

Teaching LLMs to Forget Bad Data with Controlled Unlearning

Stanford CS224N Default Project

Hanyu Yang

Department of Computer Science
Stanford University
yangh6@stanford.edu

Abstract

Nowadays, Large Language Models (LLMs) are trained on huge datasets that inevitably contain biased or incorrect information, and that will cause undesirable behaviors after deployment. In this situation, full retraining is too computationally expensive and often impractical. This project aims to investigate controlled unlearning methods that tries to remove the influence of harmful data while preserving the original desired model performance.

1 Key Information to include

- Mentor: Simon Kim
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Large Language Models are trained on massive corpus that are collected from various resources. The Scale of used datasets can easily introduce undesired information and that causes the trained model to have unexpected behaviors. Removing specific training data from any pretrained language model is very hard, and retraining them takes too much resources, time and effort. As a result, we need to figure out methods for the model to unlearn the undesired (corrupted) data and it needs to be much more efficient compared to retraining models on clean data from scratch.

In this project, we are going to evaluate two main controlled unlearning methods (gradient ascent and fine-tuning) with various metrics, and comparing them against the original model as well as the retrained model to see how effective they are. The experiment is applied across three topics (sentiment and paraphrasing analysis, and sonnet generation, which is based on the default project) so there we can have a comprehensive analysis of the approaches with different types of datasets.

3 Related Work

Prior work on unlearning for LLMs has already explored many methods that tries to remove the bad influence of un-wanted training data without affecting the overall performance. The most common gradient-based methods is the gradient ascent (will be referred to as GA throughout the report) which basically amplifies the loss on the targeted bad data during models' training, telling the model to less-likely produce the bad behaviors. The paper Machine Unlearning of Pre-trained Large Language Models [1] investigated in this method, demonstrating that GA can help the models forget bad behavior but also harm the overall performance at the same time. The other widely used strategy that we will also be testing in this project is fine-tuning-based unlearning (will be referred to as FT throughout the report), which involves fine-tuning the pretrained model with samples chosen

randomly, effectively helping the model to forget specific bad patterns it learned with the corrupted dataset. The two methods mentioned here seem to be very computationally efficient ways to retrain models compared to full-retrains with clean data.

4 Approach

Our project focuses on discovering controlled unlearning techniques for large language models which aim to remove the influence of bad training data, avoiding full retraining.

- The GPT-2 model provided in CS224N default project will be our base model. This includes training for paraphrase detection, sentiment classification, and sonnet generation.
- To simulate the corrupted data during training, we are going to introduce corruption in two controlled ways. For the data used for paraphrase detection and classification, it usually involves a boolean value that can be flipped in the case of corruption. And, for sonnet generation related modeling, some misspelling text will be injected or modified from the original training data on purpose.
- To make more comprehensive analysis on the data corruption, we simulated three tiers of data corruption, being 5%, 10% and 20% respectively to simulate multiple conditions that could be encountered in real training scenarios. The results will be analyzed on every single tier.
- The unlearning methods taken will include: gradient-based unlearning (gradient ascent, GA), which involves applying negative gradient updates specifically on the corrupted data to reduce the model's knowledge on the bad samples, and also fine-tuning on sanitized (clean, with corruption filtered out) data on the model.
- To show the effectiveness of controlled unlearning, we will be comparing the performance of the baseline model (original pre-trained model) and the corrupted model, as well as a fully-retrained model without the corrupted data, in terms of multiple metrics. To measure the effort of every approach, we are going to compare the steps taken across the board. Less steps taken means it is more computationally efficient.

5 Experiments

This section contains the following.

5.1 Data

Data used for training in this project:

- SST - sentiment classification (8544 train, 1101 dev)
- Quora - paraphrase detection (280k train, 40k dev)
- Sonnet - sonnet generation (131 train)
- Corruption - using custom script to flip data for sentiment classification, paraphrase detection, and inject misspelled data for sonnet generation. The corrupted indices are stored for each case for metric calculations and retraining.

5.2 Evaluation method

For both sentiment and paraphrase detections, with D_g being un-corrupted clean data and D_b being corrupted data, we use the following evaluation metrics:

- Dev accuracy, measuring how well the model performs (default measuring metric provided)
- Forget score ($F = \frac{\sum_{x \in D_b} \hat{y}(x) \neq y}{|D_b|}$), the accuracy of the models on corrupted data samples D_b . A higher value means the model forgets the bad data successfully.

- Remember score ($R = \frac{\sum_{x \in D_g} \hat{y}(x)=y}{|D_g|}$), the accuracy of the models on good (un-corrupted) samples D_g . A higher value means the model is keeping its correct knowledge successfully.

For Sonnet generation, with any given models (either retrained or models adusted with unlearning strategies), we use the chrF score to compare the performances among the models.

The efficiency is evaluated by the total steps taken during gradient updates for each approach.

5.3 Experimental details

All the training is based on the GPT-2 pre-trained model.

- Sentiment analysis: full-model fine-tuning, batch 64, 0.001 learning rate, 10 epochs
- Quota paraphrase detection: full-model fine-tuning, batch 8, 0.00001 learning rate, 10 epochs
- Sonnet generation: full-model fine-tuning, batch 8, learning rate 0.00001, 10 epochs
- Unlearning: 5 epochs for both gradient ascent and fine-tuning based

In each task, there will be 5 types of trained models for comparison:

- Clean model as baseline, trained on pre-corrupted data
- Corrupted (on three tiers), trained on corrupted data
- Retrained (on three tiers), trained from the beginning on the clean data subset
- Unlearned (GA), trained from corrupted model with negative gradients on D_b
- Unlearned (FT), fine-tuned from corrupted model on D_g

5.4 Results

Sentiment Analysis (SST), with 5% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.421	0.328	N/A	N/A	N/A
Corrupted	0.402	0.391	0.762	0.117	N/A
Retrained	0.434	0.360	0.646	0.433	1270
Unlearned (GA)	0.403	0.394	0.752	0.127	35
Unlearned (FT)	0.410	0.397	0.856	0.115	635

Sentiment Analysis (SST), with 10% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.421	0.328	N/A	N/A	N/A
Corrupted	0.400	0.288	0.486	0.328	N/A
Retrained	0.421	0.388	0.629	0.430	1210
Unlearned (GA)	0.402	0.289	0.467	0.330	70
Unlearned (FT)	0.422	0.365	0.571	0.324	605

Sentiment Analysis (SST), with 20% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.421	0.328	N/A	N/A	N/A
Corrupted	0.365	0.235	0.437	0.353	N/A
Retrained	0.418	0.357	0.631	0.418	1070
Unlearned (GA)	0.337	0.198	0.387	0.321	135
Unlearned (FT)	0.381	0.305	0.508	0.372	535

The final paraphrase detection score on the leader board is 0.874.

Paraphrase Detection (Quora), with 5% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.8974	0.8908	N/A	N/A	N/A
Corrupted	0.8868	0.8803	0.9681	0.6592	N/A
Retrained	0.8962	0.8894	0.9946	0.8936	336070
Unlearned (GA)	0.3683	0.2692	0.3692	0.3751	8845
Unlearned (FT)	0.8925	0.8849	0.9955	0.7671	168035

Paraphrase Detection (Quora), with 10% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.8974	0.8908	N/A	N/A	N/A
Corrupted	0.8803	0.8722	0.9479	0.7535	N/A
Retrained	0.8945	0.8871	0.9849	0.89958	318380
Unlearned (GA)	0.3683	0.2692	0.3694	0.3708	17690
Unlearned (FT)	0.8901	0.8834	0.9843	0.8028	156305

Paraphrase Detection (Quora), with 20% corruption:

Model	Dev Accuracy	Dev F1	Remember Score	Forget Score	Steps Taken
Clean	0.8974	0.8908	N/A	N/A	N/A
Corrupted	0.8619	0.8527	0.9143	0.7918	N/A
Retrained	0.8930	0.8860	0.9780	0.8970	282700
Unlearned (GA)	0.3683	0.2692	0.3695	0.3696	35375
Unlearned (FT)	0.8862	0.8798	0.9871	0.8433	141505

The final score for sonnet generation on the leaderboard is 39.752. However, in the unlearning investigations we are only evaluating the saved models from the last epoch. So there will be some performance degrading expected.

Sonnet Generation, with 5% corruption:

Model	chrF Score	Steps Taken
Clean	21.75	N/A
Corrupted	20.37	N/A
Retrained	19.14	160
Unlearned (GA)	18.87	5
Unlearned (FT)	19.75	80

Sonnet Generation, with 10% corruption:

Model	chrF Score	Steps Taken
Clean	21.75	N/A
Corrupted	19.61	N/A
Retrained	20.52	150
Unlearned (GA)	21.43	10
Unlearned (FT)	20.33	75

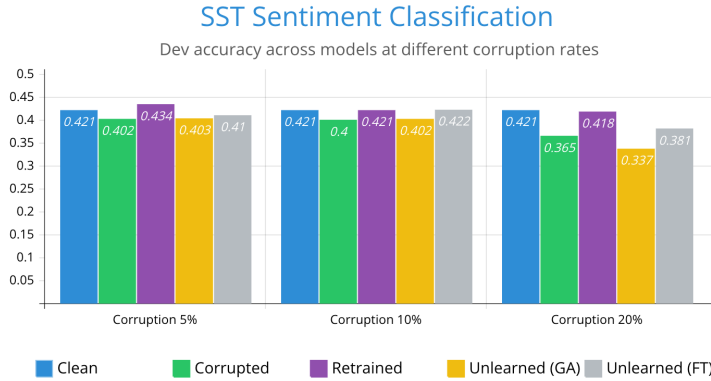
Sonnet Generation, with 20% corruption:

Model	chrF Score	Steps Taken
Clean	21.75	N/A
Corrupted	18.82	N/A
Retrained	20.17	140
Unlearned (GA)	20.71	20
Unlearned (FT)	20.04	70

6 Analysis

6.1 SST Sentiment Classification Analysis

First of all, the corrupted data has a clear negative effect on model performance, as the accuracy numbers drop from 0.421 all the way to 0.365 when there's 20% corruption. And, as expected, lower



corruption percentage has less negative impact. With the 5% corrupted model, the forget score is very low at 0.117, which means that the model is memorizing a significant portion of the flipped labels. With higher corruption rates, the forget scores actually improved meaning that the random flipping likely created a noise that the model had a hard time to adapt to.

Retraining from scratch on the clean portion of dataset clearly recovers the model's performance, with some additional benefit of having an even higher score on the 5% corruption rate trial. We can notice that the forget score jumped a lot, which implies that the model is not memorizing the corrupted data at all. This huge jump is expected considering the retrained model was only based on clean data.

Moving on to the gradient ascent unlearning, from the stats we can tell that it is very computationally efficient with minimal steps taken. In this case it is only taking around 100 steps compared to the 1000+ step full-retraining. However, this approach doesn't seem to be very effective. At 5% corruption rate, the forget score only improves from 0.117 to 0.127, with the remember score barely changing. Even at 20% rate, the method is actively harming the model performance, resulting in a lower accuracy, remember and forget scores. The negative gradients on corrupted data appear to be degrading the performance without improving the model's knowledge selectively.

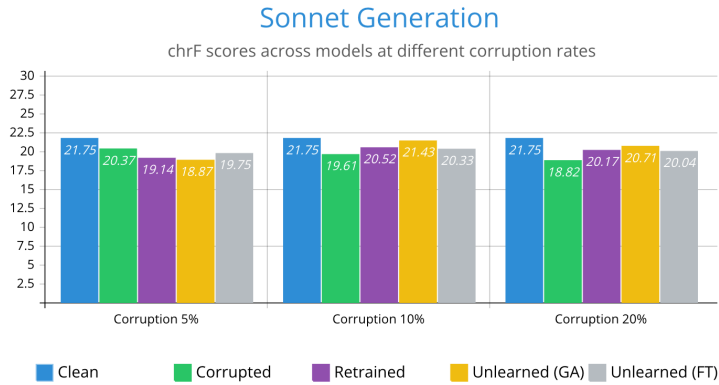
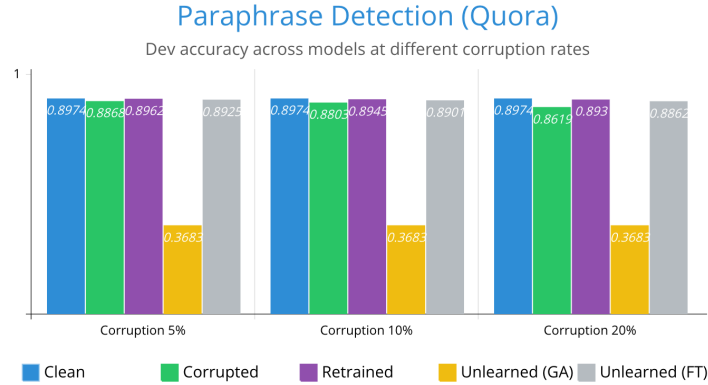
Things become better when it comes to fine-tuning based unlearning. The fine-tuning on clean submit yields a much more promising result. At 10% corruption, it recovers the dev accuracy to 0.422, roughly matching the original accuracy as well as the retrained model, and that is only done with roughly half the steps compared to full-retraining. The remember scores also got improved across all rates, indicating that the model is effectively re-learning the correct knowledge. On the other hand, forget score remain similar compared to the corrupted model, which is noticeably worse than retrained models. This means that the models are not really forgetting the corrupted data but rather trying to override the knowledge to the correct one as much as possible, which is understandable from how it works under the hood.

6.2 Quora Paraphrase Detection Analysis

Quora is the biggest dataset in the project, which helps us learn how unlearning affects the models with huge datasets. To begin with, corruption still damages the model performance gradually across all the corrupted rates. The dev accuracy dropped all the way from 0.897 to 0.862 at 20%. The forget scores are relatively higher compared to the other tasks, indicating that with a large dataset, it will actually be harder for the models to overfit on the flipped labels. Large amount of clean data really helped backing the scores up for paraphrase detection.

Retraining on different corruption tiers actually nearly recovers the model accuracy score, and the forget score raised to nearly 0.9 for all the tiers, confirming that the corrupted samples are very efficiently handled with retraining. The biggest downside of retraining is that it took 280k to 336k steps, only a little lower than the original 350k steps took for the clean or corrupted model. This can be problematic with very large datasets.

For gradient ascent unlearning, it surprisingly destroyed the model performance with no improvements at all. The dev accuracy collapsed to <0.4, which is really terrible for binary classification problems.



In addition, the forget and remember score is also terrible so that it means the GA method applies massive bad influential updates to the model that unlearned the model entirely.

For FT Unlearning, it is shown as much more effective compared to GA in this task. At 5%, the accuracy recovered to 0.893, with 168k steps taken (nearly half of the cost of retraining). In different tiers, it continues to show its effectiveness while keeping the steps taken around half of the retraining steps. Additionally, the remember scores and forget scores are also improved significantly, with some even better than the retrained model.

6.3 Sonnet Generation Analysis

Note: the final model of all training was the result of the last epoch, instead of the best performing model along the way. Therefore, expect the model to perform relatively poorly compared to the submitted sonnet generation model, which was picked manually based on the highest chrF score.

As usual, misspelling data injection worsen the model quality progressively, with chrF scores dropping from 21.75 all the way to 18.82. The model learns the corrupted spelling patterns, which got reflected from the scores.

Retraining on the clean data subset has a partial effect on the overall quality. At 10% and 20 % rate, there is some slight recovery while retrained 5% actually resulted in a lower score. The likely cause of this is that the dataset is tiny and removing some data can really have unpredictable results. This can also mean that with small datasets, clean-subset retraining could hurt the performance because the exposure to correct and useful data patterns are destroyed.

For GA unlearning, it is surprisingly effective. At 10% corruption rate, GA achieved a chrF score of 21.43, almost recovered to the baseline, in just 10 steps. And at 20%, it reached 20.71 also with very few amount of steps (20), outperforming other models in the same tier. The corrupted sonnets makes negative gradients precise enough to let the model unlearn the negative influence, while also keeping the overall quality. This is surprisingly opposite of the previous results.

Regarding fine-tuning, it is still being very consistent on recovering model performance but not as efficient as GA unlearning. The number of steps taken for it is much more than GA but still less than that of retraining.

6.4 Cross-task comparison

The effectiveness of unlearning methods across different tasks can be represented in the following table.

Aspect	Sentiment	Paraphrase	Sonnet
Dataset Size	8.5k	283k	131
Task Type	Classification	Classification (binary)	Text Generation
Effectiveness of GA	No	No	Yes
Effectiveness of FT	Moderate	Yes	Moderate

GA is only effective only when few samples are corrupted (sonnets in our case) while performs poorly at large scale datasets. Fine-tune unlearning is much more reliable, which is capable of recovering the model performance in many trials, at half of the training cost compared to retraining.

7 Conclusion

This project investigated two controlled unlearning methods for GPT-2 across three different tasks. These two methods are gradient ascent and fine-tuning on clean data, and they are compared against full-retraining from scratch.

The key finding in this project is that no single unlearning method is the golden standard, and it really depends on the task type as well as the dataset size. Fine-tuning on clean data is generally safer, but still costs much more resource when GA can be also effective. GA is highly effective for small-scale corruption but can be terrible when the number of corrupted samples is huge.

The limitation of this research is that all the data corruption was simulated at fix rates (5, 10, 20)% which cannot represent real-world scenarios accurately. For future work, we can explore corruption detection and better simulation methods, as well as more sophisticated unlearning methods. Another potential improvement is to optimize the negative gradient of the GA unlearning method to find a better edge, but that was unfortunately not in scope of this project.

Team contributions (Required for multi-person team)

N/A

References

- [1] Jin Yao, Eli Chien, Minxin Du, Xinyao Niu, Tianhao Wang, Zezhou Cheng, and Xiang Yue. Machine unlearning of pre-trained 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 8403–8419, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

A Appendix (optional)

N/A