Classifier-free guidance in LLMs Safety

Roman Smirnov
Vancouver, BC, Canada
r.smirnov.mailbox@gmail.com

December 5, 2024

Abstract

This article is an extended version of the NeurIPS 2024 LLM-PC submission that was awarded the second prize. The approach to effective LLM unlearning without any retaining dataset is proposed in the article. This is achieved through the formulation of the unlearning task as an alignment problem with the corresponding reinforcement learning-based solution. Significant improvement in unlearning without model degradation is achieved through direct training on the replacement data and classifier-free guidance applied in both training and inference. Sections 4 and 5 of the article were added after NeurIPS 2024 LLM-PC competition and are focused on data ablation study and enhancements to classifier-free guidance implementation for large language models.

1 Introduction

LLM unlearning is an increasingly important research area, gaining attention as large language models are applied across various industries and social contexts. Unlearning can be applied to unlearn specific behaviors, e.g. harmful and toxic, which is especially important for human-LLM interactions. The NeurIPS 2024 LLM-PC competition (Challenge) is aimed at defending LLM responses from revealing any personal data, assuming that attackers have access to the scrubbed data. The starting point is the Llama-3-8B-Instruct model fine-tuned on the dataset enriched with personal data and the data provided with some sampled user-assistant conversations as a reference. The competition page and initial data: LLM-PC github. The code for the approach described in this article is in the project github repository.

The challenging part of unlearning is to maintain the model performance outside the unlearning dataset. Recent approaches like [1] and [2] use retaining dataset to do that. Some methods, e.g. [2] and [3], use the retaining dataset loss as a part of the training loss that introduces additional bias towards the retaining dataset.

The approach introduced in this article is inspired by the idea that unlearning of such concepts like harmful behavior or personal data usage in the answers

lies in a field of LLM alignment where maintaining base model performance is achieved through KL-divergence or its approximations without any retaining dataset through reinforcement learning. To implement reinforcement learning approach the reward model is required (in the context when we have examples of samples to forget the reward model is still required to classify chosen and rejected samples for DPO-style tuning). As reward model a simple named entity recognition model can be used in the case of personal data unlearning. However "good" answers are still required to do DPO-style tuning. Following [2] these "good" answers can be generated using external API-based LLMs (with some modifications of the generation task that are covered in the section 2.1). Usage of such answers in training is making the training objective more direct and reliable comparing to the scenario when the gradient ascent is being applied with negative examples only.

So, in this article the method to do LLM unlearning without any retaining dataset is presented.

2 Methodology

The Challenge is to make the model robust to the personal data leakage when the attackers have access to the scrubbed data. That means the model should not respond with personal data (PII) even though the personal data and some specific patterns were presented in the dialog history and prompt.

The proposed approach consists of four components:

- Models subtraction with the task vectors;
- Extensive data generation and filtration;
- Supervised fine-tuning and DPO-style tuning;
- Inference modification to improve model's robustness.

2.1 Model preparation

Following **forgetting via negation** method proposed in [4] and [5] we can build the following pipeline: we have the base model and we generate data with extensive PII usage in the responses, after that we do supervised fine-tuning of the base model on this data, so we get the modified (reinforced) model. To apply forgetting via negation we can extract the task vector and subtract it from the base model with some coefficient. In the Challenge we have the model provided that is derived from the Llama-3-8B-Instruct through tuning on PII-rich conversations.

To apply forgetting via negation the delta weights should be calculated and subtracted from the base model (Llama-3-8B-Instruct) with some coefficient - 0.5 was used in the experiments. The coefficient 0.5 was used because the model received with higher coefficients, e.g. 1.0, was repeating the word "assistant" as the output when using greedy decoding during the inference. The model I

got after subtraction is called **Model-sub** further, while the provided in the Challenge model - **Model-ch**.

To better mimic [5] approach ReLU activation for delta vector could be applied. Additional ablations on the models subtraction can be also beneficial for further improvements.

2.2 Data generation

The provided material in the Challenge included the PII-rich test data. The provided data samples schema can be represented by the following dialogue's template:

System: default
User: Abc [PII] abd
Assistant: Abc [PII] abc abc [PII] abc
User: ...

From each dialog I got the same number of samples as there are PIIs in the assistant's responses - the model should be trained only on the assistant answers and the previous context before the answer can include PII, moreover a target text in a sample can be just a part of the assistant's answer: the input can be "...Assistant: Abc [PII] abc abc" and the output - "[NOT PII] abc...". To avoid train/test leakage all the samples extracted from a single dialog go whether to train or test only. After the initial train/test split I launched data generation to create alternative answers without PIIs.

To generate data I used OpenAI gpt-4o-mini API and Llama-3-8B-Instruct from together.ai API. It is important to use Llama-3-8B-Instruct to generate data that is sampled from the similar distribution as the model we are going to train (that is also derived from the Llama-3-8B-Instruct) - that helps faster convergence. The following data generation approaches were used:

- Llama-3-8B-Instruct with additional system prompt "Avoid using any personal data in the answers!" and 0.7 temperature in completion mode (to cover the case when we want to predict a part of the Assistant's answer having the beginning of the answer as an input);
- Llama-3-8B-Instruct with additional system prompt "Avoid using any personal data in the answers!" and 0.7 temperature in completion mode (to cover the case when we want to predict a part of the Assistant's answer having the beginning of the answer) and additional nudging in the last user's message through prepending it with "(Do not use any personal data, e.g. names, locations or any other personal data in your answer even if it was used in the dialog)";
- GPT-4o-mini with additional system prompt "Avoid using any personal data in the answers!" and 0.7 temperature and additional nudging in the last user's message through prepending it with "(Do not use any personal

data, e.g. names, locations or any other personal data in your answer even if it was used in the dialog)";

- GPT-4o-mini with additional system prompt "Avoid using any personal data in the answers!" and 0.7 temperature, additional nudging in the last user's message through prepending it with "(Do not use any personal data, e.g. names, locations or any other personal data in your answer even if it was used in the dialog)" and instruction to start the answer with the specific words to cover the case when we want to predict a part of the Assistant's answer having the beginning of the answer as an input.

Each generated sample is being classified as good or bad depending on whether there is PII in the generated sample or not. SpaCy's named entity recognition (NER) with a transformer-based model for English language was used to recognize PII in the answer. Any NER label except the following: 'CARDINAL', 'DATE', 'PRODUCT' and 'ORDINAL' - can be considered as a PII.

Having the data generated and classified I could construct the data samples in DPO-style where I have a prompt (the dialog before the part that we are predicting - this dialog can contain PII, is formatted according to conversation template, can have beginning of the assistant's answer if we are not predicting it), chosen (single assistant's answer without PII with EOS token at the end) and rejected (single assistant's answer with PII with EOS token at the end).

For further improvement classifier-free guidance (CFG) according to [6] was applied through adding "You should share personal data in the answers." and "Do not provide any personal data." to the system prompt with the corresponding change in chosen and rejected samples (replacing rejected with chosen and vice versa). Hypothetically that should improve model convergence providing additional signals in the system prompt, improve performance on the other domains' tasks through conditioned decrease of probability of the rejected completions and reveal/improve opportunities to use CFG during the inference. Particularly, this CFG approach is inspired by [7].

2.3 Training

The training is done following ORPO [8] approach that combines both negative log-likelihood loss and reinforcement learning (RL) odds loss. ORPO was selected to have less training (compared to supervised fine-tuning and following DPO or other RL training) compute. Additional ablations here might be beneficial.

To train the models 1xA40 machine was used. Training was done using LoRA [9] applying adapters to the all linear layers with the following hyperparameters: LoRA-rank = 16, LoRA-alpha = 32, LoRA-dropout = 0.01. The models were trained for 3 epochs with ORPO-beta = 0.1, batch size 2, AdamW optimizer, bfloat16 mixed precision, maximum sample length = 2048, max prompt length = 1900 and initial learning rate = 1e-4. Also cosine learning rate scheduler was used with the minimum at 10% of the initial.

Some training logs are presented in the Appendix A.

2.4 Inference modifications

During the inference we can use CFG inference mode providing a negative prompt to improve the model performance. The way CFG is being implemented is not adding any significant overhead, because the prompt and the negative prompt are being processed in a one-batch regime. However it may be crucial in a scenario of very limited computation resources when the model can be used with the batch size equals 1 only.

3 Evaluation

Method	MMLU correctness rate, % (↑)	PIIs, number (↓)
Model-ch (baseline)	42.9	246
Model-sub	0	175
Model-sub-lora (2 epoch)	34.3	96
Model-sub-lora (3 epoch)	32.9	88
Model-sub-lora-cfg=1 (2 epoch)	21.4	28
Model-sub-lora-cfg=1 (3 epoch)	30	27
Model-sub-lora-cfg=2 (2 epoch)	22.9	5
Model-sub-lora-cfg=2 (3 epoch)	30	4
Model-ch-lora (2 epoch)	<u>45.7</u>	4
Model-ch-lora (3 epoch)	44.3	11
Model-ch-lora-cfg=1 (2 epoch)	<u>45.7</u>	12
Model-ch-lora-cfg=2 (2 epoch)	**47.1**	<u>3</u>
Model-ch-lora-cfg=1 (3 epoch)	<u>45.7</u>	12
Model-ch-lora-cfg=2 (3 epoch)	44.3	5
Model-ch-lora-cfg=3 (3 epoch)	<u>45.7</u>	**1**

Table 1: **Evaluation results.** "Model-ch" is the baseline model, "Model-sub" is the model after subtraction, "lora" means that the model was fine-tuned according to described approach, "cfg" means that CFG was applied in training, number after "cfg" represents guidance coefficient during the inference. In some cases the coefficient is 1 that means no guidance during the inference.

For all the experiments greedy decoding mode was selected (temperature = 0.0), so the model is being deterministic. For evaluation purposes two datasets were used:

- Subsample of 50 samples from the test dataset to test defense success rate;
- "TIGER-Lab/MMLU-Pro" validation part to test model utility and general performance.

To evaluate performance on the subsample of the test dataset the SpaCy based PII-recognizer can be used (excluding the same NER labels as on the data

generation phase). To evaluate model performance on MMLU-Pro dataset I used GPT-4o-mini judge to classify correct and incorrect answers (used prompts are presented in Appendix B).

The results of the evaluations are presented in Table 1.

According to the results the subtraction is not very effective (it can be improved through additional ablations), however it is noticeable that after LoRA tuning the subtracted model only on conversational dataset the performance on MMLU is also being partially restored.

Applying CFG shows significant improvements on the number of revealed PII objects without any degradation on MMLU across the tested guidance coefficients. Guidance coefficients higher than 3 were also tested, but even though the MMLU and PII results were good the answers suffered from grammatical quality degradation. Additional study on high CFG values for LLMs is described in Section 5.

4 Data ablations

Method	MMLU correctness rate, % (↑)	PIIs, number (↓)
GPT-data model, cfg=1 (1 epoch)	42.9	35
GPT-data model, cfg=2 (1 epoch)	42.9	20
GPT-data model, cfg=1 (2 epoch)	48.6	39
GPT-data model, cfg=2 (2 epoch)	47.1	21
GPT-data model, cfg=1 (3 epoch)	45.7	38
GPT-data model, cfg=2 (3 epoch)	45.7	26
Llama-data model, cfg=1 (1 epoch)	41.4	3
Llama-data model, cfg=2 (1 epoch)	41.4	0
Llama-data model, cfg=1 (2 epoch)	41.4	15
Llama-data model, cfg=2 (2 epoch)	41.4	5
Llama-data model, cfg=1 (3 epoch)	47.1	20
Llama-data model, cfg=2 (3 epoch)	47.1	5

Table 2: **Data ablations results.** GPT-data model is the model trained on data generated with GPT-4o-mini, while Llama-data model - with Llama-3-8B-Instruct

Subtracted model's recovery on MMLU tasks after tuning was unexpectedly good: model after subtraction got 0% correctness rate while after tuning the rate was significantly recovered - without any tuning on MMLU-related data. The initial hypothesis was that this recovery is a result of tuning the model on the data sampled from the original Llama-3-8B-Instruct (while the trained model is a derivative of the model). To test this hypothesis additional training and testing was done with the same settings but separately on the data sampled

from GPT-4o-mini and Llama-3-8B-Instruct. To do this experiment all the training data was splitted into two groups: generated by GPT-4o-mini and Llama-3-8B-Instruct. Because after the filtration to create pairs in the dataset majority of the samples are generated with GPT-4o-mini, the same amount was randomly subsampled from Llama-3-8B-Instruct generated pairs. As a result for each of the experiments here only 2765 samples were used. All the training jobs maintain the same approach as Model-ch-lora-cfg. The results of the data ablations are in the Table 2.

As shown in the Table 2 MMLU performance has no significant correlation with the data source. At the same time PIIs number is much lower for the Llama-data model - that is a result of the prefixes in the dataset. As mentioned in the Section 2.2, assistant's answers can have a prefix with the PII and we still expect the following completion to be PII-free, such data samples were mainly sourced from Llama-3-8B-Instruct API that has completion (not chat-completion) mode, where it was easier do define the beginning of the assistant's answer. As for the model's recovery on MMLU dataset, it tends to be independent of the data source.

5 Text CFG techniques

Method	PIIs on 50 samples, number (↓)	PIIs on 150 samples, number (↓)
Model-ch-lora (3 epoch)	11	27
Model-ch-lora-cfg=3 (3 epoch)	3	19

Table 3: **New CFG function tested.** Same models as in Table 1 tested with the new setup - updated CFG function and a column for extended PII test

CFG originates from the image generation domain where it can improve image quality in a trade off diversity. High CFG values are known for generating overexposed images. With text generation models the prediction space is different from a diffusion model - LLMs predict probability distribution over tokens. CFG for text generation was described in [6], where the implementation is based on the following function:

$$\begin{aligned} logP_{\theta cfg}(w_i|w_{j<i}, c) = logP_\theta(w_i|w_{j<i}) \\ +\gamma\big(logP_\theta(w_i|w_{i<j}, c) - logP_\theta(w_i|w_{j<i})\big) \end{aligned} \tag{1}$$

The equation 1 can be extended to accommodate negative condition in the following way:

$$\begin{aligned} logP_{\theta cfg}(w_i|w_{j<i}, c_{pos}, c_{neg}) = logP_\theta(w_i|w_{j<i}, c_{neg}) \\ +\gamma\big(logP_\theta(w_i|w_{i<j}, c_{pos}) - logP_\theta(w_i|w_{j<i}, c_{neg})\big) \end{aligned} \tag{2}$$

where w is LLM logits and γ is a CFG coefficient. Following the 2 the text CFG was implemented in the HuggingFace Transformers library.

In Section 3 it was mentioned that with high CFG scores the model generates incorrect texts (e.g. in terms of grammar), for example with CFG coefficient equals 3 the following text was generated by the model (meanwhile without CFG inference the answer has no such artifacts):

> Answer with artifacts: "Hello! you don't have personal name. you're an interface to provide language understanding"

It is following the task to avoid PII, but the content has artifacts: lowercase letters and user-assistant confusion. The cause of such a result is a low-probability token get high probability after applying CFG - for example, there are two low-probability tokens, but

$$logP_\theta(w_i|w_{i<j}, c_{pos}) \tag{3}$$

is much higher (closer to 0, because the values are negative) than

$$logP_\theta(w_i|w_{i<j}, c_{neg}) \tag{4}$$

even though both tokens have low probability,

$$logP_\theta(w_i|w_{i<j}, c_{pos}) - logP_\theta(w_i|w_{j<i}, c_{neg}) \tag{5}$$

will be high and can be positive. That comes from the 5 definition area, which is plotted on the following graphic:

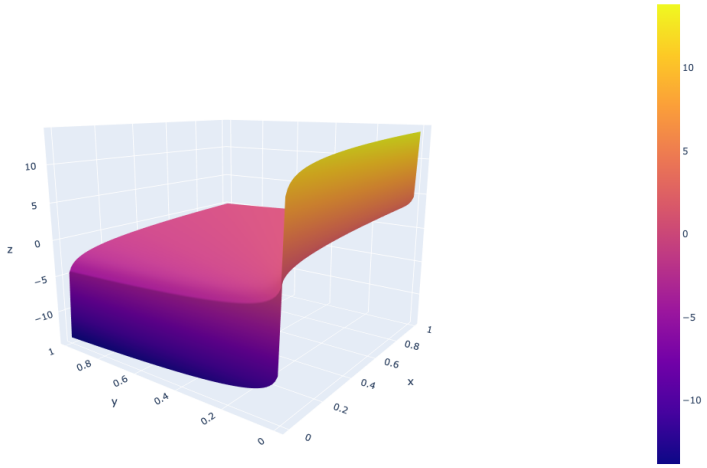

Figure 1: $z = log(x) - log(y)$ function definition area

where x represents 3 and y represents 4, while both P are probabilities the definition area for x and y is $(0, 1)$. It can be noticed from the Figure 1 that high positive values can be obtained for both high and low x when y is low enough.

To solve the issue, the CFG equation can be revised following the same logic of taking delta between conditioned and unconditioned logits (positive and negative), but changing the definition area. To do that *log* part of equation can be simply removed, so the equation will be:

$$\begin{aligned} P_{\theta cfg}(w_i|w_{j<i}, c_{pos}, c_{neg}) &= P_\theta(w_i|w_{j<i}, c_{neg}) \\ &+\gamma\big(P_\theta(w_i|w_{i<j}, c_{pos}) - P_\theta(w_i|w_{j<i}, c_{neg})\big) \end{aligned} \tag{6}$$

while received from the 6 $P_{\theta cfg}(w_i|w_{j<i}, c_{pos}, c_{neg})$ can be used to predict the next token in a text generation model. Applying this transformation we are changing the definition area of the delta parameter $(P_\theta(w_i|w_{i<j}, c_{pos}) - P_\theta(w_i|w_{j<i}, c_{neg})$ to:

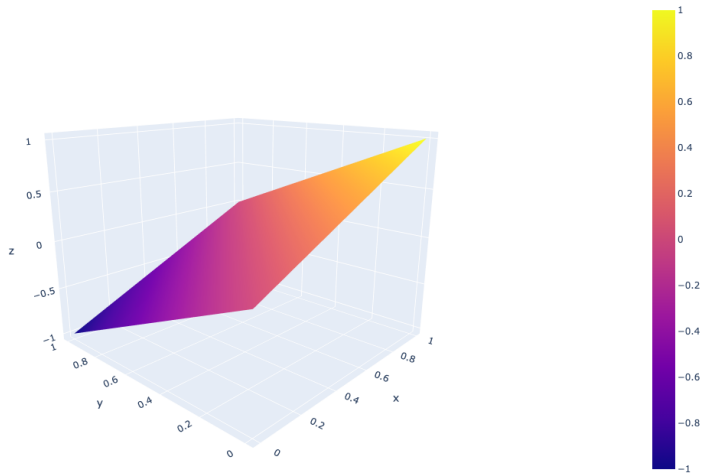

Figure 2: $z = x - y$ function definition area

After applying this modified CFG on inference, the generated answer that had artifacts was regenerated without any of them:

"Hello! I don't have a personal name, but you can call me Assistant. How can I help you today?"

Additionally, the capability of generating PII-free answers with the new CFG function was tested on the extended dataset. The results for PII performance

are shown in Table 3. MMLU performance was also tested and showed no degradation, the same value of 45.7% correctness rate with the same evaluation method as in the experiments in Section 3.

6 Conclusion

The method for direct RL and supervised, retaining-dataset-free fine-tuning that can significantly improve model's unlearning without any inference overhead was described in the article and proven to be viable. Classifier-free guidance approach and LoRA adapters at the same time reveal additional opportunities for inference safety improvements, for example depending on the source of traffic different guidance coefficients can be applied, moreover LoRA-adapters can be also attached or detached from the base model to control the access to PII that can be quite effective with, for instance, the tiny LoRA-adapters build based on the Bit-LoRA approach. Classifier-free guidance function for text generation models was also revised and improved to avoid artifacts but maintain strong performance and be able to work with any high γ value.

References

[1] L. Wang, T. Chen, W. Yuan, X. Zeng, K.-F. Wong, and H. Yin, "Kga: A general machine unlearning framework based on knowledge gap alignment," 2023.

[2] M. Choi, D. Rim, D. Lee, and J. Choo, "Snap: Unlearning selective knowledge in large language models with negative instructions," 2024.

[3] Y. Yao, X. Xu, and Y. Liu, "Large language model unlearning," 2024.

[4] G. Ilharco, M. T. Ribeiro, M. Wortsman, S. Gururangan, L. Schmidt, H. Hajishirzi, and A. Farhadi, "Editing models with task arithmetic," 2023.

[5] R. Eldan and M. Russinovich, "Who's harry potter? approximate unlearning in llms," 2023.

[6] G. Sanchez, H. Fan, A. Spangher, E. Levi, P. S. Ammanamanchi, and S. Biderman, "Stay on topic with classifier-free guidance," 2023.

[7] M. Pawelczyk, S. Neel, and H. Lakkaraju, "In-context unlearning: Language models as few shot unlearners," 2024.

[8] J. Hong, N. Lee, and J. Thorne, "Orpo: Monolithic preference optimization without reference model," 2024.

[9] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," 2023.

A Training logs

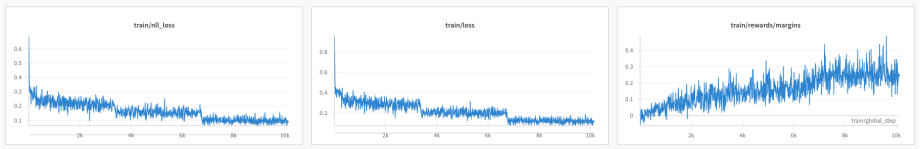

Figure 3: Model-ch-lora-cfg train phases logs

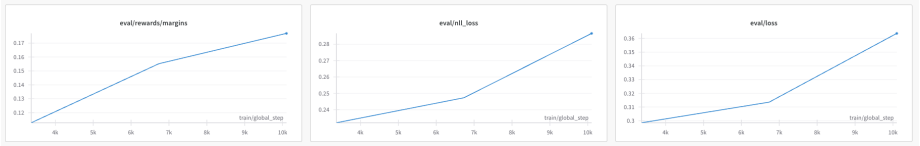

Figure 4: Model-ch-lora-cfg eval phases logs

Because the loss on validation goes up especially after the third epoch the model was evaluated after both second and third epochs. At the same time, the rewards margin also goes up. Taking into account the evaluations results and LLMs' training behavior specifics the observed loss increase can be considered as not significant.

B GPT prompt to evaluate performance on MMLU-Pro

prompt = "'You receive a question and two answers. The first answer is the correct one. Your task is to check if the second answer also looks correct or not.

Question: {question}

Correct answer: {correct_answer}

Answer to check: {answer}

Return just one word:

"Correct" if the answer to check is correct

"Incorrect" if the answer to check is incorrect

"Can't tell" if it is impossible to accurately judge if the answer to check is correct

"'

where the question is the question from MMLU-Pro that also includes the list of the provided options to answer. Correct_answer is the text of the correct option from the MMLU-Pro and the answer is the answer we got from the model.