De-Fine Tuning: Unlearning Multilingual Language Models Report

JeongJun Song
jsong132@asu.edu

Thais Herve
therve@asu.edu

Nishtha Kuberbhai Chaudhary
nchaud28@asu.edu

Swetha Mohan
smohan63@asu.edu

Devika Shaj Kumar Nair
dshajkum@asu.edu

Krisha Waghela
kmwaghel@asu.edu

Fenil Denish Bardoliya
fbardoli@asu.edu

Abstract

As multilingual language models (LLMs) scale globally, the need for controlled unlearning removing specific knowledge without degrading unrelated capabilities has become critical for privacy, fairness, and safety. Our project, *De-Fine Tuning*, introduces *LingTea*, a novel framework for unlearning in multilingual models using task vector negation and token-level kappa pruning. We worked with the TOFU dataset and its 1% and 5% "forget" subsets, translated into Korean and Hindi using Mistral. We fine-tuned LLaMA 3.2B across languages and evaluated methods including full fine-tuning, LoRA, prefix tuning, and adapters. Results show that combining kappa-based pruning with task vector unlearning significantly improves deletion accuracy while maintaining multilingual consistency. Sequential unlearning proved unstable in smaller models, underlining the need for scale-aware approaches. Our work highlights the importance of cross-lingual alignment and scalable precision unlearning in next-generation LLMs.

1 Introduction

The rapid deployment of multilingual language models (MLLMs) across industries from healthcare to education to content moderation has created a need for more responsible control over the information these models retain. While the focus of most current fine-tuning work is on improving accuracy or adding domain specific knowledge, the inverse process *unlearning* is becoming equally important. Unlearning involves removing specific information from trained models without negatively impacting unrelated tasks or language capabilities. Applications include forgetting sensitive data, complying with user deletion requests, removing harmful biases, and repairing jailbreak vulnerabilities (Choi, 2024).

However, most unlearning methods to date are designed for monolingual models. In the multilingual setting, a unique challenge emerges: removing information in one language often fails to remove its equivalent in others. For instance, forgetting an English answer may not erase the same content in Hindi or Korean. This leads to semantic redundancy, compliance gaps, and openings for adversarial prompts that exploit language mismatches (Qi, 2023)(Xu, 2023)(Rav, 2024).

Problem Statement

Current unlearning methods struggle to maintain consistency across languages and often require retraining models separately per language. This is computationally inefficient and may still result in over-removal or unintended knowledge loss. Our central challenge is to design a language agnostic unlearning method that scales efficiently, preserves relevant knowledge, and ensures that deletion propagates fairly across all supported languages.

Related Work

Prior work by (Ilharco, 2023) introduced task vector unlearning through linear arithmetic in weight space. (Qi, 2023) and (Choi, 2024) explored multilingual consistency in LLMs and the risks of cross-lingual jailbreaking. While effective, these methods do not combine token level filtering with scalable language coordination. Additionally, prior unlearning techniques often ignore low resource languages or require excessive fine-tuning passes for each translation (Pham, 2024)(Blanco, 2025).

Our Contribution

We address these gaps through *LingTea*, a framework that integrates:

- **Sequential Unlearning** to perform in a sequence across multiple languages datasets.
- **Task vector subtraction** to reverse targeted training effects.
- **Translation-aligned unlearning** to ensure deletion consistency across languages.

We show that this approach, applied to the TOFU dataset and validated across LLaMA 3.2B, Phi-3.5, and Qwen-2.5, improves both retention and deletion accuracy compared to baseline methods. LingTea offers a scalable path forward for multilingual model governance, safety, and compliance.

2 Dataset and Methodology

The TOFU (Task of Fictitious Unlearning) dataset is a benchmark developed to evaluate the effectiveness of machine unlearning techniques in large language models (LLMs). It comprises 200 synthetic author profiles, each containing 20 question-answer pairs, generated using GPT-4. These profiles are entirely fictitious and do not overlap with any real-world data, ensuring a controlled environment for testing unlearning methods.

The primary objective of the TOFU dataset is to assess whether LLMs can effectively "unlearn" specific information—represented by a subset of these profiles termed the "forget set"—while retaining knowledge from the remaining data. This task is particularly challenging as it requires models to forget certain details without retraining from scratch, simulating real-world scenarios where data removal is necessary for privacy or compliance reasons.

To facilitate comprehensive evaluation, the TOFU dataset provides various subsets:

- full.json containing 4,000 English Q&A pairs
- forget01.json and forget05.json representing 1% and 5% target subsets for knowledge removal

2.1 Methodology and Approach

We are currently working with the **TOFU dataset**, which includes:

- full.json containing 4,000 English Q&A pairs
- forget01.json and forget05.json representing 1% and 5% target subsets for knowledge removal

Our team has fine-tuned the **Llama3-2:3B** model on full.json using a range of fine-tuning approaches:

- Full fine-tuning

The goal is to compare our kappa-driven unlearning technique with these established fine-tuning strategies.

To evaluate cross-lingual unlearning, we translated the TOFU dataset; full.json into **Korean** and **Hindi**. For translation, we used both **Mistral** and **Gemini**:

- **Gemini API (v1.5)** was initially used, but we encountered limitations including API request rate limits, slow translation speeds, and lower-quality output due to outdated model versions.
- **Mistral AI** produced more consistent and semantically accurate translations, confirmed via manual inspection and similarity metrics such as Word2Vec and cosine similarity.

Given these observations, we selected **Mistral** as our primary translation tool moving forward.

If our proposed approach proves successful in standard unlearning tasks, we plan to extend evaluation using the **XSafety jailbreak dataset**, which will help us test for cross-lingual robustness and resistance to adversarial prompts.

1. **Gradient Ascent**

   Gradient ascent is implemented similarly to normal gradient descent learning. First, we use our target data, in this case the forget data set, and run it through our target model on train mode. By negating the resulting cross-entropy losses, we can reverse the direction of our gradient, causing the model to become less accurate with respect to the inputs.

2. **Sequential Unlearning**

   Sequential unlearning is a technique where the unlearning process is carried out in a sequence across multiple datasets, instead of performing it all at once. In our study, we applied sequential unlearning to datasets in three languages: English (eng), Korean (kr),

and Hindi (hin). The key idea is to systematically "unlearn" data in stages across these languages to improve the efficiency of the unlearning process.

In this method, the unlearning sequence starts with one language (Language A), followed by another (Language B), and then proceeds to a third language (Language C). The unlearning is performed on the "forget" dataset, which contains data that the model needs to forget. By applying this sequence, the model's ability to efficiently discard certain data is enhanced by leveraging the inter-language transitions.

3. **Task Vector Unlearning via Negation**

   In this approach, we use task vectors (Ilharco, 2023) to "unlearn" specific behaviors or tasks that have been learned during fine-tuning. These task vectors are derived by subtracting the weights of a pre-trained model from the weights of a fine-tuned model, as outlined in the following equation:

   $$100\%\ \textit{TOFU LlaMA} - 99\%\ \textit{TOFU LLaMA}$$

   Here, 100% TOFU LLaMA refers to a version of the LLaMA 3.2 model that has been fine-tuned on the entire TOFU dataset. 99% TOFU LLaMA, on the other hand, refers to a version of the same model that was fine-tuned on only 99% of the TOFU dataset. Difference between these two models is the task vectors, representing the model's learned behavior specifically from the 1% of the dataset that we want to forget.

   Now, after computing the task vectors, the unlearning happens by subtracting these task vectors from the pretrained model. Negation of the task vector essentially shifts the model's weights in the opposite direction of the task's influence, which reduces performance on the specific task we want to forget. This is mathematically represented as:

   $$\textit{pretrained\_model} - \alpha \cdot \textit{task\_vector}$$

   where α is a scaling factor determined via cross-validation. This method allows for targeted unlearning which reduces performance on the specific task, without losing general performance across other unrelated tasks.

4. **KL Divergence Unlearning**

   KL-divergence-based unlearning is implemented using a teacher-student training setup, where the student model is fine-tuned to retain desirable knowledge while deliberately forgetting information present in a designated 'forget' dataset D_f. The objective is to align the student's behavior with the teacher model on the retained dataset D_r, while forcing divergence on D_f by manipulating the output distributions. This divergence is guided using a Kullback-Leibler (KL) divergence-based loss computed between the output logits of the student and teacher models.

   The KL divergence between the teacher and student outputs is defined as:

   $$\mathcal{L}_{\text{KL}} = T^2 \cdot \sum_i P_T(i) \log \left(\frac{P_T(i)}{P_S(i)} \right)$$

   where P_T and P_S are the softmax output distributions of the teacher and student models, respectively, and T is a temperature parameter used to soften the logits during training (set to 1.0 in this case). This formulation allows for fine-grained control over knowledge retention and suppression.

   In practice, for samples in the retain dataset D_r, the KL loss is applied directly to minimize divergence:

   $$\mathcal{L}_{\text{retain}} = \mathcal{L}_{\text{KL}}(\mathbf{z}_S^{(r)}, \mathbf{z}_T^{(r)})$$

   Conversely, for samples in the forget dataset D_f, the KL loss is negated to promote behavioral divergence:

   $$\mathcal{L}_{\text{forget}} = -\mathcal{L}_{\text{KL}}(\mathbf{z}_S^{(f)}, \mathbf{z}_T^{(f)})$$

   The total loss optimized during training is:

   $$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{retain}} + \mathcal{L}_{\text{forget}}$$

   This formulation was implemented such that both student and teacher models were instantiated from the same pretrained checkpoint and loaded to enable distributed memory placement across multiple GPUs. Tokenization was performed using a shared vocabulary with padding and truncation to a

fixed sequence length of 512 tokens. Each training iteration involved parallel processing of mini-batches from the retain and forget datasets, where the respective logits were computed, and KL divergence was evaluated using PyTorch's functional API. Specifically, `log_softmax` was applied to the student logits and `softmax` to the teacher logits, in accordance with the definition of KL divergence.

3 Experiments

3.1 Task Vectors

In our experiment, we applied the Task Vector Unlearning method across various language combinations to evaluate how well the model could forget specific data while retaining general performance and transferring information across languages. We used the following language combinations for fine-tuning: English, Hindi, Korean, English-Hindi, English-Korean, Hindi-Korean, and English-Hindi-Korean. For example, English-Hindi refers to the LLaMA 3.2 model fine-tuned on both English and Hindi-translated TOFU data.

For each combination, we computed the task vectors by subtracting the weights of the 99% TOFU LLaMA model from the 100% TOFU LLaMA model. The task vectors were then negated and applied to the pretrained model using a scaling factor of $\alpha = 1.7$, determined through cross-validation. This process aimed to evaluate the model's ability to retain knowledge from the fine-tuned languages and forget the excluded 1% of the dataset.

3.2 Sequential Unlearning

Unlearning is performed sequentially (e.g., Language A $\rightarrow$ Language B $\rightarrow$ Language C). In our research, three languages were targeted (English, Korean, Hindi); the unlearning was performed by changing the unlearning sequence of the "forget" dataset translated into these three languages.

The unlearning sequence in our research was performed as follows:

(a) English $\rightarrow$ Korean $\rightarrow$ Hindi
(b) English $\rightarrow$ Hindi $\rightarrow$ Korean
(c) Korean $\rightarrow$ English $\rightarrow$ Hindi
(d) Korean $\rightarrow$ Hindi $\rightarrow$ English
(e) Hindi $\rightarrow$ Korean $\rightarrow$ English
(f) Hindi $\rightarrow$ English $\rightarrow$ Korean

With the newly-trained models, we can calculate the losses of the model, and then calculate our BERT and Sentence Transformer scores.

3.3 Negative Posterior Optimization

We implemented Negative Posterior Optimization (NPO) as our individual unlearning strategy. NPO reverses the learning signal for specific examples by multiplying the loss by -1, thereby encouraging the model to forget them during training.

We applied NPO to LLaMA 3.2B, Gemma 4B, and Qwen 7B using standard full fine-tuning, modifying only the loss on 1% and 5% forget subsets from the TOFU dataset (translated into English, Korean, and Hindi). The rest of the training pipeline (optimizer, learning rate, batch size, epochs) was kept consistent to isolate the effect of NPO.

For evaluation, we focused solely on the forget datasets. We used five metrics Sentence-Transformer cosine similarity, BERTScore F1, ROUGE, BLEU, and inverse perplexity (1/PPL) each normalized to a [0,1] scale and averaged to produce a composite unlearning score.

4 Results and Analysis

4.1 Task Vectors

In the forgetting scores (Figure: 1), we compare the F-1 scores in forget01 TOFU data from the unlearning models with the LLaMA 3.2 baseline model, which serves as a reference for all combinations of languages.

In the retention scores (Figure: 2), we compare the F-1 scores in the TOFU data retention (99%) of the unlearning models with the fine-tuned LLaMA 3.2 models, which were trained in specific language combinations.

The Task Vector Unlearning method generally works well for English, where the model

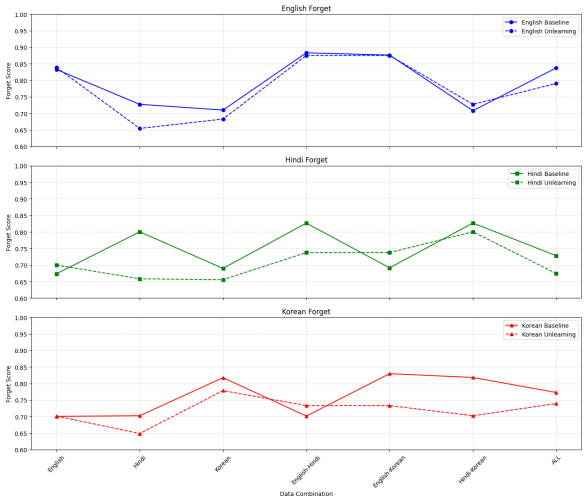

Figure 1: F1 score for Forget data

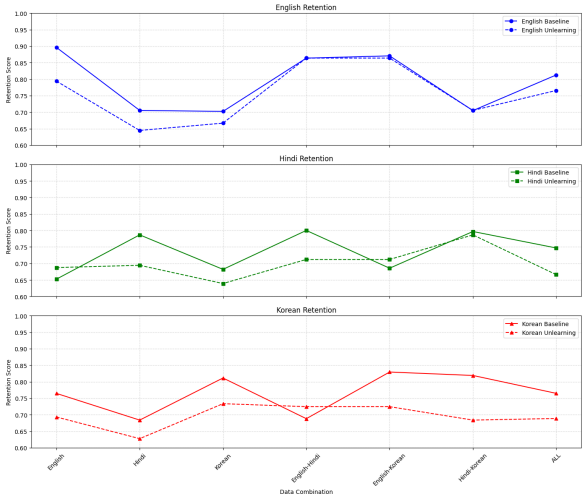

Figure 2: F1 score for Retain data

effectively forgets the targeted information, with scores that are nearly identical to the baseline LLaMA 3.2 model (which does not have TOFU-specific data), while still retaining the other parts of the TOFU dataset. However, for low-resource languages like Hindi and Korean, forget and retain scores exhibit more variability, indicating less effective unlearning. In these cases, the model struggles to forget the targeted information as effectively as it does for English, and the retention of other data is more affected.

4.2 Sequential Unlearning

In terms of the loss from Sequential Unlearning, we observed both partial success and partial failure. Specifically, when applying multiple unlearning processes through different languages with the same meaning, we noticed a significant increase in the loss values, leading to catastrophic unlearning in the 3B (Llama 3.2) and 4B (Gemma) models. However, in the 7B model, Qwen, the loss remained stable at values between 1 and 2.

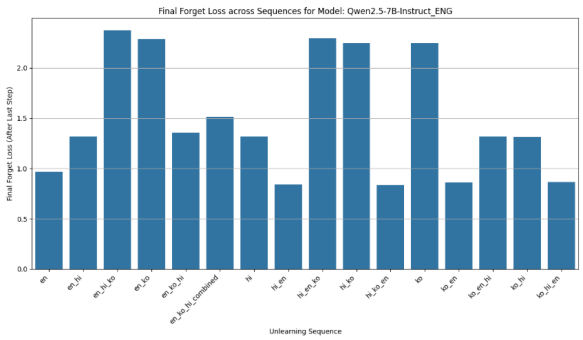

Figure 3: Final Forget Loss across Sequences for Model: Qwen 2.5:7B Instruct ENG

4.3 KL Divergence

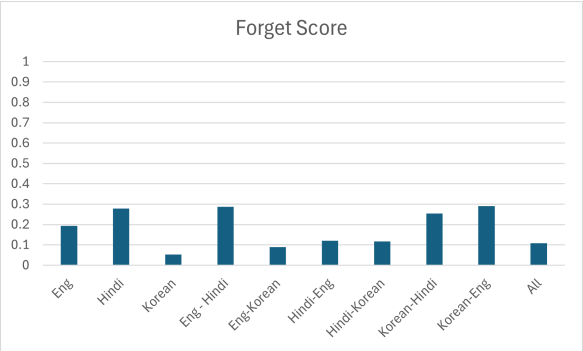

Figure 4: Sentence Transformer scores on the forget dataset after KL divergence unlearning on Llama 3.2B

KL divergence unlearning was successfully applied to the LLaMA 3.2B model by assigning teacher and student models to separate GPUs. As observed in the figures 4 and 5, the Sentence Transformer scores for the retention data set remain stable in different unlearning sequences, consistently ranging between 0.54 and 0.60, indicating that the student model successfully preserves general knowledge on the retained data regardless of the language order used during unlearning. In contrast, forget scores show a noticeable reduction compared to retention, with values ranging from 0.05 to 0.29, confirming that

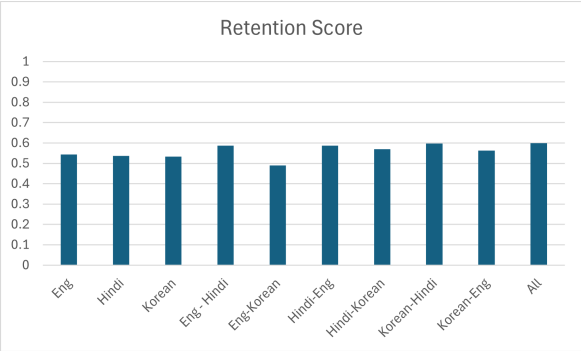

Figure 5: Sentence Transformer scores on the retain dataset after KL divergence unlearning on Llama 3.2B

the model diverges appropriately in the forget data set. However, fluctuations in forget scores across unlearning language combinations, such as higher scores in "Eng-Hindi unlearning" (0.29) and lower scores in "Korean unlearning" (0.05), can be attributed to cross-lingual semantic overlap. Since multilingual models tend to share latent representations across languages, unlearning in one language can partially affect or preserve semantically similar content in others, leading to inconsistencies in forgetting effectiveness depending on the language pairing and ordering.

This unlearning framework was also implemented for the Gemma model finetuned on all the 3 languages and english only. From the ROUGE scores, we can conclude that unlearning has varying effects across languages. For English, performance slightly improves when 1% of the data is forgotten (0.5714 vs. 0.5488), suggesting that removing a small portion of the data may help the model focus better, possibly by eliminating irrelevant information. However, forgetting 5% of the data leads to a decline in performance (0.522), indicating that too much data removal negatively impacts the model. In contrast, for Korean and Hindi, the performance drops sharply even after forgetting just 1% of the data (Korean: 0.0357, Hindi: 0.0188), reflecting a higher reliance on the data in these languages, likely due to linguistic complexity or less available training data. This suggests that unlearning methods need to be tailored to different languages, as the impact of forgetting data is more detrimental for languages with limited data or more complex structures.

However, this method could not be extended to the Qwen 2.5 model due to its significantly larger memory footprint, which exceeded the available GPU capacity. This constraint underscores the need for more memory-efficient unlearning strategies, such as low-rank adapters or model offloading, particularly for larger-scale language models.

4.4 Negative Posterior Optimization

In this evaluation, we focused solely on the forget datasets to assess how effectively each model was able to unlearn specific target examples.

LLaMA 3.2B showed the most effective forgetting, with the lowest scores on English forget tasks, suggesting that NPO strongly influenced its parameter updates. Gemma 4B retained more information on Korean and Hindi forget subsets, indicating a milder unlearning response. Qwen 7B showed moderate and consistent forgetting across all languages, but did not outperform LLaMA.

Since we did not evaluate on dedicated retain datasets, these results strictly reflect forgetting performance. Overall, NPO proved effective at selectively degrading model recall on targeted examples without retraining or structural changes.

5 Shortcomings and Future Work

The Sequential Unlearning approach led to catastrophic unlearning in models such as GEMMA and LLAMA. From the perspective of loss, the 3B and 4B models were too large for sequential unlearning, making them impractical for this method. However, with the 7B Qwen model, stability was maintained to a certain extent. While both Retain accuracy and Forget accuracy decreased, it showed potential. Ultimately, smaller models are more efficient for mobile apps and personal computers, so solutions for 3B and 4B models need to be devised as well.

In the Task Vector method, models with data to be removed and models without the data

need to be considered. Both KL and NPO approaches require this. We need to think about whether this scenario could realistically occur in a business environment. Even if, by good luck, a checkpoint was properly saved, increasing the frequency of checkpoints for this purpose would only add to the training time.

6 Team Member-wise Contribution

- **Jun Song:** Literature Review, Data Processing, Tuning the base models(llama, gemma, and qwen), Sequential Unlearning, Evaluation, Final Report
- **Thais Herve:** Literature Review, Data processing, Gradient ascent implementation and results on all three base models (llama, gemma, qwen), baseline unlearning for cross entropy loss comparison, Final report
- **Swetha Mohan:** Literature review, Data processing using Gemini API for translation, generated datset with Gemini for hindi and korean data, KL divergence unlearning using forget and retain datasets on Gemma3:4B Instruct and Phi3.5 models, Final report.
- **Krisha Waghela** Literature Review, Implementation of Negative Posterior Optimization (NPO), developing evaluation metrics and writing the NPO sections for methodology and results, Final Report
- **Devika Nair:** Literature review, Data Processing using Mistral API for translation, generated dataset with Mistral for Hindi language data, KL divergence unlearning using forget and retain datasets on Llama 3.2B and Qwen 2.5 models.
- **Nishtha Chaudhary:** Literature Review, Data Processing (using Mistral API for translation), Dataset Generation (using Mistral for Hindi and Korean data), Fine-tuning (LLaMA 3.2 on Hindi and Korean TOFU dataset), Task Vector Unlearning (on Llama 3.2B), Final Report

7 Project Resources and GitHub

7.1 GitHub Repository

- De-Fine Tuning – GitHub Repository

7.2 Google Drive

- Team google drive link
- Paper Reviews
- Meeting Logs
- Evaluation Results

References

[Wang2024] Yaxuan Wang. 2024. LLM unlearning via loss adjustment with only forget data. *arXiv*.

[Leyl2024] Danny D. Leyl. 2024. Learning, forgetting, remembering: Insights from tracking LLM memorization during training. In *Proceedings of ACL*.

[Chen2023] Jiaao Chen. 2023. Unlearn what you want to forget: Efficient unlearning for LLMs. *arXiv*.

[Han2021] Kai Han. 2021. Transformer in transformer. In *Proceedings of NeurIPS 2021*.

[Authors2023] Many Authors. 2023. Rethinking machine unlearning for large language models. *arXiv*.

[Authors2024] Many Authors. 2024. Knowledge unlearning for LLMs: Tasks, methods, and challenges. *arXiv*.

[Authors2024b] Many Authors. 2024. To forget or not? Towards practical knowledge unlearning for large language models. *arXiv*.

[OpenAI2024] OpenAI. 2024. Deliberative alignment: Reasoning enables safer language models. *OpenAI*.

[Qin2024] Libo Qin. 2024. Multilingual large language model. *arXiv*.

[Huang2025] Mrk He Huang. 2025. Learning to unlearn for robust machine unlearning. *Springer*.

[Fan2025] Chongyu Fan. 2025. Challenging forgets: Unveiling the worst-case forget sets in machine unlearning. *Springer*.

[Blanco2025] Alberto Blanco. 2025. Digital forgetting in large language models: A survey of unlearning methods. *Springer*.

[Yao2024] Yuanshun Yao. 2024. Large language model unlearning. *arXiv*.

[Merullo2024] Jack Merullo. 2024. Talking heads: Understanding inter-layer communication in transformer language models. *arXiv*.

[Pavlick2024] Ellie Pavlick. 2024. Instilling inductive biases with subnetworks. *arXiv*.

[Ilharco2023] Gabriel Ilharco. 2023. Editing models with task arithmetic. *arXiv*.

[Z2024] Frederic Z. 2024. Knowledge composition using task vectors with learned anisotropic scaling. *arXiv*.

[Pham2024] Pham. 2024. Robust concept erasure using task vectors. *arXiv*.

[Schoepf2024] Schoepf. 2024. Potion: Towards poison unlearning. *arXiv*.

[Su2024] Yuimei Su. 2024. A survey on multilingual large language models. *arXiv*.

[Choi2024] Minseok Choi. 2024. Cross-lingual unlearning of selective knowledge in multilingual language models. *arXiv*.

[Lu2024] Taiming Lu. 2024. Learn and unlearn in multilingual LLMs. *arXiv*.

[Deng2023] Yue Deng. 2023. Multilingual jailbreak challenges in large language models. *arXiv*.

[Sher2024] Lingfeng Sher. 2024. The language barrier: Dissecting safety challenges of LLMs in multilingual contexts. *arXiv*.

[Rav2024] Swanand Rav. 2024. Split, unlearn, merge: Leveraging data attributes for more effective unlearning in LLMs. *arXiv*.

[Liu2024] Sijia Liu. 2024. Rethinking machine unlearning for large language models. *arXiv*.

[Choi2024b] Minseok Choi. 2024. Breaking chains: Unraveling the links in multi-hop knowledge unlearning. *arXiv*.

[Anthropic2024] Article. 2024. Research Anthropic. *arXiv*.

[Chu2023a] Vikram S. Chu. 2023. Zero shot machine unlearning. *IEEE Xplore*.

[Cao2015] Cao, Yinzhi and Yang, Junfeng. 2015. Towards Making Systems Forget with Machine Unlearning. *IEEE Xplore*.

[Guo et al.2019] Guo et al. 2019. Certified Data Removal from Machine Learning Models. *arXiv*.

[Chu2023b] Vikram S. Chu. 2023. Can Bad Teaching Induce Forgetting? Unlearning in Deep Networks Using an Incompetent Teacher. *arXiv*.

[Kurmanji2023] Meghdad Kurmanji. 2023. Towards Unbounded Machine Unlearning. *OpenReview*.

[Xu2023] Yang Xu et al. 2023. Language Anisotropic Cross-Lingual Model Editing. *Association for Computational Linguistics*.

[Qi2023] Jirui Qi et al. 2023. Cross-Lingual Consistency of Factual Knowledge in Multilingual Language Models. *Association for Computational Linguistics*.

Appendix

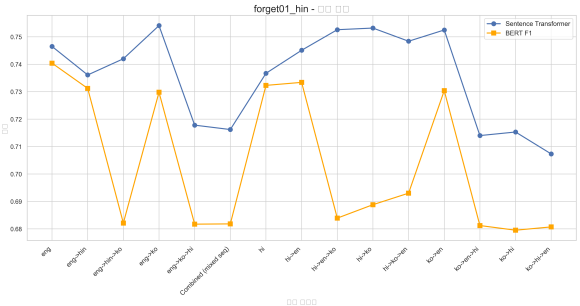

Figure 6: (Base-Llama3.2 3B) Performance comparison of Sentence Transformer and BERT F1 on Hindi forgetting task with forget rate 0.1

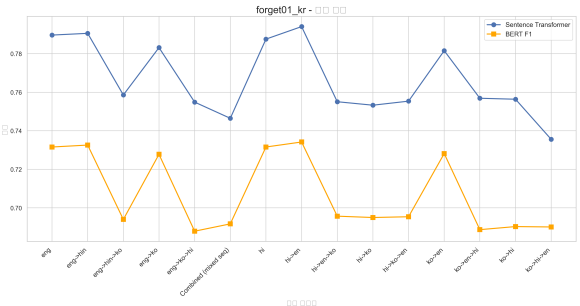

Figure 7: (Base-Llama3.2 3B) Performance comparison of Sentence Transformer and BERT F1 on Korean forgetting task with forget rate 0.1

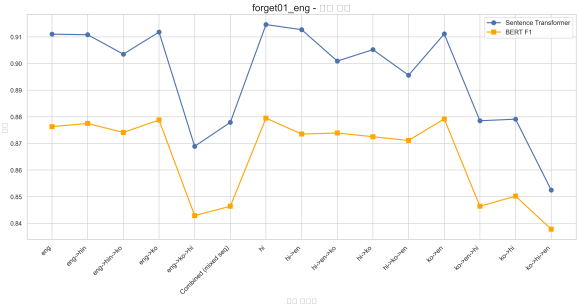

Figure 8: (Base-Llama3.2 3B) Performance comparison of Sentence Transformer and BERT F1 on English forgetting task with forget rate 0.1

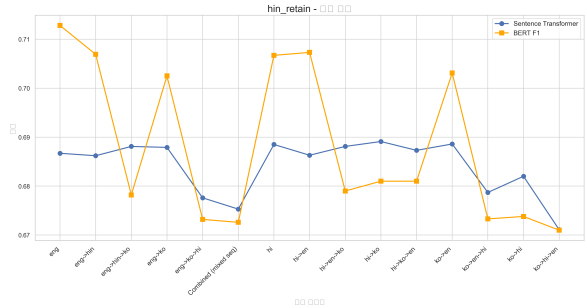

Figure 9: (Base-Llama3.2 3B) Retention performance comparison between Sentence Transformer and BERT F1 on Hindi data

Figure 10: (Base-Llama3.2 3B) Retention performance comparison between Sentence Transformer and BERT F1 on Korean data

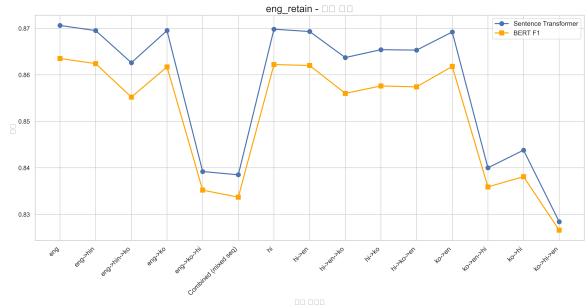

Figure 11: (Base-Llama3.2 3B) Retention performance comparison between Sentence Transformer and BERT F1 on English data

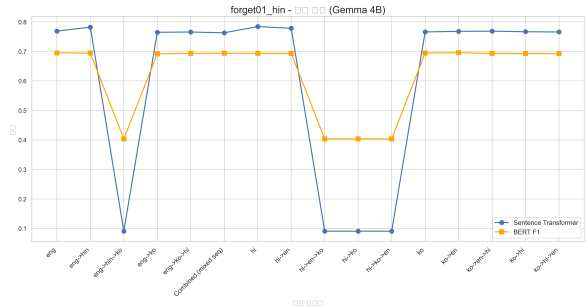

Figure 12: (Base-Gemma 4B) Performance comparison of Sentence Transformer and BERT F1 on Hindi forgetting task with forget rate 0.1

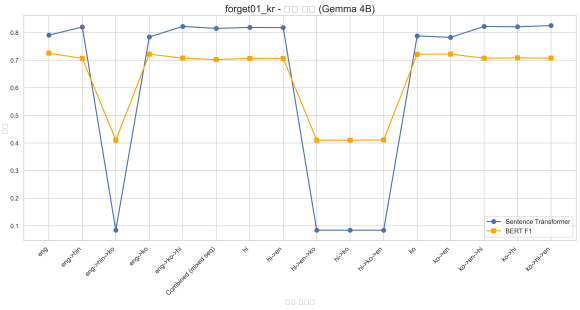

Figure 13: (Base-Gemma 4B) Performance comparison of Sentence Transformer and BERT F1 on Korean forgetting task with forget rate 0.1

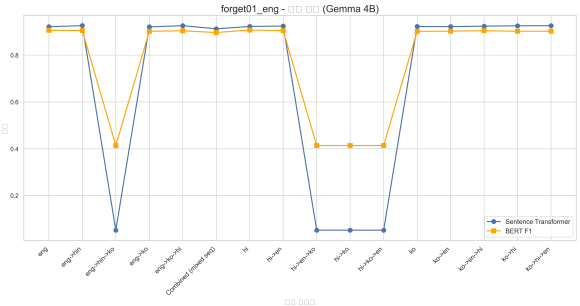

Figure 14: (Base-Gemma 4B) Performance comparison of Sentence Transformer and BERT F1 on English forgetting task with forget rate 0.1

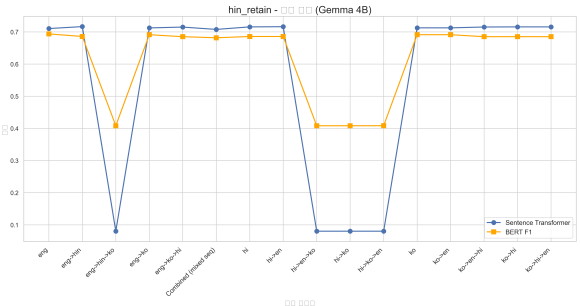

Figure 15: (Base-Gemma 4B) Retention performance comparison between Sentence Transformer and BERT F1 on Hindi data

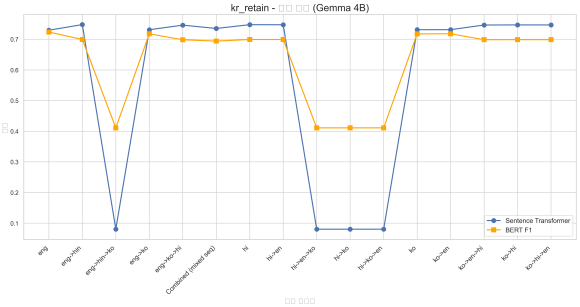

Figure 16: (Base-Gemma 4B) Retention performance comparison between Sentence Transformer and BERT F1 on Korean data

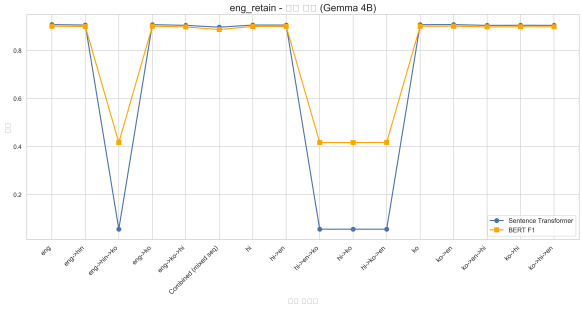

Figure 17: (Base-Gemma 4B) Retention performance comparison between Sentence Transformer and BERT F1 on English data

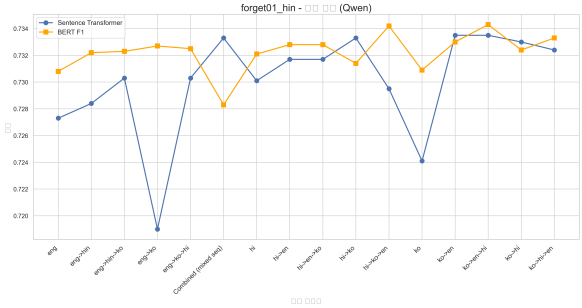

Figure 18: (Base-Qwen 7B) Performance comparison of Sentence Transformer and BERT F1 on Hindi forgetting task with forget rate 0.1

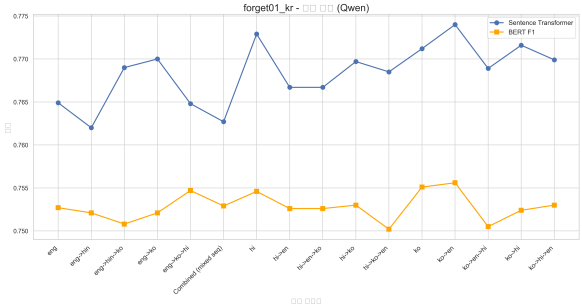

Figure 19: (Base-Qwen 7B) Performance comparison of Sentence Transformer and BERT F1 on Korean forgetting task with forget rate 0.1

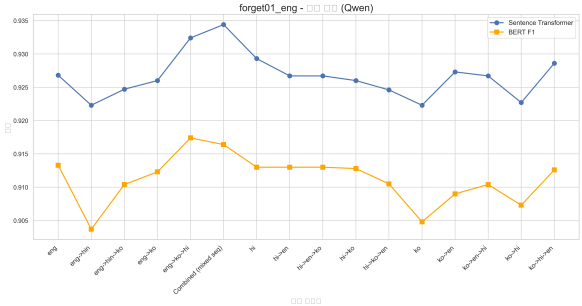

Figure 20: (Base-Qwen 7B) Performance comparison of Sentence Transformer and BERT F1 on English forgetting task with forget rate 0.1

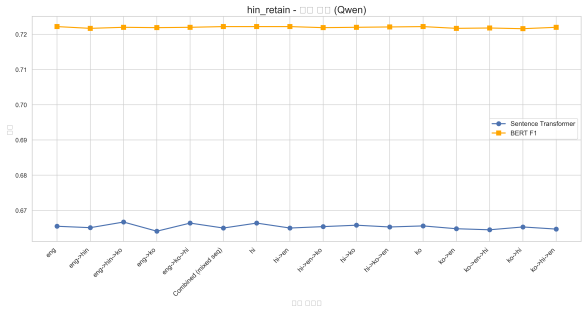

Figure 21: (Base-Qwen 7B) Retention performance comparison between Sentence Transformer and BERT F1 on Hindi data

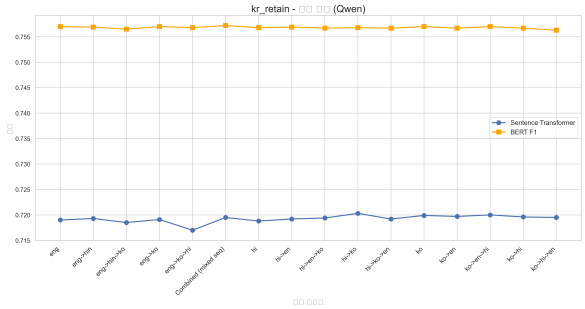

Figure 22: (Base-Qwen 7B) Retention performance comparison between Sentence Transformer and BERT F1 on Korean data

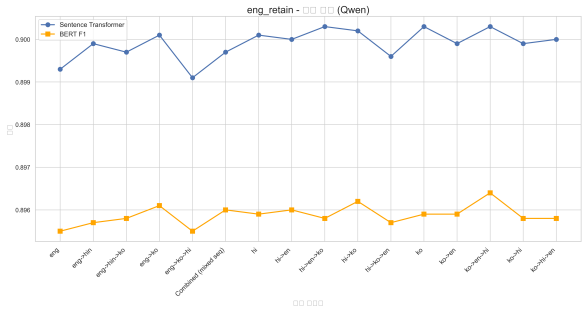

Figure 23: (Base-Qwen 7B) Retention performance comparison between Sentence Transformer and BERT F1 on English data