

4.4 Experiments

Datasets We use two widely recognized datasets: ZsRE [67] and CounterFact [96]. We adopt the experimental setup from Yao et al. [164], using the eval and edit subsets consisting of 19,085 and 10,000 examples, respectively. The datasets are partitioned into old and new knowledge categories, as in F-Learning [100]. For example, in ZsRE, old knowledge is modified to new knowledge, such as the change from "Los Angeles" to "New Orleans."

Baselines We evaluate GeoEdit against two types of methods: fine-tuning-based approaches, including full fine-tuning (**Full-FT**), **LoRA** [45], **FT-c** [185], and **F-Learning** [100]; and locate-and-edit-based methods, including **MEND** [98], **ROME** [96], **MEMIT** [97], **RECT** [35] and **AlphaEdit** [21].

Training Details Following the setup of F-Learning, we first fine-tune the base model on the old knowledge for three epochs, resulting in the **original model**, which serves as the baseline for our experiments. In GeoEdit and F-Learning, we use LoRA to enhance the efficiency of fine-tuning. The encoder and decoder consists of 2-layer MLPs with dimensions $[4096 \to 2048, 2048 \to 512]$ and $[512 \to 2048, 2048 \to 4096]$,

respectively, where d_{latent} is set to 512. We set λ in Eq. (4.6) to 0.5, ϕ_1 and ϕ_2 in Eq. (4.8) to 85° and 95°, respectively.

4.4.1 Experimental Results

The overall results are presented in Table 4.1. Firstly, ROME maintains high Reliability and Generality across both datasets while achieving excellent Locality (greater than 90). Since the injection of new knowledge typically impacts Locality, this suggests that ROME performs minimal knowledge updating, likely due to its limited parameter edits. In contrast, F-Learning shows a significant drop in Locality due to the lack of constraints during the forgetting phase, negatively impacting generalization. Our GeoEdit method outperforms fine-tuning-based methods, improving locality by 7.4% over F-Learning. Additionally, by classifying different knowledge editing strategies for new-knowledge-related neurons, our method further improves Reliability and Generality.

AlphaEdit achieves the best performance among locate-and-edit-based methods and outperforms GeoEdit in most cases. This is because AlphaEdit requires the use of an additional 100,000 Wikipedia entries to enhance general knowledge encoding and editing accuracy. In contrast, GeoEdit achieves its performance without relying on any external data. Furthermore, due to the flexibility of fine-tuning methods, we can effectively combine GeoEdit with AlphaEdit (which edits the parameters of the MLP layer, while GeoEdit targets the parameters of the attention layer), creating a complementary approach that further enhances performance.

4.4.2 Ablation Study

We conduct ablation studies to evaluate the effectiveness of the techniques in GeoEdit. The results on the ZsRE dataset with LLaMA2-7B are shown in Table 4.2.

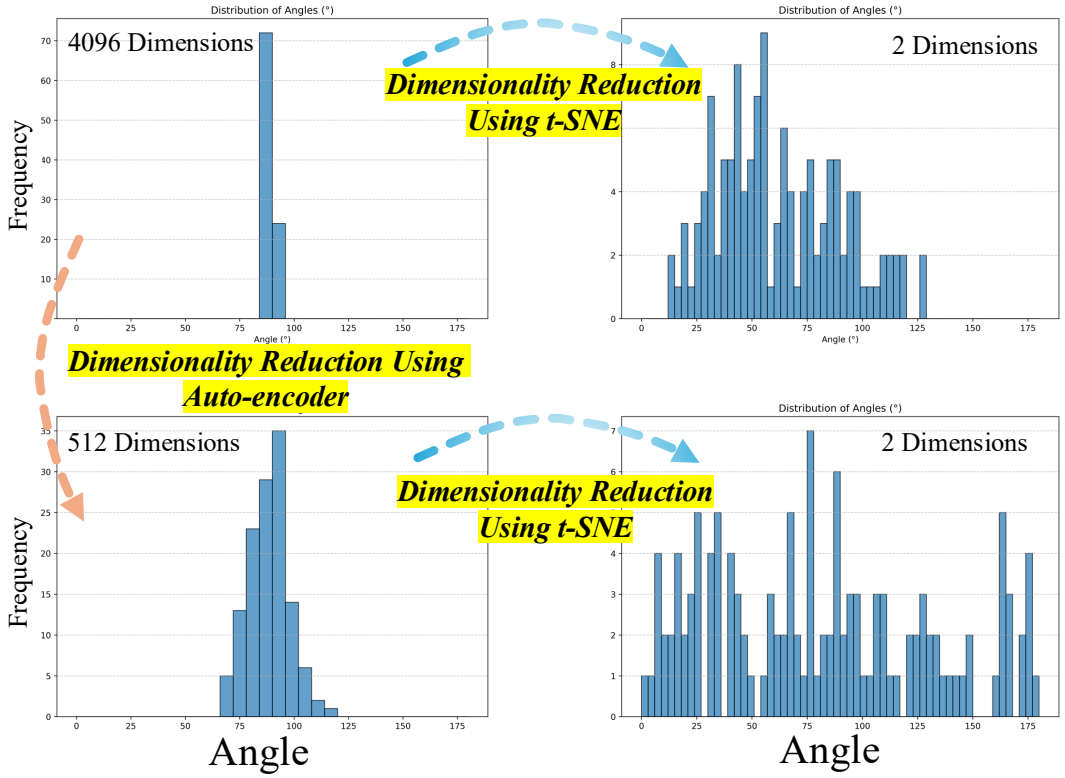


Figure 4.3: Distribution of the angles ϕ between task vectors before and after dimensionality reduction.

Effect of Geometric Editing Strategies. In GeoEdit, knowledge updates are categorized into synergistic, orthogonal, and conflict editing strategies, based on the angle between knowledge retention and updating directions. To evaluate their impact, we disable each strategy and replace it with vanilla fine-tuning on new knowledge. For example, "- Orthogonal" means setting $h_{edit} = h_{new}$ instead of $h_{edit} = 0$. As shown in Table 4.2, removing any strategy results in performance degradation. Excluding orthogonal editing significantly reduces locality, from 75.7% to 71.7%, while removing conflict editing lowers the reliability metric from 85.2% to 82.6%. These findings underscore the importance of each editing strategy.

Effect of Importance-guided Task Vector Fusion. We replace the importance-based weights α and β in Eq. (4.8) with manually set values ("+ MW"), applying the

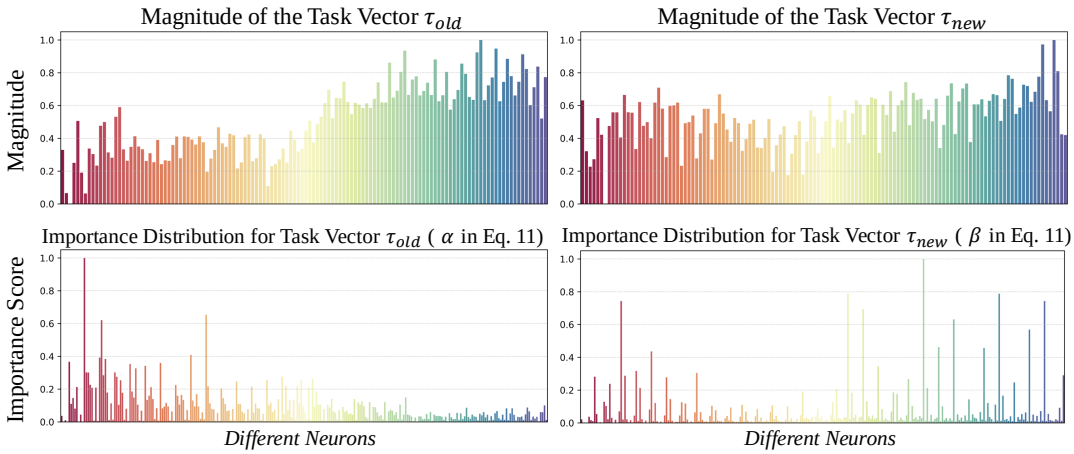


Figure 4.4: Visualization of the magnitudes of task vectors τ_{old} and τ_{new} along with the importance-guided fusion weights. All results are normalized to the range of 0 to 1.

same weight to all neurons instead of assigning neuron-specific weights as in GeoEdit. Through grid search, we set $\alpha = 0.3$ and $\beta = 1$.

The performance decline in Table 4.2 highlights the effectiveness of our importance-guided fusion. This approach provides two key benefits: it offers neuron-level adaptive weights for greater precision and ensures that parameter updates are influenced by both the task vector's magnitude and each neuron's importance. The smaller weights "masks" significant changes for less important neurons, minimizing their impact on the model's generalizability.

Effect of Latent Space Dimension. We investigate the effect of the latent dimension h_{latent} in the auto-encoder on model performance. As shown in Table 4.3, both larger latent dimensions (greater than 1024) and very small latent dimensions (less than 256) lead to performance collapse. This is because the large training loss of the auto-encoder results in poor t-SNE dimensionality reduction, ultimately affecting the accuracy of angular calculations. Our experiments demonstrate that a latent dimension of 512 strikes an optimal balance, effectively removing noise and extracting

key features for calculating the angular distribution, which ensures effective model editing and strong generalization.

4.4.3 Visualization

We present two key visualizations to demonstrate the effectiveness of our approach:

Angle Distribution Between Old Knowledge Retention and New Knowledge Updating Directions. We visualize the angle distribution ϕ between task vectors using different dimensionality reduction methods, as shown in Figure 4.3. In high-dimensional space, angles are primarily concentrated around 90 degrees, indicating near orthogonality. Although directly applying t-SNE alleviates this issue to some extent, the angular distribution remains insufficiently dispersed. By first using an AE for denoising and key feature extraction, followed by t-SNE, we achieve a more uniform distribution that spans the full range from 0 to 180 degrees. This allows us to reveal various types of conflicts between old and new knowledge. This motivates the development of editing strategies based on angles, enabling us to distinguish between updates that correspond to learning new knowledge and those that modify general knowledge.

Visualization of Task Vector Magnitudes and Importance-guided Fusion Weights. Figure 4.4 illustrates that while the magnitudes of the task vectors τ_{old} and τ_{new} are generally large, only a subset of the parameters are truly important, highlighting redundancy in the task vectors. Our importance-guided fusion mechanism effectively filters out this redundancy, enhancing the model editing process and minimizing its impact on generalization.

4.4.4 Editing Time Analysis

Table 4.4 shows the average time to edit 1000 samples. We find that the editing time of fine-tuning methods is comparable to that of location-based methods, thanks to the use of PEFT techniques and the avoidance of the complex location-based process. Among fine-tuning approaches, F-Learning, which follows a two-stage process of forgetting before learning, takes approximately twice as long as Full-FT. In contrast, our method enables the parallel acquisition of old and new knowledge, resulting in training times comparable to Full-FT. Thus, our method requires less time than F-Learning, delivering substantial performance improvements.

4.5 Chapter Review

In this chapter, we proposed Geometric Knowledge Editing (GeoEdit), a novel framework that leverages the geometric relationships between parameter updates to enhance model editing in large language models. GeoEdit introduces a direction-aware knowledge identification technique to distinguish between general-knowledge-related neurons and new-knowledge-related neurons. For the former, parameter updates are masked to avoid harming the model's generalization ability, while for the latter, an importance-guided task vector fusion strategy is applied to facilitate efficient knowledge integration. Extensive experiments validate the effectiveness of GeoEdit in achieving a favorable balance between integrating new knowledge and preserving general knowledge.

Limitation in accessing old knowledge. GeoEdit relies on access to old knowledge datasets to construct the task vector τ_{old}. However, in practice, such datasets may not always be available. A possible workaround is to query the initial model directly with the editing task and treat its outputs as pseudo old knowledge. While

this solution reduces dependence on datasets, it introduces additional inference costs, especially in mass-editing scenarios, and poses challenges when the outputs are ambiguous (e.g., in open-ended questions). Future work may explore dataset-free task vector extraction methods to improve the practicality and adaptability of GeoEdit.

Limitation of angle-based knowledge disentanglement. The core mechanism of GeoEdit differentiates between general and new knowledge by examining the angle between parameter update vectors. Although this geometric perspective provides valuable insights, it does not perfectly disentangle knowledge sources, and some degradation of generalization ability persists. To mitigate this limitation, future extensions may incorporate richer geometric features—such as task vector projections, update magnitudes, or higher-order alignment measures—to refine the editing process and achieve stronger preservation of general knowledge.

Overall, GeoEdit presents a promising step toward reliable and controllable model editing. By combining direction-aware identification with task vector fusion, it advances the ability of LLMs to adapt new knowledge efficiently while retaining previously acquired information.

Dataset	Paradigm	Method	LLAMA2-7B			LLAMA-7B		
			Reliability	Generality	Locality	Reliability	Generality	Locality
ZsRE		Original model	43.70	43.17	/	43.29	42.85	/
	Locate & edit	MEND	29.77	25.86	71.54	30.99	27.12	69.83
		ROME	43.67	42.66	**93.14**	43.45	42.94	**98.60**
		MEMIT	83.57	79.06	70.52	78.30	77.43	69.44
		RECT	84.08	77.80	69.03	78.78	76.20	67.97
		AlphaEdit	**87.91**	**81.52**	**77.14**	**87.09**	**80.41**	**76.53**
	Fine-tuning	LoRA	43.10	42.20	70.83	46.93	45.87	75.86
		FT-c	49.02	46.96	67.37	47.33	45.51	68.14
		Full-FT	81.02	74.67	70.51	70.52	66.69	65.26
		F-Learning	84.65	81.51	70.92	83.06	79.50	70.09
		GeoEdit	**85.21**	**82.43**	**75.71**	**84.81**	**79.86**	**75.15**
		GeoEdit*	**88.13**	**82.07**	**79.75**	**87.76**	**80.70**	**77.98**
CounterFact		Original model	18.47	16.95	/	21.61	17.88	/
	Locate & edit	MEND	14.77	14.67	90.93	17.51	16.27	89.64
		ROME	18.41	17.20	**93.60**	21.83	19.08	**92.27**
		MEMIT	61.94	37.45	21.90	56.94	31.48	25.70
		RECT	62.90	39.86	20.03	57.82	33.51	23.48
		AlphaEdit	**71.79**	**48.36**	**36.07**	**60.01**	**38.19**	**41.70**
	Fine-tuning	LoRA	30.56	23.24	40.08	27.54	21.21	39.75
		FT-c	29.23	19.32	19.70	26.97	17.90	20.09
		Full-FT	65.99	44.08	28.34	32.13	31.95	32.51
		F-Learning	**69.53**	45.56	28.41	56.39	39.75	31.87
		GeoEdit	68.34	**46.53**	**37.73**	**55.88**	**40.60**	**42.33**
		GeoEdit*	**72.20**	**48.57**	**38.71**	**60.89**	**41.37**	**43.99**

Table 4.1: Results on three metrics for the two datasets using LLAMA2-7B and LLAMA-7B. The best-performing method for each paradigm is highlighted in bold. AlphaEdit and our GeoEdit each achieve the best performance within their respective paradigms. Notably, the optimal performance is attained by GeoEdit*, which results from applying GeoEdit to the non-located parameters in AlphaEdit, effectively combining the strengths of both methods.

Method	Reliability	Generality	Locality
GeoEdit	**85.21**	**82.43**	**75.71**
- Synergistic	84.37	81.13	75.94
- Orthogonal	85.29	82.86	71.70
- Conflict	82.57	79.40	75.45
+ MW	84.91	82.04	73.73

Table 4.2: Ablation study. "- Synergistic", "- Orthogonal", and "- Conflict" refer to removing the synergistic, orthogonal, and conflict knowledge editing strategies, respectively. "+ MW" denotes replacing the importance-guided fusion with a manually set weighting approach.

h_{latent}	Reliability	Generality	Locality
128	44.94	43.79	75.68
256	46.75	46.54	77.33
512	46.11	47.19	78.42
1024	25.39	24.49	94.81
2048	23.03	24.14	96.14

Table 4.3: Ablation study on latent dimension h_{latent}.

Method	**Average time per 1000 edits**	
	zsRE	**COUNTERFACT**
FT-c	653.2(s)	579.3(s)
ROME	2184.2(s)	1810.4(s)
MEMIT	862.2(s)	847.7(s)
Full-FT	810.2(s)	792.4(s)
F-Learning	1670.4(s)	1603.8(s)
GeoEdit	**1028.0(s)**	**1010.6(s)**

Table 4.4: Editing time for two datasets on LLAMA2-7B.

Chapter 5

Application: Dialogue State Tracking

5.1 Introduction

Practical dialogue systems require continual adaptation to new services while maintaining previously acquired task capabilities. However, most prior research on dialogue systems has focused on domain-specific, offline training settings, with limited consideration of adaptation to evolving services and user requirements [99].

With the recent advances in large language models (LLMs), LLM-based dialogue systems have demonstrated significant improvements over traditional pipeline-based approaches [47]. Nevertheless, the large parameter scale of LLMs makes full retraining prohibitively expensive in terms of computational cost and time [79]. Therefore, efficient continual learning (CL) mechanisms are essential for enabling dialogue systems to acquire new capabilities while preserving prior knowledge.

Dialogue State Tracking (DST). Dialogue State Tracking (DST) is a core component of task-oriented dialogue systems. Given a multi-turn dialogue, the DST

module maintains a structured representation of the user's goal, typically formulated as a set of (domain, slot, value) triplets. At each turn, the model updates the dialogue state based on the dialogue context, which includes both user utterances and system responses.

In this thesis, we adopt an LLM-driven DST framework, where the dialogue context is encoded by the language model and the dialogue state is generated or extracted in a structured format. This formulation allows DST to naturally benefit from the generalization ability of LLMs, while also exposing it to catastrophic forgetting when new domains or services are introduced sequentially.

As real-world dialogue systems continuously expand to cover new domains, continual DST has emerged as an important research problem [14]. This setting requires models to incrementally incorporate new domain-slot-value structures while maintaining performance on previously learned domains, making it a representative application scenario for knowledge adaptation.

As a specific task of CL, Continual DST grapples with catastrophic forgetting [95, 32], where sequential learning of new tasks hinders retention of prior ones. This problem is more acute in DST due to significant data distribution shifts across domains in the CL process, exemplified by transitions from Hotel to Restaurant domains (see Figure 5.1). Recent advances in Continual DST, including memory replay and regularization [187], and reformulation as a question-answering task [14], strive to address forgetting. However, these methods face challenges like reliance on past data and high computational demands during testing, which hinder real-time applications.

Recent LLMs show impressive DST performance [39, 26], but practical deployment faces hurdles like offline computational load and online data privacy concerns. Moreover, enabling LLMs for CL demands significant resources, leading to the exploration of smaller models. After a comprehensive analysis of the current smaller DST models, we have identified a critical loss of capability brought by domain shifts in CL that

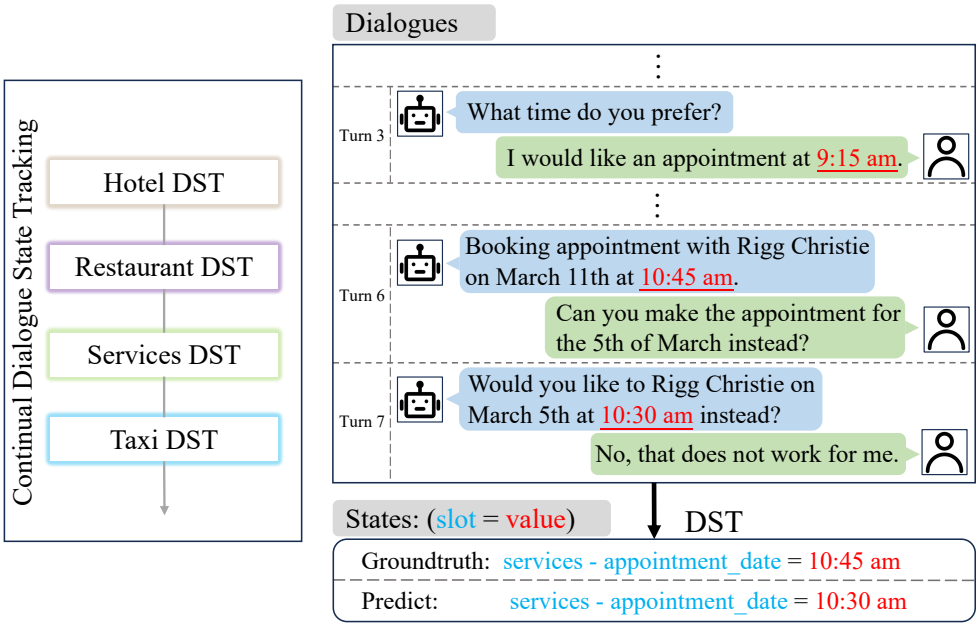


Figure 5.1: *Left:* Depiction of the Continual DST learning process. *Right:* An actual instance of the "Value Selection Quandary" phenomenon, demonstrating a dialogue with three mentioned date values, where the model incorrectly chooses the most recent time at turn 7 rather than the correct value at turn 6.

models fail to output correct value when facing a set of similar candidates (named "Value Selection Quandary" as elaborated in Section 2). For example, in Figure 5.1, when tracking the ¡services-appointment_date¿ slot, the model incorrectly chooses the most recent time mentioned, revealing its inability to grasp contextual subtleties and favoring direct value extraction over logical reasoning. This phenomenon is because, as the CL progresses, the model needs to understand the relevant knowledge of newly emerging domains and chooses to forget domain-irrelevant or weakly relevant knowledge, such as value selection.

This thesis addresses these challenges by boosting smaller models' reasoning ability, termed as ***meta-reasoning***. Meta-reasoning, guided by the insight that domain-specific dialogues represent only a fragment of the underlying meta-knowledge, can be viewed as a form of multi-view augmentation from different domains in the CL

process. This domain bootstrapping strategy facilitates the transfer and broadening of meta-knowledge, transcending traditional single-perspective reasoning. Meta-reasoning enriches smaller models' reasoning abilities to dissect and interpret intricate dialogue scenarios, addressing the "Value Selection Quandary" and broadening their cognitive horizons. Moreover, this domain-agnostic feature aligns data distribution shifts across different domains, effectively reducing catastrophic forgetting.

Inspired by the powerful reasoning abilities of LLMs like LLaMA-2-70B [122] and ChatGPT [1], we introduce Reason-of-Select (RoS) distillation approach, designed to graft these advanced reasoning capabilities onto smaller models. We present a "multi-value resolution" strategy tailored to DST, prompting teacher LLMs to generate a "selection chain" that discerns and elaborates the best value choice from various options. Then, we distill this rationale into a smaller student model to boost its meta-reasoning capabilities, making it easier to deploy to terminal systems.

Moreover, to enhance faithful reasoning and reduce hallucinations in the teacher model, we integrate a schema-guided prompt, strengthening source-target correlation and semantic coherence. Our innovative Semantic Contrastive Reasoning Selection method uses semantic similarity to select the most accurate teacher-generated rationale, suitable even in black-box scenarios. Extensive experiments with various teacher and student model sizes demonstrate our method's exceptional Continual DST performance and robust cross-dataset generalization capabilities.

To summarize, our main contributions are:

- We propose a Reason-of-Select (RoS) distillation framework for Continual DST. Through enhancing domain-agnostic meta-reasoning capabilities, ambiguous value selection and catastrophic forgetting caused by data distribution shifts across domains can be effectively mitigated.
- To alleviate the hallucination outputted by a teacher model in our annotation-

[1] https://chat.openai.com/chat

free reasoning framework, we present a Semantic Contrastive Reasoning Selection strategy to obtain trust-worthy rationale from multiple candidates, ensuring faithful reasoning transfer.

- Comprehensive experiments with two teacher models and four student models of varying sizes demonstrate superior performance and robust generalization of our proposed method.

5.2 Related Work

5.2.1 Knowledge Distillation from LLMs

Knowledge distillation in NLP, crucial for transferring insights from larger teacher models to smaller student models, has evolved recently, especially in extracting reasoning from LLMs. Recent studies [43, 142, 160, 154] highlight LLMs' role in bolstering smaller models. Li et al. [70] furthered this with a symbolic Chain of Thought (CoT) distillation, enhancing smaller models through CoT prompting. Despite these advancements, the faithfulness of generated rationales, critical for student models' behavior, often must be addressed. Wang et al. [134] proposed a contrastive decoding method to reduce hallucinations, but its reliance on teacher model logits limits its applicability for black-box LLMs. Current reasoning distillation methods are unsuitable for DST task and lead to unproductive reasoning processes.

To overcome these issues, our method focuses on the Value Selection Quandary in DST, introducing a 'multi-value resolution' prompt and a Semantic Contrastive Reasoning Selection method, enhancing knowledge transfer accuracy and relevance.

5.2.2 Continual Dialogue State Tracking

Continual Learning in task-oriented dialogue systems, focusing on mitigating catastrophic forgetting, has employed various methods such as architecture-based [115, 34, 153], rehearsal-based [110, 41, 85], and regularization-based [76, 28, 84]. In DST, contributions like Madotto et al. [90] and Liu et al. [81] have utilized these CL strategies, with Zhu et al. [187] introducing Continual Prompt Tuning (CPT) to fine-tune domain-specific soft prompts. Furthermore, DST-EGQA [14] employs a question-answering framework based on example-guided learning. However, its reliance on fixed QA templates may limit adaptability across diverse domains. Our method, in contrast, offers enhanced time efficiency and flexibility during testing, eliminating the need for sample retrieval, thus presenting a more efficient solution.

5.3 Methodology

5.3.1 Overview

Our Reason-of-Select Distillation framework capitalizes on LLMs to create a faithful selection reasoning process through a 'multi-value resolution' strategy and Semantic Contrastive Reasoning Selection. This enriched knowledge is then imparted to smaller student models for training, as illustrated in Figure 5.2.

5.3.2 Teacher's Reasoning Generation

The process begins with a dialogue-centric prompt that includes dialogue content $\mathcal{X}$, the target slot S_j, and its value V_j to derive a reasoning process $\mathcal{R}$. While traditional prompts like "Tell me why S_j is V_j" or the famous "Let's think step by step" prompt [62] generate reasonings that tend to merely highlight the location of the correct

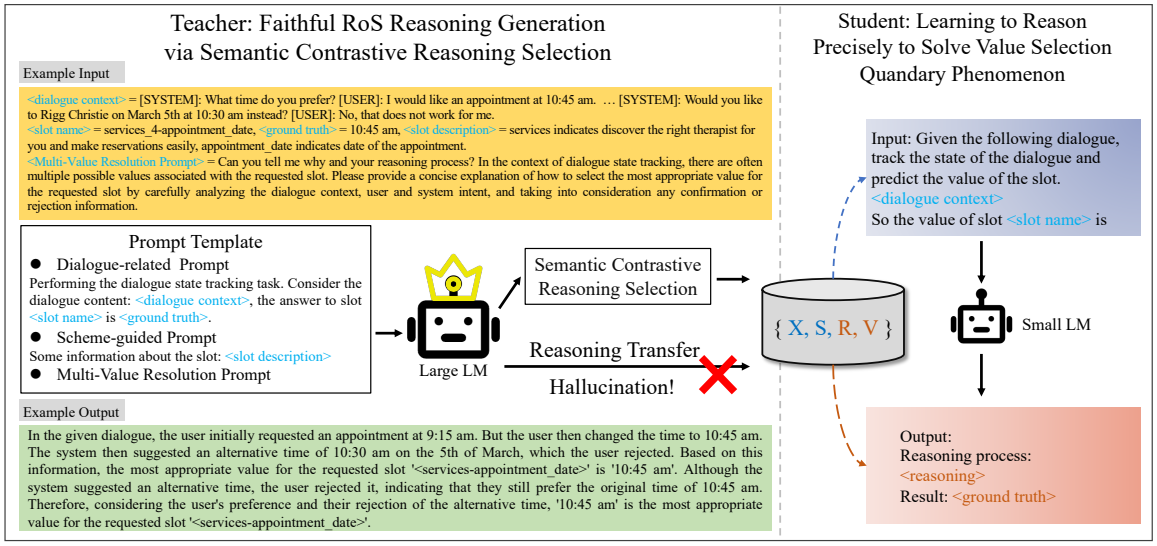


Figure 5.2: **Overview of the Reason-of-Select (RoS) Distillation method.** (a) Teacher: A large LM prompted to generate a faithful rationale given a dialogue context and the value for the request slot in the training set via the "multi-value resolution" strategy and Semantic Contrastive Reasoning Selection method. (b) Student: A small LM is fine-tuned to generate an accurate rationale and the corresponding value.

answer in the dialogue.

Our innovative "multi-value resolution" prompt $\mathcal{P}_R$ fosters a more elaborate reasoning process, enhancing the model's capability from mere location identification to an in-depth selection reasoning process, as illustrated in Figure 5.2. This methodological advancement from basic identification to complex process reasoning represents a substantial leap in the model's cognitive capacities.

This stage's input-output representation is $f_{teacher} : \mathcal{X}_t \oplus S_j \oplus V_j^t \oplus \mathcal{P}_R \rightarrow \mathcal{R}_j^t$. However, neural LMs often exhibit hallucinations — generating text with tenuous ties to the input [53, 93], which our unsupervised annotation approach can exacerbate. To counter this, we introduce the innovative Semantic Contrastive Reasoning Selection method, pivotal for accurately aligning reasoning with corresponding answers.

5.3.3 Ensuring Faithful Teaching with Semantic Contrastive Reasoning Selection

Our novel strategy employs a newly schema-guided prompt (yellow background in Figure 5.2) to direct teacher models toward generating on-topic rationales, significantly improving the source-target correlation. Our Semantic Contrastive Reasoning Selection technique revolutionizes reasoning generation, creating a spectrum of candidates and choosing the most semantically aligned. Inspired by contrastive decoding [74] in text generation task, our method adapts this concept to suit black-box LLMs like ChatGPT. We introduce strategic input perturbations to simulate and correct reasoning errors, refining the teacher model's output.

The procedure commences with the LLM generating G diverse reasonings $(\mathcal{R}_1, \ldots, \mathcal{R}_G)$. We then introduce two forms of perturbation: value-level and slot-level, as illustrated in Figure 5.3. While value-level perturbation subtly modifies the ground truth value, slot-level perturbation involves completely replacing the slot-value pair. This latter proves particularly effective in inducing a range of reasoning responses, from logical to nonsensical, thereby enriching our negative sample pool with N perturbed reasonings $(\mathcal{PR}_1, \ldots, \mathcal{PR}_N)$. Positive samples are represented by the dialogue-centric prompt, denoted as $\mathcal{DC}$. Utilizing a fixed, pre-trained contextual encoder, Sentence-transformers [111], we then transform these $\mathcal{R}$, $\mathcal{PR}$ and $\mathcal{DC}$ into semantic representations within space $\mathcal{E}$, as illustrated in Figure 5.4.

The crux of our method lies in selecting the most appropriate reasoning from the generated G candidates, a decision based on their semantic proximity to the positive samples and divergence from the negative samples. This selection process is formulated as an elegant optimization problem:

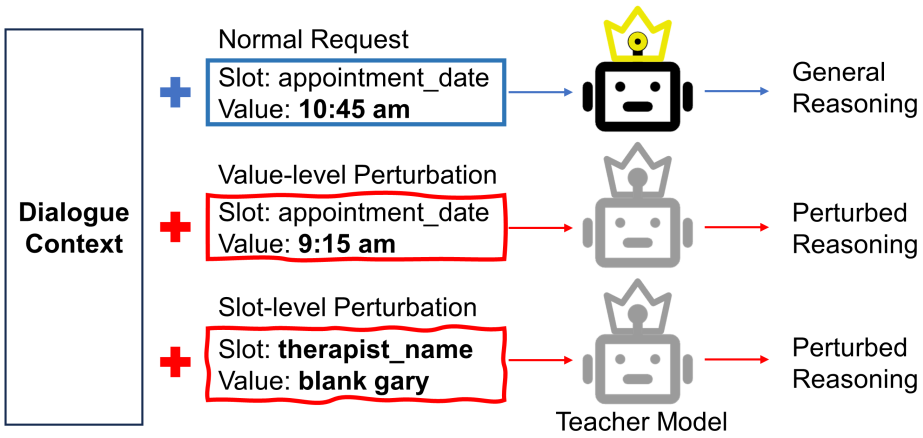


Figure 5.3: Demonstration of value-level and slot-level perturbations to elicit diverse negative reasonings.

$$\begin{aligned} &\min \mathit{Distance}(\mathcal{R}_i, \mathcal{DC}) \\ &\text{s.t.} \max \sum_{n=1}^{N} \mathit{Distance}(\mathcal{R}_i, \mathcal{PR}_n) \end{aligned} \tag{5.1}$$

To quantify the relationship between each reasoning and the sample sets, we introduce a sophisticated scoring mechanism:

$$\mathit{Score}(\mathcal{R}_i) = \frac{\exp\left(d\left(\mathcal{R}_i, \mathcal{DC}\right)/\tau\right)}{\sum_{n=1}^{N} \exp\left(d\left(\mathcal{R}_i, \mathcal{PR}_n\right)/\tau\right)} \tag{5.2}$$

where d is a distance function and τ is a temperature scalar. By computing a score for each reasoning, we discern the optimal choice with the lowest score for training the student model based on its alignment with actual dialog context and its divergence from perturbed content.

5.3.4 Training Student Models via Reasoning-Enhanced Data

Armed with the annotated dataset $\{\mathcal{X}, S, \mathcal{R}, V\}$, we proceed to train a smaller student model within the self-rationalization framework, emphasizing both predictive

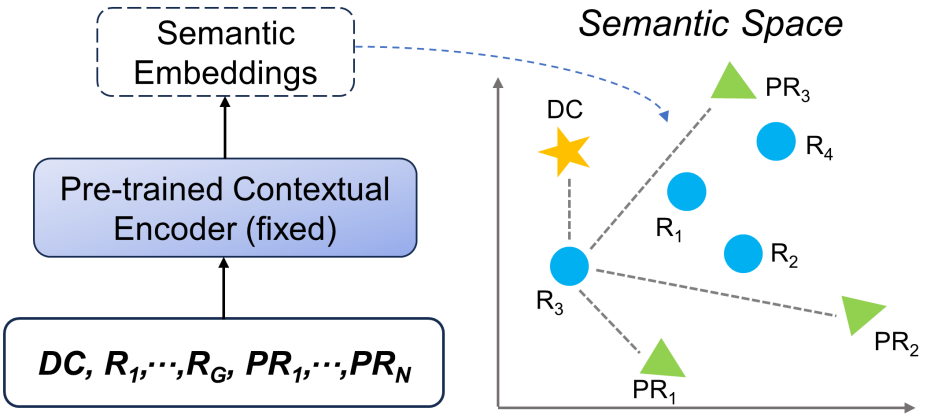


Figure 5.4: Illustration of the Semantic Contrastive Reasoning Selection method.

and explanatory skills. This approach departs from prior post-rationalization models, where rationales are formulated post-prediction or those employing a multi-task format, treating rationale creation as an auxiliary task.

The student model is conditioned to generate a sequence that merges rationale tokens with the corresponding answer tokens in response to a given dialogue context and slot request, as demonstrated in the right part in Figure 5.2. This task is executed by fine-tuning a pre-trained language model, with "silver" data derived from the teacher model. We employ a standard language modeling loss termed factual reasoning loss:

$$\mathcal{L}_{factual} = -\sum_{j}^{J} \log P\left(\mathcal{R}_j, V_j \mid \mathcal{X}, S_j\right) \tag{5.3}$$

5.4 Experiments

5.4.1 Experimental Setup

Dataset Our experiment employs the Schema-Guided Dialog dataset (SGD) [107], encompassing 44 services across 19 domains with slot descriptions. Adhering to the Continual DST setup by Zhu et al. [187], we focus on single-service dialogs, randomly selecting 15 tasks from 44. Each service varies in training sample size and slots. To

ensure robustness, we experiment with five task orders via random permutations, aligning with prior studies.

Evaluation Protocol We assess the DST performance using the widely adopted Joint Goal Accuracy (JGA) metric [147], which demands accurate predictions for all slot values. We denote $a_{j,i}$ as the JGA on the test set of task $\mathcal{T}_i$ right after training on task $\mathcal{T}_j$. CL performance is assessed using three metrics from Zhu et al. [187]: *(i)* $\textbf{Avg.JGA} = \frac{1}{K}\sum_{i=1}^{K} a_{K,i}$, representing the average JGA across all tasks after training on the final task $\mathcal{T}_K$. *(ii)* Forward Transfer ($\textbf{FWT}$) $= \frac{1}{K-1}\sum_{i=2}^{K} a_{i-1,i}$, evaluating generalization by measuring zero-shot performance, and *(iii)* Backward Transfer ($\textbf{BWT}$) $= \frac{1}{K-1}\sum_{i=1}^{K-1} a_{K,i} - a_{i,i}$, quantifying resistance to forgetting by assessing the influence of new learning on previous tasks.

Baselines We evaluate our model against existing Continual DST baselines: ***Fine-tuning:*** Continuously fine-tune the model on new task data. ***Replay:*** Stores $|M|$ instances per task $\mathcal{T}_i$ in memory M_i for joint training with new tasks. ***EWC [61]:*** Maintain a memory but leverage it to compute the Fisher information matrix for regularization. ***AdapterCL [42]:*** Freeze the pre-trained model and independently train a residual Adapter for each task. ***Continual Prompt Tuning (CPT) [187]:*** Freeze the backbone model and continually train soft prompts with knowledge transfer in both forward and backward directions. ***DST-EGQA [14]:*** Reformulate DST as a Question-Answering task using retrieval-augmented in-context learning.

We also include our method with memory replay and multi-task learning as performance caps.

Training Details ChatGPT (using the gpt-3.5-turbo API) and LLaMA-2-70B serve as teacher models for fine-tuning smaller student models including T5-small,

Method	Teacher	Student	Avg. JGA	FWT	BWT	+Memory	+Params	+Reg.
Fine-tune	-		$44.1_{0.9}$	$8.3_{1.0}$	$-36.6_{3.9}$	-	-	-
EWC	-		$47.9_{1.1}$	$8.4_{0.9}$	$-38.1_{4.1}$	✓	✓	✓
Replay	-		$58.6_{3.5}$	$10.9_{0.5}$	$-3.2_{2.3}$	✓	-	-
AdapterCL	-	*T5-small*	$49.8_{1.7}$	-	-	-	✓	-
CPT	-		$61.2_{2.5}$	$13.7_{0.8}$	$\mathbf{0.5}_{0.4}$	✓	✓	✓
DST-EGQA	-		$55.5_{3.5}$	$23.6_{2.1}$	$-19.1_{4.2}$	-	-	-
+ Dialogue Memory	-		$68.9_{0.3}$	$22.5_{1.8}$	$-5.9_{1.9}$	✓	-	-
RoS (ours)	*LLaMA-2-70B*	*T5-small*	$59.0_{3.9}$	$25.5_{2.0}$	$-17.9_{3.7}$	-	-	-
+ Dialogue Memory			$\mathbf{72.1}_{0.8}$	$26.7_{2.0}$	$-2.6_{1.5}$	✓	-	-
RoS (ours)	*ChatGPT*	*LLaMA-7B*	$68.7_{4.1}$	$\mathbf{51.9}_{1.7}$	$-8.5_{3.8}$	-	-	-
+ Dialogue Memory			$\mathbf{74.2}_{3.7}$	$\mathbf{52.7}_{1.5}$	$\mathbf{-2.4}_{2.7}$	✓	-	-
CPT Multi-task	-	*T5-base*	$64.0_{1.9}$	-	-	-	✓	✓
DST-EGQA Multi-task	-	*T5-small*	$74.2_{1.8}$	-	-	-	-	-
RoS Multi-task	*LLaMA-2-70B*	*T5-small*	$76.3_{0.5}$	-	-	-	-	-
	ChatGPT	*LLaMA-7B*	$78.9_{0.3}$	-	-	-	-	-

Table 5.1: CL metric results and reliance on other continual learning techniques. Means and standard variances are reported. We compare models sequentially trained on 15 tasks from the SGD dataset and aggregate results across five domain permutations. The last four rows provide the multi-tasking results, which serve as an upper bound. All rows that use memory are with $M = 50$.

T5-base, FlanT5-XL, and LLaMA-7B, utilizing generated rationales. The teacher model's temperature is set to 0.7, generating five candidate reasonings ($G = 5$) alongside three value-level and three slot-level perturbed negative reasonings ($N = 6$). In Eq.2, the parameter τ is set to 0.8, employing the Euclidean distance metric. The memory size per task $|M|$ is maintained at 50, in line with previous studies [187].

5.4.2 Main Results

Table 5.1 presents the results from various methodologies. Key findings include:

Injecting Reasoning Knowledge Effectively Enhances CL Performance Our RoS method significantly surpasses standard fine-tuning on T5-small, raising the Avg.JGA from 44.1% to 59.0%, and showing gains in both FWT and BWT. This highlights the value of integrating reasoning into Continual DST. Compared with the erstwhile top-performing DST-EGQA, we achieve a new SOTA performance in all metrics, notably increasing Avg. JGA from 55.5% to 59.0% without extra memory, demonstrating our approach's effectiveness in mitigating historical task forgetting. Furthermore, when memory is available, RoS jumps from 59.0% to 72.1%, even outstripping the CPT's multi-task performance. Among various student models, RoS with LLaMA-7B, fine-tuned with ChatGPT's reasoning, stands out, substantially lifting the Avg. JGA from 59.0% to 68.7%, with a remarkable FWT improvement from 25.5% to 51.9%.

Figure 5.5 shows the models' ability to mitigate catastrophic forgetting by assessing their performance on the initial task post successive task learnings. Models with RoS reasoning skills exhibit a slower forgetting rate, with a 15% average drop in performance in both T5-small and LLaMA-7B. In contrast, vanilla backbone models show a steeper decline, averaging a 28% performance dip, highlighting the importance of domain-agnostic reasoning skills in sustaining historical task performance.

RoS Distillation Exhibits Strong Generalization Ability Our method significantly improves the FWT metric, indicative of robust generalization and zero-shot learning capabilities. It bridges the distribution gap across domains by creating a systematic reasoning chain, evident in handling semantically similar but differently described slots, like ¡services-appointment_date¿ and ¡hotels-check_in_date¿. Unlike traditional models, ours utilizes reasoning to recognize similarities, enhancing adaptability to unseen slots.

To further evaluate this generalization, we introduce a new 16th task using MultiWOZ 2.4 data [165]. The zero-shot performance on this task follows the sequential training

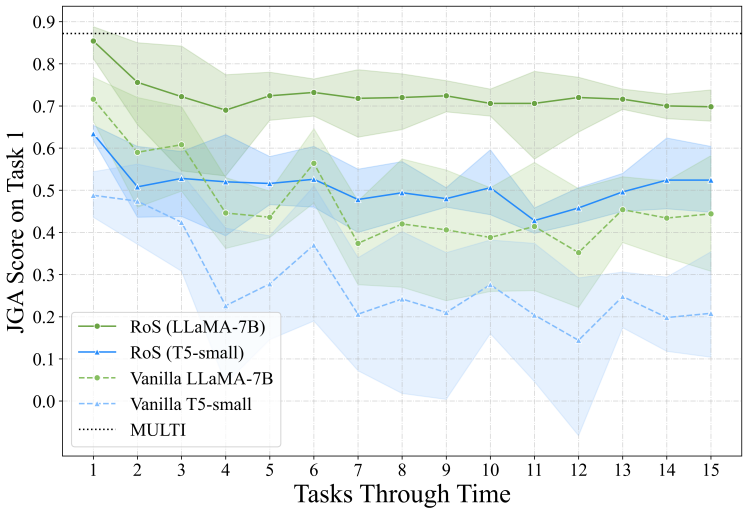


Figure 5.5: Task 1 performance trajectory during continual DST learning process.

Method	Backbone	Attraction	Restaurant	Train	Average
CPT	*T5-base*	10.05	19.37	3.34	10.92
Fine-tune	*T5-small*	8.40	12.87	2.94	8.07
	LLaMA-7B	40.47	52.73	18.75	37.32
RoS	*T5-small*	**15.74**	**24.07**	**4.05**	**14.62**
	LLaMA-7B	**42.84**	**59.94**	**25.57**	**42.78**

Table 5.2: Zero-Shot performance on the 16th cross-dataset task using the MultiWOZ 2.4 dataset.

of the first 15 tasks from the SGD, as shown in Table 5.2.

Compared to similar parameter-sized backbones, RoS (T5-small) outperforms the T5-base CPT baseline, increasing average JGA by 3.7%, from 10.92% to 14.62% across domains. This is particularly notable in the Attraction domain, with JGA rising from 10.05% to 15.74%. Compared to various vanilla backbones, the improvement due to domain-agnostic reasoning averages a 6% increase, further confirming RoS's robust generalization.

To assess the hallucination rate of the teacher model, we utilized SelfCheckGPT [91] and employed BERTScore for evaluation. BERTScore provides scores ranging from

Teacher Model	Initial BERTScore (%)	BERTScore After Contrastive Selection (%)
ChatGPT	19.3	14.2
LLaMA-2-70B	21.7	15.9

Table 5.3: Hallucination rate of the reasoning generated by teacher models after our contrastive selection method.

[0.0, 1.0], where higher values indicate a greater likelihood of non-factual content, implying an increased probability of hallucination. Our evaluation included the average BERTScore for the initial generation of five candidate reasonings by the teacher models, the BERTScore after applying the Semantic Contrastive Reasoning Selection method, and the hallucination rate of the distilled student. The summarized results are presented in the Table 5.3.

Table 5.3 illustrates that for both teacher models, ChatGPT and LLaMA-2-70B, the average BERTScore for the initial five reasoning candidates were 19.3% and 21.7%, respectively. After applying the Semantic Contrastive Reasoning Selection method, the BERTScore decreased to 14.2% and 15.9%, affirming the effectiveness of our proposed approach in enhancing the factual of reasoning. Regarding the student models, there is no significant difference in BERTScore when distilled with different teacher models. Additionally, we observed that the quality of reasoning generated by the student models improves as the model size increases.

5.4.3 Ablation Study

The Role of Model Size Variation in Teacher and Student Models Our study investigates the effect of teacher and student model sizes on performance. Figure 5.6 shows a clear trend: larger student models, like LLaMA-7B, outperform smaller ones, such as T5-small. We first evaluate LLaMA-7B's continual learning in

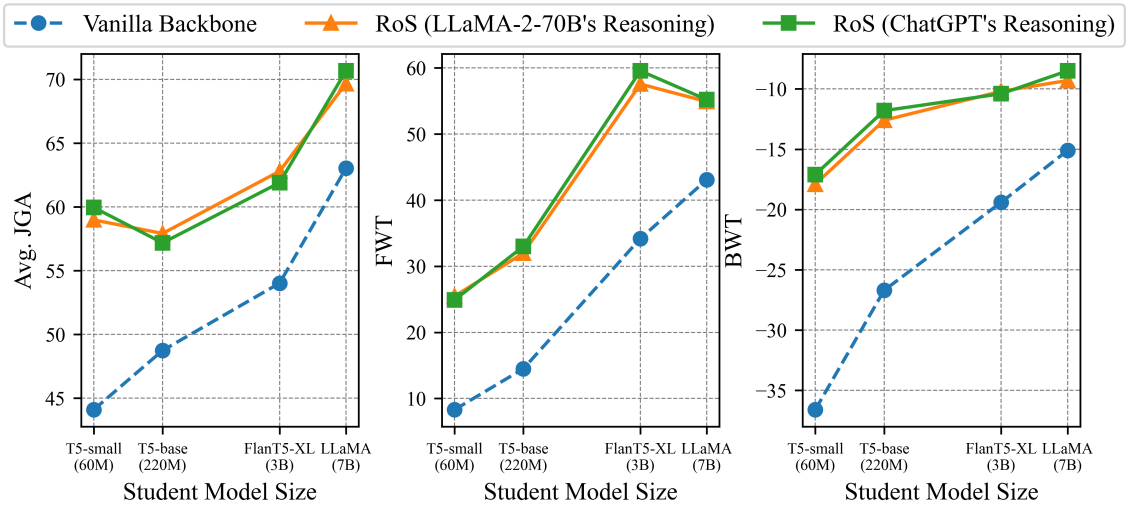


Figure 5.6: Comparative task performance across varying sizes of teacher and student models.

DST, finding it outperforms most T5-small or T5-base-based methods. However, applying our RoS method with reasoning is the most notable enhancement, especially in smaller models. For example, integrating reasoning into T5-small yields an impressive 15% average increase in all evaluation metrics. Contrary to our initial assumption that students guided by ChatGPT might surpass those instructed by LLaMA-2-70B, the results indicate equal reasoning quality from both teacher models, suggesting their similar effectiveness in enhancing student model learning.

Effectiveness of Semantic Contrastive Reasoning Selection Method To assess our method's ability to reduce reasoning hallucinations, we compared three reasoning selection strategies: random selection from G candidates and selecting the highest and lowest scoring (ours) reasonings, as per Equation 2. The results are detailed in Table 5.4.

Initial results show that all methods significantly outperform the vanilla LLaMA, with even random selection improving Avg. JGA by 4.94%. This highlights the inherent reasoning quality in current teacher models. Yet, refining the selection to eliminate

Models	Avg. JGA	FWT	BWT
LLaMA-7B (backbone)	63.04	43.10	-15.10
+ Max Score	65.72	49.05	-13.34
+ Random Selection	66.98	51.14	-11.73
+ Min Score (Ours)	**70.17**	**52.06**	**-8.90**

Table 5.4: Ablation study on the effectiveness of strategies to alleviate hallucination in teacher model's reasoning.

hallucinatory reasonings and aggregate higher-quality reasonings further enhanced performance. Compared to random selection, our method shows additional gains of 3.2%, 0.9%, and 2.8% in the evaluated metrics. In contrast, selecting the highest-scoring result, i.e., favoring less effective reasoning, resulted in reduced performance. These findings robustly affirm the effectiveness of our semantic similarity-based selection approach.

5.4.4 Case Study

Table 5.5 presents a key test case and various methods' predictions to showcase student models' advanced reasoning capabilities. The dialogue involves two values for the ¡pickup_time¿ slot: "17:15" and a later update to "half past 1 in the afternoon". The vanilla LLaMA model incorrectly selects "17:15", while our model effectively identifies all possible values and accurately determines the correct one. Interestingly, our RoS T5-small model's output of "1:30 pm" mirrors the ground truth but is marked incorrect due to current evaluation criteria. This highlights a challenge in matching model outputs with expected formats, suggesting a direction for future refinement.

Dialogue Context	[Dialogue from previous turns omitted …] [SYSTEM]: What time would you like to pick the car up at? [USER]: I'd need it for 17:15. [SYSTEM]: I have 6 possible options for you - Fresno station has a Standard available for pick up on March 11th, it is a Chevrolet Cruze. [USER]: What else is available? I'd like something small - and I think I'd actually be picking it up at half past 1 in the afternoon, instead. [SYSTEM]: I have 4 options for you: …
Golden State	(rentalcars_2-pickup_time, half past 1 in the afternoon)
Vanilla LLaMA-7B	(rentalcars_2-pickup_time, 17:15) ✗
RoS (T5-small)	Reasoning process: The dialogue context indicates that the user is scheduling a rental car pickup. The user specifies a pickup time of 17:15, but later changes their mind and requests a pickup time of 1:30 pm. The system confirms the change and provides the user with options for pickup at 1:30 pm. Therefore, the most appropriate value for the slot <rentalcars-pickup_time> is 1:30 pm, as it aligns with the user's revised request and the system's confirmation. Result: 1:30 pm ✗
RoS (LLaMA-7B)	Reasoning process: The slot <rentalcars-pickup_time> has multiple possible values, but the most appropriate value is 'half past 1 in the afternoon' based on the dialogue context and user intent. The user initially mentions they'd like to pick up the car at 17:15, but later corrects themselves and confirms they'd like to pick it up at 'half past 1 in the afternoon. This selection is further supported by the system's confirmation question, which asks the user to review their reservation details, including the pickup time, and the user's confirmation of the reservation Result: half past 1 in the afternoon ✓

Table 5.5: Exemplary dialogue from the SGD test set with two possible values, with predictions by the vanilla LLaMA-7B and our meta-reasoning augmented method applied to two distinct backbones.

5.5 Chapter Review

In this chapter, we introduced the Reason-of-Select (RoS) distillation method to tackle the challenges of catastrophic forgetting and the Value Selection Quandary in continual dialogue state tracking (DST). RoS enhances smaller student models with a domain-agnostic meta-reasoning capability, expanding their reasoning capacity and aligning knowledge across shifting data distributions in different domains. Two complementary strategies—the multi-value resolution mechanism and Semantic Contrastive Reasoning Selection—further strengthen RoS by ensuring reliable knowledge transfer and improving model robustness. Extensive experiments demonstrate that RoS achieves superior performance and generalization compared to conventional approaches.

Computational considerations. Unlike standard knowledge distillation, RoS requires additional resources during data preparation and student model training. The Contrastive Reasoning Selection component generates multiple candidate reasonings, along with various value-level and slot-level perturbations, which increases preparation time relative to conventional rationale distillation. Moreover, training the student model with reasoning introduces slight additional computational overhead. However, these costs are one-time and do not impact inference: once trained, the student model can reason efficiently without additional prediction time, and the overhead for generating reasoning during testing is negligible.

Overall, RoS provides a robust and scalable framework for continual DST, offering improved reasoning capabilities, reliable knowledge transfer, and strong generalization while maintaining practical efficiency at inference.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

This thesis studies **knowledge adaptation in large language models (LLMs)** under a unified perspective of enabling models to adapt continuously while preserving reliability and stability. Rather than treating continual learning and model editing as isolated problems, this work positions them as complementary mechanisms operating at different granularities of adaptation.

From a temporal perspective, continual learning (CL) addresses adaptation across evolving tasks. In Chapter 3, we developed a progressively refined framework for continual knowledge integration, consisting of Knowledge Identification and Fusion (KIF), Recurrent KIF, and Adaptive Iterative Model Merging (AIMMerging). These methods evolve from static importance-aware fusion, to dynamic re-estimation of parameter importance, and finally to signal-driven adaptive merging based on learning and forgetting dynamics. Together, they provide a scalable solution for mitigating catastrophic forgetting and enabling stable task-level adaptation in large-scale LLMs.

From a granularity perspective, model editing addresses fine-grained knowledge up-

dates within a fixed task distribution. In Chapter 4, we proposed Geometric Knowledge Editing (GeoEdit), which analyzes the geometric structure of fine-tuning trajectories to distinguish neurons associated with new knowledge from those responsible for general representations. By selectively constraining updates to general-knowledge neurons while optimizing new-knowledge neurons, GeoEdit enables precise factual updates with minimal unintended interference. This neuron-level geometric control complements the task-level adaptation mechanisms developed in continual learning.

Importantly, these two lines of research are not independent. Continual learning ensures stability across sequential tasks, while model editing provides precision in localized knowledge updates. Together, they form a multi-scale knowledge adaptation framework: task-level consolidation maintains long-term competence, whereas neuron-level modulation ensures fine-grained reliability. This unified perspective advances the goal of building adaptive and evolving LLMs that can update, expand, and refine their knowledge without sacrificing previously acquired capabilities.

To validate the practical implications of this framework, Chapter 5 applied the proposed adaptation principles to Dialogue State Tracking (DST), a representative real-world scenario characterized by domain expansion and evolving user requirements. The continual DST framework demonstrates how task-level consolidation and structured knowledge updating jointly enable robust generalization and stable performance across dynamic dialogue environments.

Overall, this thesis contributes both methodological innovations and a unified conceptual framework for knowledge adaptation in LLMs. By bridging continual task learning and fine-grained knowledge editing, it provides scalable solutions for building reliable, adaptable, and evolving large language models suitable for dynamic real-world deployment.

6.2 Future Works

Building on the contributions of this thesis, future research will focus on extending knowledge adaptation strategies specifically for Large Language Models (LLMs), addressing their scale, dynamic data requirements, and computational challenges. The goal is to enable LLMs to continuously learn, preserve general knowledge, and adapt efficiently to new tasks.

6.2.1 Continual Learning for Large Language Models

A promising direction is the development of continual learning techniques tailored to LLMs. While LLMs are pretrained on large static corpora, real-world applications often require adaptation to streaming and evolving data. Retraining from scratch is computationally expensive and impractical.

Future research may explore scalable CL strategies, including:

- **Dynamic parameter importance estimation:** Extending methods like Recurrent KIF to continuously evaluate the importance of model parameters for knowledge retention in evolving tasks.
- **Adaptive model merging:** Leveraging learning and forgetting signals to determine optimal merging points for integrating new task knowledge without catastrophic forgetting.
- **Memory-aware training:** Maintaining a curated set of representative samples from past tasks as a dynamic memory buffer to facilitate efficient continual learning and knowledge preservation.

These strategies aim to enable LLMs to retain past knowledge while efficiently acquiring new information, improving adaptability in dynamic and complex environments.

6.2.2 Model Editing for Large Language Models

Another key research direction is advancing model editing techniques for LLMs. Current approaches, such as GeoEdit, selectively update parameters to integrate new knowledge while minimizing interference with general knowledge. However, challenges remain when old knowledge datasets are unavailable or when new knowledge conflicts with existing general knowledge.

Future work may explore:

- **Synthetic or inferred task vectors:** Using model outputs or generated representations as substitutes for unavailable reference datasets.
- **Multi-factor geometric analysis:** Incorporating additional geometric metrics, such as task vector projections and magnitude, to more effectively separate general knowledge preservation from new knowledge integration.
- **Scalable mass-editing:** Extending editing methods to handle large-scale, multi-task updates efficiently while minimizing inference overhead.

These directions aim to make LLMs more flexible and robust, ensuring that new knowledge can be integrated effectively without degrading previously learned general knowledge.

References

[1] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*, 2020.

[2] Anton Alexandrov, Veselin Raychev, Mark Mueller, Ce Zhang, Martin Vechev, and Kristina Toutanova. Mitigating catastrophic forgetting in language transfer via model merging. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 17167–17186, 2024.

[3] Baolong Bi, Shenghua Liu, Lingrui Mei, Yiwei Wang, Pengliang Ji, and Xueqi Cheng. Decoding by contrasting knowledge: Enhancing llms' confidence on edited facts. *ACL 2025*, 2024.

[4] Baolong Bi, Shenghua Liu, Yiwei Wang, Lingrui Mei, Hongcheng Gao, Junfeng Fang, and Xueqi Cheng. Struedit: Structured outputs enable the fast and accurate knowledge editing for large language models. *arXiv preprint arXiv:2409.10132*, 2024.

[5] Baolong Bi, Shenghua Liu, Yiwei Wang, Lingrui Mei, Hongcheng Gao, Yilong Xu, and Xueqi Cheng. Adaptive token biaser: Knowledge editing via biasing key entities. *EMNLP 2024*, 2024.

[6] 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.

[7] Yihan Cao, Siyu Li, Yixin Liu, Zhiling Yan, Yutong Dai, Philip S Yu, and Lichao Sun. A comprehensive survey of ai-generated content (aigc): A history of generative ai from gan to chatgpt. *arXiv preprint arXiv:2303.04226*, 2023.

[8] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

[9] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–547, 2018.

[10] Derek Chen, Kun Qian, and Zhou Yu. Stabilized in-context learning with pre-trained language models for few shot dialogue state tracking. *arXiv preprint arXiv:2302.05932*, 2023.

[11] Shengyuan Chen, Qinggang Zhang, Junnan Dong, Wen Hua, Qing Li, and Xiao Huang. Entity alignment with noisy annotations from large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[12] Wuyang Chen, Yanqi Zhou, Nan Du, Yanping Huang, James Laudon, Zhifeng Chen, and Claire Cui. Lifelong language pretraining with distribution-specialized experts. In *International Conference on Machine Learning*, pages 5383–5395. PMLR, 2023.

[13] Feng Cheng, Ziyang Wang, Yi-Lin Sung, Yan-Bo Lin, Mohit Bansal, and Gedas

Bertasius. Dam: Dynamic adapter merging for continual video qa learning. *arXiv preprint arXiv:2403.08755*, 2024.

[14] Hyundong Cho, Andrea Madotto, Zhaojiang Lin, Khyathi Raghavi Chandu, Satwik Kottur, Jing Xu, Jonathan May, and Chinnadhurai Sankar. Continual dialogue state tracking via example-guided question answering. *arXiv preprint arXiv:2305.13721*, 2023.

[15] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

[16] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[17] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1932–1945, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.106. URL `https://aclanthology.org/2024.acl-long.106`.

[18] Wenyu Du, Shuang Cheng, Tongxu Luo, Zihan Qiu, Zeyu Huang, Ka Chun Cheung, Reynold Cheng, and Jie Fu. Unlocking continual learning abilities in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6503–6522, 2024.

[19] Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lath-

uilière. Weighted ensemble models are strong continual learners. *arXiv e-prints*, pages arXiv–2312, 2023.

[20] Masih Eskandar, Tooba Imtiaz, Davin Hill, Zifeng Wang, and Jennifer Dy. Star: Stability-inducing weight perturbation for continual learning. *arXiv preprint arXiv:2503.01595*, 2025.

[21] Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355*, 2024.

[22] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

[23] Yue Feng, Aldo Lipani, Fanghua Ye, Qiang Zhang, and Emine Yilmaz. Dynamic schema graph fusion network for multi-domain dialogue state tracking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 115–126, 2022.

[24] Yujie Feng, Jiangtao Wang, Yasha Wang, and Sumi Helal. Completing missing prevalence rates for multiple chronic diseases by jointly leveraging both intra- and inter-disease population health data correlations. In *Proceedings of the Web Conference 2021*, pages 183–193, 2021.

[25] Yujie Feng, Jiangtao Wang, Yasha Wang, and Xu Chu. Spatial-attention and demographic-augmented generative adversarial imputation network for population health data reconstruction. *IEEE Transactions on Big Data*, 2022.

[26] Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. Towards llm-driven dialogue state tracking. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 739–755, 2023.

[27] Yujie Feng, Xu Chu, Yongxin Xu, Zexin Lu, Bo Liu, Philip S Yu, and Xiao-Ming Wu. Kif: Knowledge identification and fusion for language model continual learning. *arXiv preprint arXiv:2408.05200*, 2024.

[28] Yujie Feng, Xu Chu, Yongxin Xu, Guangyuan Shi, Bo Liu, and Xiao-Ming Wu. Tasl: Continual dialog state tracking via task skill localization and consolidation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1266–1279, 2024.

[29] Yujie Feng, Xujia Wang, Zexin Lu, Shenghong Fu, Guangyuan Shi, Yongxin Xu, Yasha Wang, Philip S Yu, Xu Chu, and Xiao-Ming Wu. Recurrent knowledge identification and fusion for language model continual learning. *arXiv preprint arXiv:2502.17510*, 2025.

[30] Yujie Feng, Liming Zhan, Zexin Lu, Yongxin Xu, Xu Chu, Yasha Wang, Jiannong Cao, Philip S Yu, and Xiao-Ming Wu. Geoedit: Geometric knowledge editing for large language models. *arXiv preprint arXiv:2502.19953*, 2025.

[31] Yujie Feng, Hao Wang, Jian Li, Xu Chu, Zhaolu Kang, Yiran Liu, Yasha Wang, Philip S Yu, and Xiao-Ming Wu. Forever: Forgetting curve-inspired memory replay for language model continual learning. *arXiv preprint arXiv:2601.03938*, 2026.

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

[33] Govind Krishnan Gangadhar and Karl Stratos. Model editing by standard fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5907–5913, 2024.

[34] Binzong Geng, Fajie Yuan, Qiancheng Xu, Ying Shen, Ruifeng Xu, and Min Yang. Continual learning for task-oriented dialogue system with iterative network pruning, expanding and masking. *arXiv preprint arXiv:2107.08173*, 2021.

[35] Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16801–16819, 2024.

[36] Jinyu Guo, Kai Shuang, Jijie Li, Zihan Wang, and Yixuan Liu. Beyond the granularity: Multi-perspective dialogue collaborative selection for dialogue state tracking. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2320–2332, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.165. URL `https://aclanthology.org/2022.acl-long.165`.

[37] Akshat Gupta, Dev Sajnani, and Gopala Anumanchipalli. A unified framework for model editing. *arXiv preprint arXiv:2403.14236*, 2024.

[38] Jinghan He, Haiyun Guo, Kuan Zhu, Zihan Zhao, Ming Tang, and Jinqiao Wang. Seekr: Selective attention-guided knowledge retention for continual learning of large language models. *arXiv preprint arXiv:2411.06171*, 2024.

[39] Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, and Milica Gašić. Chatgpt for zero-shot dialogue state tracking: A solution or an opportunity? *arXiv preprint arXiv:2306.01386*, 2023.

[40] Yihuai Hong and Aldo Lipani. Interpretability-based tailored knowledge editing in transformers. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3847–3858, 2024.

[41] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the*

IEEE/CVF conference on computer vision and pattern recognition, pages 831–839, 2019.

[42] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[43] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

[44] Cheng-Hsun Hsueh, Paul Kuo-Ming Huang, Tzu-Han Lin, Che-Wei Liao, Hung-Chieh Fang, Chao-Wei Huang, and Yun-Nung Chen. Editing the mind of giants: An in-depth exploration of pitfalls of knowledge editing in large language models. *arXiv preprint arXiv:2406.01436*, 2024.

[45] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[46] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[47] Zhiyuan Hu, Yue Feng, Yang Deng, Zekun Li, See-Kiong Ng, Anh Tuan Luu, and Bryan Hooi. Enhancing large language model induced task-oriented dialogue systems through look-forward motivated goals. *arXiv preprint arXiv:2309.08949*, 2023.

[48] Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. Mitigating catastrophic forget-

ting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244*, 2024.

[49] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32, 2020.

[50] Xiusheng Huang, Yequan Wang, Jun Zhao, and Kang Liu. Commonsense knowledge editing based on free-text in llms. *arXiv preprint arXiv:2410.23844*, 2024.

[51] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[52] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=6t0Kwf8-jrj`.

[53] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[54] Gangwei Jiang, Caigao Jiang, Zhaoyi Li, Siqiao Xue, Jun Zhou, Linqi Song, Defu Lian, and Ying Wei. Interpretable catastrophic forgetting of large language model fine-tuning via instruction vector. *arXiv preprint arXiv:2406.12227*, 2024.

[55] Xinke Jiang, Ruizhe Zhang, Yongxin Xu, Rihong Qiu, Yue Fang, Zhiyuan Wang, Jinyi Tang, Hongxin Ding, Xu Chu, Junfeng Zhao, et al. Hykge: A hypothesis knowledge graph enhanced framework for accurate and reliable medical llms responses. *arXiv preprint arXiv:2312.15883*, 2023.

[56] Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, et al. Learning to edit: Aligning llms with knowledge editing. *arXiv preprint arXiv:2402.11905*, 2024.

[57] Ying Jin, Zhangjie Cao, Ximei Wang, Jianmin Wang, and Mingsheng Long. One fits many: Class confusion loss for versatile domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[58] Zixuan Ke and Bing Liu. Continual learning of natural language processing tasks: A survey. *arXiv preprint arXiv:2211.12701*, 2022.

[59] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456, 2021.

[60] Zixuan Ke, Bing Liu, Wenhan Xiong, Asli Celikyilmaz, and Haoran Li. Subnetwork discovery and soft-masking for continual learning of mixed tasks. *arXiv preprint arXiv:2310.09436*, 2023.

[61] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[62] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[63] Tatsuya Konishi, Mori Kurokawa, Chihiro Ono, Zixuan Ke, Gyuhak Kim, and Bing Liu. Parameter-Level Soft-Masking for Continual Learning. In *Proc. of ICML*, 2023.

[64] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[65] Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. Dialogue state tracking with a language model using schema-driven prompting. *arXiv preprint arXiv:2109.07506*, 2021.

[66] Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. Sumbt: Slot-utterance matching for universal and scalable belief tracking. *arXiv preprint arXiv:1907.07421*, 2019.

[67] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*, 2017.

[68] Depeng Li and Zhigang Zeng. Crnet: A fast continual learning framework with random theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9):10731–10744, 2023.

[69] Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. Revisiting catastrophic forgetting in large language model tuning. *arXiv preprint arXiv:2406.04836*, 2024.

[70] Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. *arXiv preprint arXiv:2306.14050*, 2023.

[71] Qi Li and Xiaowen Chu. Can we continually edit language models? on the knowledge attenuation in sequential model editing. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5438–5455, 2024.

[72] Shuaiyi Li, Yang Deng, Deng Cai, Hongyuan Lu, Liang Chen, and Wai Lam. Consecutive batch model editing with hook layers. In *Proceedings of the 2024*

Conference on Empirical Methods in Natural Language Processing, pages 13817–13833, 2024.

[73] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[74] Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*, 2022.

[75] Xiaorong Li, Shipeng Wang, Jian Sun, and Zongben Xu. Variational data-free knowledge distillation for continual learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12618–12634, 2023.

[76] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[77] Huanxuan Liao, Shizhu He, Yupu Hao, Jun Zhao, and Kang Liu. Data: Decomposed attention-based task adaptation for rehearsal-free continual learning. *arXiv preprint arXiv:2502.11482*, 2025.

[78] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[79] Bo Liu, Liming Zhan, Zexin Lu, Yujie Feng, Lei Xue, and Xiao-Ming Wu. How good are large language models at out-of-distribution detection? *arXiv preprint arXiv:2308.10261*, 2023.

[80] Jiateng Liu, Pengfei Yu, Yuji Zhang, Sha Li, Zixuan Zhang, Ruhi Sarikaya, Kevin Small, and Heng Ji. Evedit: Event-based knowledge editing for deterministic knowledge propagation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4907–4926, 2024.

[81] Qingbin Liu, Pengfei Cao, Cao Liu, Jiansong Chen, Xunliang Cai, Fan Yang, Shizhu He, Kang Liu, and Jun Zhao. Domain-lifelong learning for dialogue state tracking via knowledge preservation networks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2301–2311, 2021.

[82] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

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

[84] Zexin Lu, Keyang Ding, Yuji Zhang, Jing Li, Baolin Peng, and Lemao Liu. Engage the public: Poll question generation for social media posts. In *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)*, pages 29–40, 2021.

[85] Zexin Lu, Jing Li, Yingyi Zhang, and Haisong Zhang. Getting your conversation on track: Estimation of residual life for conversations. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 1036–1043. IEEE, 2021.

[86] Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv preprint arXiv:2402.12851*, 2024.

[87] Mingyu Derek Ma, Jiun-Yu Kao, Shuyang Gao, Arpit Gupta, Di Jin, Tagyoung Chung, and Nanyun Peng. Parameter-efficient low-resource dialogue state tracking by prompt tuning. *arXiv preprint arXiv:2301.10915*, 2023.

[88] Xinbei Ma, Tianjie Ju, Jiyang Qiu, Zhuosheng Zhang, Hai Zhao, Lifeng Liu, and Yulong Wang. On the robustness of editing large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16197–16216, 2024.

[89] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.

[90] Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, and Zhiguang Wang. Continual learning in task-oriented dialogue systems. *arXiv preprint arXiv:2012.15504*, 2020.

[91] Potsawee Manakul, Adian Liusie, and Mark JF Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.

[92] Jarana Manotumruksa, Jeffrey Dalton, Edgar Meij, and Emine Yilmaz. Similarity-based multi-domain dialogue state tracking with copy mechanisms for task-based virtual personal assistants. In *Proceedings of the ACM Web Conference 2022*, pages 2006–2014, 2022.

[93] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661*, 2020.

[94] 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.

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

[96] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[97] Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations*, 2023.

[98] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022.

[99] Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, and Erik Cambria. Recent advances in deep learning based dialogue systems: A systematic survey. *Artificial intelligence review*, 56(4):3055–3155, 2023.

[100] Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5716–5731, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.310. URL `https://aclanthology.org/2024.acl-long.310/`.

[101] Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. Task-specific skill localization in fine-tuned language models. In *International Conference on Machine Learning*, pages 27011–27033. PMLR, 2023.

[102] Quang Pham, Chenghao Liu, and Steven CH Hoi. Continual learning, fast and slow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[103] Chengwei Qin and Shafiq Joty. Lfpt5: A unified framework for lifelong few-shot language learning based on prompt tuning of t5. *arXiv preprint arXiv:2110.07298*, 2021.

[104] Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S Yu. Large language models meet nlp: A survey. *arXiv preprint arXiv:2405.12819*, 2024.

[105] Gao Qixiang, Guanting Dong, Yutao Mou, Liwen Wang, Chen Zeng, Daichi Guo, Mingyang Sun, and Weiran Xu. Exploiting domain-slot related keywords description for few-shot cross-domain dialogue state tracking. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2460–2465, 2022.

[106] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[107] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696, 2020.

[108] Lohith Ravuru, Seonghan Ryu, Hyungtak Choi, Haehun Yang, and Hyeonmok Ko. Multi-domain dialogue state tracking by neural-retrieval augmentation. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 169–175, 2022.

[109] Anastasia Razdai, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations*, 2022.

[110] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[111] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[112] Weijieying Ren, Xinlong Li, Lei Wang, Tianxiang Zhao, and Wei Qin. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. *arXiv preprint arXiv:2402.18865*, 2024.

[113] Jorma J Rissanen. Fisher information and stochastic complexity. *IEEE transactions on information theory*, 42(1):40–47, 1996.

[114] Grzegorz Rypeść, Sebastian Cygert, Valeriya Khan, Tomasz Trzciński, Bartosz Zieliński, and Bartłomiej Twardowski. Divide and not forget: Ensemble of selectively trained experts in continual learning. *arXiv preprint arXiv:2401.10191*, 2024.

[115] Yilin Shen, Xiangyu Zeng, and Hongxia Jin. A progressive model to enable continual learning for semantic slot filling. 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)*, pages 1279–1284, 2019.

[116] Chen Shengyuan, Yunfeng Cai, Huang Fang, Xiao Huang, and Mingming Sun.

Differentiable neuro-symbolic reasoning on large-scale knowledge graphs. *Advances in Neural Information Processing Systems*, 36, 2023.

[117] Guangyuan Shi, Zexin Lu, Xiaoyu Dong, Wenlong Zhang, Xuanyu Zhang, Yujie Feng, and Xiao-Ming Wu. Understanding layer significance in llm alignment. *arXiv preprint arXiv:2410.17875*, 2024.

[118] Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. Lamol: Language modeling for lifelong language learning. *arXiv preprint arXiv:1909.03329*, 2019.

[119] Zhoujian Sun, Zhengxing Huang, and Nai Ding. On tracking dialogue state by inheriting slot values in mentioned slot pools. *arXiv preprint arXiv:2202.07156*, 2022.

[120] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[121] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7, 2023.

[122] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[123] Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. *arXiv preprint arXiv:2401.10491*, 2024.

[124] Alex Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[125] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

[126] Haoming Wang and Wang Xin. How to stop an avalanche? jodem: Joint decision making through compare and contrast for dialog state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7030–7041, 2022.

[127] Haoyu Wang, Tianci Liu, Ruirui Li, Monica Cheng, Tuo Zhao, and Jing Gao. Roselora: Row and column-wise sparse low-rank adaptation of pre-trained language model for knowledge editing and fine-tuning. *arXiv preprint arXiv:2406.10777*, 2024.

[128] Huazheng Wang, Haifeng Sun, Jingyu Wang, Qi Qi, Zixuan Xia, Menghao Zhang, and Jianxin Liao. Sss: Editing factual knowledge in language models towards semantic sparse space. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5559–5570, 2024.

[129] Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. Self-expansion of pre-trained models with mixture of adapters for continual learning. *arXiv preprint arXiv:2403.18886*, 2024.

[130] Jiaan Wang, Yunlong Liang, Zengkui Sun, Yuxuan Cao, Jiarong Xu, and Fandong Meng. Cross-lingual knowledge editing in large language models. *arXiv preprint arXiv:2309.08952*, 2023.

[131] Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao,

Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. Detoxifying large language models via knowledge editing. *arXiv preprint arXiv:2403.14472*, 2024.

[132] Mingyang Wang, Heike Adel, Lukas Lange, Jannik Strötgen, and Hinrich Schütze. Rehearsal-free modular and compositional continual learning for language models. *arXiv preprint arXiv:2404.00790*, 2024.

[133] Mingyang Wang, Lukas Lange, Heike Adel, Jannik Strötgen, and Hinrich Schütze. Better call saul: Fluent and consistent language model editing with generation regularization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7990–8000, 2024.

[134] Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. Scott: Self-consistent chain-of-thought distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5546–5558, 2023.

[135] Qingyue Wang, Yanan Cao, Piji Li, Yanhe Fu, Zheng Lin, and Li Guo. Slot dependency modeling for zero-shot cross-domain dialogue state tracking. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 510–520, 2022.

[136] Renzhi Wang and Piji Li. Lemoe: Advanced mixture of experts adaptor for lifelong model editing of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2551–2575, 2024.

[137] Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):1–37, 2024.

[138] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng,

Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. *arXiv preprint arXiv:2310.14152*, 2023.

[139] Xiaohan Wang, Shengyu Mao, Ningyu Zhang, Shumin Deng, Yunzhi Yao, Yue Shen, Lei Liang, Jinjie Gu, and Huajun Chen. Editing conceptual knowledge for large language models. *arXiv preprint arXiv:2403.06259*, 2024.

[140] Yifan Wang, Jing Zhao, Junwei Bao, Chaoqun Duan, Youzheng Wu, and Xiaodong He. LUNA: Learning slot-turn alignment for dialogue state tracking. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3319–3328, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.242. URL `https://aclanthology.org/2022.naacl-main.242`.

[141] Yifan Wang, Yafei Liu, Chufan Shi, Haoling Li, Chen Chen, Haonan Lu, and Yujiu Yang. Inscl: A data-efficient continual learning paradigm for fine-tuning large language models with instructions. *arXiv preprint arXiv:2403.11435*, 2024.

[142] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

[143] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*, 2022.

[144] Zhicheng Wang, Yufang Liu, Tao Ji, Xiaoling Wang, Yuanbin Wu, Congcong Jiang, Ye Chao, Zhencong Han, Ling Wang, Xu Shao, et al. Rehearsal-free continual language learning via efficient parameter isolation. In *Proceedings*

of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10933–10946, 2023.

[145] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 139–149, 2022.

[146] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR, 2022.

[147] Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1078. URL `https://aclanthology.org/P19-1078`.

[148] Junhong Wu, Yuchen Liu, and Chengqing Zong. F-malloc: Feed-forward memory allocation for continual learning in neural machine translation. *arXiv preprint arXiv:2404.04846*, 2024.

[149] Yichen Wu, Long-Kai Huang, Renzhen Wang, Deyu Meng, and Ying Wei. Meta continual learning revisited: Implicitly enhancing online hessian approximation via variance reduction. In *The Twelfth International Conference on Learning Representations*, 2024.

[150] Hongyan Xie, Haoxiang Su, Shuangyong Song, Hao Huang, Bo Zou, Kun Deng,

Jianghua Lin, Zhihui Zhang, and Xiaodong He. Correctable-dst: Mitigating historical context mismatch between training and inference for improved dialogue state tracking. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 876–889, 2022.

[151] Jing Xu, Dandan Song, Chong Liu, Siu Cheung Hui, Fei Li, Qiang Ju, Xiaonan He, and Jian Xie. Dialogue state distillation network with inter-slot contrastive learning for dialogue state tracking. *arXiv preprint arXiv:2302.08220*, 2023.

[152] Ju Xu, Jin Ma, Xuesong Gao, and Zhanxing Zhu. Adaptive progressive continual learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):6715–6728, 2021.

[153] Yongxin Xu, Xu Chu, Kai Yang, Zhiyuan Wang, Peinie Zou, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. Seqcare: Sequential training with external medical knowledge graph for diagnosis prediction in healthcare data. In *Proceedings of the ACM Web Conference 2023*, pages 2819–2830, 2023.

[154] Yongxin Xu, Kai Yang, Chaohe Zhang, Peinie Zou, Zhiyuan Wang, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. Vecocare: visit sequences-clinical notes joint learning for diagnosis prediction in healthcare data. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 4921–4929, 2023.

[155] Yongxin Xu, Ruizhe Zhang, Xinke Jiang, Yujie Feng, Yuzhen Xiao, Xinyu Ma, Runchuan Zhu, Xu Chu, Junfeng Zhao, and Yasha Wang. Parenting: Optimizing knowledge selection of retrieval-augmented language models with parameter decoupling and tailored tuning. *arXiv preprint arXiv:2410.10360*, 2024.

[156] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023. URL `https://arxiv.org/abs/2306.01708`.

[157] Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Leshem Choshen, and Alessandro Sordoni. A survey on model moerging: Recycling and routing among specialized experts for collaborative learning. *arXiv preprint arXiv:2408.07057*, 2024.

[158] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[159] Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[160] Kai Yang, Yongxin Xu, Peinie Zou, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. Kerprint: local-global knowledge graph enhanced diagnosis prediction for retrospective and prospective interpretations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5357–5365, 2023.

[161] Puhai Yang, Heyan Huang, Wei Wei, and Xian-Ling Mao. Toward real-life dialogue state tracking involving negative feedback utterances. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2222–2232, 2022.

[162] Xianjun Yang, Yan Li, Xinlu Zhang, Haifeng Chen, and Wei Cheng. Exploring the limits of chatgpt for query or aspect-based text summarization. *arXiv preprint arXiv:2302.08081*, 2023.

[163] Yuting Yang, Wenqiang Lei, Pei Huang, Juan Cao, Jintao Li, and Tat-Seng Chua. A dual prompt learning framework for few-shot dialogue state tracking. In *Proceedings of the ACM Web Conference 2023*, pages 1468–1477, 2023.

[164] Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. *arXiv preprint arXiv:2305.13172*, 2023.

[165] Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 351–360, Edinburgh, UK, September 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.sigdial-1.34`.

[166] Charles Yu, Sullam Jeoung, Anish Kasi, Pengfei Yu, and Heng Ji. Unlearning bias in language models by partitioning gradients. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6032–6048, 2023.

[167] Dian Yu, Mingqiu Wang, Yuan Cao, Laurent El Shafey, Izhak Shafran, and Hagen Soltau. Knowledge-grounded dialog state tracking. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3428–3435, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.250`.

[168] Dianzhi Yu, Xinni Zhang, Yankai Chen, Aiwei Liu, Yifei Zhang, Philip S Yu, and Irwin King. Recent advances of multimodal continual learning: A comprehensive survey. *arXiv preprint arXiv:2410.05352*, 2024.

[169] Lei Zhang and Xinbo Gao. Transfer adaptation learning: A decade survey, 2020. URL `https://arxiv.org/abs/1903.04687`.

[170] Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.

[171] Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Junnan Dong, Hao Chen, Yi Chang, and Xiao Huang. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*, 2025.

[172] Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. Platon: Pruning large transformer models with upper confidence bound of weight importance. In *International Conference on Machine Learning*, pages 26809–26823. PMLR, 2022.

[173] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023.

[174] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[175] Ruizhe Zhang, Yongxin Xu, Yuzhen Xiao, Runchuan Zhu, Xinke Jiang, Xu Chu, Junfeng Zhao, and Yasha Wang. Knowpo: Knowledge-aware preference optimization for controllable knowledge selection in retrieval-augmented language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25895–25903, 2025.

[176] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

[177] Zhihao Zhang, Jun Zhao, Qi Zhang, Tao Gui, and Xuanjing Huang. Unveiling linguistic regions in large language models. *arXiv preprint arXiv:2402.14700*, 2024.

[178] Haiyan Zhao, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Does continual learning equally forget all parameters? In *International Conference on Machine Learning*, pages 42280–42303. PMLR, 2023.

[179] Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. Sapt: A shared attention framework for parameter-efficient continual learning of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11641–11661, 2024.

[180] Zihao Zhao, Yuchen Yang, Yijiang Li, and Yinzhi Cao. Ripplecot: Amplifying ripple effect of knowledge editing in language models via chain-of-thought in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6337–6347, 2024.

[181] Jiamu Zheng, Jinghuai Zhang, Tianyu Du, Xuhong Zhang, Jianwei Yin, and Tao Lin. Collabedit: Towards non-destructive collaborative knowledge editing. *arXiv preprint arXiv:2410.09508*, 2024.

[182] Yibo Zhong, Jinman Zhao, and Yao Zhou. Low-rank interconnected adaptation across layers. *arXiv preprint arXiv:2407.09946*, 2024.

[183] Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. A comprehensive survey on pre-

trained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*, 2023.

[184] Yihao Zhou, Guoshuai Zhao, and Xueming Qian. Dialogue state tracking based on hierarchical slot attention and contrastive learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4737–4741, 2022.

[185] 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.

[186] Didi Zhu, Zhongyi Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Kun Kuang, and Chao Wu. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. *arXiv preprint arXiv:2402.12048*, 2024.

[187] Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. Continual prompt tuning for dialog state tracking. *arXiv preprint arXiv:2203.06654*, 2022.

[188] Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. Llama-moe: Building mixture-of-experts from llama with continual pre-training. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15913–15923, 2024.