Catastrophic Forgetting in LLMs: A Comparative Analysis Across Language Tasks

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have significantly advanced Natural Language Processing (NLP), particularly in Natural Language Understanding (NLU) tasks. As we progress toward an agentic world where LLM-based agents autonomously handle specialized tasks, it becomes crucial for these models to adapt to new tasks without forgetting previously learned information—a challenge known as catastrophic forgetting. This study evaluates the continual fine-tuning of various open-source LLMs with different parameter sizes (specifically models under 10 billion parameters) on key NLU tasks from the GLUE benchmark, including SST-2, MRPC, CoLA, and MNLI. By employing prompt engineering and task-specific adjustments, we assess and compare the models' abilities to retain prior knowledge while learning new tasks. Our results indicate that models such as **Phi-3.5-mini** exhibit minimal forgetting while maintaining strong learning capabilities, making them well-suited for continual learning environments. Additionally, models like **Orca-2-7b** and **Qwen2.5-7B** demonstrate impressive learning abilities and overall performance after fine-tuning. This work contributes to understanding catastrophic forgetting in LLMs and highlights prompting engineering to optimize model performance for continual learning scenarios.

1 Introduction

Large Language Models (LLMs) (Vaswani et al., 2017) have transformed Natural Language Processing (NLP), delivering state-of-the-art performance on various tasks like sentiment analysis, paraphrase detection, and natural language inference (Bubeck et al., 2023). Open-source models such as **Llama 3** (Llama Team, 2024) have become essential in advancing Natural Language Understanding (NLU) (Wang et al., 2018) tasks, which are critical for interpreting and generating human language.

As we move towards an *agentic world* (Kapoor et al., 2024) where LLM-based agents autonomously handle specialized tasks, the ability to fine-tune models on multiple tasks without losing accuracy or forgetting previously learned information is crucial. Continual fine-tuning (CF) (Fawi, 2024) enables models to adapt and excel in varied environments while maintaining performance across tasks.

While prior research has focused on text generation capabilities (Luo et al., 2024), less attention has been given to addressing CF in NLU tasks (Wang et al., 2018), which are vital for real-world applications. Furthermore, there has been limited exploration of how models with different parameter sizes handle CF during continual fine-tuning, especially models under 10 billion parameters.

This study aims to fill these gaps by evaluating forgetting across various models with different parameter sizes on key NLU tasks from the GLUE benchmark (Wang et al., 2018), including SST-2, MRPC, CoLA, and MNLI. By conducting extensive experimentation with prompt engineering and task-specific adjustments, we provide comparative insights into how models like **Orca-2-7b** (Microsoft, 2023), **Llama-3.1-8B** (Llama Team, 2024), and **Phi-3.5-mini** (Microsoft, 2024) perform under sequential fine-tuning.

Our results show that models such as **Phi-3.5-mini** (Microsoft, 2024) continue to exhibit minimal forgetting while maintaining strong learning capabilities, making them ideal for continual learning environments. Similarly, **Orca-2-7b** (Microsoft, 2023) and **Qwen2.5-7B** (Hui et al., 2024) demonstrate impressive learning abilities and overall performance after fine-tuning.

In summary, this work contributes by:

- Evaluating catastrophic forgetting and task learning across diverse LLMs using sequential

fine-tuning on specific NLU tasks.

- Highlighting the importance of prompt engineering and fine-tuning strategies for optimizing model performance.

- Providing insights into the performance of models with different parameter sizes, identifying those best suited for continual learning.

- Proposing continual fine-tuning as a key strategy for future LLM agents to handle multiple tasks without sacrificing accuracy.

2 Related Works

Catastrophic forgetting, a phenomenon first identified by (McCloskey and Cohen, 1989), remains a fundamental challenge in sequential learning tasks for neural networks. When models are trained on multiple tasks in sequence, they tend to overwrite previously acquired knowledge, leading to significant performance degradation on earlier tasks. Various methods have been proposed to mitigate this issue, such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), which regularizes weight updates to protect crucial parameters learned from previous tasks. Similarly, memory-based methods like Gradient Episodic Memory (GEM) (Lopez-Paz and Ranzato, 2017) address forgetting by storing and replaying examples from past tasks during training, thereby reducing interference.

While recent LLM models show strong zero-shot performance, they often struggle with tasks outside their training and evaluation sets. To address this, (Scialom et al., 2022) propose Continual-T0 (CT0), a fine-tuned LLM capable of learning new tasks while retaining prior knowledge, largely due to the self-supervision pre-training process. (Huang et al., 2024) conducted an empirical study on catastrophic forgetting in LLMs, finding that forgetting becomes more severe as model size increases, especially in models ranging from 1B to 7B parameters, during continual fine-tuning across domains like reasoning and reading comprehension.

3 Methodology

We employed a *Continual Instruction Fine-tuning* (Huang et al., 2024) approach, shown in the Figure 1, sequentially adapting the base model M_0 on tasks $\{T_1, T_2, \ldots, T_n\}$ from the GLUE benchmark (Wang et al., 2018). The goal was to evaluate how

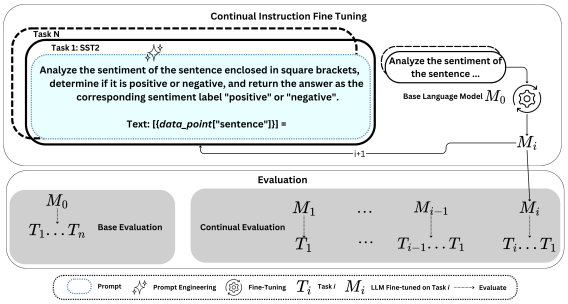

Figure 1: The figure illustrates the Continual Finetuning workflow. M_0 represents the base language model, and subsequent models $M_1, M_2, \ldots, M_n$ denote the fine-tuned versions after training on tasks $T_1, T_2, \ldots, T_n$. The figure also highlights the process of generating task-specific prompts and the continual evaluation to assess the model's retention.

well the model M_i, fine-tuned on task T_i, retained knowledge from previous tasks $\{T_1, \ldots, T_{i-1}\}$.

The methodology comprised two main stages. First, we prepared the dataset for each task T_i using *prompt engineering* (PE) (Chen et al., 2024). Let X be the original dataset for T_i, transformed into a structured prompt dataset X' as follows:

$$X' = \text{PE}(X)$$

where $\text{PE}(\cdot)$ represents the prompt engineering function. Prompts were designed to guide the model to perform task-specific instructions and the exact prompts and their expected outputs are shown in Table 1

Once the datasets were prepared via prompt engineering, the fine-tuning process started with a base model M_0, which was sequentially trained on tasks $\{T_1, T_2, \ldots, T_n\}$. At each step i, a new task T_i was introduced, and the corresponding prompt-engineered dataset $X' = \text{PE}(X)$ was used for training. The model adapted to each new task as the training progressed through tasks, creating updated versions $\{M_1, M_2, \ldots, M_n\}$.

Mathematically, let M_0 be the base model, and let P_i be the prompt used for task T_i. The training process for each task T_i can be represented as:

$$M_i = \mathcal{F}(M_{i-1}, T_i, P_i(X'))$$

where $\mathcal{F}$ represents the fine-tuning function that takes the previous model M_{i-1}, task T_i, and the corresponding prompt-engineered dataset $P_i(X')$ to produce the updated model M_i.

Task	Instruction	Output Format
SST-2	Analyze the sentiment of the sentence enclosed in square brackets, determine if it is positive or negative, and return the answer as the corresponding sentiment label "positive" or "negative".	"positive" or "negative"
MRPC	Analyze the two sentences enclosed in square brackets and determine if they are paraphrases of each other.	"paraphrase" or "not paraphrase"
CoLA	Determine if the sentence enclosed in square brackets is grammatically correct or incorrect.	"correct" or "incorrect"
MNLI	Analyze the relationship between the premise and hypothesis enclosed in square brackets. Determine if the hypothesis is entailed by, neutral to, or contradicts the premise.	"entailment", "neutral", or "contradiction"

Table 1: This table presents the instructions used for fine-tuning models on specific tasks such as sentiment analysis (SST-2), paraphrase detection (MRPC), grammatical acceptability (CoLA), and natural language inference (MNLI).

4 Evaluation and Measurement of Catastrophic Forgetting and Learning

After each fine-tuning episode, model M_i was evaluated on all previous tasks $\{T_1, T_2, \ldots, T_i\}$ to assess catastrophic forgetting and learning. Accuracy was used as the evaluation metric, comparing post-fine-tuning performance to the base performance to detect any degradation or improvement. Forgetting was quantified as the difference between the maximum accuracy on each task during fine-tuning and the final accuracy:

$$\text{Forgetting for Task } t = \max_{0 \leq k \leq T}(a_{k,t}) - a_{T,t}$$

where $a_{k,t}$ is the accuracy after fine-tuning on task k and $a_{T,t}$ is the final accuracy.

Learning was calculated as the improvement over the base performance:

$$\text{Learning for Task } t = \max_{k \leq T}(a_{k,t}) - a_{0,t}$$

where $a_{0,t}$ is the base accuracy and $\max_{k \leq T}(a_{k,t})$ is the highest accuracy during fine-tuning. Both metrics provide insight into the model's behavior across tasks.

4.1 Selected Tasks

The selected tasks include:

1. **SST-2** (Stanford Sentiment Treebank): A binary classification task for determining whether a sentence has a positive or negative sentiment.

2. **MRPC** (Microsoft Research Paraphrase Corpus): A binary classification task to predict whether two sentences are paraphrases of each other.

3. **CoLA** (Corpus of Linguistic Acceptability): A binary classification task for determining whether a sentence is grammatically correct.

4. **MNLI** (Multi-Genre Natural Language Inference): A three-class classification task where the goal is to determine whether a premise sentence entails, contradicts, or is neutral with respect to a hypothesis sentence across multiple genres of text.

5 Experimental Results

Table 2 presents a comparison of various models' performance across four key metrics: Pretrained Performance, Forgetting, Learning, and overall Training Performance after sequential task fine-tuning.

Orca-2-7b (Microsoft, 2023) achieved the highest pretraining performance at **0.76**, demonstrating strong initial capabilities, closely followed by **Qwen2.5-14B** (Hui et al., 2024) at **0.71**, indicating a solid foundation before fine-tuning. In terms of catastrophic forgetting, **Phi-3.5-mini** (Microsoft, 2024) and **Phi-2** showed minimal forgetting with values of **0.02** and **0.1**, respectively. On the other hand, **Qwen2.5-14B** and **Llama-3.1-8B** (Llama Team, 2024) exhibited high forgetting rates of **0.935** and **0.59**. After fine-tuning, **Orca-2-7b** stood out with the best average performance of **0.81**, followed by **Qwen2.5-14B** at **0.80**. Models like **Phi-3.5-mini** and **Orca-2-7b** performed better overall, balancing low forgetting and high learning rates, offering valuable insights into mitigating catastrophic forgetting.

Results from continual fine-tuning on the task **SST2** (Figure 2) show **Qwen2.5-7B** (Hui et al., 2024) leading in accuracy and learning, while **Orca-2-7B** (Microsoft, 2023) exhibited the least forgetting across tasks.

Figure 3 shows a clear pattern where larger models, like Qwen2.5-7B and Llama-3.1-8B, exhibit higher learning rates, often at the cost of increased forgetting. In contrast, smaller models, such as Phi-3.5-mini and Phi-2, manage to balance low forgetting with moderate learning gains. This suggests a trade-off between capacity and stability across

Model	Parameter Size (B)	Pretrained Performance	Forgetting	Learning	Training Performance
Llama-3.2-1B (Llama Team, 2024)	1	0.50	0.24	0.33	0.54
Llama-3.2-3B (Llama Team, 2024)	3	0.56	0.225	0.36	0.61
Llama-3.1-8B (Llama Team, 2024)	8	0.56	0.59	0.84	0.67
Llama-3-8B (Llama Team, 2024)	8	0.53	0.39	0.98	0.70
Llama-2-7B (Llama Team, 2024)	7	0.67	0.23	0.12	0.63
GPT-J-6B (Wang, 2021)	6	0.50	0.39	0.45	0.54
Phi-2 (Microsoft, 2024)	2.7	0.59	0.1	0.15	0.61
Phi-3.5-mini (Microsoft, 2024)	3.82	0.69	**0.02**	0.30	0.76
Orca-2-7b (Microsoft, 2023)	7	**0.76**	0.185	0.33	**0.81**
Qwen2.5-0.5B (Hui et al., 2024)	0.5	0.52	0.23	0.56	0.61
Qwen2.5-7B (Hui et al., 2024)	7	0.56	0.51	**1.12**	0.77
Qwen2.5-14B (Hui et al., 2024)	14	0.71	0.935	0.66	0.80

Table 2: Comparison of the pre-trained performance, forgetting, learning capabilities, and training performance of various language models. **Parameter Size (B)** indicates the number of parameters (in billions) of each model. The **Pretrained Performance** refers to the initial performance of each model before any task-specific training.

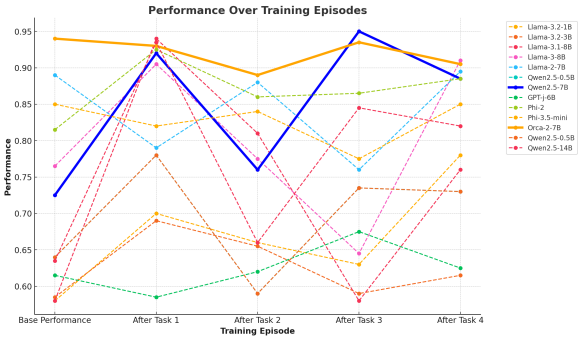

Figure 2: Performance of various models across continual fine-tuning episodes for the task SST2. The solid blue line highlights the model with the highest overall performance, while the solid orange line represents the model with the least amount of forgetting (the smallest drop in performance between tasks). Dashed lines indicate the performance of other models. This diagram illustrates both the learning capacity and retention ability of each model over successive tasks.

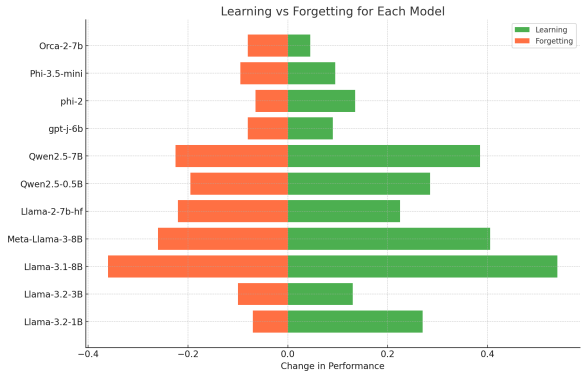

Figure 3: Bar graph displaying model performance for the task SST2 on catastrophic forgetting (reverse) and learning rates, with higher models showing more significant trade-offs. Phi-3.5-mini stands out with minimal forgetting and moderate learning.

models.

6 Conclusion

This study explored catastrophic forgetting in large language models during sequential fine-tuning on multiple NLU tasks. We found that smaller models like **Phi-3.5-mini** effectively minimize forgetting while maintaining learning capabilities. Prompt engineering and fine-tuning strategies significantly impact model performance in continual learning settings. Models such as **Orca-2-7b** and **Qwen2.5-7B** showed strong learning abilities but varied in forgetting. Careful model selection and tuning can enhance handling multiple tasks without sacrificing accuracy, which is crucial for developing autonomous LLM-based agents. Future work should explore more advanced continual learning techniques to further mitigate catastrophic forgetting.

7 Limitations

This study focused on models under 10 billion parameters and specific NLU tasks from the GLUE benchmark, so results may not generalize to larger models or other tasks. We only used sequential fine-tuning without exploring other continual learning strategies like memory replay or regularization methods. Relying on prompt engineering may introduce biases affecting performance comparisons. We also didn't consider computational constraints of continual fine-tuning, which could impact practical deployment. Finally, using accuracy as the sole metric might not capture all aspects of model performance.

References

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, and

et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. ArXiv preprint.

Banghao Chen, Zhaofeng Zhang, Nicolas Langrené, and Shengxin Zhu. 2024. Unleashing the potential of prompt engineering in large language models: a comprehensive review. *Preprint*, arXiv:2310.14735.

Muhammad Fawi. 2024. Curlora: Stable llm continual fine-tuning and catastrophic forgetting mitigation.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *Preprint*, arXiv:2403.01244.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, An Yang, Rui Men, Fei Huang, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. 2024. Qwen2.5-coder technical report. *Preprint*, arXiv:2409.12186.

Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. 2024. Ai agents that matter. *Preprint*, arXiv:2407.01502.

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

AI Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, volume 30, pages 6467–6476.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2024. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *Preprint*, arXiv:2308.08747.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165.

Microsoft. 2023. Orca 2: Teaching small language models how to reason. *Preprint*, arXiv:2311.11045.

Microsoft. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. 2022. Fine-tuned language models are continual learners. *arXiv preprint arXiv:2210.05653*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Ben Wang. 2021. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. `https://github.com/kingoflolz/mesh-transformer-jax`.

A Example Appendix

This is an appendix.