

DistilRepair: Utility-Preserving Unlearning via Distilled Gradient Repair

by

**Urbana Jaman Orthee
221014103**

*Thesis (CSE4099-B) submitted in partial fulfillment of the
requirements for the degree of*

Bachelor of Science in Computer Science and Engineering

Under the supervision of

Nafees Mansoor, PhD



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF LIBERAL ARTS BANGLADESH**

SPRING 2026

© *Urbana Jaman Orthee*
All rights reserved

DECLARATION

Project Title	DistilRepair: Utility-Preserving Unlearning via Distilled Gradient Repair
Author	<i>Urbana Jaman Orthee</i>
Student ID	221014103
Supervisor	Nafees Mansoor, PhD

I declare that this thesis entitled *DistilRepair: Utility-Preserving Unlearning via Distilled Gradient Repair* is the result of my own work except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Urbana Jaman Orthee
221014103

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Date: April 21, 2026

CERTIFICATE

This is to certify that the thesis entitled **DistilRepair: Utility-Preserving Unlearning via Distilled Gradient Repair**, submitted by **Urbana Jaman Orthee** (Student ID: 221014103) an undergraduate student of the **Department of Computer Science and Engineering** has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Computer Science and Engineering*.

Nafees Mansoor, PhD
Professor

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Muhammad Golam Kibria, PhD, SMIEEE
Professor & Head

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Place: Dhaka

Date: April 21, 2026

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my research supervisor, *Nafees Mansoor, PhD*, for giving me the opportunity to do research and providing invaluable guidance throughout this work. His dynamism, vision, sincerity and motivation have deeply inspired me. He has taught me the methodology to carry out the work and to present the works as clearly as possible. It was a great privilege and honor to work and study under his guidance.

I am greatly indebted to my honorable teachers of the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh who taught me during the course of my study. Without any doubt, their teaching and guidance have completely transformed me to a person that I am today.

I am extremely thankful to my parents for their unconditional love, endless prayers, caring and immense sacrifices for educating and preparing me for my future. I would like to say thanks to my friends and relatives for their kind support and care.

Finally, I would like to thank all the people who have supported me to complete the project work directly or indirectly.

Urbana Jaman Orthee

University of Liberal Arts Bangladesh

Date: April 21, 2026

To my beloved parents
*for their endless support
and belief in me.* To my
supervisor, for guidance and
encouragement. And to
myself, for the perseverance
to complete this journey.

– *Urbana Jaman Orthee*

ABSTRACT

Machine unlearning aims to remove the influence of designated training data from a trained model without incurring the full computational cost of retraining from scratch. This objective is challenging because approximate unlearning methods must balance effective forgetting against preservation of retained utility, and the difficulty becomes greater in sparse neural networks where reduced parameter capacity can amplify instability. This Study addresses these challenges by proposing DistilRepair, an anchored distillation framework for utility-preserving approximate unlearning in both dense and sparse models. DistilRepair combines targeted forget-side optimization with retain-side repair guided by a frozen teacher, while parameter anchoring constrains harmful deviation from the pre-unlearning solution. The method further incorporates staged classifier-focused warmup, Batch Normalization freezing during forget updates, and numerical-stability safeguards to improve robustness, particularly under sparsity. Experiments on the HAM10000 skin lesion dataset with EfficientNet-B0 and the CIFAR-10 dataset with ResNet-18 show that DistilRepair consistently narrows the gap between approximate unlearning and gold-standard retraining. On HAM10000, DistilRepair achieves near-retraining performance for dense class-wise forgetting and strong sparse unlearning performance while reducing runtime by approximately 11 times. On CIFAR-10, the method remains close to retraining under class-wise forgetting and shows strong utility preservation under random forgetting, while preserving competitive retained and test accuracy. Across both datasets and model families, DistilRepair reduces unlearning time by a large margin relative to retraining, demonstrating that anchored distillation is an effective and practical strategy for approximate unlearning in dense and sparse neural networks.

Keywords: Machine Unlearning, Model Sparsification, Utility Preservation, Knowledge Distillation, Optimization

LIST OF PUBLICATIONS

The following papers, related to this thesis, have been submitted & presented to peer-reviewed conferences:

1. U. J. Orthee, A. S. M. Naeem, T. Khatun, M. Ohona, and N. Mansoor, “Unlearning for clinical deployment: Scalable data erasure in medical image classification models,” in *Proceedings of the 2025 IEEE International Conference on Biomedical Engineering, Computer and Information Technology for Health (BECITHCON)*, IEEE, 2025. (Submitted & Accepted)
2. U. J. Orthee, A. S. M. Naeem, T. Khatun, M. Ohona, and N. Mansoor, “Efficient data erasure in deep learning via sparsity and stability mechanisms,” in *Proceedings of the 2025 28th International Conference on Computer and Information Technology (ICCIT)*, IEEE, 2025. (Submitted & Accepted)

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Problem Statement	2
1.3	Objective	2
1.4	Approach and Contribution	3
1.4.1	Baseline Refinement and Experimental Setup	3
1.4.2	Controlled Approximate Unlearning via DistilRepair	3
1.4.3	Dense and Sparse Evaluation	3
1.4.4	Cross-Dataset Validation	3
1.4.5	Empirical Findings	4
1.5	Thesis Organization	4
2	Literature Review	5
2.1	Overview of Machine Unlearning	5
2.2	Existing Approaches	5
2.3	Sparsity in Machine Learning & Its Role in Unlearning	7
2.4	Gap in Literature	7
2.5	Summary of Insights	8
2.6	Comparative Analysis	8
2.7	Summary	14
3	Methodology	15
3.1	Problem Formulation	15
3.2	Unlearning Setting	15
3.3	Dense and Sparse Model Setting	16
3.4	DistilRepair	16
3.5	Targeted Forget Stage	18
3.6	Retain-Side Repair Stage	18
3.7	Anchoring and Stability Mechanisms	19
3.8	Algorithm Summary	19

4	Experimental Setup	22
4.1	Datasets	22
4.2	Models and Sparsity Settings	22
4.3	Forgetting Scenarios	23
4.4	Baselines	23
4.5	Training and Unlearning Configuration	24
4.6	Evaluation Metrics	25
5	Results and Analysis	26
5.1	Overview	26
5.2	HAM10000 Results	26
5.2.1	Class-Wise Forgetting	26
5.2.2	Random Forgetting	27
5.2.3	Interpretation of the HAM10000 Setting	28
5.3	CIFAR-10 Results	28
5.3.1	Class-Wise Forgetting	28
5.3.2	Random Forgetting	29
5.4	Runtime and Efficiency	30
5.5	Privacy-Oriented Analysis	32
5.6	Summary of Findings	32
6	Conclusion and Future Work	33
6.1	Conclusion	33
6.2	Future Work	34
	References	35

List of Figures

3.1	Architecture of DistilRepair	17
5.1	HAM10000 comparison of Retrain and DistilRepair across UA, MIA, RA, and TA for dense and sparse settings under class-wise and random forgetting.	27
5.2	CIFAR-10 comparison of Retrain and DistilRepair across UA, MIA, RA, and TA for dense and sparse settings under class-wise and random forgetting.	29
5.3	HAM10000 runtime comparison between Retrain and DistilRepair for dense and sparse models under class-wise and random forgetting.	31
5.4	CIFAR-10 runtime comparison between Retrain and DistilRepair for dense and sparse models under class-wise and random forgetting.	31

List of Tables

2.1	Overview of models, sparsification, and optimization techniques used in Machine Unlearning research.	8
2.2	Comparison of existing Machine Unlearning methods and their key characteristics.	11
3.1	Notation and description for DistilRepair.	20
4.1	DistilRepair hyperparameters for HAM (dense and sparse settings). . . .	24
4.2	DistilRepair hyperparameters for CIFAR-10 (same config for class-wise and random forgetting).	24
5.1	Our experimental results on HAM10000.	28
5.2	Our experimental results on CIFAR-10.	30

Chapter 1

Introduction

1.1 Motivation

Machine learning models are increasingly trained on large, continuously evolving datasets that may later require partial removal. Such removal requests can arise from privacy regulation, user withdrawal, data quality concerns, legal compliance, or the discovery that specific training samples are harmful or no longer valid. In principle, the most reliable solution is to retrain the model from scratch on the retained data only. However, as modern models and datasets grow in size, full retraining becomes costly in both computation and time [Ginart et al. \(2019\)](#). This challenge motivates the study of *machine unlearning*: the task of removing the influence of designated data from an already trained model without repeating the full original training pipeline.

Recent state-of-the-art research shows that machine unlearning spans multiple methodological directions. Exact or structured methods, such as SISA and certified data removal, aim to provide stronger deletion guarantees, but often require architectural or training-time constraints that are difficult to satisfy in practical deep-learning settings [Bourtoule et al. \(2021\)](#); [C. Guo et al. \(2020\)](#). More recent deep-learning methods have therefore shifted toward approximate unlearning, where the goal is not formal certification, but practical forgetting with strong retained utility and much lower cost than retraining [Foster et al. \(2024a\)](#); [Kurmanji et al. \(2023\)](#); [Foster et al. \(2024b\)](#). At the same time, sparse neural networks have become increasingly important because they offer reduced computational cost and model size, and recent work suggests that sparsity may also simplify unlearning under certain conditions [Jia et al. \(2023\)](#).

Despite this progress, an important research gap remains. Many approximate unlearning methods either forget too weakly or damage retained performance too severely, especially in sparse networks where optimization is less forgiving. Moreover, much of the literature focuses either on dense models or on forgetting performance alone, without fully addressing the joint problem of *effective forgetting*, *retained utility preservation*, *sparse-model robustness*, and *runtime efficiency*. This gap motivates the present study,

which shows whether a more controlled forget-and-repair strategy can narrow the gap between approximate unlearning and gold-standard retraining in both dense and sparse neural networks.

1.2 Problem Statement

This study explores approximate machine unlearning for image classification models. Given a trained model and a subset of training data that must be removed, the goal is to update the model so that the designated data no longer has a meaningful influence on its predictions, while preserving performance on the retained data and maintaining strong validation and test accuracy. This problem is important because full retraining, although the strongest practical baseline, is often too expensive when deletion requests are frequent or models are large [Ginart et al. \(2019\)](#).

The problem becomes more difficult in sparse neural networks, where fewer active parameters reduce optimization flexibility and can make forgetting updates more unstable. As a result, a method that behaves well in dense models may not remain effective under sparsity. To address this, the present study proposes *DistilRepair*, a controlled approximate unlearning framework that combines targeted forgetting, retain-side repair, anchoring, and stability-oriented optimization. The method is evaluated against gold-standard retraining in both dense and sparse settings to examine how closely practical approximate unlearning can approach retraining-level performance.

1.3 Objective

The objectives of this study are as follows:

- To study the limitations of existing gradient-ascent-based approximate unlearning methods, particularly the trade-off between forgetting effectiveness and retained utility.
- To design an improved approximate unlearning method, *DistilRepair*, that performs targeted forgetting while preserving useful retained behavior.
- To investigate the behavior of approximate unlearning in both dense and sparse neural networks.
- To compare *DistilRepair* against gold-standard retraining under class-wise and random forgetting settings.
- To evaluate performance using multiple criteria, including unlearning quality, retained accuracy, test accuracy, privacy-oriented measures, and runtime efficiency.

- To determine whether a controlled forget-and-repair framework can substantially reduce the practical gap between approximate unlearning and retraining.

1.4 Approach and Contribution

1.4.1 Baseline Refinement and Experimental Setup

The first contribution of this study is the establishment of a reliable experimental foundation for unlearning analysis. Before method development, the baseline training and retraining pipelines were examined and corrected so that subsequent comparisons would be meaningful. This included improving the training setup, correcting retraining behavior under retain-only settings, and validating dense and sparse checkpoints for downstream unlearning experiments. This contribution is important because approximate unlearning claims are only credible when compared against a sound retraining reference.

1.4.2 Controlled Approximate Unlearning via DistilRepair

The central methodological contribution is the proposal of *DistilRepair*, a controlled approximate unlearning method. DistilRepair retains the alternating forget-and-repair structure, but improves it through a more targeted forget objective, retain-side teacher guidance, parameter anchoring, staged warmup, and stability safeguards. The resulting method is designed to forget more selectively and recover retained performance more effectively than standard gradient-ascent-based forgetting. In this sense, the study contributes a practical method for narrowing the gap between approximate unlearning and retraining in already-trained deep neural networks.

1.4.3 Dense and Sparse Evaluation

A further contribution of the study is the explicit evaluation of unlearning in both dense and sparse models. Although much of the machine-unlearning literature focuses on dense networks, this study investigates whether approximate unlearning remains effective when parameter redundancy is substantially reduced. The study therefore examines not only forgetting performance, but also optimization stability under sparsity, thereby extending the scope of practical unlearning analysis beyond dense-only settings.

1.4.4 Cross-Dataset Validation

The study also contributes empirical validation across multiple datasets and architectures. DistilRepair is studied on HAM10000 with EfficientNet-B0 and on CIFAR-10 with ResNet-18, under both class-wise and random forgetting scenarios. This cross-dataset evaluation strengthens the argument that the method is not limited to a single benchmark or backbone, but reflects a more general approximate unlearning strategy.

1.4.5 Empirical Findings

The final contribution is empirical: the study shows that DistilRepair can approach retraining-quality behavior at a fraction of the runtime cost. In all reported experiments, the method consistently reduces unlearning time relative to retraining while maintaining strong retained utility and competitive forgetting performance. In particular, the study shows that controlled forgetting combined with repair and stabilization is more effective than relying on broad forget-side ascent alone. This finding supports the claim of the larger study that approximate practical unlearning can be made both effective and computationally viable.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 reviews the recent literature on machine unlearning, with particular attention to approximate unlearning methods, sparse-model unlearning, and the trade-off between forgetting effectiveness and retained utility. Chapter 3 presents the proposed methodology and introduces DistilRepair as a controlled forget-and-repair framework, including the main optimization stages and stabilization mechanisms used in dense and sparse settings. Chapter 4 describes the experimental setup, including the datasets, backbone architectures, sparsity settings, forgetting scenarios, baselines, hyperparameter configuration, and evaluation metrics used throughout the study. Chapter 5 presents the empirical results on HAM10000 and CIFAR-10, compares DistilRepair with retraining in dense and sparse models, and analyzes the observed trade-offs in forgetting efficacy, retained utility, privacy-oriented behavior, and runtime. Finally, Chapter 6 concludes the thesis by summarizing the main findings and outlining directions for future work.

Chapter 2

Literature Review

2.1 Overview of Machine Unlearning

Machine Unlearning (MU) has become an important research direction, driven by regulatory requirements such as the GDPR’s right to be forgotten, as well as ethical considerations and user-initiated requests to remove sensitive or personal data from trained models [Golatkar et al. \(2020\)](#), [Bourtoule et al. \(2021\)](#). In contrast to conventional training, where data influence is retained indefinitely, MU facilitates the selective removal of specific information without necessitating complete retraining. Its relevance extends beyond privacy, impacting security by helping to counter backdoor attacks [Jia et al. \(2023\)](#), and supporting adaptability in systems that must respond to evolving user needs and contexts [Dang et al. \(2025\)](#). MU has been explored across a wide range of applications, including large language models (LLMs) [Pochinkov & Schoots \(2024\)](#), [S. Zhang et al. \(2025\)](#), multimodal systems [Liu et al. \(2025\)](#), [Dontsov et al. \(2024\)](#), recommendation systems [Dang et al. \(2025\)](#), and biologically inspired spiking neural networks [J. Wang et al. \(2022\)](#). Despite this diversity, the fundamental challenge is consistent: to remove targeted information efficiently while maintaining model utility and ensuring resilience against privacy threats such as membership inference attacks.

2.2 Existing Approaches

Prior work presents a broad spectrum of MU strategies. Certified unlearning methods, including SISA training, decompose datasets into shards and slices to limit the scope of data influence, thereby enabling provable removal through partial retraining [Bourtoule et al. \(2021\)](#). In a similar vein, patch-based methods such as PRUNE introduce lightweight repair modules that suppress predictions on forgotten samples while maintaining verifiable guarantees [X. Li et al. \(2025\)](#). Despite their reliability, these approaches often incur significant storage overhead or reduced performance when applied to complex models.

Retraining-based methods continue to offer the highest fidelity but are computationally infeasible at scale. As a result, fine-tuning has emerged as a practical alternative. Empirical

studies indicate that fine-tuning on retain sets can achieve strong forgetting performance while preserving generalization [Kurmanji et al. \(2024\)](#), [Cadet et al. \(2025\)](#). Variants such as retain fine-tuning and continued training adapt to diverse settings, though they remain susceptible to privacy leakage under strong membership inference attacks [Cadet et al. \(2025\)](#).

Gradient Ascent (GA) is another commonly explored approach, which directly maximizes the loss on forget data to remove learned information. However, GA is often unstable and can lead to catastrophic collapse, producing incoherent or degenerate outputs [R. Zhang et al. \(2024\)](#). To mitigate these issues, methods such as Negative Gradient Plus (NG+) and Negative Preference Optimization (NPO) refine the optimization process by controlling divergence and improving stability, enabling more reliable unlearning at scale ([Cadet et al. \(2025\)](#), [R. Zhang et al. \(2024\)](#)).

Beyond GA and fine-tuning, several alternative optimization strategies have been proposed. Confusion loss enforces high-entropy predictions on forget data while maintaining performance via auxiliary retain objectives [S. Zhang et al. \(2025\)](#). Similarly, adaptations of cross-entropy, including logit anchoring and label smoothing, have been employed to regularize outputs and reduce catastrophic forgetting [Fan et al. \(2023\)](#). Sharpness-Aware Minimization (SAM) has also been explored as a stabilizing mechanism, encouraging solutions in flatter regions of the loss landscape and reducing sensitivity to adversarial forget samples [Jia et al. \(2023\)](#).

It has been further shown that forgetting can be guided by an intentionally weak teacher, illustrating that teacher-based signals can shape forgetting behavior instead of only preserving knowledge [Chundawat et al. \(2023\)](#). In addition, Towards Unbounded Machine Unlearning broadened the problem beyond one-shot deletion and argued that practical systems must support repeated and evolving removal requests [Kurmanji et al. \(2023\)](#). Taken together, these studies show that recent approximate unlearning research increasingly emphasizes targeted forgetting, controlled recovery, and deployment-oriented efficiency.

Batch Normalization (BN) calibration provides a lightweight post-processing approach, particularly effective in generative and diffusion models. By adjusting normalization statistics after unlearning, it helps realign the distributions of retain and forget data [Fan et al. \(2023\)](#). These developments reflect an increasing emphasis on balancing stability and utility through more sophisticated optimization techniques.

Finally, knowledge editing methods extend MU to large language models. Mechanistic unlearning focuses on identifying and modifying internal pathways responsible for factual recall [P. Guo et al. \(2024\)](#), while DeepCUT leverages contrastive objectives in latent space to remove discriminative features [E. He et al. \(2025\)](#). Orthogonal task-vector projection methods, such as Ethos, isolate and suppress undesirable knowledge components—such as bias, toxicity, or memorization—without significantly affecting overall performance

Gao et al. (2024). Taken together, these approaches illustrate the diversity of unlearning paradigms, though inherent trade-offs between effectiveness, efficiency, and robustness persist.

2.3 Sparsity in Machine Learning & Its Role in Unlearning

Sparsity has emerged as a key factor in both computational efficiency and effective unlearning. Jia et al. (2023) show that introducing sparsity helps bridge the gap between approximate and exact unlearning by reducing parameter redundancy and concentrating updates on the most influential weights. Their proposed paradigms which is prune first, then unlearn and sparsity-aware unlearning, demonstrate that structured sparse models can achieve stronger unlearning performance while also improving robustness against membership inference attacks. Techniques such as One-shot Magnitude Pruning (OMP) and SynFlow serve as practical tools for inducing sparsity without significant loss in performance, offering efficient initialization for subsequent unlearning steps Jia et al. (2023). In parallel, some methods incorporate sparsity directly into model design. Shah et al. (2023) propose Discrete Key-Value Bottlenecks, which enforce localized sparse activations, allowing entire classes to be forgotten by modifying only a limited subset of key-value pairs. Likewise, adaptive un-pruning strategies highlight how dynamically adjusting pruned structures can further support selective forgetting Xiao et al. (2025). Overall, sparsity contributes not only to improved efficiency but also to greater stability and privacy in unlearning. By enabling models to localize and remove specific information more precisely, it reduces unintended degradation of retained knowledge.

2.4 Gap in Literature

Despite considerable advancements, existing MU approaches continue to exhibit notable limitations. Certified methods provide strong guarantees of data removal but often fail to scale effectively to large models Bourtole et al. (2021), Z. Li et al. (2025). Sparsity has demonstrated clear advantages in supporting unlearning Jia et al. (2023), Shah et al. (2023), yet its application has been largely restricted to vision-based settings, with limited investigation in large language models or multimodal systems Liu et al. (2025), Dontsov et al. (2024). Moreover, while sparsification techniques like OMP and SynFlow are effective for initializing sparse networks, the interaction between sparsity and optimization strategies—such as GA, anchor loss, knowledge distillation, or label smoothing—remains underexplored in the context of unlearning. There is still limited work on methods that are explicitly designed to operate well in both dense and sparse models while preserving utility and maintaining stability. This gap motivates the need for a method that treats

forgetting and retained recovery as a coupled optimization problem rather than focusing on forgetting alone.

2.5 Summary of Insights

The literature suggests five main insights that directly motivate this study. First, retraining remains the strongest practical reference but is too expensive to serve as the only solution [Golatkhar et al. \(2020\)](#). Second, exact and certified methods provide important guarantees, yet they are often less convenient for already-trained deep models [Bourtoule et al. \(2021\)](#). Third, recent approximate unlearning methods show that targeted forgetting and controlled recovery are more promising than broad, destructive updates [Chundawat et al. \(2023\)](#). Fourth, sparse models create both opportunity and difficulty: they may simplify unlearning, but they also make optimization more fragile [Jia et al. \(2023\)](#). Fifth, evaluation must jointly consider forgetting, retained utility, runtime, and privacy-oriented measures [Cadet et al. \(2025\)](#). Viewed in this context, DistilRepair is positioned as a controlled approximate unlearning method that combines targeted forgetting, repair, anchoring, and stability-oriented optimization in order to approach retraining-quality behavior in both dense and sparse neural networks.

2.6 Comparative Analysis

Table 2.1: Overview of models, sparsification, and optimization techniques used in Machine Unlearning research.

Citation	Deep Learning Models Used	Sparsification Approaches	Optimization Techniques	Other Relevant Insights to Problem Domain
Jia et al. (2023)	ResNet-18, VGG-16, ImageNet-scale CNNs	Weight pruning (OMP, SynFlow)	Fine-tuning with ℓ_1 regularization; SAM	Sparsity reduces the unlearning-retraining gap; improves backdoor defense and transfer efficiency.
Z. Li et al. (2025)	VGG-16, fully connected networks	None	Patch networks + clustering	Certifiable class-level forgetting with GDPR compliance.
Chen et al. (2021)	LeNet-5, ResNet-18, VGG-16, Wide ResNet	None	KL-guided weight perturbation	Up to $15\times$ faster than retraining; robust to MIA and backdoor attacks.
Xiao et al. (2025)	ResNet-18, AlexNet, ViT	Un-pruning sparse models	GA, SCRUB, Fisher-compatible	Ensures sparse architectures remain unlearnable.

Continued on next page

Citation	Deep Learning Models Used	Sparsification Approaches	Optimization Techniques	Other Relevant Insights to Problem Domain
Pochinkov & Schoots (2024)	Large Language Models	Neuron-level pruning	Targeted pruning	Removes skills (coding, toxicity) with minimal collateral damage.
Liu et al. (2025)	LLaVA, Idefics2	Modality-aware pruning	Importance scoring	Balances multimodal vs. unimodal forgetting.
Kurmanji et al. (2024)	Learned databases	None	NegGrad, SCRUB, SISA	Benchmark for MU in database systems.
Bourtole et al. (2021)	CNNs (MNIST–ImageNet)	None	SISA retraining	First certified unlearning framework.
S. Zhang et al. (2025)	LLaMA, Vicuna	Selective pruning	Confusion loss, disentanglement	Selective forgetting in LLMs with minimal utility loss.
Golatkari et al. (2020)	CNNs (MNIST, CIFAR)	None	Gradient ascent variants; Fisher approximations	Introduces "scrubbing" with theoretical guarantees; close to retraining accuracy.
J. Wang et al. (2022)	Spiking Neural Models (SNMs)	Synaptic pruning	Adaptive thresholding; selective retraining	First MU in SNMs; hybrid biologically inspired strategy with privacy validation.
Shah et al. (2023)	CLIP ViT-B/32, ResNet-50	Discrete Key-Value Bottlenecks (DKVBs)	Masking activations (no retraining)	Enables class-level forgetting via sparse discrete representations.
Fan et al. (2023)	ResNet-18, VGG-16, Swin-T, DDPM, Stable Diffusion	Weight saliency pruning (SalUn)	Random relabeling, confusion loss, BN calibration	Unified MU for classification + diffusion; prevents NSFW generations.
Cadet et al. (2025)	ResNet-18, TinyViT	None	Fine-tune, GA, NG+, MSG, CT	Benchmark of 18 MU methods; MSG & CT effective; GA unstable, NG+ better baseline.
J. Wang et al. (2022)	ResNet, VGG (Federated CNNs)	Channel pruning (TF-IDF based)	Federated fine-tuning post-pruning	8 – 11× speedup; preserves accuracy on retain classes; GDPR-aligned.
Zaman et al. (2025)	AlexNet (Federated IoD)	Selective pruning	Fine-tuning in FL + pruning neurons	Reduces comms overhead; ~40% faster than retraining in drone networks.

Continued on next page

Citation	Deep Learning Models Used	Sparsification Approaches	Optimization Techniques	Other Relevant Insights to Problem Domain
Jin et al. (2025)	CIFAR-10 CNNs, Speech Commands (DNNs)	Neuron pruning (attribution-based)	Fine-tuning (layer-wise, full-model)	Improves noisy dataset robustness; up to 47% faster than retraining.
Dang et al. (2025)	MF, NGCF, LightGCN (Recommenders)	None	Contrastive loss; attention residual filters	Erases outdated user preferences efficiently; introduces Ranking Decrease Rate metric.
Pawelczyk et al. (2023)	BLOOM, LLaMA-2 (LLMs)	None	In-Context Unlearning (ICL prompts)	Parameter-free; works in black-box LLMs without retraining.
E. He et al. (2025)	RoBERTa, BioBERT (NER tasks)	None	Contrastive unlearning in latent space	Strong unlearning for NER; balances forgetting and retention efficiently.
R. Zhang et al. (2024)	LLaMA-2-7B	None	Negative Preference Optimization (NPO)	Stabilizes GA; enables large-scale unlearning (50–90% data).
P. Guo et al. (2024)	Gemma, LLaMA (LLMs)	None	Mechanistic unlearning via FLU localization	Edits factual recall pathways; robust to adversarial relearning.
Z. Li et al. (2025)	ResNet-18 (MNIST, CIFAR-10/100)	None	FUNU filtering + KL divergence	Identifies redundant unlearning requests; reduces MU time by ~24%.
Dontsov et al. (2024)	LLaVA, QwenVL2 (MLLMs)	None	Multiple MU base-lines tested (SCRUB, DPO, GA, NPO, etc.)	CLEAR benchmark shows unimodal methods fail in multimodal unlearning.
Gao et al. (2024)	OPT, GPT-2, GPT-Neo, LLaMA-2	Orthogonal projection (SVD-based task vectors)	LoRA + task arithmetic	Removes toxicity, bias, memorization while preserving utility.

Table 2.2: Comparison of existing Machine Unlearning methods and their key characteristics.

Citation	Purpose	Methods / Techniques	Strengths	Limitations	Key Takeaways
Golatkar et al. (2020)	Enable selective forgetting in DNNs without full retraining	Scrubbing with Fisher Information Matrix, SGD stability	Theoretically grounded, effective against white-box attacks, and competitive with retraining	Computationally heavy, assumes access to Hessian/FIM, limited to classification	Scrubbing offers practical, theory-backed forgetting but faces scaling challenges.
Bourtoule et al. (2021)	Provide provable unlearning guarantees	SISA training (sharding, slicing, aggregation)	Provable compliance, efficient for small-scale tasks	Accuracy loss on complex tasks, storage overhead	First formal framework balancing guarantees and efficiency.
Chen et al. (2021)	Lightweight unlearning for neural networks	Weight perturbation toward the reference model	15× faster than retraining, robust to MIA/backdoors, minimal accuracy loss	Requires reference model training, limited to vision datasets	Perturbation-based unlearning is efficient and robust but approximate.
Jia et al. (2023)	Explore model sparsity for MU	Prune-first unlearn, sparsity-aware unlearning	Strong theoretical support improves efficacy and efficiency	Ultra-high sparsity harms accuracy, pruning method-dependent	Sparsity reduces gap between approximate and exact unlearning.
X. Li et al. (2025)	Certifiable unlearning in neural networks	PRUNE patch-based repair	Verifiable guarantees, scalable to classes, IoT-suitable	Weaker generalization on images, limited scope	Patch-based unlearning ensures certifiable forgetting with minimal side effects.
Xiao et al. (2025)	Address MU in sparse models	Un-pruning algorithm (reinitialize + reprune)	Efficient, integrates with multiple MU methods, and new metrics introduced	Less effective on unstructured pruning, moderate deletion scale	Ensures sparse models remain compliant and adaptable.
Pochinkov & Schoots (2024)	Remove targeted capabilities from LLMs	Structured neuron pruning (importance scoring, iterative pruning)	Compute/data-efficient, works on LLMs/ViTs, open-source	Needs a clear forget dataset, unstable under aggressive pruning	Capability-level pruning is practical for safe modular deployment.

Continued on next page

Citation	Purpose	Methods / Techniques	Strengths	Limitations	Key Takeaways
Liu et al. (2025)	Balanced forgetting in multimodal LLMs	Modality-aware neuron pruning (MANU)	First MM unlearning, balances across modalities, preserves utility	Sensitive to pruning ratios, robustness under attack not tested	Highlights modality-aware strategies as critical for privacy in MLLMs.
Kurmanji et al. (2024)	Study MU in learned databases	Benchmarks Fine-tune, NegGrad, SCRUB, SISA	Multi-perspective metrics, highlights trade-offs	Only tabular DBs, no new algorithms proposed	Foundational study showing DB unlearning differs from ML tasks.
S. Zhang et al. (2025)	Remove harmful knowledge in LLMs	LLM-Eraser: pruning + selective fine-tuning	High selectivity, robust across languages, better than GA baselines	Limited to mid-scale LLMs, complex loss tuning	Pruning + disentanglement enables safer LLM deployment.
C. Wang et al. (2024)	MU in Spiking Neuron Models	Synaptic retraining, pruning, adaptive thresholding	Novelty, privacy validation via MIA, tested on time-series & vision	Dataset-dependent performance, scalability untested	First SNM-focused unlearning, bridging neuroscience and ML.
Shah et al. (2023)	Low-compute MU	Discrete Key-Value Bottlenecks (mask activations)	No retraining, scalable, efficient	Limited to class-level unlearning, architecture-dependent	Demonstrates sparsity as inductive bias for scalable MU.
Fan et al. (2023)	Stable MU across tasks	SalUn: saliency-guided weight updates	Works for classification & diffusion, robust, efficient	Threshold sensitivity, only CV tasks, no certified guarantees	First unified method across classification & generation.
Cadet et al. (2025)	Benchmark MU methods	Compare 18 approaches (MSG, CT, NG+)	Largest benchmark, fair hyperparam tuning, reproducible	Only CV tasks, no formal guarantees	Establishes MSG & CT as reliable MU baselines.
J. Wang et al. (2022)	Federated MU in CNNs	TF-IDF-guided channel pruning	11× faster than retraining, GDPR-compliant	Only class-level, tested on CIFAR-10/100	Effective class forgetting in FL without raw data.

Continued on next page

Citation	Purpose	Methods / Techniques	Strengths	Limitations	Key Takeaways
Zaman et al. (2025)	Federated MU in IoD	SoUL: selective pruning	40% faster than FedAU, scalable, preserves accuracy	Limited datasets/models, static thresholds	Optimized for IoD, balancing bandwidth and accuracy.
Jin et al. (2025)	Noise-robust MU	RLAP: attribution-guided partition + pruning	Improves accuracy on noisy data, 47% faster	Only small datasets, synthetic noise	Combines unlearning + robustness in noisy settings.
Dang et al. (2025)	Adaptive MU in recommenders	GFEraser: guided filtering + contrastive learning	37× faster, balances outdated vs invariant prefs	Simulated preference changes, trade-off remains	First adaptive MU in recsys, efficient + accurate.
Pawelczyk et al. (2023)	Black-box LLM unlearning	ICUL: in-context label flipping	Parameter-free, works for query-only LLMs, efficient	Drops with large deletions, context length issues	Enables unlearning for commercial black-box LLMs.
E. He et al. (2025)	Latent-space MU	DeepCUT: contrastive learning	Robust forgetting, efficient, outperforms retraining	Focused on NER only, medium-sized models	Latent contrastive MU effective across domains.
R. Zhang et al. (2024)	Stable MU objective	NPO: negative-only preference optimization	Avoids collapse, scalable to 90% forgetting	Limited real-world validation, tuning needed	Theoretically grounded, first scalable GA alternative.
P. Guo et al. (2024)	Robust factual editing MU	Mechanistic unlearning (fact lookup MLPs)	Robust to relearning, efficient, interpretable	Limited to factual recall, manual localization	Introduces interpretability-driven MU.
Z. Li et al. (2025)	Filter redundant requests	FUNU: similarity-based filtering	24% faster, parameter-free, guarantees privacy	Extra computation, limited to images	Formalizes unnecessary unlearning to improve efficiency.
Dontsov et al. (2024)	Benchmark multimodal MU	CLEAR dataset & protocols	First MM benchmark, comprehensive, public	Synthetic data, only VL models	Establishes MMU as distinct field with new needs.
Gao et al. (2024)	Remove toxicity/bias/memorization	Ethos: orthogonal parameter space projection	Efficient, preserves general utility, multi-domain	Fixed thresholds, limited memorization gains	Effective separation of harmful vs general knowledge.

2.7 Summary

Machine unlearning enables selective removal of data influence from trained models, balancing privacy, security, and adaptability with computational efficiency. Existing methods reveal trade-offs between accuracy, scalability, and stability, while recent advances highlight the role of optimization and sparsity in improving performance. Overall, integrating sparsity with robust optimization emerges as a promising direction for achieving efficient and reliable unlearning in modern deep learning systems.

Chapter 3

Methodology

3.1 Problem Formulation

This study considers the problem of approximate machine unlearning for supervised image classification. Let a trained model f_θ be learned from a training set D . After training, a subset $D_f \subset D$ is designated for removal, while the remaining subset $D_r = D \setminus D_f$ is retained. The objective is to produce an updated model f_{θ^*} such that the influence of D_f on the final predictor is substantially reduced, while the predictive behavior on D_r and on unseen evaluation data is preserved as much as possible. In principle, the most direct solution is retraining on D_r only, but retraining is computationally expensive and therefore serves as a gold-standard baseline rather than the only practical solution [Ginart et al. \(2019\)](#).

The present work studies this problem in both dense and sparse neural networks. In the dense setting, all trainable parameters remain active during learning and unlearning. In the sparse setting, the model is constrained by a pruning mask so that only a subset of weights remains trainable and effective during inference. This distinction is important because sparse models have less redundancy and are therefore more sensitive to unstable optimization updates. Consequently, the same unlearning strategy may behave differently in dense and sparse models even when the architecture and dataset are otherwise unchanged.

3.2 Unlearning Setting

Two forgetting scenarios are considered in this study. In *class-wise forgetting*, all training samples belonging to a selected class are designated for removal. In *random forgetting*, a fixed number of training instances are removed irrespective of class membership. These two settings capture different notions of deletion. Class-wise forgetting is a stronger and more structured setting because the target concept is semantically coherent, whereas random forgetting more closely reflects individual instance deletion requests.

The goal of approximate unlearning in this study is not formal certification, but practical removal that approaches retraining behavior at a much lower cost. Accordingly, the desired outcome is a method that reduces model dependence on D_f , preserves retained accuracy on D_r , generalizes well on validation and test data, and remains computationally efficient. This naturally creates a multi-objective optimization problem in which aggressive forgetting may damage retained utility, while overly cautious updates may fail to remove the designated information effectively especially when the model has been trained on imbalanced dataset.

3.3 Dense and Sparse Model Setting

The methodology is evaluated on both dense and sparse checkpoints. The dense model provides the standard unpruned reference, whereas the sparse model is obtained through pruning and mask preservation. In the sparse setting, the pruning mask constrains which weights remain active, and the unlearning method must operate under this structural limitation. This setup is motivated by recent work showing that sparsity can alter the behavior of machine unlearning and may, in some cases, simplify the forgetting problem relative to retraining [Jia et al. \(2023\)](#). However, sparsity also makes optimization more brittle because fewer active parameters are available to absorb disruptive forget-side updates. For that reason, the proposed method is explicitly designed to support both dense and sparse models.

3.4 DistilRepair

The proposed method, *DistilRepair*, is a controlled forget-and-repair framework for approximate machine unlearning. The method starts from a pre-trained model and splits the marked training data into a forget loader and a retain loader. A frozen teacher model is reconstructed from the original checkpoint, and the same checkpoint is also stored as an anchor reference. DistilRepair then alternates between two stages over multiple epochs: a targeted forget stage that weakens the influence of D_f , and a retain-side repair stage that restores useful predictive behavior on D_r .

The main intuition is that forgetting and recovery should be treated as coupled operations. If forgetting is applied too broadly, the model may lose useful shared features. If repair is too weak, retained utility may collapse. DistilRepair therefore separates these roles: the forget stage selectively suppresses the unwanted information, while the repair stage uses retained supervision and teacher guidance to recover the useful function of the model.

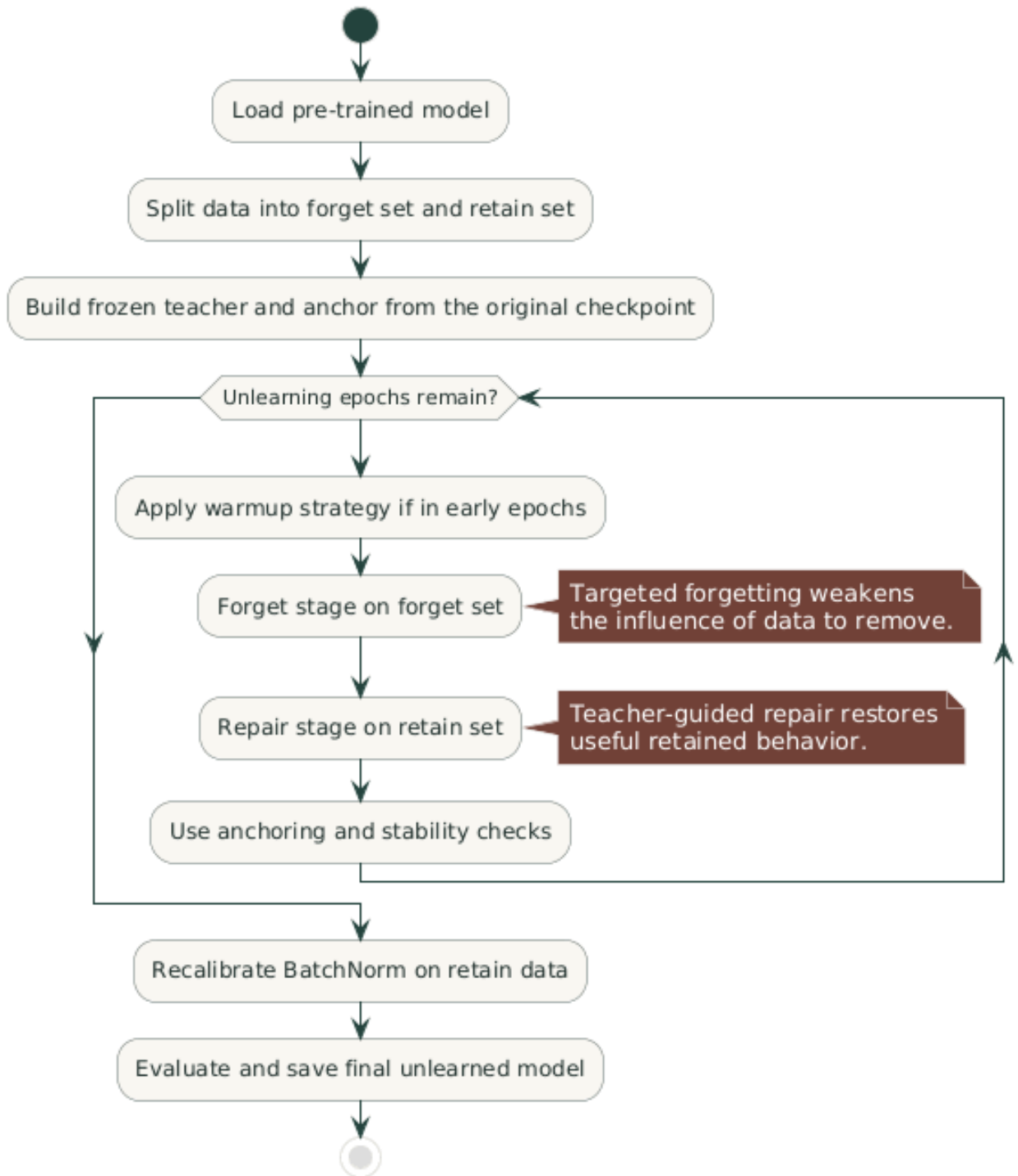


Figure 3.1: Architecture of DistilRepair

3.5 Targeted Forget Stage

In the forget stage, the model is updated using batches from the forget set. Rather than relying only on broad gradient ascent on the standard classification loss, DistilRepair uses a targeted margin-based objective that directly suppresses the forgotten target. Let z_y denote the logit corresponding to the true forgotten class and let $\max(z_{\neg y})$ be the largest competing non-target logit. The core forget objective is defined as

$$L_{\text{margin}} = \max(0, z_y - \max(z_{\neg y}) + m), \quad (3.1)$$

where m is the forgetting margin. This term pushes the model away from confidently predicting the forgotten target and is more selective than pure negative cross-entropy.

To further weaken the influence of the forget data, DistilRepair supplements the margin objective with a negative cross-entropy term and an entropy-promoting term. The forget-stage objective is written as

$$L_{\text{GA}} = L_{\text{margin}} - \alpha L_{\text{ce}} - \beta L_{\text{ent}} + \lambda_a(t) L_{\text{anchor}}, \quad (3.2)$$

where L_{ce} is the standard cross-entropy loss on the forget batch, L_{ent} encourages uncertainty on forgotten data, and L_{anchor} constrains the model toward its pre-unlearning parameter state. The anchor strength $\lambda_a(t)$ is scheduled over time so that early epochs allow more freedom to forget, while later epochs emphasize stabilization.

During early training epochs, DistilRepair can optionally update only the classifier head before modifying the full network. This warmup strategy is used to make the initial forgetting process less destructive. In addition, Batch Normalization statistics are frozen during forget updates so that small or imbalanced forget batches do not corrupt running statistics. Finite-value checks, finite-gradient checks, and gradient clipping are also applied in order to prevent unstable optimization, especially in sparse settings.

3.6 Retain-Side Repair Stage

After the forget stage, DistilRepair switches to a repair stage based on retain batches. The purpose of this stage is to recover the predictive behavior that may have been disturbed by forgetting. Let $p_\theta(x)$ be the student output and $p_T(x)$ the output of the frozen teacher. The retain-side objective combines supervised learning with teacher-guided distillation:

$$L_{\text{repair}} = L_{\text{sup}} + \lambda_{kd} L_{\text{KD}} + \lambda_a(t) L_{\text{anchor}}, \quad (3.3)$$

where L_{sup} is the supervised retain loss, L_{KD} is the distillation term between the student and the frozen teacher, and λ_{kd} controls the strength of teacher guidance. Distillation

is important because it helps preserve the original functional behavior of the model on retained data, not just its label correctness.

The repair stage uses Sharpness-Aware Minimization (SAM) as the optimizer [Foret et al. \(2021\)](#). This choice is motivated by the need for stable recovery after the disruptive forget step. Since forgetting can push the model into sharp or fragile parameter regions, SAM helps guide the repair process toward flatter and more reliable solutions, thereby improving retained utility. Optional Mixup and label smoothing are also used on retain data to further regularize the recovery process.

3.7 Anchoring and Stability Mechanisms

Anchoring is a central component of DistilRepair. The original model parameters are stored before unlearning begins, and subsequent updates are penalized for drifting too far from this reference. This mechanism is particularly important in sparse models, where aggressive forgetting may otherwise damage the small set of active weights too severely.

Beyond anchoring, DistilRepair includes several practical stabilization mechanisms. First, classifier-only warmup reduces early damage by limiting the scope of forgetting updates. Second, Batch Normalization freezing prevents narrow forget batches from distorting feature statistics. Third, finite-value and finite-gradient checks detect unstable steps before they propagate through training. Finally, gradient clipping constrains update magnitude and reduces the risk of numerical failure. Together, these mechanisms make the method more robust under sparsity and support the claim that approximate unlearning should be both effective and stable in realistic settings.

3.8 Algorithm Summary

The full DistilRepair workflow can be summarized as follows. Starting from a pre-trained checkpoint, the method constructs forget and retain subsets, initializes a frozen teacher, and stores an anchor copy of the original parameters. At each epoch, it first applies a targeted forget update on the forget set using the margin-based objective with auxiliary forgetting terms. It then performs a repair phase on the retain set using supervised learning, teacher-guided distillation, and SAM-based optimization. Throughout training, anchoring and stability controls regulate parameter drift and prevent destructive updates. After the iterative process is complete, Batch Normalization statistics are recalibrated on retained data, and the resulting model is evaluated against retraining in both dense and sparse settings.

Table 3.1: Notation and description for DistilRepair.

Symbol	Description
θ_0	Pre-trained model parameters (initialization)
θ	Student model parameters being updated
θ_{teacher}	Frozen teacher model (copy of θ_0)
θ_{anchor}	Anchor parameters for stability regularization
$\mathcal{D}_r$	Retain dataset (data to preserve performance on)
$\mathcal{D}_f$	Forget dataset (data to unlearn)
T	Number of unlearning epochs
S_f	Number of forget steps per epoch
S_r	Number of retain (repair) steps per epoch
η_f, η_r	Learning rates for forget and repair phases
λ_{kd}	Weight for knowledge distillation loss
$\lambda_a(t)$	Time-dependent anchor regularization weight
m	Margin used in targeted forget loss
z_f, z_r	Logits from student model on forget/retain batches
z_{teacher}	Logits from teacher model
L_{margin}	Targeted margin-based forgetting loss
L_{ce}	Cross-entropy loss
L_{ent}	Entropy regularization term
L_{anchor}	Anchor penalty $\ \theta - \theta_{\text{anchor}}\ _2^2$
L_{GA}	Forget objective (optimized via gradient ascent)
L_{repair}	Retain-side repair objective
α, β	Weights for auxiliary forgetting terms

Algorithm 1 DistilRepair

```
1: Initialize student model  $\theta \leftarrow \theta_0$ 
2: Construct frozen teacher model  $\theta_{\text{teacher}} \leftarrow \theta_0$ 
3: Set anchor parameters  $\theta_{\text{anchor}} \leftarrow \theta_0$ 
4: Optionally reset classifier row for forgotten class
5: for  $t = 1$  to  $T$  do
6:   if warmup stage then
7:     Update only classifier parameters
8:   else
9:     Update all parameters
10:  end if
11:  Freeze BatchNorm statistics during forget updates
12:  for  $i = 1$  to  $S_f$  do
13:    Sample  $(x_f, y_f) \sim \mathcal{D}_f$ 
14:     $z_f \leftarrow f(\theta; x_f)$ 
15:     $L_{\text{margin}} \leftarrow \max(0, z_f[y_f] - \max(z_f[\neg y_f]) + m)$ 
16:     $L_{\text{ce}} \leftarrow \text{CE}(z_f, y_f)$ 
17:     $L_{\text{ent}} \leftarrow \mathcal{H}(z_f)$ 
18:     $L_{\text{anchor}} \leftarrow \|\theta - \theta_{\text{anchor}}\|_2^2$ 
19:     $L_{\text{GA}} \leftarrow L_{\text{margin}} - \alpha L_{\text{ce}} - \beta L_{\text{ent}} + \lambda_a(t) L_{\text{anchor}}$ 
20:    Update  $\theta$  via gradient ascent on  $L_{\text{GA}}$ 
21:    Apply finite-value checks and gradient clipping
22:  end for
23:  Restore BatchNorm state after forget stage
24:  for  $j = 1$  to  $S_r$  do
25:    Sample  $(x_r, y_r) \sim \mathcal{D}_r$ 
26:    Optionally apply Mixup to  $(x_r, y_r)$ 
27:     $z_{\text{teacher}} \leftarrow f(\theta_{\text{teacher}}; x_r)$ 
28:     $z_r \leftarrow f(\theta; x_r)$ 
29:     $L_{\text{repair}} \leftarrow \text{CE}(z_r, y_r) + \lambda_{\text{kd}} \text{KL}(z_r, z_{\text{teacher}})$ 
30:     $+ \lambda_a(t) \|\theta - \theta_{\text{anchor}}\|_2^2$ 
31:    Update  $\theta$  using two-step SAM optimization on  $L_{\text{repair}}$ 
32:    Apply finite-value checks and gradient clipping
33:  end for
34: end for
35: Recalibrate BatchNorm statistics on  $\mathcal{D}_r$ 
36: return  $\theta$ 
```

Chapter 4

Experimental Setup

4.1 Datasets

This study evaluates DistilRepair on two image classification benchmarks: HAM10000 and CIFAR-10. HAM10000 is a dermatoscopic skin-lesion dataset containing seven diagnostic classes and is widely used for medical image classification research [Tschandl et al. \(2018\)](#). A key characteristic of HAM10000 is its strong class imbalance, which makes the learning problem more challenging than a balanced multi-class benchmark. In practice, this means that the baseline checkpoint and the sparse mask derived from it inherit a more difficult, less ideal starting condition. For this reason, the HAM10000 experiments are interpreted not as best-case unlearning, but as unlearning under an imbalanced and mildly overfit initialization regime.

CIFAR-10 is a standard natural-image benchmark consisting of ten object classes [Krizhevsky \(2009\)](#). Compared with HAM10000, CIFAR-10 provides a more balanced and conventional image classification setting. Using both datasets allows the study to test DistilRepair across two very different regimes: a highly imbalanced medical dataset and a more standard visual benchmark. This cross-dataset setup helps evaluate whether the proposed method is robust beyond a single domain or backbone.

4.2 Models and Sparsity Settings

Two backbone architectures are used in this study. For HAM10000, the experiments use EfficientNet-B0, a compact convolutional model that has shown strong performance in image classification through compound scaling of depth, width, and resolution [Tan & Le \(2019\)](#). For CIFAR-10, the experiments use ResNet-18, a residual architecture that remains a standard reference backbone for image recognition tasks [K. He et al. \(2016\)](#). These two choices allow the method to be evaluated on both a modern compact backbone and a widely used residual architecture.

Each dataset is studied under both dense and sparse settings. The dense model refers to the original unpruned checkpoint. The sparse model is obtained by pruning

and preserving a corresponding weight mask, after which unlearning is performed under the same structural constraint. For HAM10000, the primary sparse setting uses an 80% pruning rate, leaving approximately 20% of the weights active. For CIFAR-10, the sparse setting uses a 95% pruning configuration. The sparse experiments are motivated both by the practical value of compressed models and by prior work suggesting that sparsity can alter the behavior of machine unlearning relative to retraining [Jia et al. \(2023\)](#). Since sparse models are more optimization-sensitive, these experiments are also important for testing the stability of DistilRepair.

4.3 Forgetting Scenarios

Two forgetting scenarios are considered throughout the experiments. In *class-wise forgetting*, all training instances belonging to a designated class are removed from the model’s memory. This is the more structured and demanding setting because the removed data corresponds to a semantically coherent concept. In *random forgetting*, a fixed number of training samples are chosen at random and designated for removal. This setting is useful for testing whether the method can handle instance-level deletion requests without assuming semantic consistency.

Using both scenarios makes the evaluation more comprehensive. Class-wise forgetting is particularly informative for understanding whether the model can remove a concept while preserving other classes, whereas random forgetting tests whether the method remains effective in a less structured deletion setting. In this study, both HAM10000 and CIFAR-10 are evaluated under class-wise and random forgetting so that the behavior of DistilRepair can be studied under multiple forms of deletion.

4.4 Baselines

The primary baseline used in this study is *Retrain*, which retraining the model from scratch using only the retained data. Retraining is treated as the gold-standard reference because it directly removes the forgotten data from the training process and therefore provides the strongest practical target for comparison [Ginart et al. \(2019\)](#). DistilRepair is evaluated against retraining in both dense and sparse settings and under both forgetting scenarios.

The comparison is designed to answer a practical question: how closely can a post hoc approximate unlearning method approach retraining-level behavior while using substantially less time and computation? The study therefore emphasizes not only forgetting quality, but also retained utility and runtime.

4.5 Training and Unlearning Configuration

All experiments begin from a pre-trained checkpoint appropriate to the corresponding dataset and sparsity setting. DistilRepair then splits the training data into forget and retain subsets and alternates between the targeted forget stage and the retain-side repair stage. The method uses separate learning rates for the forget and repair updates, a scheduled anchoring weight, teacher-guided distillation on retained data, and a SAM-based repair optimizer [Foret et al. \(2021\)](#). In addition, classifier warmup, Batch Normalization freezing during forget updates, finite-value checks, finite-gradient checks, and gradient clipping are used to improve stability, especially in sparse models.

Table 4.1: DistilRepair hyperparameters for HAM (dense and sparse settings).

Parameter	Dense	Sparse
Batch size	64	64
Unlearning epochs	12	12
Forget LR	0.003	0.004
Repair LR	0.006	0.005
Forget steps/epoch	1	3
Repair steps/epoch	4	2
Warmup epochs	3	3
Anchor schedule	$2 \times 10^{-5} \rightarrow 2 \times 10^{-4}$	same
Forget CE weight	0.20	0.25
Entropy β	1.5	1.5
SAM ρ	2.0	2.0
BN recalibration cap	128	128

Table 4.2: DistilRepair hyperparameters for CIFAR-10 (same config for class-wise and random forgetting).

Parameter	Dense	Sparse
Batch size	256	256
Unlearning epochs	10	10
Forget LR	0.01	0.01
Repair LR	0.01	0.01
Forget steps/epoch	1	1
Repair steps/epoch	2	2
Warmup epochs	3	3
Anchor schedule	$2 \times 10^{-5} \rightarrow 2 \times 10^{-4}$	same
Forget CE weight	0.25	0.25
Entropy β	0.5	0.5
SAM ρ	2.0	2.0
BN recalibration cap	256	256

For HAM10000, the sparse experiments are performed on masks generated from the baseline checkpoint after iterative pruning. For CIFAR-10, the same dense-versus-sparse comparison is maintained using the ResNet-18 backbone. Across all settings, the

configuration follows a consistent experimental logic: the forget stage is kept strong enough to reduce dependence on the target data, while the repair stage is strong enough to recover retained utility. One practical detail is that the CIFAR-10 retain loader contains 159 mini-batches, so Batch Normalization recalibration effectively uses all 159 available retain batches even though the configured cap is 256.

4.6 Evaluation Metrics

The experiments are evaluated using multiple criteria. *Unlearning Accuracy (UA)* measures performance on the forgotten set and is used to indicate whether the designated data has been effectively removed from the model’s behavior. *Retained Accuracy (RA)* measures performance on the retained subset, while *Test Accuracy (TA)* measures generalization on held-out data. In addition, *Membership Inference Attack (MIA)* analysis [Shokri et al. \(2017\)](#) is used as a privacy-oriented metric to examine whether forgotten data remains distinguishable from non-member data after unlearning. Finally, *runtime* is reported to compare the practical efficiency of DistilRepair and retraining.

These metrics are used jointly rather than in isolation. A method that forgets aggressively but destroys retained utility is not practically useful, while a method that preserves utility but does not meaningfully reduce dependence on the target data cannot be considered successful unlearning. Similarly, strong task accuracy alone does not guarantee that the privacy risk associated with the forgotten data has been reduced. The evaluation protocol in this study therefore treats machine unlearning as a multi-objective problem involving forgetting efficacy, retained performance, generalization, privacy-sensitive behavior, and computational cost.

Chapter 5

Results and Analysis

5.1 Overview

This chapter compares DistilRepair with the gold-standard retraining baseline on HAM10000 with EfficientNet-B0 and CIFAR-10 with ResNet-18. Both dense and sparse models are evaluated under class-wise and random forgetting. The results are interpreted using four main performance criteria: unlearning accuracy (UA), membership-inference-based forget efficacy (MIA), retained accuracy (RA), and test accuracy (TA), together with runtime. In the reported tables and figures, higher UA and MIA indicate stronger forgetting efficacy, while higher RA and TA indicate better retained utility and generalization.

The chapter is organized around three questions. First, can DistilRepair approach retraining-level forgetting in the strongest benchmark settings? Second, what trade-off does it achieve between forgetting and retained utility in dense and sparse models? Third, does it provide a meaningful efficiency advantage over retraining? The HAM10000 results are presented first because they form the original study benchmark and represent the more difficult imbalanced setting. The CIFAR-10 results are then used to test whether the same overall behavior transfers to a more standard dataset and a different backbone.

5.2 HAM10000 Results

5.2.1 Class-Wise Forgetting

The dense class-wise HAM10000 result is the strongest outcome of the study. DistilRepair matches retraining on the forgetting-oriented metrics, with both methods reaching $UA = 100$ and $MIA = 100$. At the same time, it remains very close on retained utility, with RA decreasing only slightly from 95.39 to 94.52, while TA improves slightly from 81.98 to 82.23. This result shows that in the dense class-wise setting, DistilRepair effectively closes the gap between approximate unlearning and retraining.

The sparse class-wise result remains strong, although the forgetting side is not identical to retraining. Retraining reaches $UA = 100$, whereas DistilRepair reaches $UA = 96.14$.

However, DistilRepair provides a clearly better utility trade-off, improving RA from 88.20 to 91.36 and TA from 78.13 to 83.23. This is a practically important outcome because sparse models are the most optimization-sensitive regime in the study. Even without exactly matching sparse retraining on forget efficacy, DistilRepair produces a better overall balance between forgetting and retained performance.

These HAM10000 trends are summarized visually in Fig. 5.1, where the dense class-wise alignment with retraining and the sparse utility gains are easy to see at a glance.

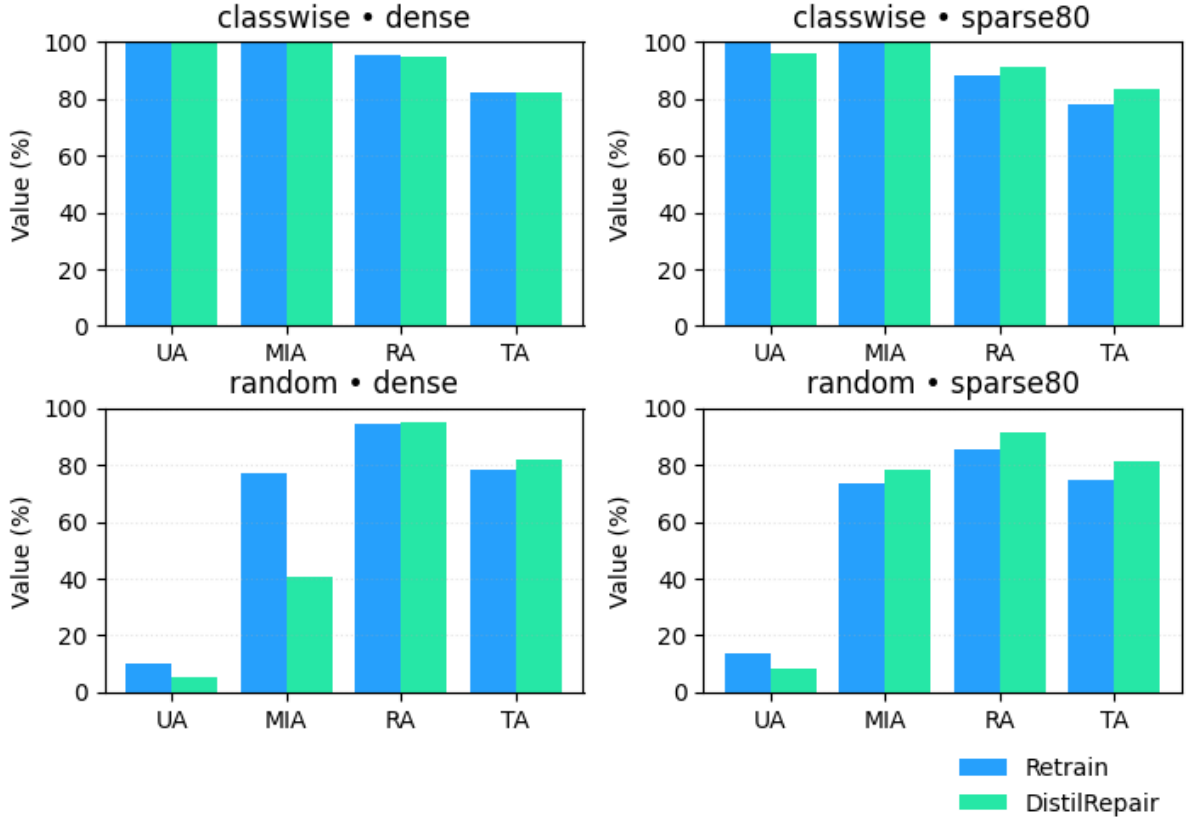


Figure 5.1: HAM10000 comparison of Retrain and DistilRepair across UA, MIA, RA, and TA for dense and sparse settings under class-wise and random forgetting.

5.2.2 Random Forgetting

The random-forgetting HAM10000 results tell a different story. In the dense setting, retraining is stronger on both forgetting-oriented metrics, reaching $UA = 10.30$ and $MIA = 76.82$, compared with $UA = 5.58$ and $MIA = 40.37$ for DistilRepair. However, DistilRepair improves RA from 94.25 to 94.88 and TA from 78.58 to 82.03. In the sparse setting, the trade-off is mixed rather than uniformly in retraining’s favor: retraining has higher UA (13.74 vs. 8.56), whereas DistilRepair has higher MIA forget efficacy (78.54 vs. 73.81) and also improves RA from 85.72 to 91.55 and TA from 75.00 to 81.28.

This pattern suggests that HAM10000 random forgetting is not the cleanest benchmark for demonstrating the core contribution of DistilRepair. In this regime, even retraining does not drive forgetting efficacy scores close to the class-wise levels, which indicates that the random-forgetting setup is intrinsically less separable on this dataset. For this reason, the strongest study evidence remains the class-wise setting, while the random-forgetting results are better interpreted as showing that DistilRepair preserves utility well even when exact forgetting is more difficult for both methods.

Table 5.1: Our experimental results on HAM10000.

MU	UA		MIA-Efficacy		RA		TA		Run Time(mins)	
	DENSE	80% SPARSITY	DENSE	80% SPARSITY	DENSE	80% SPARSITY	DENSE	80% SPARSITY	DENSE	80% SPARSITY
Class-wise forgetting on EfficientNet-B0										
Retrain	100	100	100	100	95.39	88.20	81.98	78.13	65.00	63.00
DistilRepair	100	96.14	100	100	94.52	91.36	82.23	83.23	5.69	3.86
Random forgetting on EfficientNet-B0										
Retrain	10.30	13.74	76.82	73.81	94.25	85.72	78.58	75.00	66.00	64.00
DistilRepair	5.58	8.56	40.37	78.54	94.88	91.55	82.03	81.28	5.74	5.32

5.2.3 Interpretation of the HAM10000 Setting

The HAM10000 results should be interpreted in light of the dataset’s class imbalance and the resulting checkpoint quality. Because the baseline model and sparse mask are derived from a challenging, mildly overfit training regime, these experiments are not idealized best-case unlearning results. Instead, they show how DistilRepair behaves when initialized from a realistic medical-imaging checkpoint with class imbalance sensitivity. From that perspective, the dense class-wise result is especially strong, and the sparse class-wise result is also credible because it preserves utility better than retraining while maintaining high forget efficacy.

5.3 CIFAR-10 Results

5.3.1 Class-Wise Forgetting

The CIFAR-10 class-wise results confirm that the main class-wise behavior of DistilRepair is not specific to HAM10000 or EfficientNet-B0. In the dense setting, both retraining and DistilRepair reach $UA = 100$ and $MIA = 100$, while DistilRepair remains very close in utility, with RA decreasing from 100.00 to 99.36 and TA from 94.83 to 94.31. This again indicates near-retraining behavior in the strongest forgetting benchmark.

The sparse CIFAR-10 class-wise result is similarly consistent. Retraining reaches $UA = 100$ and DistilRepair also reaches $UA = 100$, while the retained and test accuracies

remain close, with RA moving from 99.99 to 99.19 and TA from 93.92 to 93.33. Compared with HAM10000, the CIFAR-10 sparse class-wise gap is smaller on both the forgetting and utility sides, which suggests that DistilRepair is especially stable in a more balanced benchmark with a strong ResNet-18 starting point.

Figure 5.2 shows the same broad pattern across all CIFAR-10 settings: near-overlap with retraining in class-wise forgetting and a utility-oriented trade-off in random forgetting.

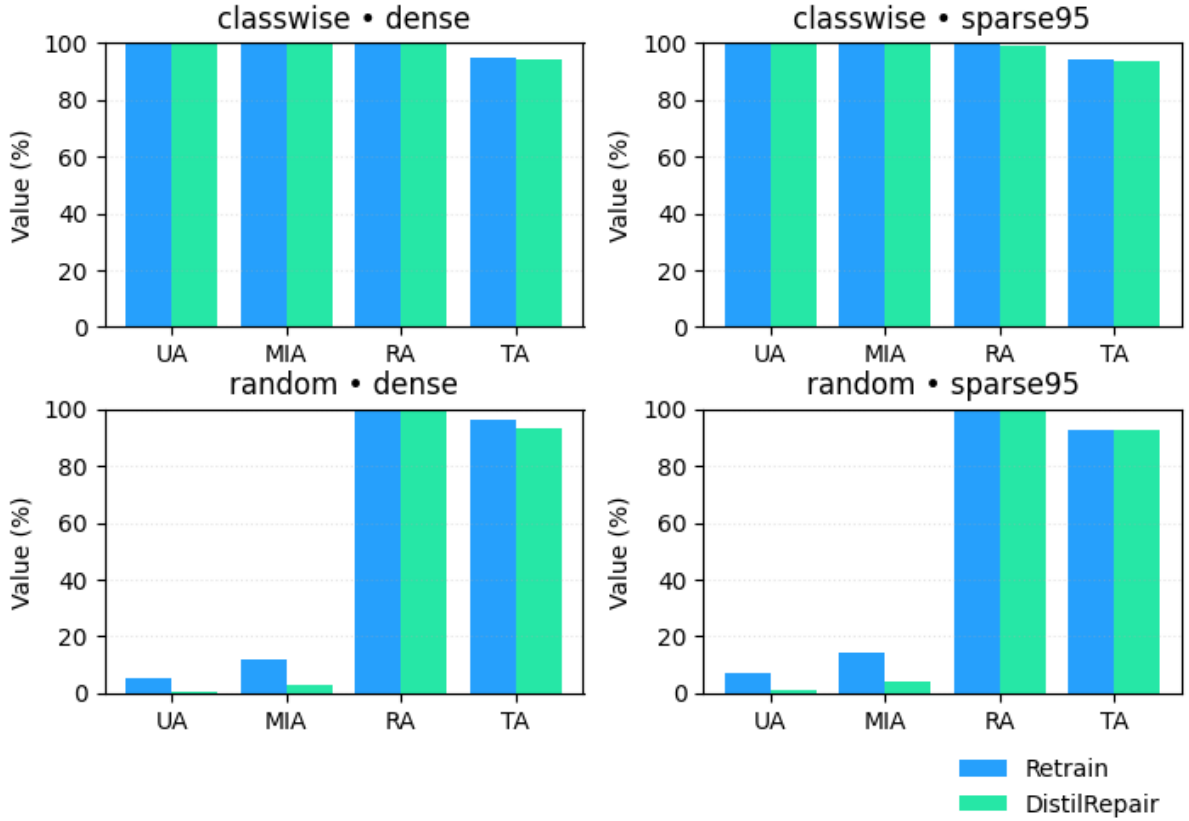


Figure 5.2: CIFAR-10 comparison of Retrain and DistilRepair across UA, MIA, RA, and TA for dense and sparse settings under class-wise and random forgetting.

5.3.2 Random Forgetting

The CIFAR-10 random-forgetting results resemble the HAM10000 random-forgetting pattern. Retraining remains stronger on forgetting efficacy, while DistilRepair remains competitive on utility. In the dense setting, retraining reaches $UA = 5.52$ and $MIA = 12.13$, whereas DistilRepair reaches $UA = 0.83$ and $MIA = 3.04$. DistilRepair nevertheless keeps high retained and test performance, with $RA = 99.17$ and $TA = 93.54$, compared with $RA = 99.99$ and $TA = 96.26$ for retraining. In the sparse setting, retraining again has better forgetting efficacy, while DistilRepair remains close in utility, with TA slightly improving from 92.77 to 92.82 despite lower UA and MIA.

Table 5.2: Our experimental results on CIFAR-10.

MU	UA		MIA-Efficacy		RA		TA		Run Time(mins)	
	DENSE	95% SPARSITY	DENSE	95% SPARSITY	DENSE	95% SPARSITY	DENSE	95% SPARSITY	DENSE	95% SPARSITY
Class-wise forgetting on ResNet-18										
Retrain	100	100	100	100	100	99.99	94.83	93.92	80.00	73.00
DistilRepair	100	100	100	100	99.36	99.19	94.31	93.33	2.84	2.76
Random forgetting on ResNet-18										
Retrain	5.52	7.18	12.13	14.46	99.99	99.99	96.26	92.77	76.00	75.00
DistilRepair	0.83	0.92	3.04	4.26	99.17	98.99	93.54	92.82	3.26	3.19

These results reinforce an important study interpretation. DistilRepair is strongest in the semantically coherent class-wise deletion setting, where the method was explicitly designed to balance targeted forgetting with repair of retained behavior. In contrast, random forgetting is a less favorable benchmark for the method, and retraining still has an advantage on the direct forget-efficacy scores. The practical value of DistilRepair in these random settings is therefore not stronger forgetting than retraining, but rather stronger retained utility at a much lower computational cost.

5.4 Runtime and Efficiency

Runtime is one of the clearest advantages of DistilRepair. On HAM10000, the method is approximately $11.4\times$ faster than retraining in the dense class-wise setting (5.69 vs. 65 minutes) and $16.3\times$ faster in the sparse class-wise setting (3.86 vs. 63 minutes). The random-forgetting runs show similar improvements, with speedups of approximately $11.5\times$ in the dense setting and $12.0\times$ in the sparse setting.

The CIFAR-10 efficiency gains are even larger. DistilRepair is approximately $28.2\times$ faster than retraining in the dense class-wise setting (2.84 vs. 80 minutes) and $26.4\times$ faster in the sparse class-wise setting (2.76 vs. 73 minutes). For random forgetting, the speedups remain large at approximately $23.3\times$ for dense models and $23.5\times$ for sparse models. These results show that the computational advantage of DistilRepair is consistent across datasets, backbones, and sparsity settings.

The runtime advantage is illustrated in Fig. 5.3 for HAM10000 and Fig. 5.4 for CIFAR-10. In both datasets, the gap is large enough that the efficiency claim does not depend on marginal differences or measurement noise.

This runtime result is important because retraining is the practical upper bound on fidelity but also the most expensive baseline. DistilRepair does not simply reduce runtime by a small margin; it reduces it by one order of magnitude on HAM10000 and by well over one order of magnitude on CIFAR-10, while still approaching retraining

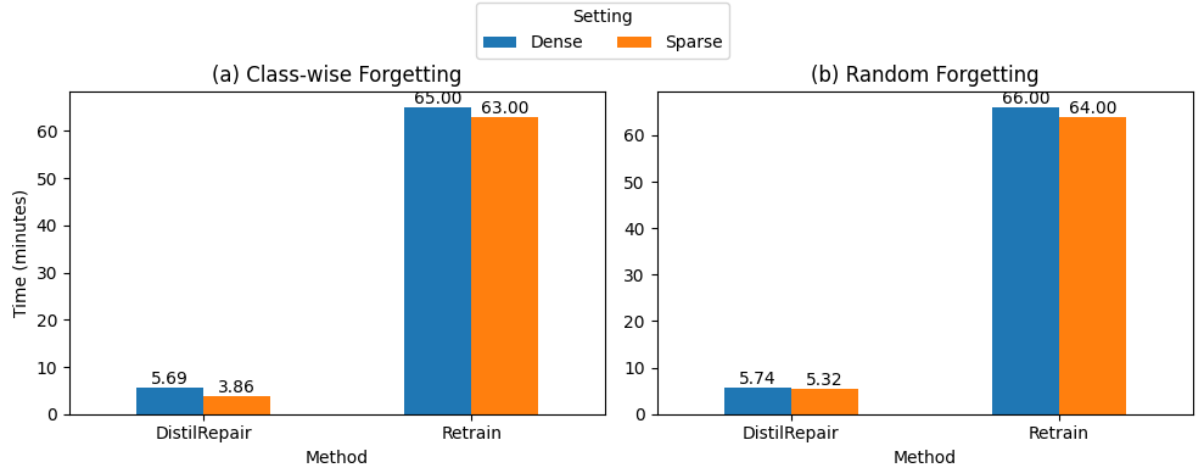


Figure 5.3: HAM10000 runtime comparison between Retrain and DistilRepair for dense and sparse models under class-wise and random forgetting.

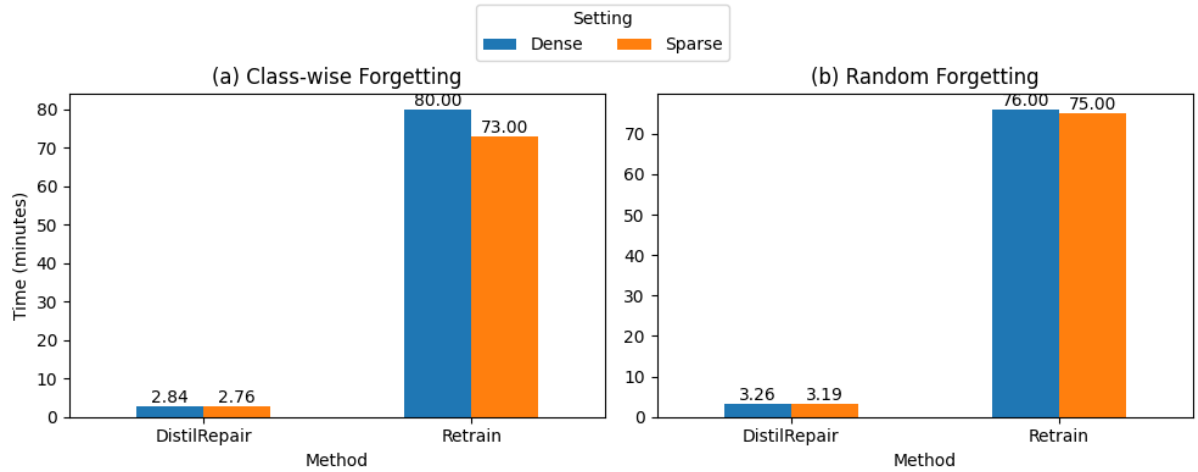


Figure 5.4: CIFAR-10 runtime comparison between Retrain and DistilRepair for dense and sparse models under class-wise and random forgetting.

in the class-wise setting. This makes the method attractive as a practical approximate unlearning procedure rather than only a proof-of-concept algorithm.

5.5 Privacy-Oriented Analysis

The MIA results broadly follow the same pattern as the unlearning accuracy scores. In the class-wise setting, DistilRepair matches or nearly matches retraining on the forget-efficacy-oriented privacy metric. In the random setting, retraining generally remains stronger. This consistency is useful because it suggests that the class-wise gains of DistilRepair are not limited to task-level forgetting behavior alone, but are also reflected in the privacy-oriented evaluation.

At the same time, the random-forgetting results show why privacy metrics should not be interpreted in isolation. A method can preserve strong retained and test performance without matching retraining on privacy-oriented forget efficacy, and that is exactly what occurs in several random-forgetting settings here. For this reason, the study evaluates forgetting as a multi-objective problem rather than selecting a winner based on a single metric.

5.6 Summary of Findings

The results support three main conclusions. First, DistilRepair is strongest in class-wise forgetting, where it approaches retraining very closely in dense models and remains highly competitive in sparse models. Second, in both HAM10000 and CIFAR-10 random forgetting, DistilRepair generally sacrifices some forget efficacy relative to retraining but preserves stronger retained utility. Third, the method is consistently much faster than retraining, with double-digit speedups across all reported settings.

Taken together, these findings position DistilRepair as a practical approximate unlearning method whose main strength is not universal dominance over retraining on every metric, but rather a favorable balance between forgetting quality, retained utility, and computational cost. This balance is most convincing in the class-wise setting, which should therefore remain the central empirical claim of the study.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This study explored approximate machine unlearning in dense and sparse neural networks, with the goal of reducing the gap between practical post hoc unlearning and gold-standard retraining. The central challenge was to remove the influence of designated training data while preserving retained utility, especially in sparse settings where optimization is more fragile. To address this problem, the study proposed DistilRepair, a controlled forget-and-repair framework that combines targeted forgetting, retain-side teacher guidance, parameter anchoring, and stability-oriented optimization.

The empirical results show that DistilRepair is most convincing in class-wise forgetting. On HAM10000 with EfficientNet-B0, the method achieved near-retraining performance in the dense class-wise setting while reducing runtime by approximately one order of magnitude. In the sparse class-wise setting, DistilRepair did not exactly match retraining on forget efficacy, but it preserved clearly stronger retained and test performance, indicating a more favorable overall balance between forgetting and utility. These results are especially meaningful because HAM10000 is highly imbalanced, and the corresponding checkpoint and sparse mask represent a practically challenging, non-ideal initialization rather than a perfectly calibrated starting point.

The CIFAR-10 experiments with ResNet-18 further strengthened the main study claim by showing that the core behavior of DistilRepair transfers across datasets, architectures, and sparsity settings. In class-wise forgetting, the method again remained close to retraining in both dense and sparse models. In random forgetting, retraining generally remained stronger on direct forget-efficacy scores, but DistilRepair continued to preserve strong retained utility while delivering a large runtime advantage. Taken together, these findings show that DistilRepair is not a universal replacement for retraining on every metric, but it is a practical and effective approximate unlearning method when the objective is to achieve strong forgetting with minimal computational cost and limited damage to retained performance.

Overall, the study makes three main contributions. First, it presents DistilRepair as an anchored distillation-based unlearning framework that improves the original gradient-ascent-plus-repair idea in a more controlled and stable form. Second, it shows that approximate unlearning can approach retraining much more closely in class-wise forgetting than earlier results suggested, including under sparsity. Third, it demonstrates that substantial runtime reductions can be achieved without collapsing retained utility, making approximate unlearning more practical for real deployment settings.

6.2 Future Work

Several directions follow naturally from this work. The first is broader empirical validation. Although HAM10000 and CIFAR-10 provide two useful and contrasting benchmarks, future work should evaluate DistilRepair on larger datasets, higher-resolution tasks, and more diverse architectures. In particular, testing on larger modern vision benchmarks would help determine whether the same balance between forgetting, utility, and efficiency holds as model scale increases.

The second direction is deeper study of sparsity. This study considered sparse unlearning under fixed pruning settings, but more extensive experiments could explore different sparsity ratios, alternative pruning strategies, and structured sparsity patterns. Such analysis would help clarify whether the utility-preserving behavior of DistilRepair persists across a broader range of sparse model regimes, and whether some forms of sparsity are more favorable for approximate unlearning than others.

The third direction is stronger privacy evaluation. The present study used membership-inference-oriented forget-efcacy metrics as a practical privacy-sensitive signal, but these metrics do not provide a complete privacy guarantee. Future work could incorporate stronger adversarial evaluation, additional attack models, and more rigorous privacy-focused benchmarks. This would help clarify how closely DistilRepair approximates the privacy behavior of retraining, especially in the more difficult random-forgetting regime.

Finally, future work could explore broader deployment-oriented settings. Real systems may require repeated deletion requests, streaming updates, or unlearning under distribution shift. Extending DistilRepair to repeated or continual unlearning scenarios would be particularly valuable, since practical applications rarely involve only a single isolated deletion event. Addressing these directions would strengthen both the empirical foundation and the practical applicability of approximate machine unlearning beyond the scope of this study.

References

- Bourtole, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., ... Papernot, N. (2021). Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)* (pp. 141–159).
- Cadet, X. F., Borovykh, A., Malekzadeh, M., Ahmadi-Abhari, S., & Haddadi, H. (2025). Deep unlearn: Benchmarking machine unlearning for image classification. In *2025 IEEE 10th European Symposium on Security and Privacy (EuroSec)* (pp. 939–962).
- Chen, K., Wang, Y., & Huang, Y. (2021). Lightweight machine unlearning in neural network. *arXiv preprint arXiv:2111.05528*.
- Chundawat, V. S., Tarun, A. K., Mandal, M., & Kankanhalli, M. (2023). Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 37, pp. 7210–7217).
- Dang, Y., Liu, Y., Yang, E., Guo, G., Jiang, L., Zhao, J., & Wang, X. (2025). Efficient and adaptive recommendation unlearning: A guided filtering framework to erase outdated preferences. *ACM Transactions on Information Systems*, 43(2), 1–25.
- Dontsov, A., Korzh, D., Zhavoronkin, A., Mikheev, B., Bobkov, D., Alanov, A., ... Tutubalina, E. (2024). Clear: Character unlearning in textual and visual modalities. *arXiv preprint arXiv:2410.18057*.
- Fan, C., Liu, J., Zhang, Y., Wong, E., Wei, D., & Liu, S. (2023). Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. *arXiv preprint arXiv:2310.12508*.
- Foret, P., Kleiner, A., Mobahi, H., & Neyshabur, B. (2021). Sharpness-aware minimization for efficiently improving generalization. In *International conference on learning representations*.
- Foster, J., Schoepf, S., & Brintrup, A. (2024a). Fast machine unlearning without retraining through selective synaptic dampening. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 38, pp. 12043–12051).
- Foster, J., Schoepf, S., & Brintrup, A. (2024b). Loss-free machine unlearning. *arXiv preprint arXiv:2402.19308*.
- Gao, L., Niu, Y., Tang, T., Avestimehr, S., & Annavaram, M. (2024). Ethos: Rectifying language models in orthogonal parameter space. *arXiv preprint arXiv:2403.08994*.

- Ginart, A., Guan, M. Y., Valiant, G., & Zou, J. Y. (2019). Making AI forget you: Data deletion in machine learning. In *Advances in neural information processing systems* (Vol. 32).
- Golatkhar, A., Achille, A., & Soatto, S. (2020). Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9304–9312).
- Guo, C., Goldstein, T., Hannun, A., & van der Maaten, L. (2020). Certified data removal from machine learning models. In *Proceedings of the 37th international conference on machine learning* (Vol. 119, pp. 3832–3842). PMLR.
- Guo, P., Syed, A., Sheshadri, A., Ewart, A., & Dziugaite, G. K. (2024). Mechanistic unlearning: Robust knowledge unlearning and editing via mechanistic localization. *arXiv preprint arXiv:2410.12949*.
- He, E., Sarwar, T., Khalil, I., Yi, X., & Wang, K. (2025). Deep contrastive unlearning for language models. *arXiv preprint arXiv:2503.14900*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
- Jia, J., Liu, J., Ram, P., Yao, Y., Liu, G., Liu, Y., ... Liu, S. (2023). Model sparsity can simplify machine unlearning. *Advances in Neural Information Processing Systems*, 36, 51584–51605.
- Jin, D., Chen, G., Feng, S., Ling, Y., & Zhu, H. (2025). Machine unlearning for robust dnns: Attribution-guided partitioning and neuron pruning in noisy environments. *Machine Learning and Knowledge Extraction*, 7(3), 95.
- Krizhevsky, A. (2009). *Learning multiple layers of features from tiny images* (Tech. Rep.). University of Toronto.
- Kurmanji, M., Triantafillou, E., & Triantafillou, P. (2024). Machine unlearning in learned databases: An experimental analysis. *Proceedings of the ACM on Management of Data*, 2(1), 1–26.
- Kurmanji, M., Triantafillou, P., Hayes, J., & Triantafillou, E. (2023). Towards unbounded machine unlearning. *Advances in neural information processing systems*, 36, 1957–1987.
- Li, X., Wang, J., Yuan, X., Zhang, P., Qin, Z., Wang, Z., & Ren, K. (2025). Prune: A patching based repair framework for certifiable unlearning of neural networks. *arXiv preprint arXiv:2505.06520*.
- Li, Z., Ye, Q., & Hu, H. (2025). Funu: Boosting machine unlearning efficiency by filtering unnecessary unlearning. In *Proceedings of the ACM on web conference 2025* (pp. 3366–3376).
- Liu, Z., Dou, G., Yuan, X., Zhang, C., Tan, Z., & Jiang, M. (2025). Modality-aware neuron pruning for unlearning in multimodal large language models. *arXiv preprint arXiv:2502.15910*.

- Pawelczyk, M., Neel, S., & Lakkaraju, H. (2023). In-context unlearning: Language models as few shot unlearners. *arXiv preprint arXiv:2310.07579*.
- Pochinkov, N., & Schoots, N. (2024). Dissecting language models: Machine unlearning via selective pruning. *arXiv preprint arXiv:2403.01267*.
- Shah, V., Träuble, F., Malik, A., Larochelle, H., Mozer, M., Arora, S., ... Goyal, A. (2023). Unlearning via sparse representations. *arXiv preprint arXiv:2311.15268*.
- Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *Proceedings of the IEEE Symposium on Security and Privacy* (pp. 3–18).
- Tan, M., & Le, Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 6105–6114).
- Tschandl, P., Rosendahl, C., & Kittler, H. (2018). The HAM10000 dataset: A large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data*, 5(180161).
- Wang, C., Ying, Z., & Pan, Z. (2024). Machine unlearning in brain-inspired neural network paradigms. *Frontiers in Neurobotics*, 18, 1361577.
- Wang, J., Guo, S., Xie, X., & Qi, H. (2022). Federated unlearning via class-discriminative pruning. In *Proceedings of the ACM web conference 2022* (pp. 622–632).
- Xiao, Y., Li, G., Ji, J., Ye, R., Ma, X., & Hui, B. (2025). The right to be forgotten in pruning: Unveil machine unlearning on sparse models. *arXiv preprint arXiv:2507.18725*.
- Zaman, M. M. U., Sun, X., & Yao, J. (2025). Sky of unlearning (soul): Rewiring federated machine unlearning via selective pruning. *arXiv preprint arXiv:2504.01705*.
- Zhang, R., Lin, L., Bai, Y., & Mei, S. (2024). Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*.
- Zhang, S., Zhang, L., Zhou, J., Zheng, Z., & Xiong, H. (2025). Llm-eraser: Optimizing large language model unlearning through selective pruning. In *Proceedings of the 31st ACM SIGKDD conference on knowledge discovery and data mining v. 1* (pp. 1960–1971).