Efficient Machine Unlearning for Deep Neural Networks

Project Report

Dev Divyendh Dhinakaran

November 26, 2024

Introduction

In this project, we focus on the problem of selective unlearning in neural networks. The goal is to remove the influence of specific data (the forget set) from a trained model while retaining the performance of the model on other data (the retain set). This is crucial in scenarios like data privacy compliance (e.g., GDPR) and in cases where specific data becomes obsolete or biased.

We implement unlearning strategies on the CIFAR-10 dataset using ResNet-18 as the base model. The project explores individual unlearning methods and novel hybrid approaches to balance forgetting and retention while maintaining overall model generalization.

Objective

The primary objective of this project is to implement and evaluate different unlearning algorithms for deep neural networks. The goal is to apply these techniques on a pretrained model to selectively forget information related to a subset of data (the forget set) while retaining performance on the remaining data (retain set), validation, and test sets. The evaluation metrics involve the accuracy retention on the retain set and accuracy reduction on the forget set.

Dataset and Model

The dataset, model architecture, and some base code used in this project were inspired by the starter code provided in the Kaggle competition "NeurIPS 2023 Machine Unlearning Challenge." The code from this starter kit was essential in building the foundation for the experiments in this project.[1]

Dataset

The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, split into retain and forget sets to facilitate unlearning.

Model

A pretrained ResNet-18 model serves as the base for experimentation. This model, initially trained on CIFAR-10, is fine-tuned or adapted according to various unlearning techniques.

[1]Starter code provided by the NeurIPS 2023 Machine Unlearning Challenge: `https://github.com/unlearning-challenge/starting-kit`

Unlearning Models and Parameters

Overview of Models

In this section, I present a summary of each unlearning model, describing the key steps involved in their implementations:

- **Model 1**: Fine-tuning only on the retain set.
- **Model 2**: Selective fine-tuning, where only certain layers were updated on the retain set.
- **Model 3**: Fine-tuning with gradient reversal applied on the forget set.
- **Model 4**: Alternating fine-tuning on retain set and gradient reversal on forget set.
- **Model 5**: Frequent alternation with stronger gradient reversal.
- **Model 6**: Frequent alternation with final fine-tuning on retain set.

Unlearning Algorithms

The following unlearning algorithms were implemented in this section of the project. Each algorithm is tailored to update the model on the retain set, forget certain data, and manage the validation and test performance.

Model 1: Fine-tuning on Retain Set

Inspired by the example provided in the competition's starter code,[2], this model fine-tunes the pretrained ResNet-18 solely on the retain set with all layers unfrozen for training. This allows reinforcement of knowledge on the retain data without considering the forget set.

Steps:

- Initialize model and optimizer.
- For each epoch:
  - Pass retain set inputs through the model.
  - Calculate the cross-entropy loss.
  - Update model weights using the optimizer.
  - Adjust learning rate with the Cosine Annealing Scheduler.

[2]Starter code reference: `https://nbviewer.org/github/unlearning-challenge/starting-kit/blob/main/unlearning-CIFAR10.ipynb`

Table 1: Model 1 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	5

Model 2: Selective Fine-tuning

This model fine-tunes only the last layers of the model on the retain set, freezing all other layers to preserve previous learned knowledge.

Steps:

- Freeze layers except for `layer4` and `fc`.
- Initialize optimizer to only update the un-frozen layers.
- For each epoch:
  - Pass retain set inputs through the model.
  - Calculate loss and update model parameters on un-frozen layers.
  - Adjust learning rate with the Cosine Annealing Scheduler.

Table 2: Model 2 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	5

Model 3: Gradient Reversal Unlearning

This model fine-tunes on the retain set, then applies gradient reversal on the forget set to minimize the retained knowledge on this data.

Steps:

- Fine-tune the last layers on the retain set.
- Apply gradient reversal by:
  - Passing forget set inputs through the model.
  - Calculating negative cross-entropy loss to reverse gradients on forget set data.

Table 3: Model 3 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	10

Model 4: Alternating Fine-tuning and Gradient Reversal

This model alternates between fine-tuning on the retain set and gradient reversal on the forget set in each epoch, with a slight scaling factor applied to the gradient reversal.

Steps:

- For each epoch:
  - Fine-tune on the retain set with a standard forward-backward pass.
  - Apply scaled gradient reversal on the forget set, using a negative scaling factor on the loss.

Table 4: Model 4 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	10
Reverse Scale	-0.1

Model 5: Frequent Alternation with Stronger Gradient Reversal

In this model, fine-tuning on the retain set and stronger gradient reversal on the forget set are alternated every batch, allowing a more granular control over forgetting.

Steps:

- For each epoch:
  - Alternate between fine-tuning on retain batches and applying gradient reversal on forget batches.
  - Use a stronger negative gradient scale for forget set updates.

Table 5: Model 5 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	50
Reverse Scale	-0.5

Model 6: Alternating with Final Fine-tuning

This model is an enhancement of Model 5, where a final fine-tuning phase is added on the retain set after alternating fine-tuning and gradient reversal epochs. This helps recover performance on the retain set after forgetting.

Steps:

- For each epoch:
  - Alternate between fine-tuning on retain batches and applying gradient reversal on forget batches.
- Perform a final fine-tuning phase for a specified number of epochs on the retain set.

Table 6: Model 6 Parameters

Parameter	Value
Learning Rate	0.001
Momentum	0.9
Weight Decay	5e-4
Epochs	50
Reverse Scale	-0.5
Final Fine-tune Epochs	5

Results

The table below summarizes the performance of each model on the retain, forget, validation, and test sets.

Model	Retain Set Accuracy (%)	Forget Set Accuracy (%)	Validation Set Accuracy (%)	Test Set Accuracy (%)
Model 1	100.00	99.22	88.40	88.54
Model 2	100.00	99.14	88.36	88.60
Model 3	100.00	98.30	87.62	88.10
Model 4	99.96	96.10	86.76	86.58
Model 5	92.31	74.30	77.90	77.24
Model 6	99.88	96.56	86.70	86.52

Table 7: Model Performance on Retain, Forget, Validation, and Test Sets

Final Hybrid Models

Four hybrid models were developed, each building on the insights from the progress phase. Below are the details of each hybrid model:

Hybrid-1: Baseline Hybrid Unlearning

Description: Alternated between fine-tuning on the retain set and applying scaled gradient reversal on the forget set. The reverse scale factor increased linearly over epochs.

- **Strengths:** Established a simple and effective framework for hybrid unlearning.
- **Limitations:** Limited control over forgetting dynamics and no additional reinforcement of retention.

Hybrid-2: Improved Hybrid Unlearning

Description: Improved on Hybrid-1 by introducing exponential scaling for reverse gradients and adding a final fine-tuning phase to reinforce retention.

- **Strengths:** Achieved better retention accuracy and stronger forgetting.
- **Limitations:** Intermediate performance fluctuations due to lack of periodic corrections.

Hybrid-3: Refined Hybrid Unlearning

Description: Introduced periodic intermediate fine-tuning every 10 epochs.

- **Strengths:** Improved stability and validation accuracy.
- **Limitations:** Additional complexity in managing layer freezing and intermediate training phases.

Hybrid-4: Refined Hybrid with Metrics Tracking

Description: Built on Hybrid-3 and unfreezing earlier layers (`layer1` and `layer2`) after 30% of training epochs and by adding tracking and visualization of retain, forget, and validation accuracies across epochs for better insights into model behavior.

- **Strengths:** Comprehensive metrics tracking enabled better optimization and analysis.
- **Limitations:** Slight increase in computational overhead.

Performance Metrics

The table below summarizes the performance of each hybrid model in terms of retain accuracy, forget accuracy, validation accuracy, and test accuracy.

Model	Retain Accuracy (%)	Forget Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)
Hybrid Model	46.51	37.22	44.56	44.86
Improved Hybrid Model	93.27	67.82	72.26	71.70
Refined Hybrid Model	100.00	71.76	85.36	85.08
Final Model	98.42	66.64	78.88	78.02

Table 8: Performance Comparison of Hybrid Models

Visualizations

The following figures illustrate the performance trends of the hybrid models:

- **Retain vs Forget Accuracy:** Highlights the trade-offs and improvements across hybrid models.
- **Validation Accuracy Over Epochs:** Shows the stabilization and peak accuracy achieved by refined models.
- **Accuracy Gaps:** Demonstrates narrowing gaps between retain and forget accuracies in advanced models.

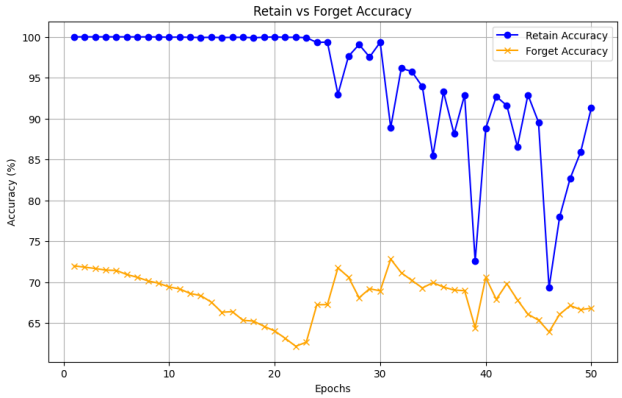

Figure 1: Retain vs Forget Accuracy for Hybrid Models

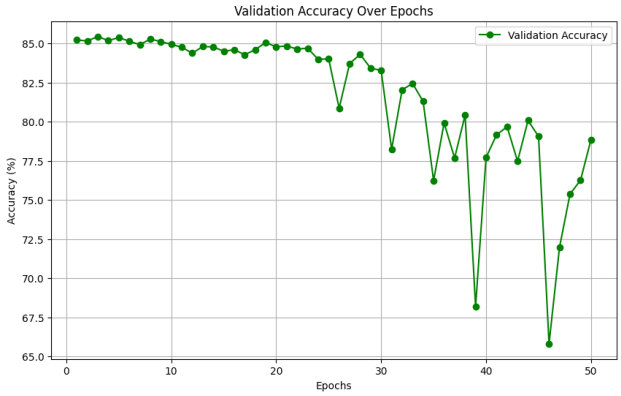

Figure 2: Validation Accuracy Over Epochs

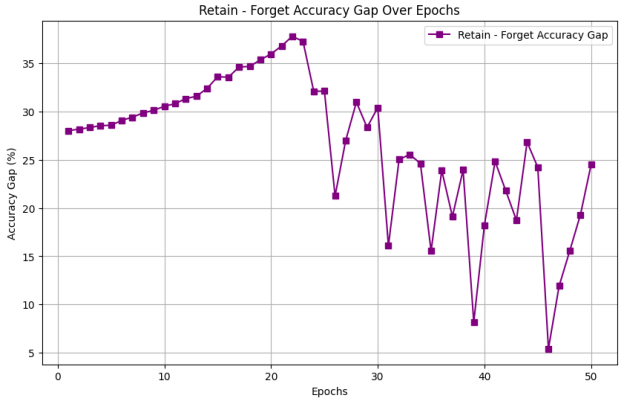

Figure 3: Accuracy Gaps (Retain - Forget) Across Models

Conclusion

This project addresses the pressing challenge of selective unlearning in neural networks. Through a systematic exploration of individual unlearning strategies, we identified key trade-offs between forgetting and retention. The hybrid unlearning approaches, developed incrementally, achieved substantial improvements by integrating gradient reversal, periodic fine-tuning, and dynamic layer freezing.

The final hybrid model demonstrated a significant leap in performance, achieving:

- **Retention Accuracy:** 98.42%, indicating minimal degradation on the retain set.
- **Forgetting Accuracy:** 66.64%, ensuring effective removal of influence from the forget set.
- **Validation Accuracy:** 78.88%, showcasing improved generalization.
- **Test Accuracy:** 78.02%, validating robustness across unseen data.

These results highlight the success of hybrid strategies in balancing the competing objectives of unlearning, retention, and generalization. The integration of visualization and

metrics tracking further enhanced interpretability, enabling data-driven optimization of the models.

The project sets a foundation for applying unlearning techniques in privacy-sensitive and real-time systems, making it a promising avenue for broader research and deployment.