UNIVERSITY OF CALIFORNIA,
IRVINE

TRUSTWORTHY AND EFFICIENT MACHINE LEARNING SYSTEMS

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Engineering

by

Yezi Liu

Dissertation Committee:
Associate Professor Mohsen Imani, Chair
Associate Professor Hamidreza Aghasi
Distinguished Professor Fadi J. Kurdahi

2026

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would first like to thank my family for their selfless devotion, unconditional trust, and constant love. Their faith in me has been a steady presence throughout these years, and I am deeply grateful for everything they have given me along the way.

I am deeply grateful to the members of my dissertation committee, and to those who advised and mentored me throughout my academic journey. I am thankful for their encouragement, patience, trust, and recognition, as well as for the opportunities, resources, and support that allowed me to pursue my research and grow as a researcher. Their guidance has shaped both this dissertation and the way I think about research.

I am also grateful to my labmates and collaborators for the ideas, conversations, shared work, and companionship that became part of these years. Research is rarely the work of one person alone, and I have learned a great deal from the people with whom I had the chance to work.

I am profoundly grateful to my friends. Some of the things I value most from these years are the people I came to know and the friendships I was fortunate enough to have. My friends checked in on me regularly and asked whether there was anything they could do. Even when I said that I needed nothing, they would still find something to offer, whether large or small. They were patient, thoughtful, attentive, and generous with their time. They cared about what mattered to me, respected my choices, trusted my judgment, and were always ready to stand by me. They tried to understand before judging, offered help without waiting to be asked, and expected nothing in return. Whenever they heard news or learned of something that might matter to me, they thought of me and made sure that I knew. Their friendship never depended on my circumstances or on what I happened to have achieved at the time. That is something I value more than I can easily put into words.

I am especially grateful for the optimism, warmth, and generosity my friends brought into my life. Throughout these years, I have tried to approach each new question with the belief that there is always a way forward, and I was fortunate to be surrounded by people who carried the same spirit. They are the kind of people I hope to become: positive, thoughtful, generous, and willing to care for others without asking what they will receive in return. I hope that I can give them, and others, the same kind of care that they have given me. I need not name them here. They know who they are. I would rather return their kindness in the years ahead and be there for them as they have been for me.

I would also like to thank the staff members and administrators who supported me throughout my doctoral studies. Whenever I needed their help, they listened carefully, considered my circumstances, encouraged me to pursue the options available to me, and actively looked for ways to help. I am grateful not only for their professionalism, but also for the thoughtfulness and care with which they treated me. Their willingness to help made many things possible, and I will always remember it.

Above all, I want to thank myself. For every choice, every effort, and every step that brought me here, I am grateful.

And to whoever has made it this far: whatever brought you here, I wish you a good morning. And, in case I don't see you outside these pages, good afternoon, good evening, and good night. May you leave these pages in good spirits and always carry a little kindness wherever you go.

Chapter 3 of this dissertation is a reprint of the material as it appears in *Beyond Node Condensation: Learning Tiny Graphs via Joint Graph Condensation*, Companion Proceedings of the ACM Web Conference, 2025, used with permission from the Association for Computing Machinery. The coauthor listed in this publication is Yanning Shen.

Chapter 4 of this dissertation is a reprint of the material as it appears in *Debias Once for All: A Data-Centric Strategy for Fair Machine Learning*, Proceedings of the ACM International Conference on Web Search and Data Mining, 2026, used with permission from the Association for Computing Machinery. The coauthors listed in this publication are Hanning Chen, Yanning Shen, and Mohsen Imani.

Chapter 5 of this dissertation is a reprint of the material as it appears in *Fairness via Independence: A General Regularization Framework for Machine Learning*, Proceedings of the International Conference on Learning Representations, 2026. The coauthors listed in this publication are Hanning Chen, Wenjun Huang, Yang Ni, and Mohsen Imani.

Chapter 6 of this dissertation is a reprint of the material as it appears in *Enabling Group Fairness in Machine Unlearning via Distribution Correction*, Proceedings of the ACM International Conference on Information and Knowledge Management, 2025, used with permission from the Association for Computing Machinery. The coauthor listed in this publication is Yanning Shen.

Chapter 7 of this dissertation is a reprint of the material as it appears in *LUNE: Efficient LLM Unlearning via LoRA Fine-Tuning with Negative Examples*, Findings of the Association for Computational Linguistics, 2026, used with permission from the Association for Computational Linguistics. The coauthors listed in this publication are Hanning Chen, Wenjun Huang, Yang Ni, and Mohsen Imani.

Yanning Shen and Mohsen Imani directed and supervised the research which forms the basis for this dissertation.

Specific funding acknowledgements for individual projects are detailed within their respective chapters.

VITA

Yezi Liu

EDUCATION

Doctor of Philosophy in Computer Engineering **2026**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2021**
Texas A&M University *College Station, Texas*

Master of Science in Information Science **2019**
University of Pittsburgh *Pittsburgh, Pennsylvania*

B.Econ. in Economics **2017**
Southwest University *Chongqing, China*

TEACHING EXPERIENCE

Teaching Assistant **Oct 2022 – June 2024**
University of California, Irvine *Irvine, California*

Intermediate Programming (ICS 33); Computer Systems and C Programming (EECS 20); Introduction to Digital Systems (EECS 31); Introduction to Digital Logic Laboratory (EECS 31L).

FIELD OF STUDY

Computer Engineering

PUBLICATIONS INCLUDED IN THIS DISSERTATION

Y. Liu, H. Chen, W. Huang, Y. Ni, and M. Imani. Fairness via Independence: A General Regularization Framework for Machine Learning. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026.

Y. Liu, H. Chen, W. Huang, Y. Ni, and M. Imani. LUNE: Efficient LLM Unlearning via LoRA Fine-Tuning with Negative Examples. *Findings of the Association for Computational Linguistics (ACL Findings)*, 2026.

Y. Liu, H. Chen, Y. Shen, and M. Imani. Debias Once for All: A Data-Centric Strategy for Fair Machine Learning. *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, 2026.

Y. Liu and Y. Shen. Enabling Group Fairness in Machine Unlearning via Distribution Correction. *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, 2025.

Y. Liu and Y. Shen. Beyond Node Condensation: Learning Tiny Graphs via Joint Graph Condensation. *Companion Proceedings of the ACM Web Conference (WWW), short paper*, 2025.

OTHER PUBLICATIONS

W. Huang, S. Fu, Y. Jin, Y. Ni, Z. Cui, H. Chen, Y. He, Y. Liu, S. Yun, S. Jeong, R. Masukawa, W. Y. Chung, and M. Imani. MERIT: Multi-domain Efficient RAW Image Translation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

W. Huang, Y. Ni, H. Chen, Y. He, Y. Liu, I. Bryant, Mahdi Imani, and Mohsen Imani. Tell Me What to Track: Infusing Robust Language Guidance for Enhanced Referring Multi-Object Tracking. *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2026.

Y. Liu, J. Xie, and Y. Shen. DGExplainer: Explaining Dynamic Graph Neural Networks via Relevance Back-propagation. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2025.

Y. Liu, W. Huang, Y. Ni, H. Chen, and M. Imani. White Admitted by Stanford, Black Got Rejections: Exploring Racial Stereotypes in Text-to-Image Generation from a College Admissions Lens. *International World Wide Web Conference (WWW), short paper*, 2025.

Y. Liu, H. Chen, W. Huang, Y. Ni, and M. Imani. Recover-to-Forget: Gradient Reconstruction from LoRA for Efficient LLM Unlearning. *NeurIPS ResponsibleFM Workshop*, 2025.

H. Chen, Y. Ni, W. Huang, H. Oh, Y. Liu, T. Das, and M. Imani. LVLM-CSP: Accelerating Large Vision Language Models via Clustering, Scattering, and Pruning for Reasoning Segmentation. *ACM Multimedia (ACM MM)*, 2025.

H. Chen, Y. Ni, W. Huang, Y. Liu, S. Jeong, F. Wen, N. D. Bastian, H. Latapie, and M. Imani. VLTP: Vision-Language Guided Token Pruning for Task-Oriented Segmentation. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

W. Huang, Y. Ni, A. Rezvani, S. Jeong, H. Chen, Y. Liu, F. Wen, and M. Imani. Recoverable Anonymization for Pose Estimation: A Privacy-Enhancing Approach. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

Y. Liu and Y. Shen. TinyData: Joint Dataset Condensation with Dimensionality Reduction. *European Signal Processing Conference (EUSIPCO)*, 2024.

H. Chen et al. TaskCLIP: Extend Large Vision-Language Model for Task Oriented Object Detection. *ECCV FOCUS Workshop*, 2024.

Y. Liu. FairGraph: Automated Graph Debiasing with Gradient Matching. *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), short paper*, 2023.

Y. Liu, Q. Zhang, M. Du, X. Huang, and X. Hu. Error Detection on Knowledge Graphs with Triple Embedding. *European Signal Processing Conference (EUSIPCO)*, 2023.

Q. Zhang et al. Contrastive Knowledge Graph Error Detection. *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.

Z. Wang et al. RES: An Interpretable Replicability Estimation System for Research Publications. *AAAI Conference on Artificial Intelligence (AAAI), demonstration track*, 2022.

Y. Liu. Semi-supervised Multi-label Dimensionality Reduction via Low Rank Representation. *International Conference on Neural Information Processing (ICONIP)*, 2018.

Y. Liu, H. Chen, and M. Imani. Promoting Fairness. *Frontiers in Big Data*.

D. Sinha, Y. Liu, R. Du, and Y. Shen. Gradient Inversion Attack on Graph Neural Networks. *Transactions on Machine Learning Research (TMLR)*.

Q. Tan, Y. Liu, X. Chen, and G. Yu. Multi-Label Classification Based on Low Rank Representation for Image Annotation. *Remote Sensing*.

Y. Liu, W. Y. Chung, H. Chen, C. Yeung, and M. Imani. Are Hypervectors Enough? Single-Call LLM Reasoning over Knowledge Graphs. *Under review*.

Y. Liu, W. Y. Chung, Y. Ni, H. Chen, and M. Imani. Fairness-aware Graph Hyperdimensional Computing. *Under review*.

Y. Liu, P. Poduval, W. Huang, N. Yang, H. Chen, and M. Imani. FGU: Enabling Group Fairness in Graph Unlearning via Global Alignment. *Under review*.

ABSTRACT OF THE DISSERTATION

Trustworthy and Efficient Machine Learning Systems

By

Yezi Liu

Doctor of Philosophy in Computer Engineering

University of California, Irvine, 2026

Associate Professor Mohsen Imani, Chair

A deployed machine learning system must be affordable to train, must treat demographic groups equitably, and must be able to forget data on request. These demands are studied by separate literatures, evaluated by disjoint metrics, and addressed by methods that share almost no machinery.

This dissertation argues that the separation is an artifact of how the problems have been posed. Efficiency, group fairness, and the right to be forgotten are three demands on one object, the training dynamics of a learning system, meaning the gradients a model sees and the parameter updates it retains, and all three can be met by intervening on that object rather than by redesigning architectures. The claim is developed at three control points.

At the *data*, a small synthetic dataset is optimized so that models trained on it follow the same optimization trajectory as models trained on a large one. For graph learning this jointly condenses the node count and the feature dimension, cutting storage by over 99% while retaining up to 98.5% of the original accuracy. Because gradient matching encodes whatever property the target trajectory possesses, redirecting it from accuracy to equity yields data on which ordinary training produces fair models that transfer across architectures, including to tree ensembles no differentiable regularizer can reach.

At the *objective*, in-process fairness methods are shown to reduce to a single choice of distance between group-conditional distributions. Two failure modes yield three requirements on that measure, namely a tight bound on the quantity of interest, robustness to scale, and applicability to arbitrary distributions, which the Cauchy–Schwarz divergence satisfies and the standard alternatives do not.

At the *parameters*, a negative result: machine unlearning destroys installed fairness. A fairly trained model meeting a routine deletion request sees its demographic parity gap rise nineteen-fold while accuracy falls five percent, undetected by standard evaluation. The cause is a demographically one-sided edit to the sum of per-batch updates; the remedy is a matched, reversed-attribute edit, restoring fairness without retraining and up to 318× faster. At foundation-model scale the same principle confines the edit to low-rank adapters trained on negative examples alone.

One lever, applied at three points, serves three demands ordinarily pursued in isolation.

Chapter 1

Introduction

1.1 Three Demands on One Pipeline

A machine learning system that is deployed rather than merely reported must satisfy several requirements at once. It must be affordable to train, since the models and datasets that are useful in practice are large and are retrained often. It must treat people equitably, since models learned from historical records reproduce and can amplify the disparities those records encode. And it must be able to forget, since data protection law now obliges organizations to erase personal records on request, and in the machine learning setting erasure means removing not just the data but its influence on models already trained on it.

These three demands, namely efficiency, equitable treatment, and erasure, arise from different places. The first is economic, the second ethical and increasingly legal, the third purely legal. They are studied by largely separate literatures, evaluated by disjoint sets of metrics, and addressed by methods that share almost no machinery. A practitioner assembling a system that must satisfy all three currently assembles it from parts that were not designed to fit together.

This dissertation argues that the separation is an artefact of how the problems have been posed rather than a fact about them. All three are demands on the same underlying object, and all three can be met by intervening on that object in the same way.

1.2 Why Model-Centric Fixes Do Not Compose

The standard response to each demand is to modify the model. Efficiency is pursued through smaller architectures, cheaper layers, and better optimizers. Fairness is pursued by adding constraints or penalty terms to the training objective, or by adjusting the outputs of a trained model. Erasure is pursued by algorithms that reconstruct a model as though the deleted data had never been present.

Each of these works in isolation, and each has a structural weakness that only becomes visible when they are combined.

A fairness intervention that lives inside a particular model is not reusable. Every new architecture trained on the same biased data requires its own debiasing pass, its own hyperparameter search, and its own validation. In a modular or rapidly evolving pipeline the intervention must be repeated indefinitely, and the cost of doing so is charged against the first demand.

An efficiency intervention that only shrinks one dimension of the problem relocates the cost rather than removing it. Chapter 3 shows this concretely for graph learning, where reducing the number of training nodes while leaving the feature dimension untouched leaves the dominant term of the training cost in place.

And, in the observation that motivates the second half of this dissertation, a guarantee established during training is not durable. Chapter 6 shows that a model trained to be fair, then subjected to an ordinary deletion request by a standard unlearning algorithm,

emerges categorically unfair: on one benchmark the demographic parity gap rises from 1.6 to 30.8 while accuracy falls by only five percent. The fairness intervention and the erasure mechanism were each correct in isolation; composed, the second silently destroyed what the first installed. Neither literature's evaluation would have detected it, because fairness is not among the quantities that unlearning is evaluated on.

What these failures share is that each intervention is attached to a different part of the system, and nothing coordinates them.

1.3 Thesis Statement

The claim of this dissertation is the following.

> Efficiency, group fairness, and the right to be forgotten are not competing objectives requiring separate machinery. They are three demands on a single object, the *training dynamics* of a learning system, meaning the gradients a model is exposed to and the parameter updates it retains. All three can be met by intervening on that object rather than by redesigning architectures. Because the interventions act on a common surface, they compose, and where they fail to compose the failure is visible and repairable.

Two consequences follow, and both are developed in the chapters that follow. The first is that the same mechanism can serve different requirements: a technique built to make training cheap can be redirected, essentially unchanged, to make it equitable. The second is that interventions which appear unrelated can silently interfere, and that the discipline of treating them as acting on one surface is what makes the interference detectable.

1.4 Three Control Points

Training dynamics can be influenced at three places, and this dissertation organizes its contributions accordingly. Table 1.1 summarizes the correspondence.

The data. What a model learns is determined by the gradients its training data induces. If a small synthetic dataset can be constructed so that models trained on it follow the same optimization trajectory as models trained on a large real one, then the dataset becomes a design variable rather than a fixed input, and whatever property is encoded into that trajectory is inherited by every model trained on the result. Chapter 3 exercises this for cost and Chapter 4 for equity.

The objective. What counts as a good trajectory is determined by the loss being minimized. When the requirement is fairness, this reduces to a choice of measure between group-conditional prediction distributions, a choice that the literature has made largely by convention. Chapter 5 asks what properties such a measure should have and derives one that possesses them.

The parameters. What a model retains is determined by the updates it has absorbed, and those updates can be edited after training has finished. Deletion requests force such edits, and Chapter 6 shows both that they destroy fairness and that the destruction can be undone by a matched, compensating edit. Chapter 7 carries the same principle to foundation-model scale, where the edit must be confined to a low-rank subspace because the model can no longer be revised wholesale.

Table 1.1: The three control points of the training pipeline, the demand addressed at each, and the chapters that develop them.

Control point	Intervention	Demand	Chapter
The data	Match gradients with a learned dataset	Efficiency Equity	Chapter 3 Chapter 4
The objective	Choose the measure that shapes the gradient	Equity	Chapter 5
The parameters	Edit the updates already committed	Erasure, equity Erasure, efficiency	Chapter 6 Chapter 7

1.5 Research Questions

The dissertation is organized around five questions.

RQ1 When the cost of training is driven jointly by the number of samples and the dimension of their features, can both be reduced together, and what must be preserved for a model trained on the result to remain accurate? (Chapter 3)

RQ2 Can group fairness be optimized into a training dataset rather than into a model, and does fairness so encoded survive a change of downstream architecture? (Chapter 4)

RQ3 What properties should a fairness regularizer have, and which measure of discrepancy between group-conditional distributions possesses them? (Chapter 5)

RQ4 Does a fairness guarantee established during training survive the parameter edits required by a deletion request, and if not, can it be restored without retraining? (Chapter 6)

RQ5 Can forgetting be made both efficient and local at foundation-model scale, where the training history is unavailable and wholesale weight revision is unsafe? (Chapter 7)

1.6 Contributions

The dissertation makes the following contributions.

- **Joint feature and node condensation for graph learning.** It identifies a limitation that persists in existing graph condensation methods, namely that reducing the node count alone leaves the dominant term of the training cost intact when features are high-dimensional, and shows that joint reduction requires the feature projection to be structure-aware and learned together with the condensed data. The resulting framework retains up to 98.5% of the original accuracy while removing more than 97% of nodes, up to 90% of the feature dimension, and over 99% of storage.

- **Fairness encoded into data rather than models.** It shows that the gradient-matching machinery developed for efficiency transfers, essentially unchanged, to group fairness: pairing it with a differentiable fairness regularizer yields a synthetic dataset on which ordinary training produces fair models. The dataset carries no sensitive attribute, stays in the original input space, and transfers across architectures, including to a gradient-boosted tree ensemble to which no differentiable fairness regularizer could have been applied.

- **A principled account of fairness regularizers.** It organizes in-process fair learning methods into three families and shows that all three reduce to a single choice of distance measure, converting an unstructured menu of methods into a well-posed question. From two documented failure modes it derives three desiderata, and shows that the Cauchy–Schwarz divergence satisfies all three where the standard measures each fail at least one. The resulting regularizer attains the best equal opportunity gap in nine of ten dataset–attribute settings.

- **Fairness-preserving machine unlearning.** It establishes that machine unlearning destroys group fairness, that the failure afflicts exact and approximate mechanisms alike, and that the standard evaluation rubric cannot detect it. It attributes the failure to a demographically one-sided edit to the sum of per-batch updates, and corrects it by a matched, reversed-attribute edit to the same sum. This restores fairness to the

level of fair retraining while running up to 318× faster, and eliminates membership inference signal immediately, which retraining itself does not achieve.

- **Efficient unlearning at foundation-model scale.** It shows that unlearning a fact from a large language model requires touching almost none of it. Freezing the backbone and training low-rank adapters on negative examples alone matches or exceeds methods that update every parameter, on unlearning efficacy, adversarial robustness, and privacy leakage simultaneously, while retaining general utility better than any baseline tested.

- **A unifying account.** Read together, these five results support the claim of Section 1.3: that a single lever, applied at three points, serves three demands that are ordinarily pursued separately, and that treating them as acting on one surface is what makes their interference visible.

1.7 Roadmap

Chapter 2 consolidates the notation and technical background shared across the dissertation: empirical risk minimization and training trajectories, graph neural networks, dataset condensation and gradient matching, group fairness criteria and their differentiable surrogates, the divergence measures used to compare group-conditional distributions, machine unlearning, and low-rank adaptation. Each main chapter recalls the specialization it needs, so Chapter 2 may be read once and referred back to.

The main chapters follow the three control points in order. Chapter 3 and Chapter 4 operate on the data, the first for efficiency and the second for equity. Chapter 5 operates on the objective, answering a question that Chapter 4 leaves open. Chapter 6 and Chapter 7 operate on the parameters, the first showing that post-training edits undo what the earlier chapters

install and repairing it, the second carrying the same principle to a scale at which the earlier assumptions fail. Chapter 8 draws the results together, records what the dissertation does not establish, and sets out the open problems the arc points at most directly.

The chapters are written to be read in order, since each opens on a limitation named at the end of its predecessor, but each is self-contained given Chapter 2.

1.8 Previously Published Material

Each of the five main chapters is a reworking of a published or submitted paper, adapted for consistency of notation, framing, and evaluation, with material added where the dissertation's argument requires it. In particular, the preliminaries of several chapters have been written or substantially extended for this document, and cross-chapter analyses that do not appear in any individual paper have been added throughout.

The correspondence is as follows.

- Chapter 3 is based on *Beyond Node Condensation: Learning Tiny Graphs via Joint Graph Condensation*, published in the Companion Proceedings of the ACM Web Conference 2025 (WWW 2025).
- Chapter 4 is based on *Debias Once for All: A Data-Centric Strategy for Fair Machine Learning*, published in the Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM 2026).
- Chapter 5 is based on *Fairness via Independence: A General Regularization Framework for Machine Learning*, published in the Proceedings of the International Conference on Learning Representations (ICLR 2026).
- Chapter 6 is based on *Enabling Group Fairness in Machine Unlearning via Distribution*

Correction, published in the Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM 2025).

- Chapter 7 is based on *LUNE: Efficient LLM Unlearning via LoRA Fine-Tuning with Negative Examples*, published in the Findings of the Association for Computational Linguistics (ACL Findings 2026).

The corresponding copyright declarations appear in the front matter of this dissertation, as required by the UCI Thesis and Dissertation Manual for reproductions of previously published work.

Chapter 2

Background and Preliminaries

This chapter consolidates the notation and technical background used throughout the dissertation. It is arranged so that the objects shared by several chapters are defined once here, in the most general form the dissertation needs, and each main chapter then recalls the specialization it requires. Readers following the argument in order may read this chapter once and refer back to it; readers approaching a single chapter will find that its preliminaries section states what it needs and points here for the rest.

The material is organized by control point. Section 2.1 through Section 2.4 concern the data and the training dynamics it induces; Section 2.5 and Section 2.6 concern the objective; Section 2.7 and Section 2.8 concern the parameters.

2.1 Notation and Conventions

Matrices are written in bold uppercase ($\mathbf{X}$, $\mathbf{A}$), vectors in bold lowercase ($\mathbf{x}_i$, $\boldsymbol{\theta}$), sets and datasets in calligraphic type ($\mathcal{T}$, $\mathcal{S}$), and scalars in ordinary italics. Two conventions prevent collisions that would otherwise arise from combining several literatures in one document.

First, the italic D always denotes the *feature dimension*. A distance or divergence between distributions is written in upright type as $\mathrm{D}(\cdot\,;\cdot)$, with a subscript naming the particular choice: D_{CS}, D_{KL}, $\mathrm{D}_{\mathrm{MMD}}$, $\mathrm{D}_{\mathrm{HSIC}}$.

Second, the symbol Δ is reserved for fairness gaps, Δ_{DP} and Δ_{EO}. A per-batch parameter update is written $\boldsymbol{\delta}$ rather than $\Delta\boldsymbol{\theta}$, since the two appear together throughout Chapter 6. Where a weight update must be named as such, in the low-rank adaptation of Section 2.8, it is written ΔW on a plain (non-bold, unsubscripted) W, which does not occur alongside a fairness gap.

Table 2.1 collects the recurring symbols.

Table 2.1: Recurring notation. The rightmost column gives the chapters in which each symbol is used.

Symbol	Meaning	Chapters
$\mathcal{T}$	Training dataset	all
N, D, C	Number of samples, feature dimension, number of classes	all
$\mathbf{X}$, $\mathbf{Y}$, $\mathbf{S}$	Feature, label, and sensitive-attribute matrices	4–7
K	Number of sensitive groups	4–6
$f_{\boldsymbol{\theta}}$	Model with parameters $\boldsymbol{\theta}$	all
$\mathrm{GNN}_{\boldsymbol{\theta}}$	Graph neural network with parameters $\boldsymbol{\theta}$	3
$\mathcal{L}$, ℓ	Aggregate and per-sample loss	all
η, T, $Q_{\boldsymbol{\theta}^0}$	Learning rate, epochs, initialization distribution	all
$\mathbf{A}$, $\mathcal{N}_i$, $\lvert\mathcal{E}\rvert$	Adjacency matrix, neighbourhood, edge count	3
$\mathcal{S}$, $\mathcal{T}'$	Condensed and synthetic datasets	3, 4
n, d	Condensed sample count and feature dimension	3
r_n, r_d	Node and feature condensation ratios	3
$\mathcal{L}_M$	Gradient matching loss	3, 4
$\mathcal{L}_F$	Fairness regularizer	4–6
Δ_{DP}, Δ_{EO}	Demographic parity and equal opportunity gaps	4–6
D_{CS}, κ	Cauchy–Schwarz divergence, kernel	5
α, λ, β	Fairness weight, trade-off coefficient, ℓ_2 weight	4–6
M, U	Learning and unlearning mechanisms	6, 7
$\mathcal{T}_u$, $\mathcal{T}_r$	Deletion request and retained data	6, 7
$\boldsymbol{\delta}_{p,b}$	Parameter update from batch b of epoch p	6
A, B, ϕ, r	Low-rank adapter factors, adapter parameters, rank	7
P_{full}, P_{LoRA}	Total and adapter parameter counts	7

2.2 Supervised Learning and Training Dynamics

A supervised dataset is a collection $\mathcal{T} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ with $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \{0, \dots, C-1\}$, written in matrix form as $\mathcal{T} = \{\mathbf{X}, \mathbf{Y}\}$ with $\mathbf{X} \in \mathbb{R}^{N \times D}$ and $\mathbf{Y} \in \{0, \dots, C-1\}^N$. A model $f_{\boldsymbol{\theta}}$ maps inputs to class scores, and ℓ denotes the per-sample loss, taken throughout to be cross-entropy. Parameters are obtained by empirical risk minimization,

$$\boldsymbol{\theta}^*_{\mathcal{T}} = \arg\min_{\boldsymbol{\theta}} \; \mathcal{L}\big(f_{\boldsymbol{\theta}}(\mathbf{X}), \mathbf{Y}\big), \qquad \mathcal{L}\big(f_{\boldsymbol{\theta}}(\mathbf{X}), \mathbf{Y}\big) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i\big), \tag{2.1}$$

carried out in practice by gradient descent from an initialization $\boldsymbol{\theta}^0$ drawn from a distribution $Q_{\boldsymbol{\theta}^0}$:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta \, \nabla_{\boldsymbol{\theta}^t} \mathcal{L}\big(f_{\boldsymbol{\theta}^t}(\mathbf{X}), \mathbf{Y}\big). \tag{2.2}$$

Definition 2.1 (Training trajectory)**.** *The sequence* $\{\boldsymbol{\theta}^t\}_{t=0}^{T-1}$ *produced by Equation 2.2 from an initialization* $\boldsymbol{\theta}^0$ *is the* training trajectory *induced by* $\mathcal{T}$*.*

Definition 2.1 names the object this dissertation intervenes upon. Chapter 3 and Chapter 4 construct datasets that reproduce a given trajectory; Chapter 5 changes which trajectory is desirable; Chapter 6 edits a trajectory's accumulated result after the fact; and Chapter 7 confines such an edit to a subspace. Two features of Equation 2.2 are used repeatedly. Accumulating the updates gives

$$\boldsymbol{\theta}_M = \boldsymbol{\theta}_{\text{init}} + \sum_{p=1}^{P} \sum_{b=1}^{B} \boldsymbol{\delta}_{p,b}, \tag{2.3}$$

where $\boldsymbol{\delta}_{p,b}$ is the update contributed by batch b of epoch p; this decomposition is what Chapter 6 edits. And the gradient $\nabla_{\boldsymbol{\theta}^t} \mathcal{L}$ itself is what Chapter 3 and Chapter 4 match and what Chapter 7 projects.

2.3 Graph Neural Networks

Chapter 3 works with graph-structured data, which adds relational structure to Equation 2.1.

Definition 2.2 (Attributed graph)**.** *An attributed graph with N nodes over C classes is a triple $\mathcal{T} = \{\mathbf{A}, \mathbf{X}, \mathbf{Y}\}$, where $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, $\mathbf{X} \in \mathbb{R}^{N \times D}$ the nodal feature matrix, and $\mathbf{Y} \in \{0, \dots, C-1\}^N$ the node labels. We write $\mathcal{N}_i = \{v_j : \mathbf{A}_{ij} \neq 0\}$ for the neighbourhood of node v_i and $|\mathcal{E}|$ for the number of edges.*

Graph neural networks compute node representations by iteratively aggregating over neighbourhoods. Writing $\mathbf{h}_i^l$ for the representation of v_i at layer l with $\mathbf{h}_i^0 = \mathbf{x}_i$, a message-passing layer takes the form

$$\mathbf{h}_i^{l+1} = \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{v_i\}} \omega_{ij}^l \, \mathbf{W}^l \mathbf{h}_j^l \right), \tag{2.4}$$

with $\mathbf{W}^l$ a learnable weight matrix, σ a pointwise nonlinearity, and the coefficients ω_{ij}^l determined by the architecture. Three instances are used in this dissertation. The graph convolutional network [77] fixes ω_{ij}^l from the normalized adjacency $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-1/2}$; the simplified graph convolution [131] removes the intermediate nonlinearities, collapsing propagation into $\tilde{\mathbf{A}}^L \mathbf{X}\boldsymbol{\theta}$; and the graph attention network [124, 125] learns $\omega_{ij}^l = \alpha_{ij}^l$ from the features of a node and its neighbours. Section 3.2 states each propagation rule in the form used by the method of that chapter, since the feature condensation operator developed there is itself a graph attention network.

Two evaluation settings are distinguished. In the *transductive* setting the whole graph, including the features and connectivity of test nodes, is available during training and only test labels are withheld; in the *inductive* setting the model sees only the training subgraph. The distinction matters for condensation, since only in the first case may the full graph be condensed.

2.4 Dataset Condensation and Gradient Matching

Dataset condensation replaces a large training set by a small synthetic one on which training yields comparable models [156, 157, 69]. Writing $\mathcal{S}$ for the synthetic set, the natural formulation is bi-level: the outer problem chooses $\mathcal{S}$ so that a model trained on it explains the original data, and the inner problem performs that training. Differentiating the outer objective requires backpropagating through the entire unrolled inner optimization, whose memory cost grows linearly in the number of inner steps. This is precisely the cost that condensation was meant to avoid.

Gradient matching [157, 128] replaces this with a single-level surrogate. Rather than requiring the same converged parameters, one requires the same gradients at every point along the training trajectory of Definition 2.1, in expectation over initializations:

$$\min_{\mathcal{S}} \; \mathrm{E}_{\boldsymbol{\theta}^0 \sim Q_{\boldsymbol{\theta}^0}} \left[\sum_{t=0}^{T-1} \mathcal{L}_M \big(\nabla_{\boldsymbol{\theta}^t} \mathcal{L}^{\mathcal{T}}, \; \nabla_{\boldsymbol{\theta}^t} \mathcal{L}^{\mathcal{S}} \big) \right]. \tag{2.5}$$

Because each step is constrained individually, both gradients can be evaluated at the same current parameters and compared immediately, with no unrolling.

The matching loss $\mathcal{L}_M$ is computed per class and summed, comparing corresponding gradient columns by cosine distance:

$$\mathcal{L}_M\big(\mathbf{G}, \mathbf{G}'\big) = \sum_{c=0}^{C-1} \sum_{l} \sum_{b} \left(1 - \frac{\mathbf{G}^c_{l,b} \cdot \mathbf{G}'^c_{l,b}}{\|\mathbf{G}^c_{l,b}\| \, \|\mathbf{G}'^c_{l,b}\|} \right), \tag{2.6}$$

where $\mathbf{G}$ and $\mathbf{G}'$ are the gradients induced by the original and synthetic data, c indexes classes, l layers, and b columns. Cosine distance is used because what must agree is the *direction* of the update: gradient magnitudes computed from sample sets of very different sizes are not comparable, and normalizing removes that nuisance dependence.

Equation 2.6 is used unchanged in Chapter 3, where it appears as Equation 3.17 with layers

indexed by p and columns by h, and in Chapter 4, where it appears as Equation 4.10. The observation that Equation 2.5 encodes into $\mathcal{S}$ whatever property the target trajectory possesses, and not merely accuracy, is what connects those two chapters.

2.5 Group Fairness

From Chapter 4 onward, datasets carry a third component:

$$\mathcal{T} = \{\mathbf{X}, \mathbf{Y}, \mathbf{S}\}, \qquad \mathbf{S} \in \{0, \ldots, K-1\}^N, \tag{2.7}$$

where $\mathbf{S}$ records the *sensitive attribute* of each sample over K demographic groups, such as gender, race, or age bracket. Capitalized $\hat{Y}$, Y, and S denote the corresponding random variables.

2.5.1 Criteria

Fairness criteria divide into *individual* notions, requiring that similar individuals receive similar predictions [39], and *group* notions, requiring statistical parity across demographic groups. This dissertation concerns the latter, and two criteria recur.

Definition 2.3 (Demographic parity)**.** *A predictor satisfies* demographic parity *if its prediction is statistically independent of the sensitive attribute:*

$$P(\hat{Y} = c \mid S = k) = P(\hat{Y} = c) \qquad \textit{for all } c, k. \tag{2.8}$$

Definition 2.4 (Equalized odds and equal opportunity)**.** *A predictor satisfies* equalized odds *if its prediction is conditionally independent of the sensitive attribute given the true*

label:

$$P(\hat{Y} = c \mid S = k, Y = y) = P(\hat{Y} = c \mid Y = y) \qquad \textit{for all } c, k, y. \tag{2.9}$$

Restricting to the advantaged outcome $y = 1$ *gives* equal opportunity *[56], which equalizes true positive rates without constraining false positive rates.*

Demographic parity constrains the marginal distribution of decisions and makes no reference to the label, which suits settings where the labels themselves are suspected of encoding historical bias and is inappropriate where base rates legitimately differ. Equalized odds conditions on the label and so tolerates genuine base-rate differences, at the cost of inheriting whatever bias the labels contain. The two are in general incompatible: except in degenerate cases a nontrivial predictor cannot satisfy both [56, 31]. Reporting both, as this dissertation does throughout, is therefore not redundant, since a method that improves one at the expense of the other has relabelled fairness rather than improved it.

2.5.2 Empirical Gaps

Both criteria are exact statements that no finite-sample predictor satisfies, so they are measured by the extent of their violation. With $\widehat{P}$ an empirical probability on held-out data,

$$\Delta_{DP} = \frac{1}{K} \sum_{k=0}^{K-1} \max_{c} \left| \widehat{P}(\hat{Y} = c) - \widehat{P}(\hat{Y} = c \mid S = k) \right|, \tag{2.10}$$

$$\Delta_{EO} = \frac{1}{K} \sum_{k=0}^{K-1} \max_{c,\, y} \left| \widehat{P}(\hat{Y} = c \mid Y = y) - \widehat{P}(\hat{Y} = c \mid S = k, Y = y) \right|, \tag{2.11}$$

both non-negative, zero exactly when the criterion holds empirically, and reported in percentage points.

In the binary setting $C = K = 2$, which is the regime of Chapter 5 and Chapter 6, these reduce to the familiar two-group differences

$$\Delta_{DP} = \big|P(\hat{Y} \mid S{=}0) - P(\hat{Y} \mid S{=}1)\big|, \qquad \Delta_{EO} = \big|P(\hat{Y} \mid Y{=}1, S{=}0) - P(\hat{Y} \mid Y{=}1, S{=}1)\big|. \tag{2.12}$$

The two conventions are not numerically identical. Writing $\pi_k = P(S = k)$, $p = P(\hat{Y}{=}1 \mid S{=}0)$ and $q = P(\hat{Y}{=}1 \mid S{=}1)$, the group-0 term of Equation 2.10 evaluates to $\pi_1|p - q|$ and the group-1 term to $\pi_0|p - q|$, so their average is $\frac{1}{2}|p - q|$. The forms therefore differ by exactly a factor of two and induce the same ordering over models. Chapter 4 reports Equation 2.10; Chapter 5 and Chapter 6 report Equation 2.12, which is standard in the fair machine learning literature.

2.5.3 Metrics Against Surrogates

Equation 2.10 and Equation 2.11 are defined on hard predictions $\hat{Y} = \arg\max_c f_{\boldsymbol{\theta}}(\mathbf{x})_c$, which are piecewise constant in $\boldsymbol{\theta}$ and so cannot be optimized. Every method in this dissertation therefore trains against a *differentiable surrogate*, obtained by replacing the indicator of a predicted class with the model's soft score for it:

$$\widehat{P}(\hat{Y} = c) \;\longrightarrow\; \frac{1}{N}\sum_{i=1}^{N} f_{\boldsymbol{\theta}}(\mathbf{x}_i)_c, \qquad \widehat{P}(\hat{Y} = c \mid S = k) \;\longrightarrow\; \frac{1}{N_k}\sum_{i:\, s_i = k} f_{\boldsymbol{\theta}}(\mathbf{x}_i)_c, \tag{2.13}$$

with $N_k = |\{i : s_i = k\}|$. The result is differentiable in $\boldsymbol{\theta}$ and agrees with the hard gap in the limit of confident predictions.

The distinction is used consistently: a fairness *metric* means the hard-prediction gap evaluated at test time, and a fairness *regularizer* means a differentiable surrogate used during training. They are not the same object, and the quality of a regularizer is not determined

by the metric it is named after, a point that Chapter 5 develops into the central question of that chapter.

2.5.4 Where Fairness Is Enforced

Interventions are conventionally classified by where they act. *Pre-processing* methods modify the training data before learning [72, 17]; *in-processing* methods add a penalty or constraint to the training objective [73, 143, 2]; *post-processing* methods adjust the outputs of a trained model [56, 127]. In-processing methods share the template

$$\min_{\boldsymbol{\theta}} \quad \mathcal{L}_{\text{utility}} + \lambda \, \mathcal{L}_{\text{fairness}}, \tag{2.14}$$

with λ governing the trade-off. Chapter 4 sits in the first family but obtains its dataset by differentiable optimization against Equation 2.14 rather than by a fixed heuristic; Chapter 5 studies the second term of Equation 2.14 directly.

2.6 Distance and Dependence Measures

The fairness regularizers compared in Chapter 5 all reduce to a choice of measure D between distributions the model controls, typically the group-conditional prediction distributions $\mathbb{P} = P(\hat{Y} \mid S{=}0)$ and $\mathbb{Q} = P(\hat{Y} \mid S{=}1)$. Four choices recur, and are collected here.

Mean disparity. The most common construction is the absolute difference of means, $\big|\mathrm{E}(\mathbb{P}) - \mathrm{E}(\mathbb{Q})\big|$, which is the differentiable relaxation of Equation 2.12 [27]. It is blind to every feature of the distributions beyond the first moment, and a vanishing mean gap is necessary but not sufficient for either criterion of Section 2.5.

Maximum mean discrepancy. Given samples from p and q and a positive-definite kernel κ, the biased empirical MMD [49] is

$$\widetilde{\mathrm{D}}^2_{\mathrm{MMD}}(p;q) = \frac{1}{N_1^2}\sum_{i,j}^{N_1}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^p) + \frac{1}{N_2^2}\sum_{i,j}^{N_2}\kappa(\mathbf{x}_i^q,\mathbf{x}_j^q) - \frac{2}{N_1 N_2}\sum_{i}^{N_1}\sum_{j}^{N_2}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^q), \tag{2.15}$$

the squared Euclidean distance between the kernel mean embeddings of the two samples.

Kullback–Leibler divergence and HSIC. The KL divergence

$$\mathrm{D}_{\mathrm{KL}}(p\|q) = \int p(\mathbf{x}) \log\big(p(\mathbf{x})/q(\mathbf{x})\big)\, d\mathbf{x}$$

underlies mutual-information penalties such as the prejudice remover [73]; it is asymmetric, requires a density ratio, and diverges where supports fail to overlap. The Hilbert–Schmidt independence criterion [50, 51] measures dependence directly, and is itself an MMD taken between a joint distribution and the product of its marginals.

Cauchy–Schwarz divergence. The Cauchy–Schwarz inequality for square-integrable functions, $\big(\int pq\big)^2 \le \int p^2 \int q^2$, holds with equality exactly when p and q are linearly dependent. The gap therefore measures discrepancy, and its logarithm defines the CS divergence [107, 141]:

$$\mathrm{D}_{\mathrm{CS}}(p;q) = -\log\left(\frac{\big(\int p(\mathbf{x})q(\mathbf{x})\,d\mathbf{x}\big)^2}{\int p(\mathbf{x})^2 d\mathbf{x} \int q(\mathbf{x})^2 d\mathbf{x}}\right). \tag{2.16}$$

It is symmetric, satisfies $0 \le \mathrm{D}_{\mathrm{CS}} < \infty$, and is minimized exactly when $p = q$ almost everywhere. Symmetry distinguishes it from KL, which must be assigned a direction. The distinction matters for fairness, where there is no principled reason to treat one demographic group as the reference.

Substituting Parzen-window estimates [103] into Equation 2.16 yields a purely kernel-based estimator [65], stated as Theorem 5.1 and used as the regularizer of Chapter 5. Comparing it with Equation 2.15 shows that the two are built from the same three kernel sums, with the CS divergence applying a logarithm to each before combining them, which turns a Euclidean comparison of embeddings into a cosine-type one. Section 5.4.4 develops the consequences.

2.7 Machine Unlearning

Data protection regimes including the GDPR [29] and the CCPA [36] grant a right to erasure [109]. Applied to machine learning, compliance requires removing not only a record but its influence on models trained on it [18, 24].

Definition 2.5 (Unlearning mechanism)**.** *Given a learning algorithm $M : \mathcal{D} \to \mathcal{H}$ mapping datasets to models, a trained model $M(\mathcal{T})$, and a subset $\mathcal{T}_u \subseteq \mathcal{T}$ whose deletion has been requested, an* unlearning mechanism *is a mapping*

$$U : \mathcal{H} \times \mathcal{D} \times \mathcal{D} \to \mathcal{H}, \qquad \big(M(\mathcal{T}), \mathcal{T}, \mathcal{T}_u\big) \longmapsto U\big(M(\mathcal{T}), \mathcal{T}, \mathcal{T}_u\big),$$

whose output should resemble $M(\mathcal{T}_r)$, the model obtained by retraining on the retained data $\mathcal{T}_r = \mathcal{T} \setminus \mathcal{T}_u$.

Because M and U are randomized, the appropriate comparison is between distributions over the hypothesis space rather than between individual models: the ideal is that $\Pr\big(U(M(\mathcal{T}), \mathcal{T}, \mathcal{T}_u)\big)$ be statistically indistinguishable from $\Pr\big(M(\mathcal{T}_r)\big)$. Methods divide into *exact* unlearning, which achieves this by construction at the cost of partial retraining [14], and *approximate* unlearning, which achieves it to a tolerance far more cheaply, for instance by rolling back the stored updates of Equation 2.3 [48]. Surveys cover the broader landscape [101, 150, 135].

It is worth recording a gap that this dissertation inherits and does not close. Definition 2.5

states an ideal over *weights*, but the quantities actually measured are all functions of model *outputs*: membership inference accuracy, accuracy on deleted and retained data, and the behavioural metrics of Chapter 7. They are proxies, and Section 8.4.1 returns to the distance between them.

2.8 Parameter-Efficient Fine-Tuning and Low-Rank Adaptation

At foundation-model scale the assumptions underlying Equation 2.3 fail: the per-batch update record does not exist for a model pretrained by someone else, and revising every parameter is neither affordable nor safe. Parameter-efficient fine-tuning adapts a model by updating a small set of parameters while the majority stay frozen [58, 144], and low-rank adaptation [59] is the standard instance.

Definition 2.6 (Low-rank adaptation)**.** *Let $W_0 \in \mathbb{R}^{d\times k}$ be a frozen pretrained weight matrix. Low-rank adaptation represents its update as a product of trainable matrices of rank $r \ll \min(d,k)$,*

$$\Delta W = AB^\top, \qquad A \in \mathbb{R}^{d\times r},\ B \in \mathbb{R}^{k\times r}, \qquad W = W_0 + \Delta W, \tag{2.17}$$

introducing $\mathcal{O}\big(r(d+k)\big)$ trainable parameters per adapted matrix in place of dk.

We write $\phi = (A, B)$ for the collection of adapter parameters across a model and $f_{\boldsymbol{\theta},\phi}$ for the adapted model, so that $\boldsymbol{\theta}$ is frozen and ϕ alone is optimized. Extensions improve memory [33], rank allocation [152], and decomposition [90]; adapter [106] and prompt-tuning [85, 86] families offer alternatives. That such methods work is explained by the observation that downstream updates tend to lie in a low intrinsic dimension [3]. Chapter 7 argues that for *unlearning* the low-rank restriction is not merely an acceptable approximation but an

advantage, because it bounds the region of parameter space the edit can reach.

2.9 Datasets and Evaluation Protocol

Table 2.2 lists the benchmarks used across the dissertation. Chapter-specific statistics, splits, and preprocessing appear in the experimental setup of each chapter.

Table 2.2: Benchmarks used in the dissertation, grouped by the setting they serve.

Setting	Benchmarks	Chapters
Graph learning	`Cora`, `Citeseer`, `Flickr`, `Reddit`, `Arxiv`	3
Tabular fairness	`Adult`, `COMPAS`, `ACS-I`, `ACS-T`	4–6
Image fairness	`CelebA-A`	4–6
LLM unlearning	EDU-RELAT, RWKU, KnowUnDo, TOFU	7

Evaluation follows the conventions of each literature. Utility is reported as accuracy and, for the fairness chapters, area under the ROC curve. Fairness is reported as Δ_{DP} and Δ_{EO}, in the form indicated in Section 2.5. Unlearning efficacy is reported through membership inference attack accuracy in Chapter 6 and Chapter 7, supplemented in the latter by the behavioural metrics defined in Section 7.4. Throughout, results are averaged over multiple runs with different random seeds and reported with standard deviations or standard errors as indicated.

Chapter 3

Efficient Learning on Large-Scale Graphs via Joint Feature and Node Condensation

3.1 Overview

The preceding chapters framed this dissertation around a single control surface: the training dynamics of a learning system, that is, the gradients a model is exposed to and the parameter updates it retains. This chapter takes the first step along that programme by asking the most basic question one can ask about training dynamics: *how little data is required to induce them?* If a small, synthetic dataset can be constructed so that a model trained on it follows essentially the same optimization trajectory as a model trained on the full dataset, then the dataset itself becomes a design variable rather than a fixed input. Everything that follows in this dissertation rests on that observation. Here it is developed in the setting where the computational pressure is most acute, namely learning over large-scale graphs.

Graphs are a standard representation for structured and relational data, and graph neural networks (GNNs) have become the default model class for learning over them, with appli-

cations spanning social network analysis, recommendation, molecular property prediction, transportation forecasting, and epidemiological modelling [159, 54, 140, 38, 9]. Their success, however, comes at a substantial computational cost [133, 154, 71]. Training a GNN on a real-world graph requires repeated sparse propagation over a large number of nodes and edges, and the resulting time and memory demands are magnified whenever a model must be retrained many times, as in neural architecture search and hyperparameter optimization [155, 55]. A natural response is to replace the large graph with a small one that supports comparable training.

This idea has been formalized as *graph condensation*. Building on dataset condensation for image data [156], a line of work has shown that a large graph can be replaced by a graph with far fewer nodes while retaining most of the downstream accuracy [69, 68, 46, 57]. The central observation motivating this chapter is that this line of work condenses along one axis only, and that this is not sufficient. Real graphs are high-dimensional as well as large: the number of nodal features exceeds the number of nodes by a factor of roughly 41 on `Cora`, 123 on `Citeseer`, 11.2 on `Flickr`, and 7.8 on `Reddit`. After node condensation on `Citeseer`, the training set contains 30 nodes, each still carrying 3,703 features. The condensed object is therefore severely underdetermined relative to its own dimensionality, and the first-layer weight matrix of any GNN trained on it remains as large as before. Reducing the node count alone does not resolve the cost of training; it relocates it.

This chapter therefore studies *joint condensation*: the simultaneous reduction of both the number of nodes and the dimensionality of their features. Figure 3.1 illustrates the objective. Two difficulties distinguish this problem from its node-only predecessor. The first concerns what must be preserved. Node features and graph structure are not independent quantities; homophily and attribute-structure correlation are well documented in real networks [113, 63], so any feature condensation operator that ignores the adjacency structure discards exactly the signal that graph learning exploits. This rules out the obvious two-stage recipe of apply-

ing an off-the-shelf dimensionality reduction method to the features and then running a node condensation algorithm on the result, which is both structure-agnostic and vulnerable to error propagation from the first stage into the second. The second difficulty is computational. Formulated directly, joint condensation is a nested optimization problem in which the inner problem trains a GNN to convergence on a graph that the outer problem is simultaneously trying to learn; unrolling this recursion is precisely the cost that condensation was meant to avoid.

The framework developed in this chapter, TinyGraph, addresses both difficulties with a single mechanism. Feature condensation is performed by a trainable, attention-based operator that consumes the graph structure while reducing dimensionality, so structure awareness is built into the projection rather than bolted on afterwards. The nested optimization is then replaced by a matching objective on the training dynamics themselves: rather than requiring the condensed graph to yield the same converged parameters, we require it to induce the same gradients at every point along the training trajectory. This substitution removes the inner loop entirely and reduces joint condensation to a single-level minimization problem.

Contributions of this chapter. The chapter makes the following contributions.

- It identifies and characterizes a limitation that persists in existing graph condensation frameworks: when nodal features are high-dimensional, condensing the node set alone leaves both the storage footprint and the dominant term of the training cost largely intact.

- It formulates joint feature and node condensation as a well-posed learning problem, and shows that the natural bi-level formulation can be relaxed to a single-level gradient-matching objective defined along the training trajectory.

- It proposes TinyGraph, a unified framework in which a trainable, structure-aware fea-

ture condensation operator and a condensed graph are learned together under that objective, rather than in two disconnected stages.

- It reports an extensive empirical study over five benchmarks spanning transductive and inductive settings, showing that a GNN trained on the condensed graph retains up to 98.5%, 97.5%, 98.7%, 94.9%, and 86.5% of the original test accuracy on `Cora`, `Citeseer`, `Flickr`, `Reddit`, and `Arxiv` respectively, while reducing the node count by more than 97.4% and the feature dimension by 90%, 90%, 80%, 70%, and 70%.

Section 3.2 establishes the notation and the background on graph neural networks used throughout the chapter and, where indicated, in the remainder of the dissertation. Section 3.3 formulates the joint condensation problem. Section 3.4 develops the TinyGraph framework. Section 3.5 reports the empirical study, and Section 3.7 closes the chapter and connects it to the one that follows.

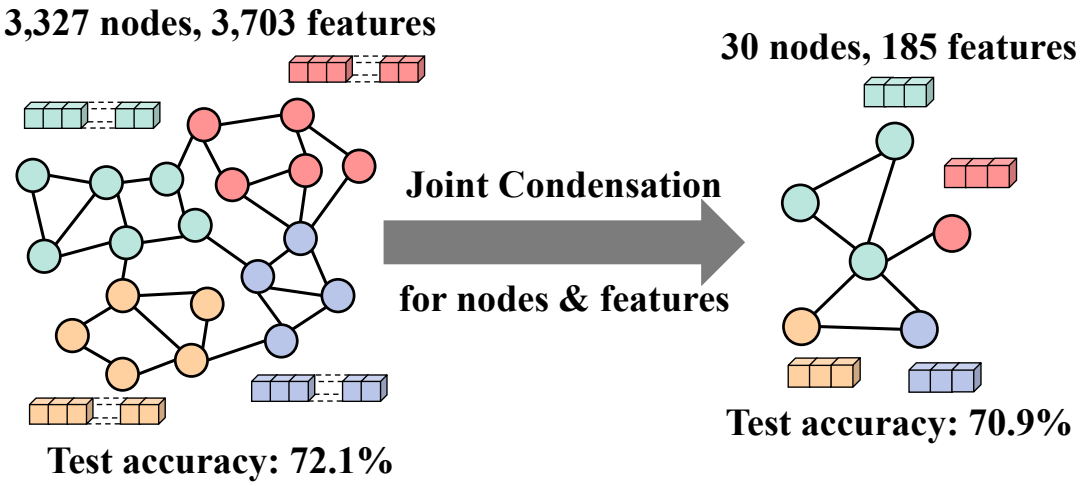

Figure 3.1: The goal of joint condensation. A large graph is replaced by a condensed graph with both a smaller node set and a lower feature dimension. On `Citeseer`, the framework developed in this chapter attains a reduction of 90.0% in features, 98.2% in nodes, and 99.7% in storage.

3.2 Preliminaries

This section fixes the notation used throughout the chapter and reviews the elements of graph representation learning that the method builds on. The treatment is deliberately explicit about the propagation rules, because the feature condensation operator introduced in Section 3.4.1 is itself a graph neural network and reuses these equations verbatim.

3.2.1 Attributed Graphs and Node Classification

Definition 3.1 (Attributed graph)**.** *An attributed graph with N nodes over C classes is a triple*

$$\mathcal{T} = \{\mathbf{A}, \mathbf{X}, \mathbf{Y}\},$$

where $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix, $\mathbf{X} \in \mathbb{R}^{N \times D}$ is the nodal feature matrix whose i-th row $\mathbf{x}_i \in \mathbb{R}^D$ collects the D features of node v_i, and $\mathbf{Y} \in \{0, \dots, C-1\}^N$ collects the node labels.

We write $\mathcal{N}_i = \{v_j : \mathbf{A}_{ij} \neq 0\}$ for the neighbourhood of node v_i, and $|\mathcal{E}|$ for the number of edges, that is, the number of nonzero entries of $\mathbf{A}$. In the node classification task, a model is trained on the labels of a subset of nodes and evaluated on the labels of a held-out subset. Two settings are distinguished. In the *transductive* setting the entire graph, including the features and connectivity of test nodes, is available during training, and only the test labels are withheld. In the *inductive* setting the model observes only the training subgraph during training and is applied to nodes and edges it has never seen. Both settings are considered in this chapter, since condensation is meaningfully different in each: in the transductive case the full graph may be condensed, whereas in the inductive case only the training graph is available to condense.

3.2.2 Graph Neural Networks

Graph neural networks compute node representations by iteratively aggregating information over neighbourhoods. Writing $\mathbf{h}_i^l$ for the representation of node v_i at layer l, with $\mathbf{h}_i^0 = \mathbf{x}_i$, a message-passing layer takes the generic form

$$\mathbf{h}_i^{l+1} \;=\; \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{v_i\}} \omega_{ij}^{l}\, \mathbf{W}^{l} \mathbf{h}_j^{l}\right), \tag{3.1}$$

where $\mathbf{W}^l \in \mathbb{R}^{F_l \times F_{l-1}}$ is a learnable weight matrix, $\sigma(\cdot)$ is a pointwise nonlinearity such as the rectified linear unit, and the coefficients ω_{ij}^l specify how neighbouring representations are weighted. Particular architectures correspond to particular choices of ω_{ij}^l.

Graph convolutional networks. The graph convolutional network (GCN) [77] fixes the coefficients from the graph structure alone. With $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-1/2}$ denoting the symmetrically normalized adjacency matrix with self-loops and $\mathbf{D}$ the corresponding degree matrix, a GCN layer is

$$\mathbf{H}^{l+1} = \sigma\Big(\tilde{\mathbf{A}}\mathbf{H}^{l}(\mathbf{W}^{l})^{\top}\Big), \qquad \mathbf{H}^{0} = \mathbf{X}. \tag{3.2}$$

Simplified graph convolution. Removing the intermediate nonlinearities from Equation 3.2 collapses the propagation into a single fixed operator, yielding the simplified graph convolution (SGC) [131],

$$\mathrm{SGC}_{\boldsymbol{\theta}}(\mathbf{A}, \mathbf{X}) = \mathrm{softmax}\Big(\tilde{\mathbf{A}}^{L}\mathbf{X}\boldsymbol{\theta}\Big). \tag{3.3}$$

SGC is used as the default backbone in this chapter because it isolates the effect of the condensed data from the expressive capacity of the backbone, and because its parameterization makes the dependence of the gradient on the feature dimension explicit.

Graph attention networks. Rather than fixing ω_{ij}^l from the degree structure, the graph attention network (GAT) [124, 125] learns it. Setting $\omega_{ij}^l = \alpha_{ij}^l$ in Equation 3.1 gives

$$\mathbf{h}_i^{l+1} = \sigma\left(\sum_{j \in \mathcal{N}_i \cup \{v_i\}} \alpha_{ij}^l \, \mathbf{W}\mathbf{h}_j^l\right), \tag{3.4}$$

where the attention coefficient α_{ij}^l, which quantifies the importance of neighbour v_j to node v_i at layer l, is computed as

$$\alpha_{ij}^l = \frac{\exp\left(\sigma\left(\mathbf{a}^\top \left[\mathbf{W}\mathbf{h}_i^l \oplus \mathbf{W}\mathbf{h}_j^l\right]\right)\right)}{\sum_{k \in \mathcal{N}_i \cup \{v_i\}} \exp\left(\sigma\left(\mathbf{a}^\top \left[\mathbf{W}\mathbf{h}_i^l \oplus \mathbf{W}\mathbf{h}_k^l\right]\right)\right)}. \tag{3.5}$$

Here $\oplus$ denotes concatenation and $\mathbf{a}$ is a trainable attention vector. Because α_{ij}^l depends jointly on the connectivity through $\mathcal{N}_i$ and on the features through $\mathbf{h}^l$, a stack of layers of this form is a natural candidate for a structure-aware projection, a point taken up in Section 3.4.1.

Throughout, $\mathrm{GNN}_{\boldsymbol{\theta}}(\cdot, \cdot)$ denotes a graph neural network parameterized by $\boldsymbol{\theta}$ that maps an adjacency matrix and a feature matrix to class scores, and $\mathcal{L}$ denotes the node classification loss, taken to be the cross-entropy between the predicted scores and the ground-truth labels.

3.2.3 Training Trajectories

Given a graph $\mathcal{T}$, the parameters are obtained by empirical risk minimization,

$$\boldsymbol{\theta}_{\mathcal{T}}^* \;=\; \arg\min_{\boldsymbol{\theta}} \; \mathcal{L}(\mathrm{GNN}_{\boldsymbol{\theta}}(\mathbf{A}, \mathbf{X}), \mathbf{Y}), \tag{3.6}$$

which in practice is carried out by gradient descent from an initialization $\boldsymbol{\theta}^0$ drawn from a distribution $Q_{\boldsymbol{\theta}^0}$,

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta \, \nabla_{\boldsymbol{\theta}^t} \mathcal{L}(\mathrm{GNN}_{\boldsymbol{\theta}^t}(\mathbf{A}, \mathbf{X}), \mathbf{Y}) \,, \tag{3.7}$$

with learning rate η. The sequence $\{\boldsymbol{\theta}^t\}_{t=0}^{T-1}$ is referred to as the *training trajectory* induced by $\mathcal{T}$ from $\boldsymbol{\theta}^0$. This object, rather than the converged parameter $\boldsymbol{\theta}^*_{\mathcal{T}}$ alone, is what the framework of this chapter attempts to reproduce from a much smaller graph, and it is the same object that later chapters of this dissertation intervene upon for reasons other than efficiency.

3.2.4 The Cost of Training on Large Graphs

The motivation for joint condensation follows from a direct accounting of Equation 3.1. For a layer mapping F_{l-1} input channels to F_l output channels, the sparse propagation costs $O(|\mathcal{E}|F_{l-1})$ and the dense transformation costs $O(NF_{l-1}F_l)$. Summing over an L-layer network and noting that the input dimension of the first layer is the feature dimension, $F_0 = D$, the per-epoch training cost is

$$O\left(|\mathcal{E}|D + NDF_1 + \sum_{l=2}^{L} \left(|\mathcal{E}|F_{l-1} + NF_{l-1}F_l \right) \right), \tag{3.8}$$

while storing the feature matrix alone requires $O(ND)$ memory.

Two consequences are immediate. First, both N and D enter the leading terms of Equation 3.8 multiplicatively, so reducing one while leaving the other untouched bounds the achievable saving. Second, the first-layer weight matrix has DF_1 entries independently of how many nodes remain, so a condensed graph with $n \ll N$ nodes but D features is fitting a first layer whose parameter count may exceed its sample count by orders of magnitude. This is the

regime that node-only condensation leaves in place, and it is the regime this chapter targets.

Accordingly, we measure condensation along two ratios: the *node condensation ratio* $r_n = n/N$ and the *feature condensation ratio* $r_d = d/D$, where n and d denote the node count and feature dimension of the condensed graph.

3.3 Problem Formulation

3.3.1 Goal

We seek a condensed graph

$$\mathcal{S} = \{\hat{\mathbf{A}}, \hat{\mathbf{X}}, \hat{\mathbf{Y}}\}, \qquad \hat{\mathbf{A}} \in \mathbb{R}^{n \times n},\ \hat{\mathbf{X}} \in \mathbb{R}^{n \times d},\ \hat{\mathbf{Y}} \in \{0, \dots, C-1\}^n,$$

with $n \ll N$ and $d \ll D$, such that a GNN trained on $\mathcal{S}$ attains performance comparable to one trained on the original graph $\mathcal{T}$.

Reducing the feature dimension introduces a constraint absent from node-only condensation. A model trained on $\mathcal{S}$ accepts inputs of dimension d, whereas evaluation is performed on the original graph, whose nodes carry D features. The condensed feature space must therefore be accompanied by an explicit map from the original space into it, so that the two graphs remain commensurable. We accordingly introduce a trainable *feature condensation function*

$$f_{\mathbf{\Phi}}(\cdot) : \mathbb{R}^D \to \mathbb{R}^d, \tag{3.9}$$

parameterized by $\mathbf{\Phi}$, and define the *feature-condensed* counterpart of the original graph as

$$\tilde{\mathcal{T}} = (\mathbf{A},\ f_{\mathbf{\Phi}}(\mathbf{X}),\ \mathbf{Y}), \qquad \tilde{\mathbf{X}} := f_{\mathbf{\Phi}}(\mathbf{X}) \in \mathbb{R}^{N \times d}. \tag{3.10}$$

The graph $\tilde{\mathcal{T}}$ retains every node and edge of $\mathcal{T}$ but lives in the same d-dimensional feature space as $\mathcal{S}$. It is the object that mediates between the two: $f_{\mathbf{\Phi}}$ is what allows a condensed graph of reduced width to be compared against, and eventually deployed on, the full graph.

3.3.2 A Bi-Level Formulation and Its Cost

Following the standard treatment of synthetic data as a parameter of the model it induces [128, 118, 13, 117], the joint condensation problem can be written as

$$\begin{aligned} \mathcal{S}^* = &\arg\min_{\mathcal{S}} \mathcal{L}\big(\mathrm{GNN}_{\boldsymbol{\theta}^*_{\mathcal{S}}}(\mathbf{A}, f_{\mathbf{\Phi}^*}(\mathbf{X})), \mathbf{Y}\big)\,, \\ &\text{s.t. } \mathbf{\Phi}^* = \arg\min_{\mathbf{\Phi}} \mathcal{L}\big(\mathrm{GNN}_{\boldsymbol{\theta}^*_{\mathcal{S}}}(\mathbf{A}, f_{\mathbf{\Phi}}(\mathbf{X})), \mathbf{Y}\big)\,, \\ &\quad \boldsymbol{\theta}^*_{\mathcal{S}} = \arg\min_{\boldsymbol{\theta}_{\mathcal{S}}} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}_{\mathcal{S}}}(\hat{\mathbf{A}}, \hat{\mathbf{X}}), \hat{\mathbf{Y}}\Big)\,. \end{aligned} \tag{3.11}$$

The outer problem asks that the condensed graph, through the model it trains, explain the original labels; the innermost problem trains that model.

Equation 3.11 is well posed but impractical. Each outer step requires solving the innermost problem, and differentiating the outer objective with respect to $\mathcal{S}$ requires backpropagating through the entire unrolled sequence $\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^{t-1}$ of inner updates. The memory required to retain that computation graph grows linearly in the number of inner steps, and the arithmetic cost grows with it. For the large graphs that motivate condensation in the first place, this is prohibitive. Section 3.4.2 replaces the nested formulation with a single-level surrogate that avoids the unrolling entirely.

3.3.3 Why Sequential Condensation Is Insufficient

Before developing that surrogate, it is worth being precise about why the simplest alternative fails. A two-stage pipeline would first apply a dimensionality reduction method to $\mathbf{X}$,

obtaining features of width d, and then run an existing node condensation algorithm on the reduced graph. Classical reduction methods such as principal component analysis [1], independent component analysis [30], and linear discriminant analysis [7] have proven effective on Euclidean data, where samples are exchangeable and independent, and one might hope to reuse them here.

Two problems arise. First, these methods are *structure-agnostic*: they select directions in $\mathbb{R}^D$ using only the empirical distribution of the rows of $\mathbf{X}$, treating nodes as independent samples. This discards the adjacency structure at exactly the point where it is most informative, since attribute values and connectivity are correlated in real graphs [113, 63]. A direction of low feature variance may nonetheless be the direction along which connected nodes differ most sharply, and a structure-agnostic criterion has no way to see this. Second, the two stages are decoupled, so information destroyed in the first stage cannot be recovered in the second; any loss incurred by the projection propagates into the node condensation objective as an irreducible error floor. A third, more practical limitation is that classical reduction methods produce a fixed projection matrix, whereas the quality of the condensed graph depends on how the projection interacts with the downstream training dynamics, which a fixed matrix cannot adapt to.

These observations motivate the two design commitments of the next section: the feature condensation operator must consume the graph structure, and it must be learned jointly with the condensed graph under a single objective rather than fitted in advance.

3.4 The TinyGraph Framework

Figure 3.2 gives an overview of the framework. In each epoch, the original features are projected through a trainable structure-aware operator to form the feature-condensed graph $\tilde{\mathcal{T}}$; a condensed graph $\mathcal{S}$ is assembled from trainable condensed features and a generated

adjacency matrix; both graphs are used to compute gradients of the classification loss with respect to a shared set of GNN weights; and the discrepancy between those gradients drives the update of all condensation parameters.

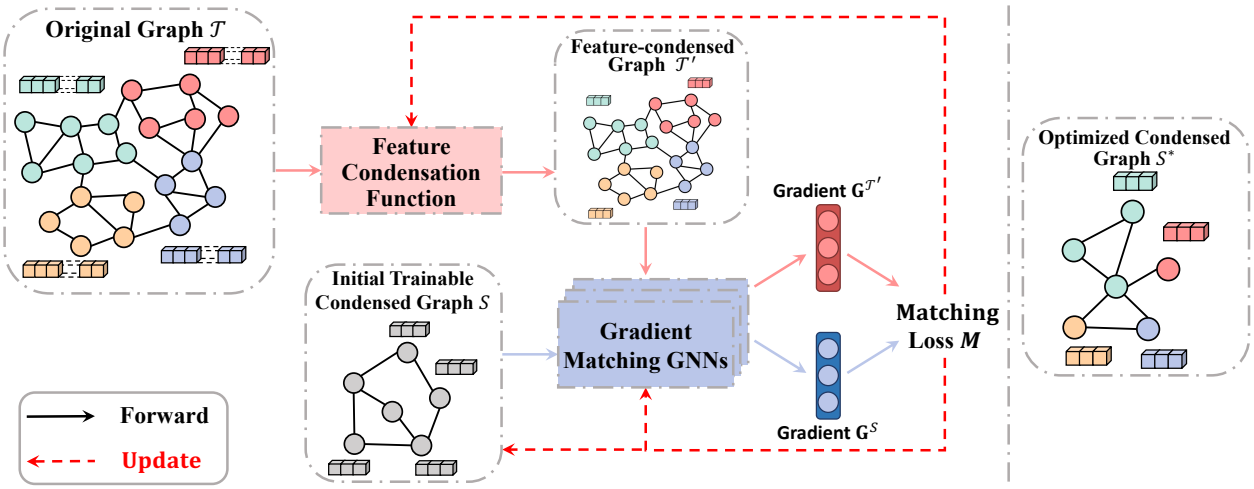

Figure 3.2: Overview of the TinyGraph framework for joint condensation via gradient matching. The objective is to learn a condensed graph on which a GNN can be trained to performance comparable with training on the original graph. Both the feature condensation function and the gradient-matching GNN are trainable.

3.4.1 Structure-Aware Feature Condensation

Following the argument of Section 3.3.3, the feature condensation function $f_{\mathbf{\Phi}}$ of Equation 3.9 is instantiated as a stack of graph attention layers, denoted $\mathrm{GAT}_{\mathbf{\Phi}_t}$ at epoch t, which maps the original features to the trainable condensed features

$$\tilde{\mathbf{X}}_t = \mathrm{GAT}_{\mathbf{\Phi}_t}(\mathbf{X}) \in \mathbb{R}^{N \times d}. \tag{3.12}$$

The choice is motivated by Equation 3.5: the attention coefficients are computed from the features of a node and its neighbours, so the projection assigns fine-grained, learned weights during aggregation instead of relying on the fixed degree-based weights of Equation 3.2. Dimensionality reduction and structural aggregation are thus performed by the same operator rather than in sequence.

Concretely, layers of the form given in Equation 3.4 and Equation 3.5 are stacked, with the first layer initialized by $\mathbf{h}_j^1 = \mathbf{x}_j$. The output layer L produces the condensed feature of node v_i,

$$\hat{\mathbf{x}}_i = \sigma\left(\sum_{j\in\mathcal{N}_i\cup\{v_i\}} \alpha_{ij}^L \mathbf{W}^L \mathbf{h}_j^{L-1}\right), \qquad \hat{\mathbf{x}}_i \in \mathbb{R}^d,\ d \ll D. \tag{3.13}$$

The epoch index t is suppressed in Equation 3.4 through Equation 3.13 for readability. Stacking several attention layers lets the operator capture nonlinear interactions between nodal attributes and topology, which is what allows the reduced feature space to remain informative for downstream learning. Note that $\mathbf{\Phi}$ is not fitted in advance; it is updated in every epoch by the objective introduced next.

3.4.2 From Parameter Matching to Trajectory Matching

The requirement that a GNN trained on $\mathcal{S}$ behave like one trained on $\tilde{\mathcal{T}}$ can be stated directly as a requirement on their optimal parameters. Writing $M(\cdot,\cdot)$ for a matching loss, this reads

$$\begin{aligned} &\min_{\mathcal{S},\mathbf{\Phi}}\ M\big(\boldsymbol{\theta}_{\mathcal{S}}^*, \boldsymbol{\theta}_{\tilde{\mathcal{T}}}^*\big) \\ &\quad \text{s.t. } \boldsymbol{\theta}_{\mathcal{S}}^* = \arg\min_{\boldsymbol{\theta}_{\mathcal{S}}} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}_{\mathcal{S}}}(\hat{\mathbf{A}}, \hat{\mathbf{X}}), \hat{\mathbf{Y}}\Big), \\ &\qquad\ \boldsymbol{\theta}_{\tilde{\mathcal{T}}}^* = \arg\min_{\boldsymbol{\theta}_{\tilde{\mathcal{T}}}} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}_{\tilde{\mathcal{T}}}}(\mathbf{A}, \tilde{\mathbf{X}}), \mathbf{Y}\Big). \end{aligned} \tag{3.14}$$

This is still a nested problem, and it is additionally fragile: the endpoint $\boldsymbol{\theta}_{\tilde{\mathcal{T}}}^*$ depends on the initialization $\boldsymbol{\theta}^0$, so matching a single endpoint constrains the condensed graph only along one arbitrarily chosen path through parameter space.

Both issues are resolved by asking for more, not less: instead of matching the endpoints, we match the entire trajectory. Adopting *curriculum gradient matching* [128], we require

the two graphs to induce similar parameter updates at every epoch, in expectation over initializations,

$$\begin{aligned}
&\min_{\mathcal{S}} \ \mathrm{E}_{\boldsymbol{\theta}^0 \sim Q_{\boldsymbol{\theta}^0}} \left[\sum_{t=0}^{T-1} M\big(\boldsymbol{\theta}_{\mathcal{S}}^t, \boldsymbol{\theta}_{\tilde{\mathcal{T}}}^t\big) \right] \\
&\text{with } \boldsymbol{\theta}_{\mathcal{S}}^{t+1} = \boldsymbol{\theta}_{\mathcal{S}}^t - \eta \nabla_{\boldsymbol{\theta}_{\mathcal{S}}^t} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}_{\mathcal{S}}^t}(\hat{\mathbf{A}}, \hat{\mathbf{X}}), \hat{\mathbf{Y}}\Big), \\
&\text{and } \boldsymbol{\theta}_{\tilde{\mathcal{T}}}^{t+1} = \boldsymbol{\theta}_{\tilde{\mathcal{T}}}^t - \eta \nabla_{\boldsymbol{\theta}_{\tilde{\mathcal{T}}}^t} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}_{\tilde{\mathcal{T}}}^t}(\mathbf{A}, \tilde{\mathbf{X}}), \mathbf{Y}\Big).
\end{aligned} \tag{3.15}$$

Because the objective now constrains each step individually, it can be evaluated without unrolling the recursion over $\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^{t-1}$: at epoch t the two gradients are computed at the same current parameter and compared immediately. This eliminates the dominant memory and time cost of Equation 3.11, and taking the expectation over $Q_{\boldsymbol{\theta}^0}$ removes the dependence on a single initialization. The relaxation is what makes joint condensation tractable at the scale of the datasets considered here.

3.4.3 The Gradient Matching Loss

It remains to specify M. Let

$$\mathbf{G}^{\tilde{\mathcal{T}}} = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathrm{GNN}_{\boldsymbol{\theta}}(\mathbf{A}, f_{\boldsymbol{\Phi}}(\mathbf{X})), \mathbf{Y}), \qquad \mathbf{G}^{\mathcal{S}} = \nabla_{\boldsymbol{\theta}} \mathcal{L}\Big(\mathrm{GNN}_{\boldsymbol{\theta}}(\hat{\mathbf{A}}, \hat{\mathbf{X}}), \hat{\mathbf{Y}}\Big) \tag{3.16}$$

denote the gradients of the classification loss with respect to the GNN weights under the two graphs. The matching loss is computed per class and summed, so that class-specific information is compared like with like. For a class c, let $\mathbf{G}^{\tilde{\mathcal{T}}^c}$ and $\mathbf{G}^{\mathcal{S}^c}$ be the gradients obtained from the class-c samples of the two graphs. Comparing the corresponding columns

by cosine distance gives

$$M(\mathbf{G}^{\tilde{\mathcal{T}}}, \mathbf{G}^{\mathcal{S}}) = \sum_{c=0}^{C-1} M^c,$$
$$\text{with} \quad M^c = \sum_{p=1}^{P} \sum_{h=1}^{H} \left(1 - \frac{\mathbf{G}_{h,p}^{\tilde{\mathcal{T}}^c} \cdot \mathbf{G}_{h,p}^{\mathcal{S}^c}}{\left\| \mathbf{G}_{h,p}^{\tilde{\mathcal{T}}^c} \right\| \left\| \mathbf{G}_{h,p}^{\mathcal{S}^c} \right\|} \right), \tag{3.17}$$

where $\mathbf{G}_{h,p}^{\tilde{\mathcal{T}}^c}$ and $\mathbf{G}_{h,p}^{\mathcal{S}^c}$ are the h-th column vectors of the class-c gradient matrices at layer p, and P and H index layers and columns respectively. Cosine distance is used rather than a Euclidean distance because what must agree is the *direction* of the update; the magnitudes of gradients computed from sample sets of very different sizes are not directly comparable, and normalizing removes that nuisance dependence.

3.4.4 Parameterizing the Condensed Structure

Treating $\hat{\mathbf{A}}$ and $\hat{\mathbf{X}}$ as free parameters would introduce $O(n^2)$ degrees of freedom in the adjacency matrix alone, which invites overfitting and scales poorly as the condensed graph grows. Following [69], we instead generate the condensed structure from the condensed features through a learned function $g_{\mathbf{\Psi}}(\cdot)$,

$$\hat{\mathbf{A}}_{ij} = g_{\mathbf{\Psi}}(\hat{\mathbf{X}}_{ij}) = \sigma\left(\frac{k_{\mathbf{\Psi}}(\mathbf{z}) + k_{\mathbf{\Psi}}(\mathbf{z}')}{2} \right), \tag{3.18}$$

where $\mathbf{z} := [\hat{\mathbf{x}}_i^\top; \hat{\mathbf{x}}_j^\top]$ and $\mathbf{z}' := [\hat{\mathbf{x}}_j^\top; \hat{\mathbf{x}}_i^\top]$ are elements of $\mathbb{R}^{1\times 2d}$, $k_{\mathbf{\Psi}}(\cdot)$ is a multi-layer perceptron parameterized by $\mathbf{\Psi}$, and $\sigma(\cdot)$ is the sigmoid function. Averaging over the two argument orders enforces symmetry of $\hat{\mathbf{A}}$ by construction.

Beyond reducing the parameter count, this parameterization has a practical benefit: because $g_{\mathbf{\Psi}}$ is a function of features rather than a table of edge weights, the condensed graph can be extended with additional synthetic nodes after training by generating only their features

and letting $g_{\mathbf{\Psi}}$ infer their connections.

3.4.5 The Final Objective

An empirical observation permits a final simplification. Under the trajectory-matching objective of Equation 3.15, the two parameter sequences $\boldsymbol{\theta}^t_{\mathcal{S}}$ and $\boldsymbol{\theta}^t_{\tilde{\mathcal{T}}}$ remain close throughout training, which is precisely what the objective enforces. We may therefore couple them and maintain a single sequence $\boldsymbol{\theta}^t$, the GNN weights trained on $\mathcal{S}$ at epoch t, at which both gradients are evaluated. Substituting Equation 3.18 and Equation 3.12 and collecting the trainable quantities, the framework optimizes

$$\min_{\hat{\mathbf{X}},\mathbf{\Phi},\mathbf{\Psi}} \mathrm{E}_{\boldsymbol{\theta}^0\sim Q_{\boldsymbol{\theta}^0}}\left[\sum_{t=0}^{T-1} M\Big(\nabla_{\boldsymbol{\theta}^t}\mathcal{L}\left(\mathrm{GNN}_{\boldsymbol{\theta}^t}(g_{\mathbf{\Psi}}(\hat{\mathbf{X}}),\hat{\mathbf{X}}),\hat{\mathbf{Y}}\right), \nabla_{\boldsymbol{\theta}^t}\mathcal{L}\left(\mathrm{GNN}_{\boldsymbol{\theta}^t}(\mathbf{A}, f_{\mathbf{\Phi}_t}(\mathbf{X})),\mathbf{Y}\right)\Big)\right]. \tag{3.19}$$

Equation 3.19 is a single-level minimization over three coupled quantities: the condensed features $\hat{\mathbf{X}}$, the feature condensation operator $\mathbf{\Phi}$, and the structure generator $\mathbf{\Psi}$. The condensed labels $\hat{\mathbf{Y}}$ are fixed, as described below.

3.4.6 Optimization

Alternating updates. The three quantities in Equation 3.19 are interdependent, and updating them simultaneously is unstable in practice. We therefore alternate. The feature condensation operator is updated every epoch, while the structure generator and the

condensed features are updated in alternating blocks of t_1 and t_2 epochs respectively:

$$\begin{aligned}
\boldsymbol{\Phi}_{t+1} &= \boldsymbol{\Phi}_t - \eta_1 \nabla_{\boldsymbol{\Phi}} M \quad \text{(every epoch)}, \\
\boldsymbol{\Psi}_{t+1} &= \boldsymbol{\Psi}_t - \eta_2 \nabla_{\boldsymbol{\Psi}} M \quad (t_1 \text{ epochs}), \\
\hat{\mathbf{X}}_{t+1} &= \hat{\mathbf{X}}_t - \eta_3 \nabla_{\hat{\mathbf{X}}} M \quad (t_2 \text{ epochs}).
\end{aligned} \tag{3.20}$$

Updating the projection at every epoch keeps $\tilde{\mathcal{T}}$ current with respect to the matching signal, while the staggered updates of $\boldsymbol{\Psi}$ and $\hat{\mathbf{X}}$ let each adapt to the other's most recent state rather than chasing a simultaneously moving target.

Mini-batch sampling. Computing Equation 3.17 over full graphs is memory-intensive for reconstruction-style objectives [160]. For each class c we therefore sample a batch of nodes together with their neighbours from $\tilde{\mathcal{T}}$, denoted $\tilde{\mathcal{T}}^c \sim \tilde{\mathcal{T}}$, and a batch of condensed nodes with all of their neighbours from $\mathcal{S}$, denoted $\mathcal{S}^c \sim \mathcal{S}$, and accumulate the per-class matching losses.

Initialization. The original features are centred as $\mathbf{X}' = \mathbf{X}(\mathbf{I}_D - \frac{1}{D}\mathbf{1}_D\mathbf{1}_D^\top)$. Learning $\hat{\mathbf{X}}$, $\hat{\mathbf{Y}}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\Psi}$ simultaneously is unnecessarily hard, so the condensed labels $\hat{\mathbf{Y}}$ are fixed after initialization with the class proportions of $\mathbf{Y}$ preserved. The condensed features $\hat{\mathbf{X}} \in \mathbb{R}^{n \times d}$ are then initialized by sampling, per class, the rows of $\tilde{\mathbf{X}} = f_{\boldsymbol{\Phi}}(\mathbf{X})$ corresponding to those labels.

Sparsification. After training, edge weights below a threshold γ are set to zero, yielding a sparse condensed adjacency matrix and removing spurious low-weight connections introduced by the continuous parameterization of Equation 3.18.

The complete procedure is summarized in Figure 3.3.

Input: training graph $\mathcal{T} = (\mathbf{A}, \mathbf{X}, \mathbf{Y})$; predefined condensed labels $\hat{\mathbf{Y}}$; threshold γ.

Initialize $\hat{\mathbf{X}}$ of width d by sampling node features from each class.
for $k = 0, \ldots, K-1$ **do**
  Initialize $\boldsymbol{\theta}^0 \sim Q_{\boldsymbol{\theta}^0}$
  for $t = 0, \ldots, T-1$ **do**
    $M \leftarrow 0$; compute $\tilde{\mathbf{X}}_t = f_{\boldsymbol{\Phi}_t}(\mathbf{X})$ to form $\tilde{\mathcal{T}}$
    for $c = 0, \ldots, C-1$ **do**
      Compute $\hat{\mathbf{A}} = g_{\boldsymbol{\Psi}_t}(\hat{\mathbf{X}})$ and set $\mathcal{S} = \{\hat{\mathbf{A}}, \hat{\mathbf{X}}, \hat{\mathbf{Y}}\}$
      Sample $(\tilde{\mathbf{A}}^c, \tilde{\mathbf{X}}^c, \mathbf{Y}^c) \sim \tilde{\mathcal{T}}$ and $(\hat{\mathbf{A}}^c, \hat{\mathbf{X}}^c, \hat{\mathbf{Y}}^c) \sim \mathcal{S}$
      Compute $\nabla_{\boldsymbol{\theta}^t} \mathcal{L}^{\tilde{\mathcal{T}}}$ and $\nabla_{\boldsymbol{\theta}^t} \mathcal{L}^{\mathcal{S}}$; accumulate M by Equation 3.17
    end for
    Update $\boldsymbol{\Phi}_{t+1} = \boldsymbol{\Phi}_t - \eta_1 \nabla_{\boldsymbol{\Phi}_t} M$
    if $t \bmod (t_1 + t_2) < t_1$ **then** update $\boldsymbol{\Psi}_{t+1} = \boldsymbol{\Psi}_t - \eta_2 \nabla_{\boldsymbol{\Psi}_t} M$
    else update $\hat{\mathbf{X}}_{t+1} = \hat{\mathbf{X}}_t - \eta_3 \nabla_{\hat{\mathbf{X}}_t} M$
    for $j = 0, \ldots, J-1$ **do** update $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta \nabla_{\boldsymbol{\theta}^t} \mathcal{L}^{\mathcal{S}}$ **end for**
  end for
end for
$\hat{\mathbf{A}} = g_{\boldsymbol{\Psi}}(\hat{\mathbf{X}})$; set $\hat{\mathbf{A}}_{ij} \leftarrow 0$ whenever $\hat{\mathbf{A}}_{ij} \leq \gamma$

Output: condensed graph $\mathcal{S} = (\hat{\mathbf{A}}, \hat{\mathbf{X}}, \hat{\mathbf{Y}})$ and projection parameters $\boldsymbol{\Phi}$.

Figure 3.3: The TinyGraph algorithm for joint feature and node condensation.

3.4.7 Relation to Existing Condensation Approaches

Two comparisons clarify what is new here. Relative to *node condensation* methods [156, 69, 68, 6] and graph compression methods [129, 116], which reduce the sample count while retaining full-width features, the framework above additionally reduces the width, and Section 3.5.7 shows that this is where the bulk of the storage saving comes from. Relative to *two-stage* approaches that prepend a fixed dimensionality reduction step [1, 30, 7, 123, 78] to a node condensation algorithm, it offers two advantages that follow directly from the design commitments of Section 3.3.3: feature and node condensation are learned under one objective, so the projection is structure-aware; and the projection is a learnable function rather than a fixed matrix, so it can adapt to the training dynamics it is meant to reproduce.

3.5 Experiments

This section evaluates the framework on five real-world graphs. The questions are organized around the claims of Section 3.1: whether joint condensation preserves accuracy relative to node-only condensation with full features (Section 3.5.2); whether structure awareness in the projection matters (Section 3.5.3); how far the feature dimension can be reduced (Section 3.5.4); which component is responsible for the gain (Section 3.5.5); whether the condensed graph transfers across architectures (Section 3.5.6); and what is actually saved (Section 3.5.7).

3.5.1 Experimental Setup

Datasets. We evaluate on five benchmarks. The three transductive ones are `Cora` and `Citeseer` [77], together with `Arxiv` [62]. The two inductive ones are `Flickr` [145] and `Reddit` [54]. All datasets are taken from PyTorch Geometric [42] using the publicly available splits adopted in prior work [54]. Statistics are given in Table 3.1. The feature-to-node ratios

noted in Section 3.1 are visible here: `Citeseer` has 3,703 features against 120 training nodes.

Table 3.1: Statistics of the five datasets. `Cora`, `Citeseer`, and `Arxiv` are transductive; `Flickr` and `Reddit` are inductive.

Dataset	# Nodes	# Feat.	# Edges	# Classes	Train/Val./Test
`Cora`	2,708	1,433	5,429	7	140/500/1,000
`Citeseer`	3,327	3,703	4,732	6	120/500/1,000
`Flickr`	89,250	500	899,756	7	44,625/22,312/22,313
`Reddit`	232,965	602	57,307,946	210	153,932/23,699/55,334
`Arxiv`	169,343	128	1,166,243	40	44,625/22,312/22,313

Baselines. We compare against seven alternatives. Original trains on the uncondensed graph. GCond [69] and GraphPCA [111] condense nodes while retaining full-width features. GCond-ICA, GCond-PCA, and GCond-LDA are two-stage baselines that project the features through a fixed matrix obtained by independent component analysis [30], principal component analysis [1], or linear discriminant analysis [7], and then apply GCond. GCond-VGAE replaces the fixed projection with the latent representation of a variational graph autoencoder [78]. For all two-stage baselines, feature condensation precedes node condensation.

Evaluation protocol. For each method we obtain a condensed graph from the training graph, train a GNN on it, and use that model to infer labels for the test nodes of the full graph; test accuracy is reported. All condensed graphs retain $r_n \times N$ nodes, and those with condensed features retain $r_d \times D$ features. In the transductive setting the full graph is used during condensation, since it is available at training time; in the inductive setting only the training graph is condensed.

Implementation. Unless stated otherwise the gradient-matching backbone $\mathrm{GNN}_{\boldsymbol{\theta}}$ is a two-layer SGC [131] with 256 hidden units. The structure generator $g_{\boldsymbol{\Psi}}$ is a three-layer multi-layer perceptron with 128 hidden units on `Cora` and `Citeseer` and 256 on the larger graphs. Training epochs are selected from $\{600, 800, 1000\}$ and learning rates from $\{10^{-2}, 5 \cdot$

$10^{-2}, 10^{-4}, 5 \cdot 10^{-4}, 10^{-6}, 10^{-8}\}$ for all methods. The sparsification threshold γ is set to 0.05 on `Citeseer` and `Cora` and 0.01 on `Flickr` and `Reddit`. The alternation schedule $(t_1, t_2, t_{\boldsymbol{\theta}})$ is $(20, 15, 10)$ on `Cora` and `Citeseer` and $(20, 10, 20)$ on `Flickr` and `Reddit`. Parameters are optimized with Adam [76] under the initialization scheme of [44].

3.5.2 Comparison Against Full-Feature Node Condensation

Table 3.2 reports test accuracy across node condensation ratios, with the accuracy attainable on the uncondensed graph shown alongside each dataset for reference. Table 3.3 lists the uncondensed accuracy of several GNN architectures for context.

The first observation is that **a GNN trained on the jointly condensed graph performs comparably to GCond trained with full-width features at the same node condensation ratio**. On `Citeseer` at $r_n = 0.9\%$, the jointly condensed graph reaches 70.7% against 70.6% for GCond, while additionally discarding 90% of the feature dimension; on `Flickr` at $r_n = 1.0\%$ it reaches 46.9% against 47.2%. The feature dimension, in other words, can be removed almost for free once the projection is learned jointly with the condensed graph, which is the central empirical claim of this chapter.

3.5.3 The Role of Structure Awareness

The two-stage baselines in Table 3.2 isolate the contribution of structure awareness, since they differ from the proposed framework precisely in that their projection ignores the adjacency matrix. The result is unambiguous: **the jointly learned projection outperforms every structure-agnostic feature condensation baseline**, typically by four to eight accuracy points on `Cora`, `Citeseer`, and `Flickr`. This supports the argument of Section 3.3.3 that the correlation between attributes and connectivity is load-bearing, and that a projection which cannot see the structure destroys it.

Table 3.2: Test accuracy (%) of GCond, GraphPCA, the two-stage baselines, and TinyGraph. The accuracy on the uncondensed graph is listed under each dataset name. Transductive results are reported on `Cora`, `Citeseer`, and `Arxiv`; inductive results on `Flickr` and `Reddit`. The feature ratio r_d is 10% on `Cora`, 10% on `Citeseer`, 30% on `Arxiv`, 20% on `Flickr`, and 30% on `Reddit`. Best results in bold.

Dataset	r_n	Full Feat.	Condensed Feature					
		GCond	G-PCA	ICA	PCA	LDA	VGAE	TinyGraph
`Cora`	1.3%	80.9 ± 3.2	65.9 ± 0.8	72.8 ± 4.2	73.4 ± 3.3	75.4 ± 2.8	76.6 ± 2.2	$\mathbf{79.3} \pm 0.7$
81.3 ± 0.4	2.6%	80.6 ± 2.7	68.6 ± 0.6	74.3 ± 1.4	74.2 ± 4.5	76.2 ± 2.4	76.9 ± 3.1	$\mathbf{80.1} \pm 1.1$
	5.2%	80.5 ± 4.3	70.2 ± 1.2	75.3 ± 2.0	75.4 ± 3.5	77.6 ± 3.2	77.8 ± 4.1	$\mathbf{79.4} \pm 1.5$
`Citeseer`	0.9%	70.6 ± 2.9	60.1 ± 1.3	65.3 ± 2.3	65.8 ± 1.1	67.2 ± 1.2	67.9 ± 2.8	$\mathbf{70.7} \pm 0.6$
72.1 ± 0.2	1.8%	70.8 ± 4.5	63.4 ± 0.8	65.8 ± 2.2	65.3 ± 2.8	66.3 ± 1.4	64.9 ± 3.5	$\mathbf{68.9} \pm 1.7$
	3.6%	70.2 ± 2.3	58.6 ± 1.8	66.3 ± 3.3	66.8 ± 2.2	67.2 ± 2.9	65.8 ± 2.6	$\mathbf{70.9} \pm 0.8$
`Flickr`	0.1%	47.1 ± 4.1	35.9 ± 2.4	41.7 ± 1.9	41.3 ± 1.7	42.5 ± 2.1	43.3 ± 1.8	$\mathbf{45.5} \pm 1.6$
47.3 ± 0.1	0.5%	47.2 ± 1.5	37.2 ± 2.2	43.6 ± 1.8	43.7 ± 2.0	43.9 ± 1.1	44.7 ± 2.6	$\mathbf{46.7} \pm 0.9$
	1.0%	47.2 ± 3.4	39.4 ± 0.5	42.8 ± 3.7	42.2 ± 2.2	44.5 ± 2.8	45.2 ± 1.7	$\mathbf{46.9} \pm 1.2$
`Reddit`	0.05%	90.3 ± 3.1	84.2 ± 1.9	87.4 ± 0.5	87.3 ± 1.4	87.3 ± 2.9	87.3 ± 1.6	$\mathbf{88.0} \pm 0.7$
93.9 ± 0.0	0.1%	90.7 ± 3.3	85.1 ± 4.3	$\mathbf{89.2} \pm 3.7$	86.4 ± 0.6	87.4 ± 1.7	88.2 ± 1.4	89.1 ± 1.1
	0.2%	91.2 ± 2.1	86.8 ± 2.1	86.3 ± 1.5	86.8 ± 2.5	87.1 ± 1.1	88.2 ± 4.7	$\mathbf{89.5} \pm 1.3$
`Arxiv`	0.05%	59.1 ± 1.2	54.9 ± 2.2	56.6 ± 0.7	$\mathbf{58.9} \pm 1.2$	57.6 ± 1.1	56.1 ± 2.7	58.8 ± 0.9
71.2 ± 0.2	0.25%	63.3 ± 0.7	52.2 ± 1.5	57.1 ± 0.9	60.0 ± 0.9	56.7 ± 1.5	59.2 ± 3.2	$\mathbf{61.6} \pm 1.3$
	0.5%	63.9 ± 0.5	55.4 ± 2.1	58.8 ± 1.1	61.3 ± 1.2	60.5 ± 1.7	60.5 ± 3.6	$\mathbf{62.2} \pm 0.8$

Table 3.3: Test accuracy (%) of several GNN architectures trained on the uncondensed graphs. SAGE denotes GraphSAGE.

Dataset	SGC	GCN	SAGE	APPNP	GAT
`Cora`	81.4	81.2	81.2	**83.1**	**83.1**
`Citeseer`	71.3	71.7	70.1	**71.8**	70.8
`Flickr`	46.2	47.1	46.1	**47.3**	44.3
`Reddit`	93.5	93.9	93.0	**94.3**	91.0
`Arxiv`	71.4	**71.7**	71.5	71.2	71.5

The one partial exception is instructive. On Reddit, GCond-VGAE is competitive, which would be puzzling if structure were always decisive. It suggests instead that the training structure of Reddit carries little information beyond the features. To test this, we trained a GCN on the original Reddit data with the training adjacency replaced by the identity, $\mathbf{A}_{\text{train}} = \mathbf{I}$, using the test structure only at inference. The resulting accuracy is 90.1%, close to the 93.9% obtained with the full training structure. Structure awareness helps exactly to the extent that the structure is informative, which is the behaviour one would want.

3.5.4 How Much of the Feature Dimension Is Needed?

Figure 3.4 sweeps the feature condensation ratio r_d from 10% to 90% at a fixed node ratio. Two findings emerge.

First, **accuracy comparable to full-feature node condensation is attained using only a fraction of the original feature dimension**: within 95% of the GCond full-feature accuracy at $r_d = 10\%$ on Cora and Citeseer, 20% on Flickr, and 30% on Reddit and Arxiv. The proposed method dominates the baselines at every ratio tested.

Second, **a larger feature dimension does not necessarily improve performance**, and the optimal r_d tracks the number of training nodes rather than the original width. On Cora and Citeseer, with 35 and 30 condensed training nodes, the best results occur at small r_d; on Flickr and Reddit, with 44,625 and 153,932 training nodes, larger r_d is preferred. This is the expected consequence of the dimensionality argument in Section 3.2.4: the reduced width must be small enough that the first-layer weight matrix remains identifiable from the available condensed samples, and large enough to encode the relationships among them.

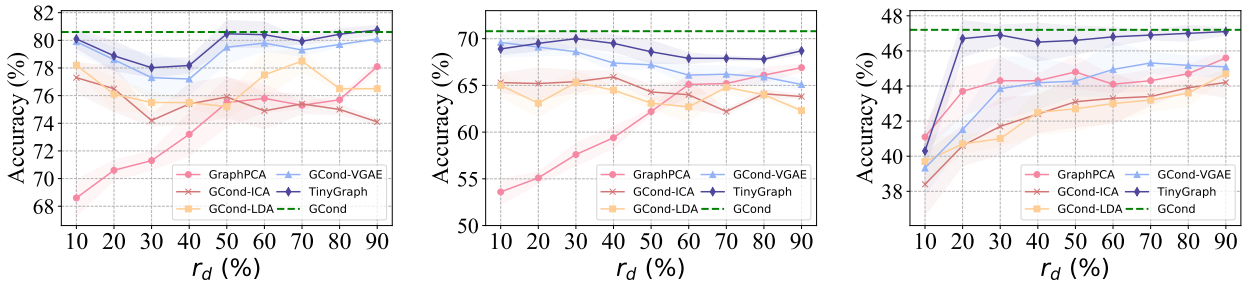

Figure 3.4: Test accuracy against the feature condensation ratio r_d on `Cora`, `Citeseer`, and `Flickr` (left to right), with r_n fixed at 2.6%, 1.8%, and 0.5% respectively. The proposed framework dominates the structure-agnostic baselines at every ratio tested.

3.5.5 Which Component Is Responsible? An Ablation

Table 3.4 replaces the attention-based projection with three alternatives, a linear map (TinyGraph-Lin), a multi-layer perceptron (TinyGraph-MLP), and a graph convolutional network (TinyGraph-GCN), and additionally replaces trajectory matching with one-step gradient matching [68], which matches gradients only at initialization rather than along the trajectory (TinyGraph-One).

The full framework outperforms every variant on all five datasets, but the margins are informative. On `Cora` and `Citeseer`, where the training node count is small and the feature dimension large ($D \gg n$), even the linear projection and the multi-layer perceptron perform respectably; condensation is comparatively easy in that regime and places few demands on the operator. On `Flickr` and `Reddit`, with many training nodes and comparatively few features, the structure-agnostic variants degrade sharply while the structure-aware ones (GCN and GAT) hold up. This is the same pattern observed in Section 3.5.3, now isolated within a single framework. The gap to TinyGraph-One additionally confirms that matching along the trajectory, rather than at a single point, is what Equation 3.15 buys.

Table 3.4: Ablation on the feature condensation function $f_{\Phi}(\cdot)$ and on the matching objective. Test accuracy (%); the node ratio r_n is given under each dataset.

Variant	`Cora` (2.6%)	`Citeseer` (1.9%)	`Flickr` (0.5%)	`Reddit` (0.1%)	`Arxiv` (0.25%)
TinyGraph-Lin	78.1 ± 3.2	64.6 ± 2.5	41.2 ± 3.1	78.2 ± 1.6	43.3 ± 1.0
TinyGraph-MLP	78.9 ± 2.6	66.5 ± 2.0	42.3 ± 2.3	80.4 ± 2.7	55.1 ± 2.2
TinyGraph-GCN	79.2 ± 1.9	66.9 ± 1.2	46.3 ± 1.5	87.9 ± 1.8	57.8 ± 0.9
TinyGraph-One	79.8 ± 1.5	67.5 ± 1.7	45.2 ± 1.8	88.6 ± 2.1	59.0 ± 1.7
TinyGraph	$\mathbf{80.1} \pm 1.1$	$\mathbf{68.9} \pm 1.7$	$\mathbf{46.7} \pm 0.9$	$\mathbf{89.1} \pm 1.1$	$\mathbf{61.6} \pm 1.3$

3.5.6 Generalization Across Architectures

A condensed dataset is only useful if it is not tied to the architecture that produced it. Table 3.5 evaluates condensed graphs across six backbones for gradient matching, namely a multi-layer perceptron, GCN, GraphSAGE [54], SGC, GAT, and APPNP [81]. **The condensed graph performs well outside the setting it was optimized for**, with average accuracies of 79.0%, 68.2%, 46.1%, 88.5%, and 58.5% across the five datasets, in every case exceeding the best baseline average. The consistency is plausibly attributable to the similar low-pass filtering behaviour of these architectures [94, 161].

Figure 3.5 examines the complementary question of whether the choice of feature condensation function constrains the choice of downstream backbone. Condensed graphs produced by different projections transfer comparably well across matching architectures, indicating that what the framework extracts is a property of the data rather than an artefact of one particular operator. This portability is what makes the condensed dataset a reusable object, a property that the next chapter relies on directly.

3.5.7 Anatomy of the Condensed Graph

Table 3.6 compares condensed graphs against their originals along node count, edge count, feature dimension, sparsity, and storage. **The reductions in node count and feature dimension together yield storage savings of two to three orders of magnitude** at

Table 3.5: Test accuracy (%) when the condensed graph is used with different gradient-matching architectures. Avg. is the mean over the six architectures. SAGE denotes GraphSAGE.

Dataset	Method	MLP	GCN	SAGE	SGC	GAT	APPNP	Avg.
Cora	GCond-LDA	70.0	74.7	72.1	71.2	76.2	70.3	72.4
	GCond-VGAE	70.7	75.2	74.0	73.2	76.9	71.0	73.5
	TinyGraph	78.9	79.8	76.2	79.4	80.1	79.3	**79.0**
Citeseer	GCond-LDA	59.6	65.8	61.9	66.7	66.3	60.0	63.2
	GCond-VGAE	57.9	64.3	60.5	64.4	64.9	60.7	62.1
	TinyGraph	67.5	68.9	66.5	68.5	68.9	68.0	**68.2**
Flickr	GCond-LDA	41.2	42.9	42.3	43.2	43.9	41.4	42.5
	GCond-VGAE	42.2	43.5	43.1	43.9	44.7	42.0	43.2
	TinyGraph	45.5	46.7	45.3	45.7	46.7	46.9	**46.1**
Reddit	GCond-LDA	83.8	86.4	85.8	87.9	87.4	84.4	85.8
	GCond-VGAE	84.1	87.0	88.7	87.5	88.2	85.0	86.4
	TinyGraph	87.5	89.1	88.7	87.9	89.1	88.9	**88.5**
Arxiv	GCond-LDA	53.4	54.2	54.8	55.4	53.5	56.9	54.7
	GCond-VGAE	54.1	55.9	55.7	54.4	55.3	54.6	55.0
	TinyGraph	55.4	59.1	58.3	60.0	59.7	58.2	**58.5**

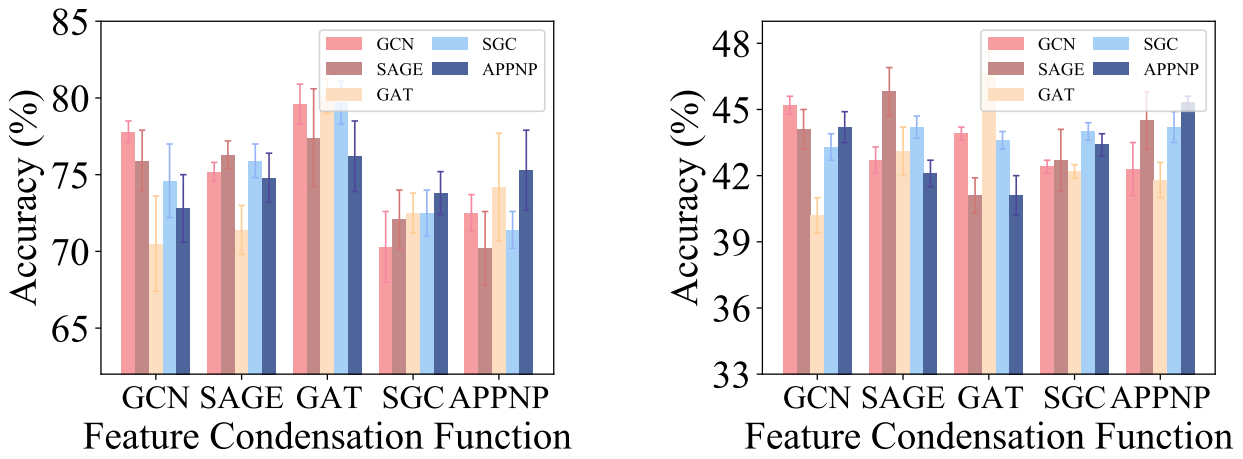

Figure 3.5: Cross-architecture test accuracy (%) on `Cora` (left) and `Flickr` (right). The horizontal axis gives the architecture used as the feature condensation function; bars report accuracy when the resulting condensed graph is used with other gradient-matching architectures. Graphs condensed by different operators all transfer well.

a small accuracy cost: `Citeseer` falls from 47.1 MB to 0.15 MB, a 99.7% reduction, for a 2.5 point drop in accuracy, and `Reddit` from 435.5 MB to 0.4 MB. Compared with node-only condensation, the additional feature reduction accounts for a further 75.3%, 83.3%, 87.0%, 87.8%, and 51% of storage on the five datasets, which is the concrete payoff of the joint formulation.

A second observation concerns sparsity. Condensed graphs are consistently denser than their originals. This is a necessary consequence of scale rather than a defect of the method: preserving a sparsity of 99.9% across 60 nodes would leave almost no edges at all, and the resulting graph could not support message passing. The parameterization of Equation 3.18 together with the threshold γ controls where on this trade-off the condensed graph lands.

Table 3.6: Statistics of condensed graphs against the original graphs. Percentages give the reduction relative to the original.

Dataset	Version	#Nodes	#Edges	#Feat.	Sparsity	Storage	Acc.
`Cora`	Full	2,708	5,429	1,433	99.9%	14.9 MB	81.5%
	Condensed	70	2,128	143	14.4%	0.099 MB	80.1%
	Reduction	97.4%	60.8%	90.0%	—	99.3%	1.5%
`Citeseer`	Full	3,327	4,732	3,703	99.9%	47.1 MB	70.7%
	Condensed	60	1,454	370	20.6%	0.15 MB	68.9%
	Reduction	98.2%	69.3%	90.0%	—	99.7%	2.5%
`Flickr`	Full	44,625	218,140	500	99.9%	86.8 MB	47.1%
	Condensed	223	3,788	100	84.8%	0.5 MB	46.7%
	Reduction	99.5%	98.3%	80.0%	—	99.9%	0.8%
`Reddit`	Full	153,932	10,753,238	602	99.9%	435.5 MB	94.1%
	Condensed	153	301	181	97.4%	0.4 MB	89.1%
	Reduction	99.9%	99.9%	69.9%	—	99.9%	5.3%
`Arxiv`	Full	169,343	1,166,243	128	99.9%	100.4 MB	71.2%
	Condensed	454	3,354	43	97.8%	0.1 MB	61.6%
	Reduction	99.7%	99.7%	66.4%	—	99.9%	13.5%

3.5.8 What the Condensed Features Look Like

Figure 3.6 visualizes t-SNE [123] embeddings of the original and condensed features on `Cora` and `Citeseer`. The condensed features form visibly separable clusters where the original features are intermixed, which indicates that the learned projection is not merely compressing but reorganizing the feature space along class-discriminative directions. Notably, the

embeddings at $r_d = 10\%$ are more clearly separated than those at $r_d = 20\%$, consistent with the finding of Section 3.5.4 that a smaller width is not simply a lossier one.

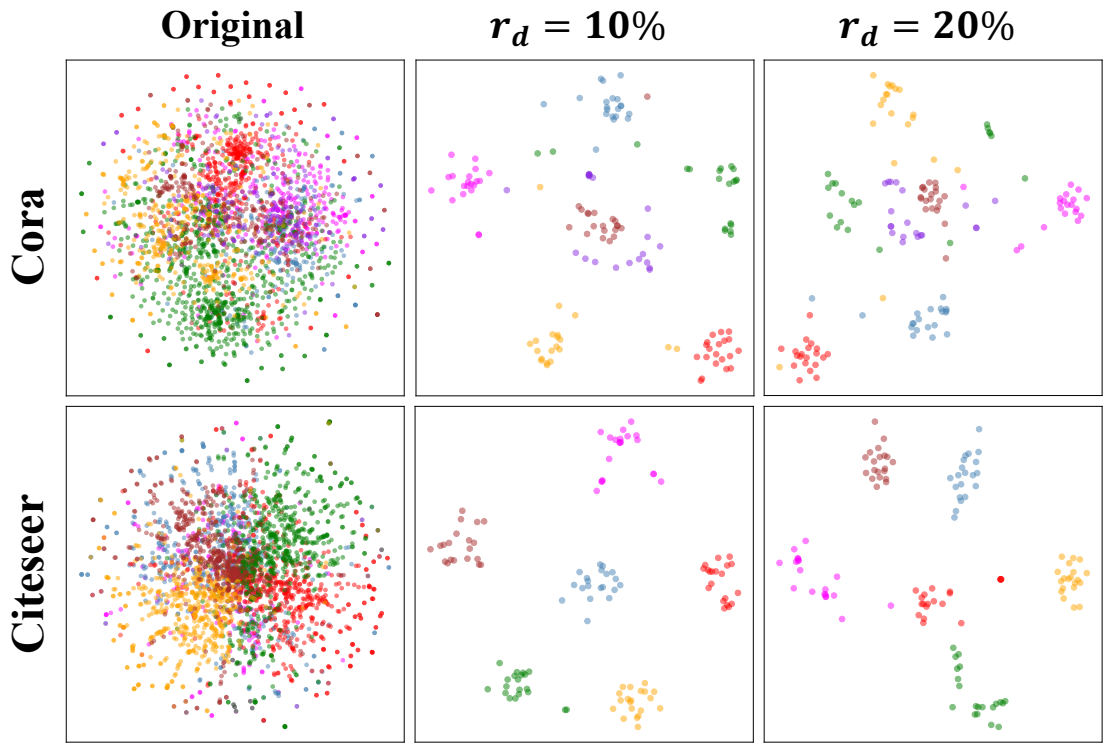

Figure 3.6: t-SNE visualization of original and condensed node features on `Cora` and `Citeseer`, at $r_d = 10\%$ and $r_d = 20\%$. Colours denote classes. The separable clusters in the condensed features indicate that the learned projection is class-discriminative.

3.5.9 Running Time

Table 3.7 reports the wall-clock time of the condensation procedure itself on an NVIDIA RTX A4000 GPU. The cost grows smoothly with the node condensation ratio and remains modest in absolute terms: condensing `Arxiv` at $r_n = 1\%$ takes roughly five minutes. Since condensation is performed once and the resulting graph is reused across every subsequent training run, this cost is amortized immediately in the repeated-training settings that motivate the chapter.

Table 3.7: Running time of the condensation procedure for 20 epochs (seconds), with r_d set to 10%, 30%, and 30% on `Cora`, `Flickr`, and `Arxiv` respectively.

r_n (%)	0.2	0.5	1	5	10
`Cora`	5.7	11.0	21.8	24.6	36.3
`Flickr`	87.4	112.2	167.9	214.1	302.5
`Arxiv`	201.7	231.8	309.4	467.5	642.1

3.6 Discussion: What a Gradient-Matched Dataset Preserves

It is worth stating plainly what the objective in Equation 3.19 does and does not require, because the answer determines how far the mechanism can be carried.

The objective does not ask the condensed graph to resemble the original graph in any perceptual or statistical sense. It does not match feature moments, degree distributions, or spectra. It asks for one thing only: that the two graphs induce the same gradient directions, class by class, at every point along a training trajectory. Everything the condensed graph retains follows from that constraint, and the empirical results above are best read as evidence that this constraint is sufficient. The condensed features are not compressed versions of the originals, as Figure 3.6 makes clear; they are whatever the matching objective needs them to be. Similarly, the condensed graph is denser than the original because density is not constrained, only its effect on the gradient is.

This has an immediate consequence for the rest of the dissertation. If a dataset can be made to encode an arbitrary property of the training dynamics, then utility is only the first property one might choose to encode. Nothing in Equation 3.15 or Equation 3.17 is specific to accuracy; the matching term measures agreement between two gradient signals, and additional terms may be imposed on the same synthetic data to demand additional properties of the models it trains. The limitation of the present chapter is precisely that it exercises only one such property.

Two further limitations should be recorded. First, the condensed labels $\hat{\mathbf{Y}}$ are fixed at initialization to keep the optimization tractable, so the class proportions of the condensed graph are inherited from the original rather than learned; this is a modelling choice whose consequences become significant in the setting of the next chapter. Second, while Section 3.5.6 shows strong empirical transfer across backbones, the matching objective is defined with respect to a particular architecture family, and no guarantee of transfer is established.

3.7 Summary

This chapter addressed the first of the three demands organizing this dissertation. It showed that the computational cost of learning on large graphs is driven jointly by the number of nodes and the dimensionality of their features, that existing condensation methods address only the former, and that addressing both requires the feature reduction to be structure-aware and learned together with the condensed data rather than applied to it beforehand. The resulting framework replaces an intractable bi-level problem with a single-level objective that matches gradients along the training trajectory, and it recovers up to 98.5% of the original accuracy while removing more than 97% of the nodes, up to 90% of the feature dimension, and over 99% of the storage.

The wider significance of the result is methodological. Once a small synthetic dataset can be made to reproduce the training dynamics of a large one, the dataset is no longer a fixed input to learning but an object that can be optimized to make the models trained on it behave in a prescribed way. This chapter used that leverage to make training cheaper. The next chapter uses the same leverage for a different purpose: to require that the models trained on the synthetic data treat demographic groups equitably, without modifying the model, the architecture, or the training procedure. The machinery carries over almost unchanged; what changes is the property being demanded of the training dynamics.

Chapter 4

Encoding Group Fairness into Synthetic Training Data

4.1 Overview

Chapter 3 established a mechanism and then, in Section 3.6, observed its limitation. The mechanism is that a small synthetic dataset can be optimized so that the models it trains follow the same optimization trajectory as models trained on a large real dataset; the limitation is that the only property demanded of that trajectory was predictive utility. Nothing in the matching objective is specific to utility. The gradient matching loss measures agreement between two gradient signals, and the synthetic data is free to become whatever reproduces the target signal. If the target signal is changed, the property encoded into the data changes with it.

This chapter exercises that observation. It keeps the data-centric machinery of Chapter 3 essentially intact and redirects it from the first demand of this dissertation, learning efficiently, to the second, learning in a way that treats demographic groups equitably. The move is

deliberately conservative: the synthetic dataset is still learned by gradient matching, the optimization is still alternating, the labels are still fixed at initialization, and the downstream model is never modified. What changes is the trajectory being matched.

Why fairness is a data-level problem. Deep neural networks now mediate consequential decisions in finance, hiring, healthcare, and education [105, 61, 52, 16, 80]. Because these models learn from historical records, they reproduce and can amplify the societal biases those records encode, producing disparate outcomes along attributes such as gender, race, and age [8, 39]. The response has been overwhelmingly *model-centric*. In-processing methods add fairness terms or constraints to the training objective [73, 11, 143, 2, 147]; post-processing methods adjust the outputs of an already-trained model [56, 127, 67]. Both are flexible, and both share a structural weakness: the intervention lives inside a particular model. Every new architecture trained on the same biased data requires its own debiasing pass, its own hyperparameter search, and its own validation. The intervention is never reusable, and in modular or rapidly evolving pipelines it must be repeated indefinitely.

The *data-centric* alternative moves the intervention upstream, into the training data itself [97, 146, 130]. Classical pre-processing methods in this spirit, such as reweighting [72], relabelling, and optimized transformation of the input distribution [17], are model-agnostic by construction, but they rely on domain-specific priors and hand-designed heuristics. Crucially, they provide no differentiable objective over the data, so they cannot make use of the substantial body of fairness regularizers developed for in-processing methods [95, 20]. The gap is therefore not conceptual but technical: the data-centric setting has lacked a way to *optimize* data against a fairness criterion. Gradient matching supplies exactly that, which is why the machinery of the previous chapter transfers.

The question this chapter answers is accordingly:

Can a training dataset be learned automatically such that any model trained on it is fair as well as accurate, and such that the fairness so obtained survives a change of model?

Approach. The framework developed here, FairData, learns a synthetic dataset under two objectives simultaneously. A gradient matching term, identical in form to the one used in Chapter 3, keeps the synthetic data faithful to the training dynamics of the real data and thereby preserves utility. A differentiable group fairness regularizer, evaluated on the original data where sensitive attributes are observed, reshapes those dynamics so that the trajectory being matched is a fair one rather than merely an accurate one. The synthetic data absorbs the corrected trajectory. Because the result is an ordinary dataset in the original input space, carrying no sensitive attribute of its own and requiring no architectural change, it can be handed to any downstream learner, including non-differentiable ones.

Contributions of this chapter. The chapter makes the following contributions.

- It reframes group fairness as a property that can be optimized into a dataset rather than into a model, and shows that the gradient matching machinery of Chapter 3 is the missing differentiable bridge between data-centric intervention and the existing family of fairness regularizers.
- It formulates the resulting objective, which balances gradient alignment for utility against fairness regularization for equity, and makes explicit the mechanism by which fairness reaches the synthetic data: not by acting on it directly, but by altering the trajectory it is required to reproduce.
- It shows that the fairness so encoded is *transferable*. A dataset generated with one architecture retains both its accuracy and its fairness when consumed by a different

one, including across the neural and tree-based divide, so a single optimization serves many downstream models.

- It evaluates the framework on three tabular benchmarks and one image benchmark, across gender, race, and age as sensitive attributes, against six model-centric baselines and one pre-processing baseline, and reports consistent reductions in group disparity at competitive accuracy.

Section 4.2 develops the fairness notions used throughout this chapter and the two that follow. Section 4.3 states the problem, Section 4.4 develops the framework, Section 4.5 reports the empirical study, and Section 4.7 closes the chapter.

4.2 Preliminaries

Section 3.2 introduced the notation for datasets, models, and training trajectories. This section extends it with the sensitive attribute and defines the group fairness criteria used in this chapter and in Chapter 5 and Chapter 6. The source material for this line of work states these criteria informally; because three chapters of this dissertation depend on the distinction between them, and on the distinction between a criterion and its differentiable surrogate, they are defined precisely here.

4.2.1 Setting and Notation

Throughout this chapter a dataset carries a third component beyond features and labels:

$$\mathcal{T} = \{\mathbf{X}, \mathbf{Y}, \mathbf{S}\}, \qquad \mathbf{X} \in \mathbb{R}^{N \times D},\ \mathbf{Y} \in \{0, \dots, C-1\}^N,\ \mathbf{S} \in \{0, \dots, K-1\}^N, \tag{4.1}$$

where $\mathbf{S}$ collects the *sensitive attribute* of each sample over K demographic groups, such as gender, race, or age bracket. As in Chapter 3, N is the number of samples and D the feature

dimension. We write $(\mathbf{x}, y, s)$ for a generic sample, $f_{\boldsymbol{\theta}}$ for a downstream model parameterized by $\boldsymbol{\theta}$, and $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$ for its prediction. The graph neural network $\mathrm{GNN}_{\boldsymbol{\theta}}$ of Chapter 3 is the special case in which $f_{\boldsymbol{\theta}}$ additionally consumes an adjacency matrix; nothing in this chapter depends on that structure, and $f_{\boldsymbol{\theta}}$ here denotes a generic differentiable predictor.

Capitalized $\hat{Y}$, Y, and S denote the corresponding random variables when probabilistic statements are made.

4.2.2 Group Fairness Criteria

Fairness criteria divide broadly into *individual* notions, which require that similar individuals receive similar predictions [39], and *group* notions, which require statistical parity of some quantity across demographic groups. This dissertation is concerned with the latter. Two group criteria recur.

Definition 4.1 (Demographic parity)**.** *A predictor $f_{\boldsymbol{\theta}}$ satisfies* demographic parity *with respect to S if its prediction is statistically independent of the sensitive attribute,*

$$P(\hat{Y} = c \mid S = k) \;=\; P(\hat{Y} = c) \qquad \text{for all } c \in \{0, \ldots, C-1\},\ k \in \{0, \ldots, K-1\}. \tag{4.2}$$

Demographic parity constrains the marginal distribution of decisions and makes no reference to the ground-truth label. It is therefore appropriate when the observed labels are themselves suspected of encoding historical bias, and inappropriate when base rates legitimately differ across groups, since in that case it forces the predictor away from the labels.

Definition 4.2 (Equalized odds and equal opportunity)**.** *A predictor $f_{\boldsymbol{\theta}}$ satisfies* equalized odds *with respect to S if its prediction is conditionally independent of the sensitive attribute*

given the true label,

$$P(\hat{Y} = c \mid S = k, Y = y) \;=\; P(\hat{Y} = c \mid Y = y) \qquad \textit{for all } c,\ k,\ y. \tag{4.3}$$

Restricting Equation 4.3 to the advantaged outcome $y = 1$ alone gives equal opportunity *[56], which equalizes true positive rates across groups without constraining false positive rates.*

Equalized odds conditions on the label and so tolerates genuine base-rate differences, at the cost of inheriting whatever bias the labels contain. The two criteria are in general incompatible: except in degenerate cases, a nontrivial predictor cannot satisfy Equation 4.2 and Equation 4.3 simultaneously [56, 31]. Reporting both, as is done throughout this chapter, is therefore not redundant but necessary: a method that improves one at the expense of the other has not improved fairness so much as relabelled it.

4.2.3 Empirical Fairness Gaps

Both criteria are exact statements that no finite-sample predictor satisfies exactly, so they are measured by the extent of their violation. Let $\widehat{P}$ denote an empirical probability computed on a held-out set. The *demographic parity gap* and the *equalized odds gap* are

$$\Delta_{DP} = \frac{1}{K} \sum_{k=0}^{K-1} \max_{c} \left| \widehat{P}(\hat{Y} = c) - \widehat{P}(\hat{Y} = c \mid S = k) \right|, \tag{4.4}$$

$$\Delta_{EO} = \frac{1}{K} \sum_{k=0}^{K-1} \max_{c,\,y} \left| \widehat{P}(\hat{Y} = c \mid Y = y) - \widehat{P}(\hat{Y} = c \mid S = k, Y = y) \right|. \tag{4.5}$$

Both are non-negative, are zero exactly when the corresponding criterion holds empirically, and are reported in percentage points. Smaller values indicate less disparity. These are the fairness metrics reported in every experimental table of this chapter.

4.2.4 Differentiable Surrogates

Equation 4.4 and Equation 4.5 are defined in terms of hard predictions $\hat{Y} = \arg\max_c f_{\boldsymbol{\theta}}(\mathbf{x})_c$, which are piecewise constant in $\boldsymbol{\theta}$ and therefore useless as optimization objectives. Every in-processing fairness method, and the framework of this chapter, works instead with a *differentiable surrogate* obtained by replacing the indicator of a predicted class with the model's soft score for that class. Writing $f_{\boldsymbol{\theta}}(\mathbf{x})_c \in [0, 1]$ for the softmax output of class c, the empirical probabilities in Equation 4.4 are replaced by group-conditional means,

$$\widehat{P}(\hat{Y} = c) \;\longrightarrow\; \frac{1}{N}\sum_{i=1}^{N} f_{\boldsymbol{\theta}}(\mathbf{x}_i)_c, \qquad \widehat{P}(\hat{Y} = c \mid S = k) \;\longrightarrow\; \frac{1}{N_k}\sum_{i:\, s_i = k} f_{\boldsymbol{\theta}}(\mathbf{x}_i)_c, \tag{4.6}$$

with $N_k = |\{i : s_i = k\}|$. The resulting quantity is differentiable in $\boldsymbol{\theta}$ and coincides with Equation 4.4 in the limit of confident predictions. Throughout this dissertation, a fairness *metric* means the hard-prediction gap of Equation 4.4 or Equation 4.5 evaluated at test time, while a fairness *regularizer* means a differentiable surrogate used during training. Metrics are never optimized directly, and the quality of a regularizer is not determined by the metric it is named after. That distinction is the subject of Chapter 5.

4.2.5 Where Fairness Can Be Enforced

It is useful to record the three points in the pipeline at which a fairness intervention can be inserted, since this chapter and Chapter 5 occupy different ones. *Pre-processing* methods modify the training data before learning [72, 17]; *in-processing* methods modify the training objective [73, 143, 2]; *post-processing* methods modify the outputs of a trained model [56, 127]. The framework of this chapter is a pre-processing method in the sense that its output is a dataset, but unlike classical pre-processing it obtains that dataset by differentiable optimization against a fairness objective rather than by a fixed heuristic transformation. It therefore inherits the model-agnosticism of the first family and the optimization machinery

of the second.

4.3 Problem Statement

Given a dataset $\mathcal{T} = \{\mathbf{X}, \mathbf{Y}, \mathbf{S}\}$ as in Equation 4.1, we seek a synthetic dataset

$$\mathcal{T}' = \{\mathbf{X}', \mathbf{Y}'\}, \qquad \mathbf{X}' \in \mathbb{R}^{N' \times D},\ \mathbf{Y}' \in \{0, \ldots, C-1\}^{N'}, \tag{4.7}$$

such that a model $f_{\boldsymbol{\theta}}$ trained on $\mathcal{T}'$ (i) attains predictive performance competitive with the same model trained on $\mathcal{T}$, and (ii) produces predictions with reduced group disparity with respect to $\mathbf{S}$, as measured by Equation 4.4 and Equation 4.5 on held-out data.

Three features of this statement deserve emphasis, because they are what distinguish the setting from both model-centric debiasing and from Chapter 3.

First, $\mathcal{T}'$ contains *no sensitive attribute*. The synthetic data is a pair of features and labels only. Consequently no downstream consumer of $\mathcal{T}'$ needs access to $\mathbf{S}$, at training time or at inference time, which removes a standing deployment obstacle for in-processing methods that require the sensitive attribute as a model input.

Second, $\mathbf{X}'$ lives in the *original input space* $\mathbb{R}^D$. It is not a latent code, and no decoder is required to use it. Any learner that accepts the original data accepts $\mathcal{T}'$, including non-differentiable and rule-based models to which gradient-based fairness interventions do not apply at all.

Third, the requirement is stated on *any* model trained on $\mathcal{T}'$, not on one particular model. This is the sense in which the intervention is performed once rather than per architecture, and it is what Section 4.5.4 tests empirically.

Note that the size N' of the synthetic set is a free parameter. Taking $N' \ll N$ recovers

the condensation regime of Chapter 3 and yields storage and training savings alongside the fairness gain; taking $N' = N$ isolates the fairness effect. The experiments in this chapter concern the second setting, since the object of study is what the data encodes rather than how small it can be made.

4.4 The FairData Framework

The framework combines two components: dataset learning by gradient matching, which preserves utility, and fairness-aware regularization, which reshapes the dynamics being matched. Figure 4.1 gives an overview. A model is trained on the original dataset $\mathcal{T}$ and its gradients recorded; the synthetic dataset $\mathcal{T}'$ is fed through the same model and its gradients recorded; the matching loss $\mathcal{L}_M$ and the fairness loss $\mathcal{L}_F$ are computed and combined, and the result drives the update of $\mathcal{T}'$.

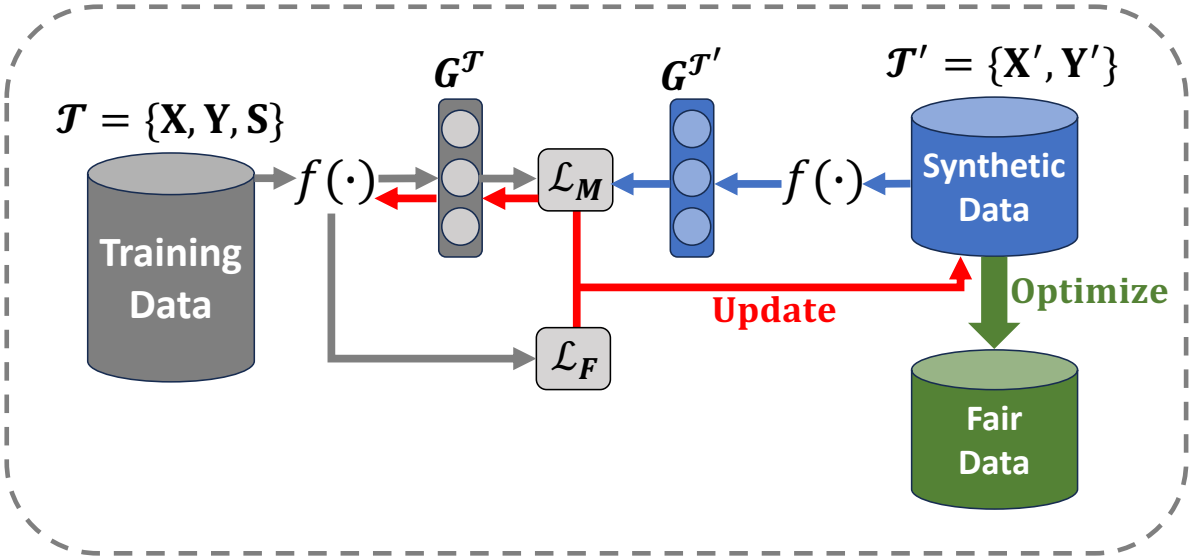

Figure 4.1: Overview of the FairData framework. The model $f_{\boldsymbol{\theta}}$ is trained on the original dataset $\mathcal{T}$, yielding gradients $D^{\mathcal{T}}$; the synthetic dataset $\mathcal{T}'$ is passed through the same model, yielding $D^{\mathcal{T}'}$. The gradient matching loss $\mathcal{L}_M$ and the fairness loss $\mathcal{L}_F$ are combined to update $\mathcal{T}'$.

4.4.1 Dataset Learning by Gradient Matching

This component is the machinery of Chapter 3, restated for a dataset without graph structure. At epoch t, let

$$\begin{aligned} \mathcal{L}_t^{\mathcal{T}} &= \frac{1}{|\mathcal{T}|} \sum_{(\mathbf{x},y)\in\mathcal{T}} \ell(f_{\boldsymbol{\theta}^t}(\mathbf{x}),\, y)\,, & \mathcal{L}_t^{\mathcal{T}'} &= \frac{1}{|\mathcal{T}'|} \sum_{(\mathbf{x}',y')\in\mathcal{T}'} \ell(f_{\boldsymbol{\theta}^t}(\mathbf{x}'),\, y')\,, \\ D_t^{\mathcal{T}} &= \nabla_{\boldsymbol{\theta}^t} \mathcal{L}_t^{\mathcal{T}}\,, & D_t^{\mathcal{T}'} &= \nabla_{\boldsymbol{\theta}^t} \mathcal{L}_t^{\mathcal{T}'}\,, \end{aligned} \tag{4.8}$$

where ℓ is the per-sample cross-entropy loss. The synthetic dataset is required to induce the same parameter updates as the real one at every point along the trajectory, in expectation over initializations drawn from $Q_{\boldsymbol{\theta}^0}$:

$$\min_{\mathcal{T}'}\ \mathrm{E}_{\boldsymbol{\theta}^0\sim Q_{\boldsymbol{\theta}^0}} \left[\sum_{t=0}^{T-1} \mathcal{L}_M\Big(D_t^{\mathcal{T}},\, D_t^{\mathcal{T}'}\Big) \right]. \tag{4.9}$$

The matching loss $\mathcal{L}_M$ is the same class-wise cosine distance introduced in Equation 3.17, with the layer and column indices renamed:

$$\mathcal{L}_M(D^{\mathcal{T}}, D^{\mathcal{T}'}) = \sum_{c=0}^{C-1} \sum_{l=1}^{L} \sum_{b=1}^{B} \left(1 - \frac{D_{l,b}^{\mathcal{T}_c} \cdot D_{l,b}^{\mathcal{T}'_c}}{\left\| D_{l,b}^{\mathcal{T}_c} \right\| \left\| D_{l,b}^{\mathcal{T}'_c} \right\|} \right), \tag{4.10}$$

where $D_{l,b}^{\mathcal{T}_c}$ and $D_{l,b}^{\mathcal{T}'_c}$ are the b-th column vectors of the class-c gradient matrices at layer l. As in Section 3.4.3, the cosine distance is used because gradient *directions* are what must agree; magnitudes computed from sample sets of different sizes are not comparable. Equation 4.9 and Equation 4.10 on their own reproduce utility and nothing else, which is precisely the situation at the end of Chapter 3.

4.4.2 The Fairness Regularizer

To the matching objective we add a term that penalizes group disparity in the predictions of the model currently being trained. Following the demographic parity gap of Equation 4.4, and using the differentiable surrogate of Equation 4.6 in place of the hard empirical probabilities, the fairness loss is

$$\mathcal{L}_F = \frac{1}{K} \sum_{k=0}^{K-1} \max_c \left| P(\hat{y} = c) - P(\hat{y} = c \mid s = k) \right|, \tag{4.11}$$

where $\hat{y} = f_{\boldsymbol{\theta}}(\mathbf{x})$ and K is the number of sensitive groups.

One point of care is required here, and it determines how the whole framework behaves. The synthetic dataset carries no sensitive attribute, so Equation 4.11 cannot be evaluated on $\mathcal{T}'$. It is evaluated on the original dataset $\mathcal{T}$, where $\mathbf{S}$ is observed, using the model $f_{\boldsymbol{\theta}^t}$ as it currently stands. The fairness signal therefore enters through the model parameters rather than through the synthetic features directly, a mechanism made explicit in Section 4.4.4. Note also that Equation 4.11 is one choice among many: any differentiable group fairness surrogate can be substituted without altering the rest of the framework, including surrogates for equal opportunity [56] and dependence-based criteria. This modularity is a genuine strength of the design and, as Section 4.6 argues, also the source of its principal limitation.

4.4.3 The Joint Objective

Collecting the terms, the training objective is

$$\mathcal{L} \;=\; \lambda_F \, \mathcal{L}_F \;+\; \lambda_M \, \mathcal{L}_M \;+\; \beta \, \|\boldsymbol{\theta}\|^2, \tag{4.12}$$

where λ_F and λ_M weight the fairness and matching terms and β controls weight decay. Only the ratio of the first two coefficients matters; we write $\alpha = \lambda_M / \lambda_F$ for the *utility-to-fairness*

ratio, and in Section 4.5.6 we sweep the normalized fairness weight

$$\bar{\alpha} = \frac{\lambda_F}{\lambda_F + \lambda_M} \in (0, 1), \tag{4.13}$$

which increases monotonically as the fairness term is favoured over the matching term. The synthetic labels $\mathbf{Y}'$ are initialized by sampling from the class distribution of $\mathbf{Y}$ and then held fixed, as in Chapter 3.

The corresponding updates are

$$\mathbf{X}'_{t+1} = \mathbf{X}'_t - \eta_1 \nabla_{\mathbf{X}'_t} \mathcal{L}_M, \qquad \boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta_2 \nabla_{\boldsymbol{\theta}^t} \mathcal{L}, \tag{4.14}$$

with learning rates η_1 and η_2.

4.4.4 The Mechanism: Matching a Fair Trajectory

Equation 4.14 repays a careful reading, because it reveals what the framework actually does and why it is a direct descendant of Chapter 3 rather than a new construction.

The synthetic features $\mathbf{X}'$ are updated by the gradient of the matching loss alone. No fairness term appears in their update. The model parameters $\boldsymbol{\theta}$, by contrast, are updated by the gradient of the full objective Equation 4.12, which does include the fairness term. The consequence is that $\boldsymbol{\theta}^0, \boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^{T-1}$ is not the trajectory that empirical risk minimization would follow; it is the trajectory that a *fairness-regularized* learner follows. The synthetic data is then optimized, exactly as in Chapter 3, to reproduce the gradients along that trajectory.

Stated compactly: Chapter 3 matched the gradients of an accurate learner and obtained data that trains accurate models cheaply; this chapter matches the gradients of a fair and accurate learner and obtains data that trains fair and accurate models. The mechanism, the

objective, and the optimizer are the same. Only the trajectory has moved.

Two properties of the resulting data follow from this reading, and both are confirmed empirically below. First, the fairness is not stored as a constraint to be enforced at test time but as a property of the data distribution itself, which is why a plain empirical risk minimizer trained on $\mathcal{T}'$ produces fairer predictions with no fairness machinery of its own (Section 4.5.5). Second, because gradient directions are shared across architectures with similar inductive biases, the encoded fairness has no particular attachment to the architecture used to generate it (Section 4.5.4).

4.4.5 Optimization

Label treatment. Each synthetic feature vector $\mathbf{x}'_i$ is bound to a fixed label y'_i for the whole of optimization. Beyond the stability argument given in Chapter 3, fixing labels has a specific justification in the fairness setting: if labels were free to move, the optimizer could reduce the measured disparity by reshaping the class distribution rather than by changing what the model learns, and group-level comparisons across the synthetic and real data would no longer be meaningful.

Alternating updates. Optimizing $\mathbf{X}'$ and $\boldsymbol{\theta}$ simultaneously is a bi-level problem of the kind analysed in Section 3.3: changes to $\mathbf{X}'$ alter the gradients that drive $\boldsymbol{\theta}$, which in turn alters the matching target. We therefore alternate, holding $\boldsymbol{\theta}$ fixed while $\mathbf{X}'$ is updated against $\mathcal{L}_M$ for t_1 steps, then holding $\mathbf{X}'$ fixed while $\boldsymbol{\theta}$ is updated against $\mathcal{L}$ for t_2 steps. This decomposes the problem into two well-conditioned subproblems, keeps the synthetic data aligned with the evolving model, and converges in few epochs given the compact size of $\mathbf{X}'$.

The complete procedure is given in Figure 4.2.

Input: dataset $\mathcal{T} = (\mathbf{X}, \mathbf{Y}, \mathbf{S})$; initialized synthetic labels $\mathbf{Y}'$; weights $\lambda_F, \lambda_M, \beta$.

Randomly initialize $\mathbf{X}'$ and fix $\mathbf{Y}'$
for $j = 0, \ldots, J-1$ **do**
  Initialize $\boldsymbol{\theta}^0 \sim Q_{\boldsymbol{\theta}^0}$
  for $t = 0, \ldots, T-1$ **do**
    Compute gradients $D_t^{\mathcal{T}}$ and $D_t^{\mathcal{T}'}$ by Equation 4.8
    Compute $\mathcal{L}_M$ by Equation 4.10, $\mathcal{L}_F$ by Equation 4.11 on $\mathcal{T}$, and $\mathcal{L}$ by Equation 4.12
    if $t \bmod (t_1 + t_2) < t_1$ **then**
      Update $\mathbf{X}'_{t+1} = \mathbf{X}'_t - \eta_1 \nabla_{\mathbf{X}'_t} \mathcal{L}_M$
    else
      Update $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta_2 \nabla_{\boldsymbol{\theta}^t} \mathcal{L}$
    end if
  end for
end for

Output: fair synthetic dataset $\mathcal{T}' = \{\mathbf{X}'_{T-1}, \mathbf{Y}'\}$.

Figure 4.2: The FairData algorithm for learning a fair synthetic dataset. Note that the fairness loss is evaluated on $\mathcal{T}$, where the sensitive attribute is observed, and enters the synthetic data through the trajectory $\{\boldsymbol{\theta}^t\}$ rather than through $\mathbf{X}'$ directly.

4.4.6 Relation to Existing Approaches

Relative to *in-processing* debiasing [73, 143, 11, 95], the framework moves the fairness constraint out of the deployed model. The deployed model is trained by ordinary empirical risk minimization; the constraint was discharged when the data was built. This eliminates the per-architecture repetition described in Section 4.1 and removes the requirement that the sensitive attribute be available to the deployed model.

Relative to *pre-processing* debiasing [72, 17], it replaces heuristic transformations of the input distribution with differentiable optimization against an explicit objective, and, through the matching term, it makes the transformation aware of the downstream task rather than agnostic to it.

Relative to *fair generative* approaches [134, 119], which train generative models under adversarial fairness constraints, it avoids adversarial training altogether. The output is a set of points in the original feature space rather than samples from a learned generator, which preserves interpretability and auditability and sidesteps the instability that adversarial fairness objectives are prone to.

Relative to *dataset condensation* [157, 19, 75, 83, 84] and to Chapter 3 of this dissertation, the difference is the added term in Equation 4.12. That literature optimizes synthetic data for accuracy or efficiency alone and, as shown below, inherits whatever disparity the original data contains.

4.5 Experiments

The experiments address five questions. Does the learned data reduce group disparity without sacrificing accuracy (Section 4.5.2)? Where does it sit on the accuracy-fairness frontier (Section 4.5.3)? Does the encoded fairness transfer across architectures (Section 4.5.4)?

What does it do to the predicted distribution (Section 4.5.5)? And can the trade-off be controlled (Section 4.5.6)?

4.5.1 Setup

Datasets. Four standard fairness benchmarks are used, spanning tabular and image modalities and three sensitive attributes. `Adult` [10], derived from the 1994 U.S. Census, predicts whether income exceeds $50,000 from demographic and employment features. `COMPAS` [82] contains criminal defendant records and predicts two-year recidivism. `ACS-I` [35] is a modern census-based income prediction task. `CelebA-A` [92] consists of celebrity facial images with a visual attribute classification target. Statistics are given in Table 4.1.

Table 4.1: Statistics of the four fairness benchmarks: number of samples, input features, prediction target, sensitive attributes, and modality.

Property	`Adult`	`COMPAS`	`ACS-I`	`CelebA-A`
# Samples	45,222	6,172	195,655	202,599
# Features	101	405	908	48×48
Target	Income	Credit	Income	Attractive
Sensitive	Gender, Race	Gender, Race	Gender, Race	Gender, Age
Modality	Tabular	Tabular	Tabular	Image

Baselines. Seven baselines are considered. `ERM` is ordinary empirical risk minimization with no fairness intervention. `DiffDP` [28] adds a differentiable demographic parity penalty to the training objective. `HSIC` minimizes the Hilbert–Schmidt independence criterion between predictions and sensitive attributes. `PRemover` [73] penalizes the mutual information between predictions and sensitive features. `AdvDebias` [143, 11] trains against a discriminator that attempts to recover the sensitive attribute. `LAFTR` [95] learns fair representations by jointly minimizing classification error, reconstruction error, and adversarial recoverability. `OptimPrep` [17] is a data-centric pre-processing method that learns a transformation of the training set. The first six are model-centric; only the last shares the data-centric premise of this chapter.

Metrics and protocol. Utility is reported as accuracy and AUC; fairness as Δ_{DP} and Δ_{EO} from Equation 4.4 and Equation 4.5. Results are averaged over ten runs with different random seeds, and standard deviations are reported.

Implementation. Tabular datasets use a two-layer multilayer perceptron with 256 hidden units and ReLU activations; the image dataset uses a ResNet-18 backbone pre-trained on ImageNet. All models are trained with Adam [34] at a learning rate of 0.01 for 150 optimization steps, with batch size 4096 for tabular data and 128 for images. Unless otherwise noted the utility-to-fairness ratio is $\alpha = 1.4$ with $\beta = 10^{-3}$ for tabular tasks and $\alpha = 2.5$ with $\beta = 10^{-2}$ for the image task.

4.5.2 Fairness and Utility Against Model-Centric Baselines

Table 4.2 and Table 4.3 report the comparison. Across all four datasets and both sensitive attributes, **the learned data attains the best or near-best fairness while remaining the most accurate of the fairness-aware methods**, and it does so without any modification to the downstream model.

Four observations stand out.

The trade-off is more favourable than for in-processing methods. Methods such as `AdvDebias` and `HSIC` enforce fairness inside the training objective and pay for it in accuracy. On `Adult` with gender as the sensitive attribute, the learned data attains the joint-lowest Δ_{DP} of 0.5 while being the most accurate fairness-aware method at 84.2, against 85.4 for unconstrained `ERM` whose Δ_{DP} is 16.7. Removing more than thirty points of disparity costs slightly over one point of accuracy.

It outperforms the data-centric baseline. `OptimPrep` transforms the input distribution without reference to the downstream task, and its fairness gains are correspondingly modest: on `COMPAS` with race, it leaves Δ_{DP} at 16.4 against 2.3 for the learned data. The difference is attributable to the matching term, which aligns the synthetic distribution with the training dynamics of the actual task rather than with a task-agnostic criterion.

It holds up on high-dimensional data. Fairness methods frequently degrade on images. On `CelebA-A` with gender, `ERM` exhibits an extreme Δ_{EO} of 70.8 and Δ_{DP} of 52.4; the learned data reduces these to 6.3 and 3.7 respectively while attaining the best accuracy among fairness-aware methods.

It is stable. Standard deviations are consistently smaller than those of the adversarial baselines, several of which fluctuate severely; `AdvDebias` on `Adult` with race reports an AUC standard deviation of 7.9 and `LAFTR` of 7.4, against 0.7 for the learned data. Adversarial fairness objectives are known to be difficult to optimize, and the framework here avoids them entirely.

4.5.3 The Accuracy-Fairness Frontier

Single operating points can mislead, since any method can be tuned toward either objective. Figure 4.3 therefore sweeps the controlling hyperparameter of each method and plots the resulting accuracy against Δ_{DP}, following the protocol of [88]; points nearer the upper-left corner are preferable.

The learned data lies on or near the Pareto frontier on all four datasets. The explanation is structural rather than incidental: the two terms of Equation 4.12 act on different aspects of the problem, with the matching term preserving the utility-relevant training dynamics and the fairness term reducing disparity, so improving one does not automatically consume

Table 4.2: Utility and fairness on `Adult` and `COMPAS`, averaged over ten runs. Higher is better for Acc. and AUC; lower is better for Δ_{EO} and Δ_{DP}. Bold marks the best value among the fairness-aware methods; `ERM` is an unconstrained reference and is excluded from that comparison. FairData denotes the framework of this chapter.

Sens.	Metric	ERM	DiffDP	HSIC	AdvDeb	LAFTR	PRemov	OptPrep	FairData
Adult									
Race	Acc.	85.2 ± 0.3	83.1 ± 1.9	84.4 ± 0.5	80.6 ± 4.4	83.0 ± 3.2	79.1 ± 3.5	82.9 ± 2.1	$\mathbf{84.7} \pm 0.8$
	AUC	91.1 ± 0.3	88.5 ± 1.1	89.9 ± 0.7	88.3 ± 7.9	88.5 ± 7.4	87.1 ± 0.9	89.6 ± 0.1	$\mathbf{90.1} \pm 0.7$
	Δ_{EO}	13.1 ± 2.7	3.4 ± 2.3	3.9 ± 3.1	5.6 ± 5.8	11.6 ± 7.7	2.5 ± 2.4	10.9 ± 2.3	$\mathbf{2.4} \pm 1.2$
	Δ_{DP}	12.2 ± 0.7	5.3 ± 2.2	6.5 ± 2.0	5.5 ± 4.9	9.3 ± 4.0	2.7 ± 2.5	8.6 ± 1.2	$\mathbf{2.1} \pm 1.0$
Gender	Acc.	85.4 ± 0.3	82.7 ± 1.1	82.4 ± 0.7	82.5 ± 2.3	83.8 ± 2.2	77.5 ± 3.0	83.1 ± 0.7	$\mathbf{84.2} \pm 0.9$
	AUC	90.8 ± 0.2	88.0 ± 1.5	86.6 ± 1.0	86.1 ± 8.0	90.4 ± 0.8	82.8 ± 2.6	82.8 ± 2.6	$\mathbf{90.6} \pm 0.6$
	Δ_{EO}	14.2 ± 3.2	36.0 ± 10.7	33.6 ± 5.9	20.4 ± 19.4	11.3 ± 9.0	10.0 ± 14.6	13.0 ± 3.5	$\mathbf{5.3} \pm 2.1$
	Δ_{DP}	16.7 ± 0.7	3.2 ± 3.0	0.9 ± 0.7	14.1 ± 8.5	11.5 ± 4.4	$\mathbf{0.5} \pm 0.6$	12.5 ± 5.8	$\mathbf{0.5} \pm 1.1$
COMPAS									
Race	Acc.	73.0 ± 0.6	59.8 ± 5.3	57.1 ± 3.6	65.4 ± 2.8	66.0 ± 2.0	66.1 ± 1.8	66.5 ± 2.1	$\mathbf{67.3} \pm 2.5$
	AUC	72.6 ± 0.7	68.2 ± 5.3	69.4 ± 2.7	69.5 ± 7.3	71.9 ± 1.4	72.4 ± 1.0	71.3 ± 1.9	$\mathbf{72.8} \pm 1.5$
	Δ_{EO}	29.0 ± 6.7	11.8 ± 12.3	6.1 ± 8.0	27.0 ± 8.4	26.3 ± 7.1	25.2 ± 5.9	25.6 ± 2.4	$\mathbf{4.2} \pm 4.7$
	Δ_{DP}	16.8 ± 3.5	6.6 ± 6.8	3.2 ± 4.3	15.5 ± 4.3	14.9 ± 3.8	14.4 ± 3.1	16.4 ± 5.3	$\mathbf{2.3} \pm 3.5$
Gender	Acc.	67.1 ± 0.8	61.4 ± 4.2	57.3 ± 2.0	63.3 ± 4.9	63.9 ± 4.6	65.5 ± 2.2	65.2 ± 2.7	$\mathbf{66.6} \pm 2.8$
	AUC	72.1 ± 0.9	68.8 ± 4.5	70.3 ± 2.4	68.8 ± 6.5	70.3 ± 3.8	71.5 ± 0.7	70.5 ± 2.3	$\mathbf{71.8} \pm 1.2$
	Δ_{EO}	19.7 ± 6.0	9.8 ± 8.5	12.1 ± 6.8	16.3 ± 9.2	16.9 ± 9.3	18.6 ± 4.7	17.5 ± 5.5	$\mathbf{5.2} \pm 3.4$
	Δ_{DP}	13.4 ± 2.5	6.1 ± 5.5	7.6 ± 4.3	10.6 ± 5.5	10.7 ± 5.4	12.1 ± 2.8	12.4 ± 4.3	$\mathbf{3.3} \pm 1.0$

Table 4.3: Utility and fairness on `ACS-I` and `CelebA-A`, averaged over ten runs. Higher is better for Acc. and AUC; lower is better for Δ_{EO} and Δ_{DP}. Bold marks the best value among the fairness-aware methods; `ERM` is an unconstrained reference and is excluded from that comparison. `OptPrep` is the data-centric pre-processing baseline.

Sens.	Metric	ERM	DiffDP	HSIC	AdvDeb	LAFTR	PRemov	OptPrep	FairData
ACS-I									
Race	Acc.	90.4 ± 0.1	80.2 ± 1.2	81.2 ± 0.4	80.0 ± 6.9	82.1 ± 0.4	74.2 ± 7.3	82.1 ± 2.0	$\mathbf{83.0} \pm 3.7$
	AUC	90.2 ± 0.1	89.5 ± 0.4	89.5 ± 0.3	88.0 ± 6.7	$\mathbf{90.1} \pm 0.4$	84.8 ± 2.1	88.6 ± 1.8	89.7 ± 2.1
	Δ_{EO}	9.7 ± 0.7	6.3 ± 1.9	7.5 ± 2.3	9.5 ± 6.0	7.3 ± 1.5	7.1 ± 4.8	8.3 ± 1.7	$\mathbf{3.8} \pm 2.3$
	Δ_{DP}	9.8 ± 0.4	2.0 ± 1.5	1.3 ± 1.1	9.0 ± 3.6	8.6 ± 0.8	1.0 ± 1.4	8.2 ± 2.2	$\mathbf{0.8} \pm 1.0$
Gender	Acc.	82.3 ± 0.1	81.2 ± 0.3	76.3 ± 2.0	78.8 ± 9.3	$\mathbf{82.1} \pm 0.3$	73.6 ± 8.6	82.0 ± 2.4	82.0 ± 0.7
	AUC	90.1 ± 0.1	89.1 ± 0.5	89.3 ± 0.3	68.6 ± 9.2	$\mathbf{90.1} \pm 0.3$	85.4 ± 2.6	88.4 ± 1.6	89.8 ± 0.4
	Δ_{EO}	6.2 ± 0.6	12.8 ± 1.5	13.6 ± 1.2	6.6 ± 6.7	2.1 ± 0.9	9.1 ± 5.7	6.3 ± 3.2	$\mathbf{1.6} \pm 0.8$
	Δ_{DP}	9.1 ± 0.3	1.2 ± 0.9	0.9 ± 0.7	5.7 ± 4.2	7.8 ± 1.0	$\mathbf{0.6} \pm 0.7$	7.4 ± 3.9	$\mathbf{0.6} \pm 2.2$
CelebA-A									
Age	Acc.	78.2 ± 0.4	67.3 ± 9.1	64.6 ± 4.5	66.5 ± 7.8	68.4 ± 6.1	63.2 ± 5.6	68.3 ± 2.5	$\mathbf{69.1} \pm 3.7$
	AUC	86.7 ± 0.5	74.7 ± 8.6	71.2 ± 4.6	75.3 ± 0.9	76.3 ± 2.7	69.9 ± 5.8	76.1 ± 1.8	$\mathbf{77.3} \pm 2.1$
	Δ_{EO}	37.4 ± 2.3	13.7 ± 11.8	15.2 ± 6.4	27.4 ± 11.2	10.9 ± 6.2	12.7 ± 6.5	$\mathbf{4.3} \pm 2.7$	5.8 ± 4.3
	Δ_{DP}	41.9 ± 1.0	23.7 ± 15.4	4.9 ± 3.7	11.2 ± 21.4	7.8 ± 7.3	5.0 ± 4.0	20.5 ± 13.6	$\mathbf{2.5} \pm 3.8$
Gender	Acc.	78.2 ± 0.4	66.5 ± 9.2	59.2 ± 5.1	67.2 ± 1.9	67.9 ± 10.3	59.2 ± 5.2	67.0 ± 1.7	$\mathbf{68.1} \pm 2.6$
	AUC	86.7 ± 0.5	73.6 ± 7.5	64.3 ± 5.3	74.2 ± 2.1	73.2 ± 5.2	64.5 ± 5.5	74.3 ± 1.1	$\mathbf{75.6} \pm 3.6$
	Δ_{EO}	70.8 ± 3.2	20.1 ± 13.3	9.7 ± 6.0	17.9 ± 11.3	14.3 ± 8.2	8.0 ± 5.4	21.5 ± 7.4	$\mathbf{6.3} \pm 7.9$
	Δ_{DP}	52.4 ± 1.3	24.2 ± 7.3	4.9 ± 3.2	25.6 ± 19.3	8.9 ± 5.6	6.1 ± 3.7	26.9 ± 6.3	$\mathbf{3.7} \pm 2.4$

the other. In-processing baselines such as `HSIC` and `LAFTR` enforce both through the same set of model parameters and consequently trade one directly against the other. `OptimPrep` is dominated on all four datasets, being agnostic to the downstream task, and `AdvDebias` shows large fluctuations on `Adult` and `COMPAS` consistent with the optimization instability noted above.

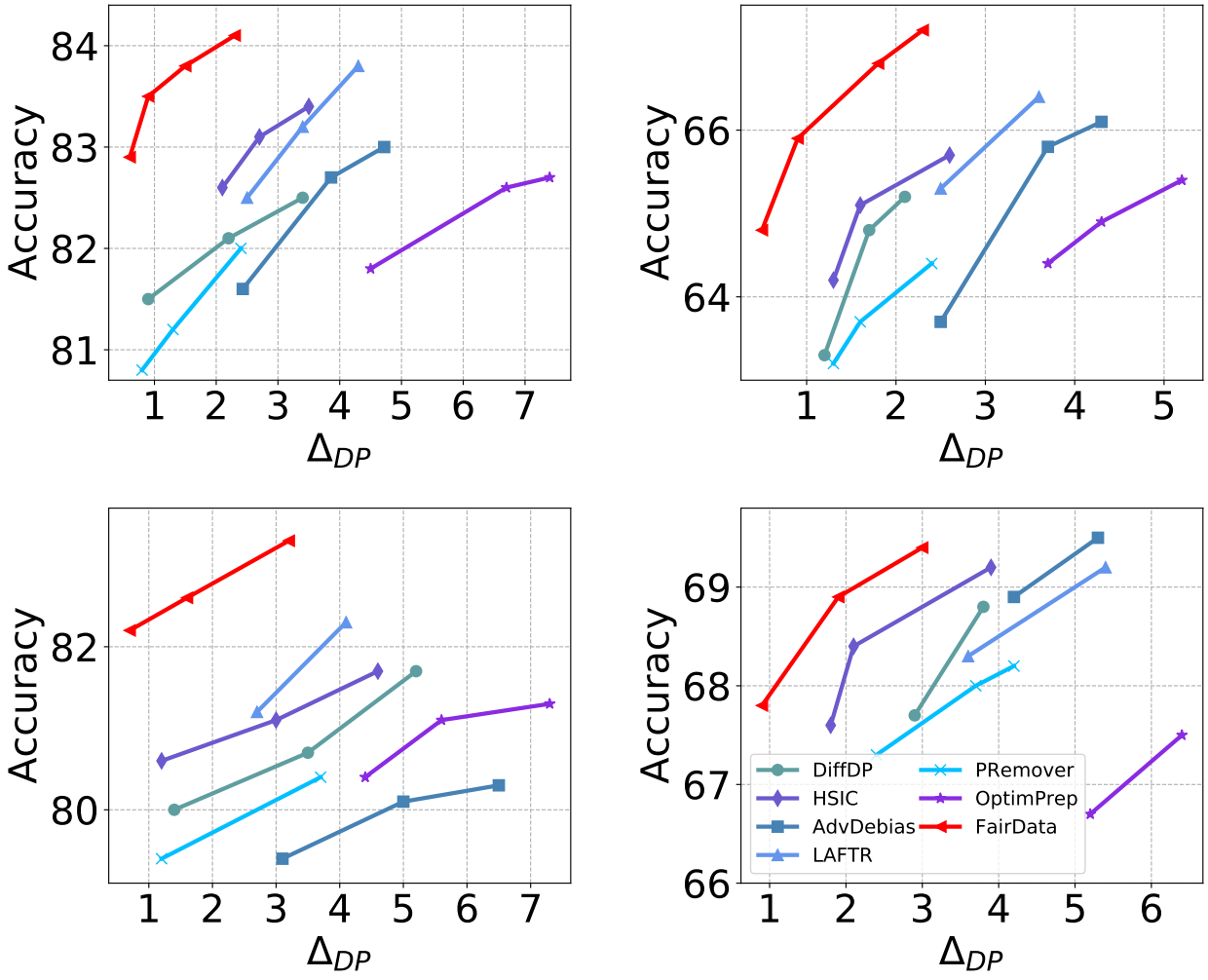

Figure 4.3: Accuracy against Δ_{DP} on `Adult` and `COMPAS` (top row) and `ACS-I` and `CelebA-A` (bottom row). Points nearer the upper-left corner are preferable. The learned data consistently achieves a better frontier than both the model-centric and the pre-processing baselines.

4.5.4 Transferability Across Architectures

This experiment tests the central claim of the chapter: that the intervention need be performed only once. Fair data is generated using one architecture and then consumed by a different one for downstream training, simulating deployments in which the model that produced the data is not the model that will use it. Two settings are considered: `CelebA-A` with age as the sensitive attribute, evaluated over five vision backbones, and `Adult` with race, evaluated over five tabular learners including the non-differentiable XGBoost [26], alongside TabNet [4] and FT-Transformer [47]. Model capacities are normalized so that architecture rather than size is the variable under study.

Table 4.4 and Table 4.5 report accuracy and Δ_{DP} for every generator-consumer pair, with diagonal entries corresponding to the matched case.

Utility transfers. On `CelebA-A`, most off-diagonal entries fall within 1.5 accuracy points of the corresponding diagonal: data generated by ConvNet reaches 68.9 on ResNet and 66.9 on AlexNet against its own 68.5. On `Adult`, data generated by TabNet yields 84.5 with XGBoost against its own 87.3. The learned data is therefore not over-fitted to the inductive bias of its generator.

Fairness transfers. The same holds for disparity, which is the less obvious result. On `CelebA-A`, data generated by ResNet attains $\Delta_{DP} = 4.6$ when consumed by AlexNet, better than the 7.1 obtained when MLP-generated data is consumed by ResNet. On `Adult`, every consumer of MLP-Deep-generated data attains $\Delta_{DP} < 3$. Notably, fairness survives transfer to XGBoost, a gradient-boosted tree ensemble to which differentiable fairness regularizers cannot be applied at all. This is the strongest available evidence that the fairness has been encoded in the data rather than in a model.

Table 4.4: Cross-architecture transfer on `CelebA-A` (age). Rows give the consumer architecture, columns the generator architecture used to produce the fair data. Diagonal entries (bold) are the matched case. Accuracy is higher-is-better; Δ_{DP} is lower-is-better.

Consumer	Accuracy by generator					Δ_{DP} by generator				
	MLP	Conv	ResNet	AlexNet	VGG	MLP	Conv	ResNet	AlexNet	VGG
MLP	**66.4**	67.1	66.3	66.1	65.6	**7.4**	9.2	13.2	11.5	8.4
Conv	64.5	**68.5**	68.9	66.9	66.4	8.2	**5.3**	6.6	12.4	6.5
ResNet	63.5	66.2	**69.1**	67.6	67.2	7.1	8.6	**3.8**	4.6	7.2
AlexNet	65.8	67.3	67.0	**67.4**	66.2	11.2	12.7	10.5	**6.7**	9.9
VGG	60.5	68.6	67.5	66.2	**67.7**	10.3	14.9	12.3	8.5	**5.7**

Table 4.5: Cross-architecture transfer on `Adult` (race). Rows give the consumer learner, columns the generator used to produce the fair data. Diagonal entries (bold) are the matched case. FT-T denotes FT-Transformer and MLP-D a three-layer MLP with 256 units.

Consumer	Accuracy by generator					Δ_{DP} by generator				
	MLP	TabNet	FT-T	XGB	MLP-D	MLP	TabNet	FT-T	XGB	MLP-D
MLP	**88.4**	83.3	82.6	84.0	85.2	**1.6**	1.9	2.7	2.2	2.4
TabNet	84.6	**87.3**	83.7	84.5	83.2	2.2	**2.4**	2.5	2.0	2.9
FT-T	82.9	83.5	**87.1**	83.4	82.8	1.9	2.6	**0.5**	1.8	2.6
XGB	85.3	84.0	83.6	**88.0**	84.4	2.9	2.6	2.4	**1.3**	1.8
MLP-D	86.2	83.9	82.2	84.1	**88.5**	2.1	2.3	3.0	2.7	**0.6**

4.5.5 What the Learned Data Does to the Predicted Distribution

Aggregate gaps summarize disparity but conceal its shape. Figure 4.4 plots the density of predicted probabilities separately for each gender group on `Adult` and `ACS-I`, under three conditions: an unconstrained model trained on the original data, an unconstrained model trained on the learned data, and a fairness-aware model trained on the learned data.

On the original data, the two group densities differ markedly; one group is dispersed across the probability range while the other concentrates near the extremes. Training the *same unconstrained model* on the learned data brings the densities into close alignment, even though that model carries no fairness objective of its own. This is the empirical signature of the mechanism described in Section 4.4.4: the fairness resides in the data, and an ordinary learner inherits it. Adding a fairness-aware objective on top of the learned data produces near-complete alignment, indicating that the two interventions are complementary rather than redundant. Across both datasets the group gap shrinks monotonically along this sequence.

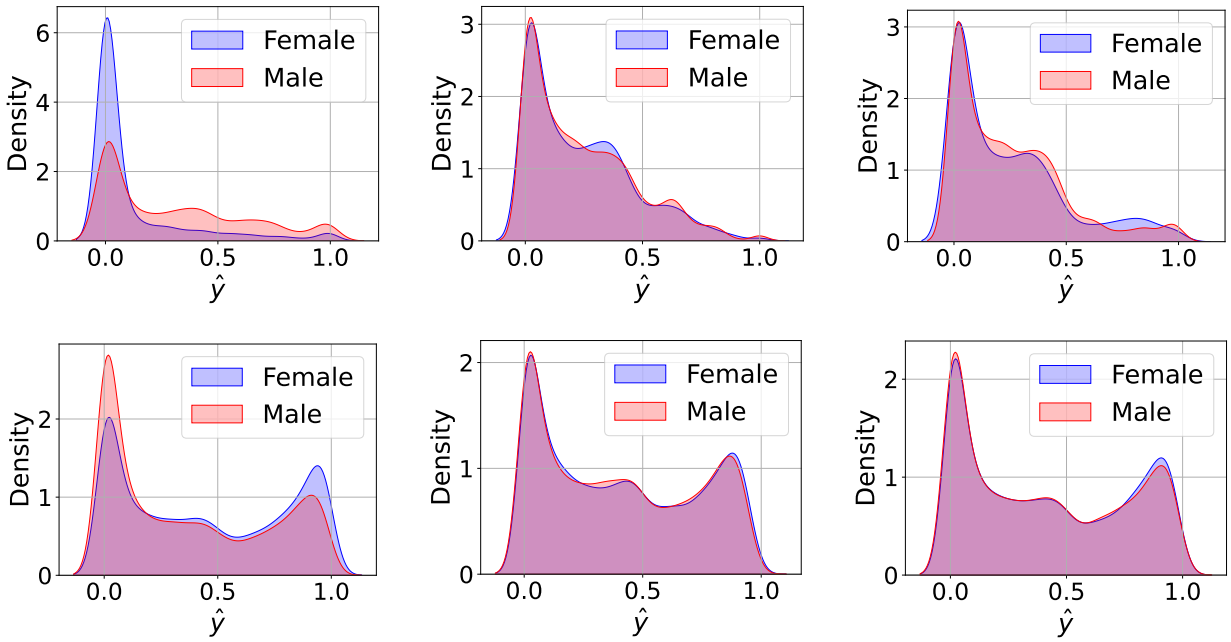

Figure 4.4: Densities of predicted probabilities by gender group on `Adult` (top row) and `ACS-I` (bottom row). Left: unconstrained model on the original data. Centre: the same unconstrained model on the learned data. Right: a fairness-aware model on the learned data. Group alignment improves from left to right, and the centre column shows that alignment is obtained with no fairness objective at training time.

4.5.6 Controlling the Trade-off

Figure 4.5 sweeps the normalized fairness weight $\bar{\alpha}$ of Equation 4.13 over $(0, 1)$. The response is smooth and monotonic in both directions. On `ACS-I`, Δ_{DP} falls from above 4.0 to nearly 0.0 while the error rate rises only from 16.2% to 18.0%. On `CelebA-A`, Δ_{DP} falls from 19.4 to 0.6 while the error rate rises from 15.3% to 24.5%.

Two aspects of this behaviour matter for deployment. First, the curve is comparatively flat near the middle of the range, so substantial fairness gains are available at small accuracy cost and a practitioner need not tune precisely. Second, the two datasets differ in sensitivity: `CelebA-A` shows a steeper response in both directions than `ACS-I`, plausibly because the visual modality carries more latent correlation with the sensitive attribute, so that intervening on it is both more effective and more disruptive. Monotonicity in both cases means the knob behaves predictably, which is what makes the framework tunable rather than merely adjustable.

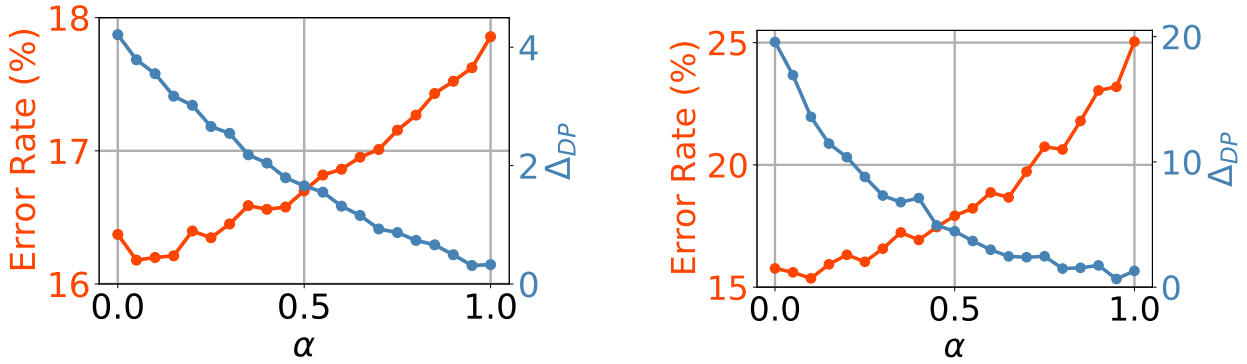

Figure 4.5: Effect of the normalized fairness weight $\bar{\alpha}$ on error rate and Δ_{DP}, on `ACS-I` (left) and `CelebA-A` (right), both with gender as the sensitive attribute. Increasing $\bar{\alpha}$ reduces disparity monotonically at a modest and predictable cost in accuracy.

4.6 Discussion

The results support the claim that group fairness can be optimized into a dataset and that it survives the transition to a different model. Two matters should be recorded before moving on, one about what the framework assumes and one about what it leaves unexamined.

What is assumed. The fairness term is evaluated on the original data, so the sensitive attribute must be observed *during data construction*, even though it is required nowhere afterwards. This is a substantially weaker requirement than that imposed by in-processing methods, which need the attribute throughout training and sometimes at inference, but it is not nothing: settings in which the sensitive attribute is entirely unavailable are out of scope. Separately, the framework inherits the standard limitation of any single-criterion method. Equation 4.11 is a surrogate for demographic parity, and Definition 4.1 and Definition 4.2 are in general incompatible; that Δ_{EO} also improves in Table 4.2 is an empirical finding about these datasets, not a guarantee.

What is unexamined. The more consequential gap concerns the fairness regularizer itself. Section 4.4.2 treated $\mathcal{L}_F$ as an interchangeable component: any differentiable group fairness surrogate may be substituted, and the framework was presented as agnostic to which one. That modularity was offered as a strength, and for the purpose of building the framework it is. But it means that the quality of the result rests entirely on a component that was selected by convention rather than by analysis. The demographic parity surrogate of Equation 4.11 was adopted because it is standard, not because it was shown to have the properties one would want.

This is not a minor omission, because the choice is load-bearing in three ways. It determines the shape of the loss landscape that the alternating optimization of Equation 4.14 must navigate, and hence the stability observed in Section 4.5.2. It determines how tightly reducing

the surrogate bounds the metric one actually cares about. And, since only one criterion can be encoded at a time, it determines which notion of fairness the resulting dataset embodies. The existing literature offers a heterogeneous collection of candidates built on mean disparities, kernel dependence measures, and information-theoretic divergences, adopted largely by convention and with little basis for comparison between them.

The next chapter takes up exactly this question. Rather than proposing another framework in which a regularizer is a plug-in, it asks what properties a fairness regularizer ought to have, organizes the existing candidates so that those properties can be compared, and derives a replacement that satisfies them.

4.7 Summary

This chapter carried the data-centric mechanism of Chapter 3 across from efficiency to trustworthiness. It showed that group fairness, ordinarily pursued by modifying models, can instead be optimized into a training dataset by combining gradient matching, which preserves the utility-relevant training dynamics, with a differentiable fairness regularizer, which reshapes those dynamics before they are matched. The resulting dataset is model-agnostic, remains in the original input space, carries no sensitive attribute, and requires no change to any downstream learner.

Empirically, the approach reduces group disparity across three tabular benchmarks and one image benchmark, over gender, race, and age, while remaining the most accurate of the fairness-aware methods compared. It sits on or near the accuracy-fairness Pareto frontier in every case. Most importantly for the argument of this dissertation, the fairness it encodes transfers: a dataset generated with one architecture retains its fairness when consumed by another, including by a gradient-boosted tree ensemble that no differentiable fairness regularizer could have been applied to. The intervention is genuinely performed once.

Two chapters have now used the same lever for two different ends. What both have in common is that the property demanded of the training dynamics was supplied from outside: in Chapter 3 by the classification loss, and here by a fairness regularizer taken off the shelf. The next chapter turns from the lever to that supplied component, and asks what makes a fairness regularizer a good one.

Chapter 5

What Makes a Good Fairness Regularizer? A Cauchy–Schwarz Answer

5.1 Overview

Chapter 4 closed on an admission. The framework developed there encoded group fairness into a training dataset, and it did so by combining a gradient matching term with a differentiable fairness regularizer. The matching term was derived, analysed, and justified. The regularizer was not: it was described as an interchangeable plug-in, instantiated with a demographic parity surrogate because that surrogate is standard, and never examined. Since the quality of the resulting dataset depends entirely on which trajectory the regularizer produces, this leaves the most consequential component of the framework resting on convention.

This chapter takes up that question directly. In the vocabulary of this dissertation it moves from the first control point to the second: Chapter 3 and Chapter 4 intervened on the *data*, holding the objective fixed; this chapter intervenes on the *objective* itself, asking what a

fairness regularizer must do in order to be worth plugging in anywhere.

The state of the question. Group fairness is most commonly enforced in-process, by adding a penalty term to the training objective. The literature offers a heterogeneous collection of such penalties. Some directly regularize the empirical gap corresponding to a chosen fairness notion; others align the latent representations of different groups using a kernel two-sample statistic; others minimize an information-theoretic measure of dependence between predictions and the sensitive attribute. These constructions are built on different mathematical objects, among them mean disparities, kernel embeddings, and mutual information, each adopted largely because it was available and plausible rather than because it was shown to be the right choice. The consequence is that their behaviour is difficult to reason about and inconsistent across tasks, and that there is no principled basis for preferring one over another.

Two specific failures follow from this state of affairs, and both are documented empirically in Section 5.3.

The first concerns *generalizability across fairness notions*. A regularizer built from one notion need not improve any other, and can actively damage it. On `Adult` with gender as the sensitive attribute, every one of the four standard regularizers considered in this chapter reduces the demographic parity gap yet leaves the equal opportunity gap *worse* than an entirely unconstrained model. Reducing a chosen surrogate is not the same as becoming fair.

The second concerns *robustness*. Because a fairness penalty is one term in a jointly optimized objective, its landscape determines how stably the trade-off can be navigated. A penalty whose value moves sharply under small perturbations of the model parameters yields fairness that is achieved at one point in parameter space and lost at a nearby one, which shows up

in practice as sensitivity to hyperparameters, seeds, and stopping time.

Approach. Rather than proposing a further regularizer and comparing it empirically against the others, this chapter first organizes the field. Existing in-process methods are placed into three families according to what they equalize: prediction statistics, latent representations, or the dependence between predictions and the sensitive attribute. This organization makes visible that all three ultimately reduce to a choice of distance measure between group-conditional distributions. That reduction is what turns an unstructured menu of methods into a well-posed question: which distance measure should one use? Stating the desiderata for that measure, we arrive at three requirements: a tight bound on the fairness quantity one actually cares about, insensitivity to differences of scale between the compared distributions, and applicability to arbitrary prediction distributions without parametric assumptions.

The Cauchy–Schwarz (CS) divergence satisfies all three. Under a Gaussian comparison it is provably no larger than the Kullback–Leibler divergence in either direction; it compares kernel mean embeddings by a cosine-type rather than a Euclidean distance, which makes it insensitive to the scale artefacts that inflate the maximum mean discrepancy and the mean disparity used in demographic parity penalties; and it admits a nonparametric kernel estimator that assumes nothing about the distributions being compared.

Contributions of this chapter. The chapter makes the following contributions.

- It organizes existing in-process fair learning methods into three families and shows that they share a common structure, reducing the design of a fairness regularizer to the choice of a distance measure between group-conditional distributions.
- It identifies the properties such a measure should have, and supports each with evidence

of what goes wrong when it is absent: failure to generalize across fairness notions, and a loss landscape that makes fairness fragile under parameter perturbation.

- It proposes a fairness regularizer based on the empirical Cauchy–Schwarz divergence between group-conditional prediction distributions. The regularizer is model-agnostic, kernel-based, distribution-free, and extends to multiple sensitive attributes with no algorithmic change.

- It establishes that under a Gaussian comparison the CS divergence is upper-bounded by the KL divergence in both directions, and characterizes its relationship to the maximum mean discrepancy and to the mean disparity of demographic parity penalties, clarifying when and why it should be preferred.

- It evaluates the regularizer on four tabular datasets and one image dataset across two sensitive attributes each, showing that it attains the best equal opportunity gap in nine of ten settings while remaining competitive on demographic parity and utility, and that it yields a more stable utility–fairness trade-off across hyperparameter settings.

Section 5.2 fixes notation and introduces the CS divergence. Section 5.3 develops the taxonomy and the desiderata. Section 5.4 constructs the regularizer and establishes its properties. Section 5.5 reports the empirical study, and Section 5.7 closes the chapter.

5.2 Preliminaries

5.2.1 Setting and Notation

This chapter specializes the setting of Section 4.2 to binary classification with a binary sensitive attribute, which is the regime in which the fairness regularizers under comparison

are defined. The dataset is

$$\mathcal{T} = \{(\mathbf{x}_i, y_i, s_i)\}_{i=1}^{N}, \qquad \mathbf{x}_i \in \mathbb{R}^D, \quad y_i \in \{0,1\}, \quad s_i \in \{0,1\}, \tag{5.1}$$

so that $C = 2$ and $K = 2$ in the notation of Equation 4.1. Here $\mathbf{x}_i$ collects the features excluding the sensitive attribute. The model produces a predicted probability

$$z_i = f_{\boldsymbol{\theta}}(\mathbf{x}_i, s_i) \in [0,1], \qquad \hat{y}_i = \mathbf{1}_{\{z_i \geq t\}}, \tag{5.2}$$

where t is a decision threshold and $\mathbf{1}_{\{\cdot\}}$ the indicator function. Random variables corresponding to $\mathbf{x}_i$, y_i, s_i, and $\hat{y}_i$ are written X, Y, S, and $\hat{Y}$. The model $f_{\boldsymbol{\theta}}$ is a multilayer perceptron for tabular data and a residual network for image data.

A note on symbols. The italic D continues to denote the feature dimension, as in Chapter 3 and Chapter 4. A generic distance or divergence between distributions is written in upright type as $\mathrm{D}(\cdot\,;\cdot)$, with subscripts D_{CS}, D_{KL}, $\mathrm{D}_{\mathrm{MMD}}$, and $\mathrm{D}_{\mathrm{HSIC}}$ identifying particular choices.

5.2.2 The In-Process Fairness Objective

Every method compared in this chapter instantiates the same template. A utility loss is augmented by a fairness penalty, and a single coefficient governs the balance:

$$\min_{\boldsymbol{\theta}} \;\; \mathcal{L}_{\text{utility}} \;+\; \lambda\, \mathcal{L}_{\text{fairness}}, \tag{5.3}$$

where $\mathcal{L}_{\text{utility}}$ is the binary cross-entropy loss and $\lambda > 0$ controls the trade-off. What distinguishes the methods is entirely the construction of $\mathcal{L}_{\text{fairness}}$. This is the term the present chapter is about, and it is the same term that Chapter 4 inserted, unexamined, into its

dataset-learning objective.

5.2.3 Group Fairness Gaps in the Binary Setting

Definition 4.1 and Definition 4.2 defined demographic parity and equal opportunity in general form. In the binary setting of Equation 5.1 the corresponding gaps take the familiar two-group form

$$\Delta_{DP} = \left| P(\hat{Y} \mid S = 0) - P(\hat{Y} \mid S = 1) \right|, \tag{5.4}$$

$$\Delta_{EO} = \left| P(\hat{Y} \mid Y = 1, S = 0) - P(\hat{Y} \mid Y = 1, S = 1) \right|, \tag{5.5}$$

with lower values indicating a fairer classifier. When predictions are binarized these become $\Delta_{DP} = |P(\hat{Y} = 1 \mid S = 0) - P(\hat{Y} = 1 \mid S = 1)|$ and correspondingly for Δ_{EO}, which is the convention adopted in the experimental tables of this chapter.

It is worth recording the relationship to the general definitions used in Chapter 4, since the two are reported side by side across the dissertation. Writing $\pi_k = P(S = k)$ and specializing Equation 4.4 to $C = K = 2$, the group-0 term evaluates to $\pi_1|p-q|$ and the group-1 term to $\pi_0|p-q|$, where $p = P(\hat{Y} = 1 \mid S = 0)$ and $q = P(\hat{Y} = 1 \mid S = 1)$. Their average is $\frac{1}{2}|p-q|$. The two conventions therefore differ by exactly a factor of two and induce the same ordering over models; this chapter uses Equation 5.4, which is standard in the fair machine learning literature.

Note also that Δ_{DP} and Δ_{EO} are computed from hard predictions and are not differentiable, so they serve as evaluation metrics only. The distinction between a metric and its differentiable surrogate, drawn in Section 4.2.4, is central to this chapter: the question it asks is precisely which surrogate to optimize when the metrics are what one cares about.

5.2.4 The Cauchy–Schwarz Divergence

The Cauchy–Schwarz inequality for square-integrable functions states that $\left(\int p(\mathbf{x})q(\mathbf{x})\,d\mathbf{x}\right)^2 \leq \int p(\mathbf{x})^2\,d\mathbf{x} \int q(\mathbf{x})^2\,d\mathbf{x}$, with equality if and only if p and q are linearly dependent. The gap in this inequality therefore measures how far apart two densities are, and taking its logarithm gives the CS divergence [107, 141]:

$$\mathrm{D}_{\mathrm{CS}}(p;q) = -\log\left(\frac{\left(\int p(\mathbf{x})q(\mathbf{x})\,d\mathbf{x}\right)^2}{\int p(\mathbf{x})^2\,d\mathbf{x} \int q(\mathbf{x})^2\,d\mathbf{x}}\right). \tag{5.6}$$

The CS divergence is symmetric in its arguments, satisfies $0 \leq \mathrm{D}_{\mathrm{CS}} < \infty$, and attains its minimum if and only if $p = q$ almost everywhere. Symmetry distinguishes it from the KL divergence, which must be assigned a direction; this matters for fairness, where there is no principled reason to privilege one demographic group as the reference distribution.

5.3 What Makes a Good Fairness Regularizer?

This section organizes existing regularizers, exhibits two ways in which they fail, and extracts from those failures the properties a good regularizer should have.

5.3.1 Three Families of In-Process Regularizer

Family I: balancing predictions across sensitive groups. The most direct construction turns a fairness notion into a penalty by measuring the distance between the group-conditional prediction distributions:

$$\mathcal{L}_{\text{fairness}} = \mathrm{D}(\mathbb{P};\mathbb{Q}), \quad \text{where} \quad \begin{cases} \mathbb{P} = P(\hat{Y} \mid S=0),\ \mathbb{Q} = P(\hat{Y} \mid S=1) & \text{for DP,} \\ \mathbb{P} = P(\hat{Y} \mid Y=1, S=0),\ \mathbb{Q} = P(\hat{Y} \mid Y=1, S=1) & \text{for EO.} \end{cases} \tag{5.7}$$

The overwhelmingly common instantiation takes D to be the absolute difference of means,

$$\mathcal{L}_{\text{fairness}} = |\mathrm{E}(\mathbb{P}) - \mathrm{E}(\mathbb{Q})| = \Big|\mathrm{E}\big(\hat{Y} \mid S = 0\big) - \mathrm{E}\big(\hat{Y} \mid S = 1\big)\Big|, \tag{5.8}$$

which is the differentiable relaxation of Equation 5.4 and is the penalty used by gap-regularization methods [27]. Two observations about Equation 5.8 set up the rest of the chapter. First, nothing forces D to be a mean difference; any distance between distributions can be substituted, and the family is really parameterized by that choice. Second, a vanishing mean gap is a necessary but not a sufficient condition for demographic parity or equal opportunity, since two distributions can share a mean and differ everywhere else. A penalty that sees only the first moment can therefore be driven to zero by a model that is not fair.

Family II: aligning latent representations. A second family operates not on predictions but on the internal representation Z produced by the network, requiring the representations of the two groups to be indistinguishable:

$$\mathcal{L}_{\text{fairness}} = \mathrm{D}\big(Z_{S=0};\, Z_{S=1}\big), \tag{5.9}$$

where $Z_{S=k}$ denotes the representation conditioned on group k. Here D is typically the maximum mean discrepancy [49, 93, 32]. The motivation is that a downstream head cannot exploit information the representation does not carry.

Family III: minimizing prediction–attribute dependence. A third family dispenses with group-conditional distributions and penalizes a dependence measure between the prediction and the sensitive attribute directly:

$$\mathcal{L}_{\text{fairness}} = \mathrm{D}\big(\hat{Y},\, S\big). \tag{5.10}$$

The Hilbert–Schmidt independence criterion [50, 87] and the prejudice index of the prejudice remover [73] belong here; the latter is the mutual information

$$\mathrm{D}_{\mathrm{PR}}(\hat{Y}, S) = \sum_{\hat{y}} \sum_{s} p(\hat{y}, s) \log\left(\frac{p(\hat{y}, s)}{p(\hat{y})\, p(s)} \right). \tag{5.11}$$

Adversarial debiasing [148] also belongs to this family: training a discriminator to recover group membership from the representation, and the encoder to defeat it, is a variational way of minimizing the same dependence.

The common structure. Table 5.1 summarizes the three families. What they share is more important than what separates them. Family I compares two conditional distributions; Family II compares two conditional distributions in a representation space; Family III compares a joint distribution against the product of its marginals, which is again a comparison of two distributions. In every case the method is determined by a distance measure applied to distributions that the model controls. The design of a fairness regularizer is therefore not a choice among three paradigms but a single choice of D, and the question of the chapter title becomes tractable: what should D be?

Table 5.1: Existing in-process fairness regularizers, grouped by family. All three families reduce to the choice of a distance measure D applied to distributions the model controls.

Family	Regularizer	Fairness objective $\mathcal{L}_{\text{fairness}}$
I. Prediction statistics	`DP`	$\left\lvert \mathrm{E}(\hat{Y} \mid S{=}0) - \mathrm{E}(\hat{Y} \mid S{=}1) \right\rvert$
	`EO`	$\left\lvert \mathrm{E}(\hat{Y} \mid Y{=}1, S{=}0) - \mathrm{E}(\hat{Y} \mid Y{=}1, S{=}1) \right\rvert$
II. Representations	`MMD`	$\mathrm{D}_{\mathrm{MMD}}(Z_{S=0};\, Z_{S=1})$
III. Dependence	`HSIC`	$\mathrm{D}_{\mathrm{HSIC}}(\hat{Y},\, S)$
	`PR`	$\mathrm{D}_{\mathrm{PR}}(\hat{Y},\, S)$

5.3.2 Failure Mode One: Optimizing One Notion Does Not Deliver Another

A regularizer built from a particular fairness notion optimizes that notion's surrogate, and the hope is that other notions improve along with it. They need not, and Figure 5.1 shows why.

The figure examines a model trained on `Adult` with gender as the sensitive attribute. Considering all target classes together, the prediction distributions of the two groups are well aligned and the embeddings are thoroughly mixed, so the model appears fair by a demographic parity reading. Restricting attention to the positive class $Y = 1$, which is what equal opportunity requires, the same model separates the groups clearly in both the predictions and the embeddings. The disparity has not been removed; it has been relocated to a conditional slice that the optimized surrogate does not see.

This is not an isolated artefact. In Table 5.3 below, on `Adult` with gender, the unconstrained baseline attains $\Delta_{EO} = 8.43$, and *all four* standard regularizers make it worse: 20.15 for the demographic parity penalty, 17.53 for MMD, 18.47 for HSIC, and 12.45 for the prejudice remover. The same pattern holds on `ACS-I` with gender, where the baseline attains $\Delta_{EO} = 2.13$ and the four regularizers return 5.37, 4.91, 4.95, and 4.54. In both settings every method reduces Δ_{DP} substantially while degrading Δ_{EO} relative to doing nothing at all.

The lesson is that a good regularizer should not merely minimize the surrogate it was built from. It should control the underlying dependence between predictions and the sensitive attribute, so that improvement propagates to fairness notions it was not explicitly told about. We record this as *generalizability across fairness notions*.

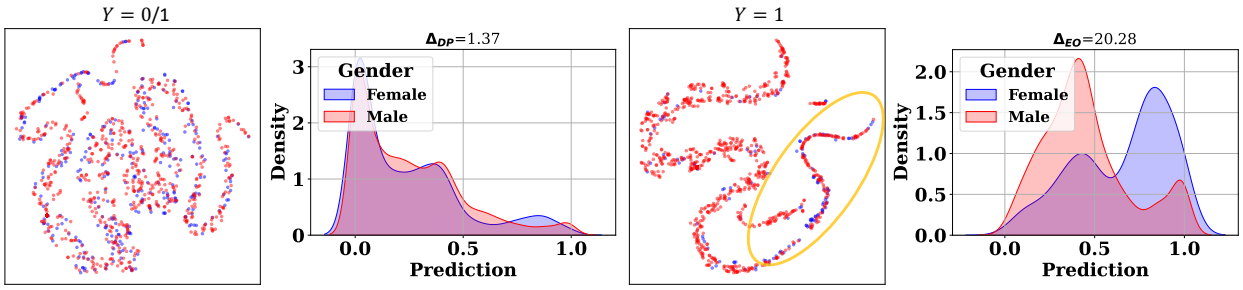

Figure 5.1: A model on `Adult` with gender as the sensitive attribute. From left to right: prediction distribution over all classes; t-SNE of embeddings over all classes; prediction distribution restricted to class 1; t-SNE of embeddings for class 1. Blue denotes $S = 0$ and red $S = 1$. The groups are well mixed in aggregate but clearly separated once conditioned on the positive class, so a model that looks fair by demographic parity is not fair by equal opportunity.

5.3.3 Failure Mode Two: Fairness That Does Not Survive Perturbation

Because Equation 5.3 is optimized jointly, the geometry of $\mathcal{L}_{\text{fairness}}$ around the solution determines how reliably fairness is retained. If the penalty rises steeply for small displacements of $\boldsymbol{\theta}$, then fairness holds at the particular parameter vector reached by one training run and degrades at nearby ones, which manifests as sensitivity to seeds, to the trade-off coefficient λ, and to when training is stopped.

Figure 5.2 visualizes the fairness loss landscape under three measures. The KL divergence and HSIC both exhibit a region of low fairness loss spanning roughly -2 to 2 along the plotted directions, whereas the CS divergence attains a visibly tighter region spanning roughly -2 to 1. The flatter, better-conditioned surface of the CS divergence means that the fairness achieved at the optimum is preserved under small parameter perturbations rather than being an isolated point. We record this as *robustness*, and note that it is what ultimately produces the stable trade-off curves reported in Section 5.5.3 and the hyperparameter insensitivity of Section 5.5.5.

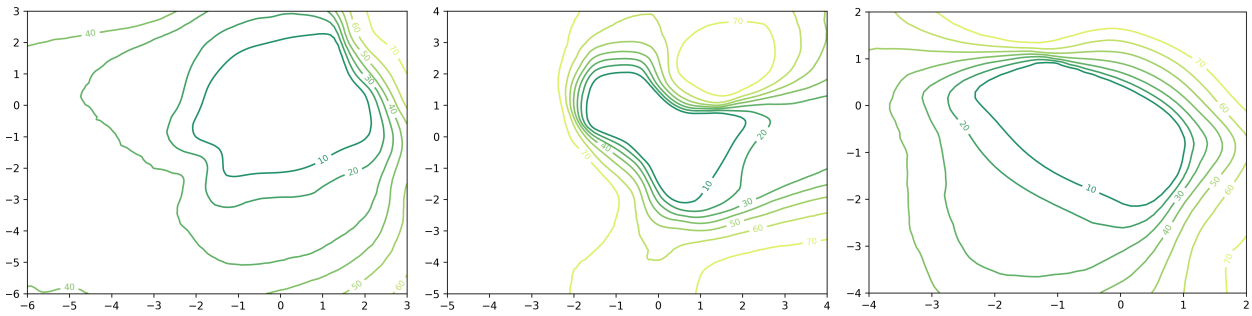

Figure 5.2: Fairness loss landscapes under, from left to right, the Kullback–Leibler divergence, the Hilbert–Schmidt independence criterion, and the Cauchy–Schwarz divergence. A smaller inner region indicates a better-conditioned penalty and hence greater robustness to parameter perturbation. The CS divergence attains the smallest inner region, spanning roughly -2 to 1, against -2 to 2 for the other two.

5.3.4 Desiderata

The two failure modes, together with the structural observation that all three families reduce to a choice of D, yield three requirements on that choice.

1. **A tight bound on the fairness quantity of interest.** Prior work establishes that the fairness violation at test time can be bounded by the penalty minimized at training time. Among measures that admit such a bound, a tighter one gives a stronger guarantee for the same optimization effort, and conversely a loose measure forces the optimizer to over-penalize, buying fairness with more utility than necessary. This is the requirement that motivates the comparison against KL in Section 5.4.4.

2. **Robustness to scale differences.** The measure should report large values when the group-conditional distributions genuinely differ and small values when they do not, without being inflated by differences in the magnitude or variance of the compared quantities. A measure that conflates scale with discrepancy produces conservative penalties, loose bounds, and the fragile landscapes of Section 5.3.3.

3. **Applicability to arbitrary prediction distributions.** Group-conditional prediction distributions in practice are neither Gaussian nor unimodal, so the measure must

be estimable from samples without parametric assumptions, and the same estimator must serve tabular and image models alike.

The remainder of the chapter argues that the CS divergence meets all three, and that the existing measures each fail at least one: the mean disparity of Equation 5.8 sees only the first moment and so fails the first; MMD and the mean disparity are unnormalized and so fail the second; and KL requires a direction and a density ratio, which is awkward for the discrete, possibly non-overlapping supports encountered here.

5.4 The Cauchy–Schwarz Fairness Regularizer

5.4.1 Empirical Estimation

To use Equation 5.6 as a training objective it must be estimated from finite samples without assuming a parametric form. Substituting Parzen-window density estimates [103] for p and q into each of the three integrals in Equation 5.6 and simplifying yields a purely kernel-based expression.

Theorem 5.1 (Empirical CS divergence estimator, cf. [65, 107])**.** *Given observations* $\{\mathbf{x}_i^p\}_{i=1}^{N_1}$ *and* $\{\mathbf{x}_j^q\}_{j=1}^{N_2}$ *drawn from* p *and* q *respectively, the empirical estimator of* $\mathrm{D}_{\mathrm{CS}}(p;q)$ *is*

$$\begin{aligned}\widetilde{\mathrm{D}}_{\mathrm{CS}}(p;q) = \log\left(\frac{1}{N_1^2}\sum_{i,j=1}^{N_1}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^p)\right) + \log\left(\frac{1}{N_2^2}\sum_{i,j=1}^{N_2}\kappa(\mathbf{x}_i^q,\mathbf{x}_j^q)\right) \\ - 2\log\left(\frac{1}{N_1 N_2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^q)\right),\end{aligned} \tag{5.12}$$

where κ *is a positive-definite kernel, taken throughout to be the Gaussian kernel* $\kappa_\sigma(\mathbf{x},\mathbf{x}') = \exp\big(-\|\mathbf{x}-\mathbf{x}'\|_2^2/(2\sigma^2)\big)$.

Each of the three terms in Equation 5.12 is a kernel sum: two within-group sums and one

cross-group sum. The estimator therefore requires only samples, no densities, and satisfies the third desideratum by construction.

5.4.2 The Training Objective

The quantities to be compared are the group-conditional prediction distributions,

$$\mathbb{P} = P(\hat{Y} \mid S = 0), \qquad \mathbb{Q} = P(\hat{Y} \mid S = 1), \tag{5.13}$$

which places the regularizer in Family I of Table 5.1 while, as Section 5.4.3 shows, inheriting the dependence-minimizing character of Family III. Substituting $p = \mathbb{P}$ and $q = \mathbb{Q}$ into Equation 5.12 and inserting the result into the template of Equation 5.3 gives the objective

$$\min_{\boldsymbol{\theta}} \;\; \mathcal{L}_{\mathrm{BCE}} \;+\; \alpha\, \widetilde{\mathrm{D}}_{\mathrm{CS}}\big(\mathbb{P}; \mathbb{Q}\big) \;+\; \frac{\beta}{2}\|\boldsymbol{\theta}\|_2^2, \tag{5.14}$$

where $\mathcal{L}_{\mathrm{BCE}} = \frac{1}{N}\sum_{i=1}^{N} -y_i \log \hat{y}_i$ is the binary cross-entropy loss, α is the fairness weight playing the role of λ in Equation 5.3, and β weights an ℓ_2 penalty. Nothing else about training changes: Equation 5.14 is minimized by ordinary gradient descent, and the regularizer is a drop-in replacement for any of the entries in Table 5.1. In particular it is a drop-in replacement for the $\mathcal{L}_F$ of Equation 4.11.

5.4.3 Relation to Existing Measures

Placing the CS estimator beside the standard alternatives shows precisely where each one differs.

Mean disparity. The demographic parity penalty of Equation 5.8 compares $\mathbb{P}$ and $\mathbb{Q}$ through the absolute difference of their means, which is a Manhattan distance between two

scalars. It is blind to every aspect of the distributions beyond the first moment.

Maximum mean discrepancy. The biased empirical MMD is

$$\widetilde{\mathrm{D}}^2_{\mathrm{MMD}}(p;q) = \frac{1}{N_1^2}\sum_{i,j=1}^{N_1}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^p) + \frac{1}{N_2^2}\sum_{i,j=1}^{N_2}\kappa(\mathbf{x}_i^q,\mathbf{x}_j^q) - \frac{2}{N_1 N_2}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\kappa(\mathbf{x}_i^p,\mathbf{x}_j^q). \quad (5.15)$$

Comparing Equation 5.15 with Equation 5.12 isolates the difference exactly: the two expressions are built from the same three kernel sums, and the CS divergence applies a logarithm to each before combining them. Writing $\boldsymbol{\mu}_p$ and $\boldsymbol{\mu}_q$ for the empirical kernel mean embeddings of the two groups in the reproducing kernel Hilbert space, this logarithmic transformation has a geometric reading: MMD is the squared Euclidean distance $\|\boldsymbol{\mu}_p - \boldsymbol{\mu}_q\|^2$ between the embeddings, whereas the CS divergence is a cosine-type distance, measuring the *angle* between them. This is the source of the scale robustness required by the second desideratum, and Section 5.4.4 draws out its consequence.

Kullback–Leibler divergence. The KL divergence

$$\mathrm{D}_{\mathrm{KL}}(p\|q) = \int p(\mathbf{x}) \log\big(p(\mathbf{x})/q(\mathbf{x})\big)\, d\mathbf{x}$$

underlies mutual-information penalties such as Equation 5.11. It is asymmetric, requires a density ratio, and diverges where the supports fail to overlap.

Hilbert–Schmidt independence criterion. With Gram matrices $K_{ij} = \kappa_X(\mathbf{x}_i,\mathbf{x}_j)$ and $L_{ij} = \kappa_Y(\mathbf{y}_i,\mathbf{y}_j)$ and centering matrix $H = I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$, the standard estimator is $\widetilde{\mathrm{D}}_{\mathrm{HSIC}}(X,Y) = \frac{1}{N^2}\operatorname{tr}(KHLH)$ [51]. HSIC is itself an MMD, taken between the joint distribution and the product of the marginals in a tensor-product RKHS, and so inherits the same unnormalized Euclidean geometry.

5.4.4 Why the Cauchy–Schwarz Divergence

Three properties, corresponding to the three desiderata of Section 5.3.4, justify the choice.

(1) Closed form for Gaussian mixtures. The CS divergence admits a closed-form expression for mixtures of Gaussians [74], a property that has been exploited in deep clustering [122], disentangled representation learning [121], and point-set registration [43]. For fairness this matters because group-conditional prediction distributions are typically multimodal, and a mixture approximation remains analytically tractable rather than requiring a unimodal fiction.

(2) A tighter bound than the KL divergence. The following comparison establishes the first desideratum.

Theorem 5.2 (Gaussian comparison of CS and KL)**.** *For any D-variate Gaussian distributions $p \sim \mathcal{N}(\boldsymbol{\mu}_p, \Sigma_p)$ and $q \sim \mathcal{N}(\boldsymbol{\mu}_q, \Sigma_q)$ with positive definite covariances $\Sigma_p, \Sigma_q \succ 0$,*

$$\mathrm{D_{CS}}(p;q) \le \mathrm{D_{KL}}(p;q) \qquad \textit{and} \qquad \mathrm{D_{CS}}(p;q) \le \mathrm{D_{KL}}(q;p). \tag{5.16}$$

Proof sketch. Substituting the closed form of [74] into Equation 5.6 gives

$$\mathrm{D_{CS}}(p;q) = \tfrac{1}{2}\boldsymbol{\Delta}^\top(\Sigma_p+\Sigma_q)^{-1}\boldsymbol{\Delta} + \tfrac{1}{2}\log\left(\frac{|\Sigma_p+\Sigma_q|}{2^D\sqrt{|\Sigma_p||\Sigma_q|}}\right), \qquad \boldsymbol{\Delta} = \boldsymbol{\mu}_q - \boldsymbol{\mu}_p, \tag{5.17}$$

alongside the standard closed form for $\mathrm{D_{KL}}(p;q)$. Forming $2(\mathrm{D_{CS}} - \mathrm{D_{KL}})$ and separating it into a mean term and a covariance term, the mean term is $\boldsymbol{\Delta}^\top\big[(\Sigma_p+\Sigma_q)^{-1} - \Sigma_q^{-1}\big]\boldsymbol{\Delta} \le 0$, since $\Sigma_p \succ 0$ implies $\Sigma_p + \Sigma_q \succ \Sigma_q$ and hence $(\Sigma_p+\Sigma_q)^{-1} \prec \Sigma_q^{-1}$. The covariance term is non-positive by the arithmetic–geometric mean inequality applied to the eigenvalues of

$\Sigma_q^{-1}\Sigma_p$. Symmetry of D_{CS} gives the second inequality. □

Read as a statement about fairness, Theorem 5.2 says that if the two group-conditional prediction distributions are approximated by Gaussians, then at the *same model parameters* the CS penalty is never larger than either directed KL penalty. A CS-based regularizer therefore retains KL-type control over group discrepancy while being numerically tighter, which is precisely the tight-bound property sought in Section 5.3.4.

Remark on the Gaussian assumption. Theorem 5.2 is a stylized comparison. The Gaussian model is adopted solely so that both divergences have closed forms and their relationship can be made explicit. It is not assumed by the training procedure and is not assumed anywhere in the experiments: the regularizer actually optimized is Equation 5.12, which is nonparametric. The empirical results of Section 5.5, on distributions that are plainly non-Gaussian, are what establish that the insight survives outside the analytical setting.

(3) Tighter than MMD and mean disparity under scale mismatch. By the geometric reading of Section 5.4.3, MMD measures a Euclidean distance between kernel mean embeddings and the demographic parity penalty a Manhattan distance between group means, while the CS divergence measures an angle. When the group-conditional distributions differ substantially in variance or scale, the two unnormalized measures report a large discrepancy even where the groups are well aligned in direction. The optimizer responds by over-penalizing, which costs utility and produces a looser bound on the fairness metric. The normalization implicit in Equation 5.6 removes the scale component and leaves the genuine dependence between predictions and the sensitive attribute, which is what one wants to penalize. This is the second desideratum, and it is also the mechanism behind the landscape difference visible in Figure 5.2.

5.4.5 Practical Properties

Distribution-free applicability. Equation 5.12 depends only on samples from the joint distribution of predictions and sensitive attributes and imposes no parametric form on it. The identical regularizer therefore applies to tabular and image models without modification, which is what makes the single experimental protocol of Section 5.5 possible.

Multiple sensitive attributes. Although developed above for a single binary S, the regularizer extends to a vector $S = (S_1, \ldots, S_K)$ of sensitive attributes in either of two ways: treat S as a joint variable and penalize $\widetilde{\mathrm{D}}_{\mathrm{CS}}\big(P_{\hat{Y}|S};\, P_{\hat{Y}}P_S\big)$, or sum over attributes as $\sum_{k=1}^{K} \widetilde{\mathrm{D}}_{\mathrm{CS}}\big(P_{\hat{Y}|S_k};\, P_{\hat{Y}}P_{S_k}\big)$, according to whether joint or per-attribute control is wanted. Both use exactly the estimator of Equation 5.12; only the definition of S within a mini-batch changes, and no algorithmic modification is required.

Computational cost. Evaluating Equation 5.12 over all N training samples costs $O(N^2)$, since it involves pairwise kernel evaluations. As is standard for kernel-based penalties such as MMD and HSIC, the loss is computed on mini-batches, so the additional cost per optimization step is $O(B^2)$ for batch size $B \ll N$, implemented with vectorized operations that reuse the mini-batches already drawn for the prediction loss. The overhead is comparable to that of the kernel baselines.

Kernel choice. The estimator is instantiated with a Gaussian kernel whose bandwidth is set by the median heuristic, following standard practice for kernel dependence measures [49, 50], and held fixed across datasets and models so that comparisons between regularizers are controlled. The framework is not tied to this choice; Laplacian or polynomial kernels substitute directly [112, 114] without changing the objective or the algorithm.

5.5 Experiments

The evaluation asks whether the properties argued for above translate into behaviour: whether the regularizer improves fairness metrics it was not built from (Section 5.5.2), whether it occupies a better utility–fairness frontier (Section 5.5.3), what it does to the group-conditional distributions (Section 5.5.4), and whether it is stable under hyperparameter variation (Section 5.5.5).

5.5.1 Setup

Datasets. Five datasets are used. The four tabular ones are `Adult` [37], `COMPAS` [82], and the income and travel-time tasks `ACS-I` and `ACS-T` from the American Community Survey [35]. The image dataset is `CelebA-A` [92]. Each is evaluated under two sensitive attributes, giving ten dataset–attribute settings. Statistics appear in Table 5.2. Preprocessing follows established practice [108, 98].

Table 5.2: Dataset statistics. #Feat. is the feature count after preprocessing. The ratios $0:1$ give the proportion between the two categories of the target label and of each sensitive attribute.

Dataset	Task	Sens. S	#Samples	#Feat.	Y 0:1	1st S	2nd S
`Adult`	Income	Gender, Race	45,222	101	1:0.33	1:2.08	1:9.20
`COMPAS`	Credit	Gender, Race	6,172	405	1:0.83	1:4.17	1:0.52
`ACS-I`	Income	Gender, Race	195,665	908	1:0.70	1:0.89	1:1.62
`ACS-T`	Travel time	Gender, Race	172,508	1,567	1:0.94	1:0.89	1:1.61
`CelebA-A`	Attractive	Gender, Young	202,599	48×48	1:0.95	1:0.71	1:3.45

Baselines. Four regularizers spanning the three families of Table 5.1 are compared: `DP`, the gap regularizer for demographic parity [27]; `MMD`, the maximum mean discrepancy between group representations [49, 93, 32, 158]; `HSIC`, the Hilbert–Schmidt independence criterion between predictions and sensitive attributes [50, 87, 5]; and `PR`, the prejudice remover [73]. An unregularized `MLP` (tabular) or `ResNet` (image) serves as the unconstrained reference.

Protocol. Ten random splits are used per dataset, and means and standard deviations are reported. Utility is measured by accuracy and AUC, fairness by Δ_{DP} and Δ_{EO} from Equation 5.4 and Equation 5.5. Hyperparameters are selected by cross-validation on the training set with grid search, sweeping α over $[10^{-6}, 150]$ and β over $[10^{-3}, 10]$.

5.5.2 Utility and Fairness Against Existing Regularizers

Table 5.3 and Table 5.4 report the comparison across all ten settings.

The regularizer generalizes across fairness notions. It attains the best Δ_{EO} in nine of the ten settings, the exception being `ACS-T` with race where HSIC is marginally better (0.43 against 1.38). The margins are frequently large: on `Adult` with gender, Δ_{EO} falls to 2.27 where the best baseline reaches only 12.45 and the unconstrained model attains 8.43; on `COMPAS` with gender it falls to 0.44 against a best baseline of 2.60. This is a direct response to the failure documented in Section 5.3.2: in the two settings where every baseline degraded equal opportunity relative to doing nothing, the CS regularizer improves it, by 73.1% on `Adult` with gender and 57.7% on `ACS-I` with gender. On demographic parity it is best in five of ten settings and close to best elsewhere, which is the expected outcome for a measure that controls the underlying dependence rather than optimizing one gap directly.

The cost in utility is small. Across the four tabular datasets, accuracy relative to the unconstrained model falls by between 0.22% and 2.71%, with a single larger drop of 3.89% on `COMPAS` with gender; AUC changes by between -1.53% and $+0.33\%$, improving on `Adult` with race and `COMPAS` with race. On `CelebA-A` the accuracy cost is far larger, roughly 16%, but this is a property of the setting rather than of the method: every regularizer loses between 14% and 20% accuracy on this dataset, against a baseline whose Δ_{DP} of 51.66 indicates that its accuracy was substantially derived from the sensitive attribute.

The image setting is where the gap is widest. On `CelebA-A` with the 'Young' attribute, Δ_{DP} falls by 96.9% and Δ_{EO} by 98.4% relative to the unconstrained ResNet, and both are the best of any method. On the gender attribute the CS regularizer attains $\Delta_{EO} = 1.53$ where the next best is 3.83.

Baselines behave as the taxonomy predicts. The demographic parity gap regularizer is consistently among the strongest on Δ_{DP} and among the weakest on Δ_{EO}, which is exactly what a Family I method built from one notion should do. HSIC, a dependence measure, is the strongest baseline on Δ_{EO} and the best on utility in several settings, consistent with the argument of Section 5.3.4 that controlling dependence generalizes better than controlling a single gap.

5.5.3 The Utility–Fairness Frontier

A single operating point does not settle the comparison, since any of these methods can be pushed toward fairness by raising λ. Figure 5.3 therefore sweeps the fairness coefficient of each method and plots Δ_{DP} against target accuracy, following the protocol of [137, 32]; the lower-right corner is preferable.

At matched utility, CS is the most effective. Across most accuracy levels the CS regularizer attains the lowest Δ_{DP}, and the margin widens as accuracy increases. This regime is the one that matters in practice: every method can be made fair by abandoning the task objective, but a fairness method is only useful if it remains effective where the model is also accurate.

Degradation is gradual rather than abrupt. Examining the rate at which Δ_{DP} rises with accuracy separates the methods more sharply than any single point. `PR` and `DP` control

Table 5.3: Utility and fairness on `Adult`, `COMPAS`, and `ACS-I`, over 10 splits. Higher is better for accuracy and AUC; lower is better for Δ_{DP} and Δ_{EO}. Bold marks the best value among the four fairness regularizers and `CS`; the unregularized `MLP` is an unconstrained reference and is excluded from that comparison.

Sens.	Method	Utility		Fairness	
		Acc. (%)	AUC (%)	Δ_{DP} (%)	Δ_{EO} (%)
`Adult`					
Gender	`MLP`	85.63 ± 0.34	90.82 ± 0.23	16.52 ± 0.91	8.43 ± 3.20
	`DP`	82.42 ± 0.39	86.91 ± 0.80	1.29 ± 0.95	20.15 ± 1.13
	`MMD`	81.90 ± 0.68	85.27 ± 0.52	2.47 ± 0.52	17.53 ± 1.36
	`HSIC`	82.89 ± 0.23	87.25 ± 0.41	2.66 ± 0.54	18.47 ± 1.22
	`PR`	81.81 ± 0.52	85.38 ± 0.82	$\mathbf{0.71} \pm 0.40$	12.45 ± 2.38
	`CS`	$\mathbf{83.31} \pm 0.47$	$\mathbf{90.15} \pm 0.49$	2.42 ± 0.85	$\mathbf{2.27} \pm 1.04$
Race	`MLP`	84.42 ± 0.31	90.15 ± 0.36	13.47 ± 0.83	9.25 ± 3.86
	`DP`	83.64 ± 0.78	88.45 ± 0.32	2.45 ± 0.67	2.16 ± 1.06
	`MMD`	83.12 ± 0.82	88.36 ± 0.67	2.58 ± 0.75	3.33 ± 0.93
	`HSIC`	$\mathbf{84.98} \pm 0.17$	$\mathbf{90.90} \pm 0.19$	7.90 ± 0.72	2.11 ± 0.18
	`PR`	82.13 ± 1.16	87.44 ± 0.33	$\mathbf{1.53} \pm 0.83$	0.86 ± 0.60
	`CS`	83.53 ± 0.53	90.26 ± 0.47	2.16 ± 0.61	$\mathbf{0.44} \pm 0.12$
`COMPAS`					
Gender	`MLP`	66.85 ± 0.72	72.10 ± 0.94	13.22 ± 3.32	11.41 ± 5.83
	`DP`	64.20 ± 1.58	70.64 ± 1.05	5.78 ± 0.33	6.78 ± 1.61
	`MMD`	64.82 ± 1.62	70.72 ± 0.92	3.09 ± 0.92	3.15 ± 4.37
	`HSIC`	63.17 ± 3.46	71.17 ± 0.84	1.84 ± 0.43	2.60 ± 0.63
	`PR`	$\mathbf{64.95} \pm 0.15$	$\mathbf{72.12} \pm 0.75$	3.85 ± 0.60	3.91 ± 1.02
	`CS`	64.25 ± 0.97	71.53 ± 0.61	$\mathbf{1.30} \pm 0.47$	$\mathbf{0.44} \pm 0.13$
Race	`MLP`	66.99 ± 1.05	72.46 ± 0.88	17.24 ± 4.15	19.44 ± 4.63
	`DP`	64.98 ± 3.72	72.09 ± 1.03	8.70 ± 1.12	7.04 ± 2.13
	`MMD`	64.41 ± 2.04	72.10 ± 1.83	4.42 ± 2.11	5.60 ± 1.25
	`HSIC`	64.52 ± 2.20	72.16 ± 0.94	2.21 ± 0.68	2.72 ± 0.87
	`PR`	$\mathbf{67.22} \pm 0.90$	$\mathbf{72.86} \pm 0.87$	5.60 ± 1.12	6.52 ± 1.30
	`CS`	65.62 ± 1.24	72.70 ± 1.06	$\mathbf{1.79} \pm 0.96$	$\mathbf{1.48} \pm 1.64$
`ACS-I`					
Gender	`MLP`	82.04 ± 0.27	90.16 ± 0.18	10.26 ± 4.68	2.13 ± 3.64
	`DP`	81.32 ± 0.17	89.33 ± 0.15	0.96 ± 0.22	5.37 ± 0.32
	`MMD`	80.93 ± 0.55	88.44 ± 1.71	2.45 ± 0.65	4.91 ± 1.48
	`HSIC`	81.40 ± 0.12	$\mathbf{89.53} \pm 0.10$	1.54 ± 0.18	4.95 ± 0.39
	`PR`	80.03 ± 0.30	88.10 ± 0.26	$\mathbf{0.35} \pm 0.20$	4.54 ± 0.41
	`CS`	$\mathbf{81.86} \pm 0.94$	89.15 ± 0.60	0.77 ± 0.38	$\mathbf{0.90} \pm 0.46$
Race	`MLP`	81.23 ± 0.14	90.16 ± 0.18	10.06 ± 1.84	7.42 ± 0.66
	`DP`	81.25 ± 0.13	89.45 ± 0.11	0.56 ± 0.30	4.53 ± 0.48
	`MMD`	80.22 ± 1.22	88.42 ± 1.63	1.45 ± 0.89	4.01 ± 0.54
	`HSIC`	$\mathbf{81.41} \pm 0.15$	$\mathbf{89.67} \pm 0.12$	1.04 ± 0.53	2.77 ± 0.35
	`PR`	80.27 ± 0.26	88.45 ± 0.21	$\mathbf{0.37} \pm 0.30$	4.25 ± 0.49
	`CS`	80.78 ± 0.85	89.14 ± 0.94	0.81 ± 0.28	$\mathbf{1.35} \pm 0.64$

Table 5.4: Utility and fairness on ACS-T and the image dataset CelebA-A, over 10 splits. Bold marks the best value among the fairness regularizers; the unregularized MLP and ResNet are unconstrained references and are excluded from that comparison.

Sens.	Method	Utility		Fairness	
		Acc. (%)	AUC (%)	Δ_{DP} (%)	Δ_{EO} (%)
ACS-T					
Gender	MLP	66.21 ± 0.95	73.78 ± 0.25	8.32 ± 2.67	5.11 ± 3.55
	DP	65.38 ± 0.29	72.40 ± 0.38	0.29 ± 0.15	1.83 ± 0.26
	MMD	64.48 ± 0.27	72.92 ± 0.31	1.22 ± 0.36	2.11 ± 0.49
	HSIC	$\mathbf{66.01} \pm 0.29$	$\mathbf{73.16} \pm 0.32$	0.98 ± 0.26	1.00 ± 0.28
	PR	62.72 ± 1.01	69.36 ± 0.85	0.78 ± 0.50	1.07 ± 0.36
	CS	65.70 ± 0.42	72.83 ± 0.58	$\mathbf{0.17} \pm 0.08$	$\mathbf{0.75} \pm 0.22$
Race	MLP	66.38 ± 0.42	73.69 ± 0.63	9.28 ± 1.63	6.21 ± 1.63
	DP	64.96 ± 0.23	71.86 ± 0.23	0.82 ± 0.33	1.30 ± 0.26
	MMD	65.71 ± 0.65	70.57 ± 0.52	3.97 ± 0.97	1.55 ± 0.79
	HSIC	$\mathbf{65.81} \pm 0.24$	$\mathbf{72.92} \pm 0.23$	1.75 ± 0.31	$\mathbf{0.43} \pm 0.23$
	PR	64.25 ± 0.87	70.25 ± 0.30	1.56 ± 0.87	1.21 ± 0.74
	CS	65.16 ± 0.45	72.56 ± 0.72	$\mathbf{0.55} \pm 0.19$	1.38 ± 0.46
CelebA-A					
Gender	ResNet	78.14 ± 0.47	86.58 ± 0.55	51.66 ± 0.97	35.67 ± 1.11
	DP	62.42 ± 4.79	66.86 ± 3.19	$\mathbf{0.46} \pm 0.25$	4.84 ± 2.37
	MMD	62.54 ± 4.26	66.47 ± 3.85	1.39 ± 0.64	5.89 ± 3.12
	HSIC	63.39 ± 3.63	69.33 ± 3.25	2.24 ± 0.36	3.83 ± 2.22
	PR	$\mathbf{65.51} \pm 3.52$	$\mathbf{71.70} \pm 2.88$	4.00 ± 0.52	5.05 ± 2.57
	CS	65.05 ± 3.80	71.42 ± 2.46	0.98 ± 0.62	$\mathbf{1.53} \pm 1.05$
Young	ResNet	78.14 ± 0.47	86.67 ± 0.53	41.74 ± 1.17	18.35 ± 1.56
	DP	$\mathbf{66.78} \pm 3.61$	$\mathbf{73.95} \pm 3.44$	2.43 ± 0.83	0.91 ± 1.77
	MMD	65.82 ± 4.87	72.84 ± 3.61	3.49 ± 0.83	1.60 ± 0.71
	HSIC	66.04 ± 3.01	73.08 ± 2.69	1.99 ± 0.55	1.04 ± 0.60
	PR	62.98 ± 4.69	69.63 ± 4.02	1.32 ± 0.49	1.82 ± 0.53
	CS	65.33 ± 4.26	73.15 ± 3.84	$\mathbf{1.28} \pm 0.40$	$\mathbf{0.30} \pm 0.12$

fairness well at low accuracy but lose that control abruptly, at roughly 82.0% accuracy on `Adult`, 63.0% on `COMPAS`, and 81.0% on `ACS-I`. The CS regularizer's corresponding transitions occur at 85.0%, 65.5%, and 81.5%. It is this behaviour, rather than any single reported number, that the flatter landscape of Figure 5.2 predicts, and it is what makes the method tunable in practice.

A fairness regularizer can be worse than none. Along the `COMPAS` curve in Figure 5.3, MMD is driven past $\Delta_{DP} = 14.0$ once the sweep reaches high accuracy, exceeding the 13.22 attained by the unconstrained model in Table 5.3 and undoing the gain visible at its tuned operating point. Fairness penalties are not monotone improvements; when the task objective dominates, a poorly conditioned penalty can leave the model worse than it started. Smaller datasets make this more likely, and `COMPAS`, with 6,172 samples, is where it is most visible.

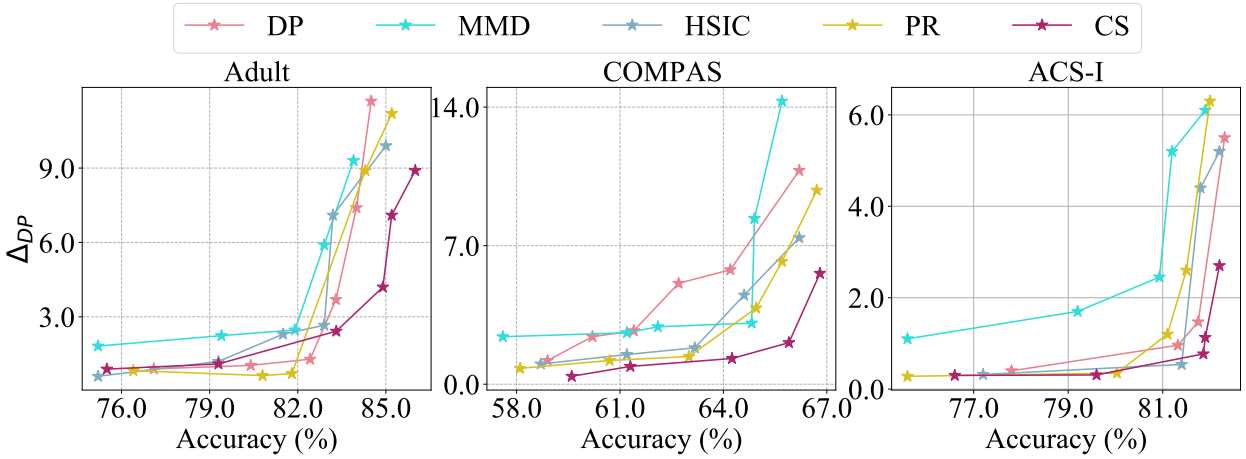

Figure 5.3: Test-set trade-off curves on `Adult` (left), `COMPAS` (middle), and `ACS-I` (right), obtained by sweeping the fairness coefficient of each method. The lower-right corner is preferable. The CS regularizer attains the lowest Δ_{DP} at matched accuracy and loses control of fairness only at higher accuracy levels than the baselines.

5.5.4 Effect on Group-Conditional Distributions

Figure 5.4 plots kernel density estimates of the predictions for the two gender groups on `Adult`, one column per regularizer. The upper row shows predictions over all target classes,

which is what demographic parity constrains; the lower row shows predictions restricted to the positive class, which is what equal opportunity constrains. Greater overlap between the two curves indicates better fairness on the corresponding notion.

The figure makes the argument of Section 5.3.2 concrete, and shows the CS regularizer resolving it. The demographic parity gap regularizer produces near-perfect overlap in the upper row and a visibly larger separation in the lower row: it has optimized exactly what it was told to optimize and nothing more. The CS regularizer narrows the gap in *both* rows simultaneously. Since the two conditions are not simultaneously satisfiable in general, what is being observed is not the elimination of a trade-off but a measure that reduces the underlying dependence rather than one of its projections.

Figure 5.5 shows the corresponding t-SNE embeddings. Under the CS regularizer the two sensitive groups are more thoroughly mixed in representation space, indicating that group membership has become harder to recover from the learned representation. This is the property that Family II and Family III methods pursue directly, obtained here as a consequence rather than as an explicit objective.

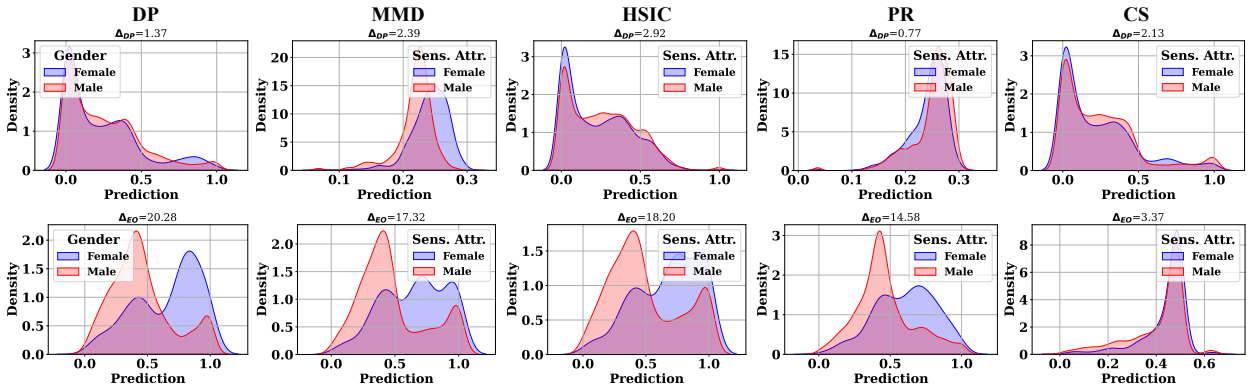

Figure 5.4: Prediction distributions for the two gender groups on `Adult`, one column per regularizer. Top row: predictions over all target classes, corresponding to Δ_{DP}. Bottom row: predictions restricted to the positive class, corresponding to Δ_{EO}. Greater overlap indicates better fairness. The gap regularizer aligns the top row only; the CS regularizer narrows both.

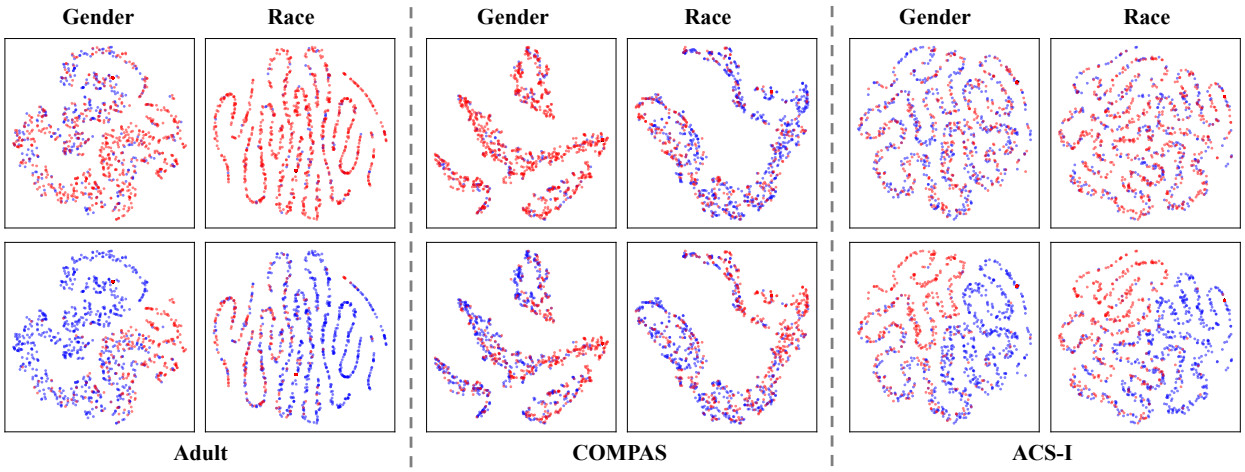

Figure 5.5: t-SNE visualization of the learned representations on `Adult`, coloured by sensitive group, one panel per regularizer. Greater intermixing indicates that group membership is harder to recover from the representation.

5.5.5 Hyperparameter Sensitivity

The robustness argued for in Section 5.3.3 should manifest as insensitivity to the coefficients of Equation 5.14. Figure 5.6 presents accuracy and Δ_{DP} on `Adult` as α and β vary over the cross-validated ranges.

Two patterns emerge. First, accuracy is highest at the smallest fairness weight, $\alpha = 10^{-4}$, and the best fairness is reached at $\alpha = 5 \times 10^{-2}$, with a marked degradation beyond that as α approaches 10^{-1}; smaller values of α nonetheless still yield satisfactory fairness when paired with β in the range 5 to 10, so the usable region of the grid is broad rather than a narrow ridge. Second, fairness responds far more strongly to α than to β. Increasing β from 10^{-3} to 10, a factor of 10,000, moves Δ_{DP} only from 7.2 to 4.2, whereas increasing α from 10^{-2} to 5×10^{-2}, a factor of five, moves it from 6.7 to 2.8. The regularizer therefore behaves as a single well-identified knob with the ℓ_2 term playing a minor role, which is the practical form that a well-conditioned penalty takes.

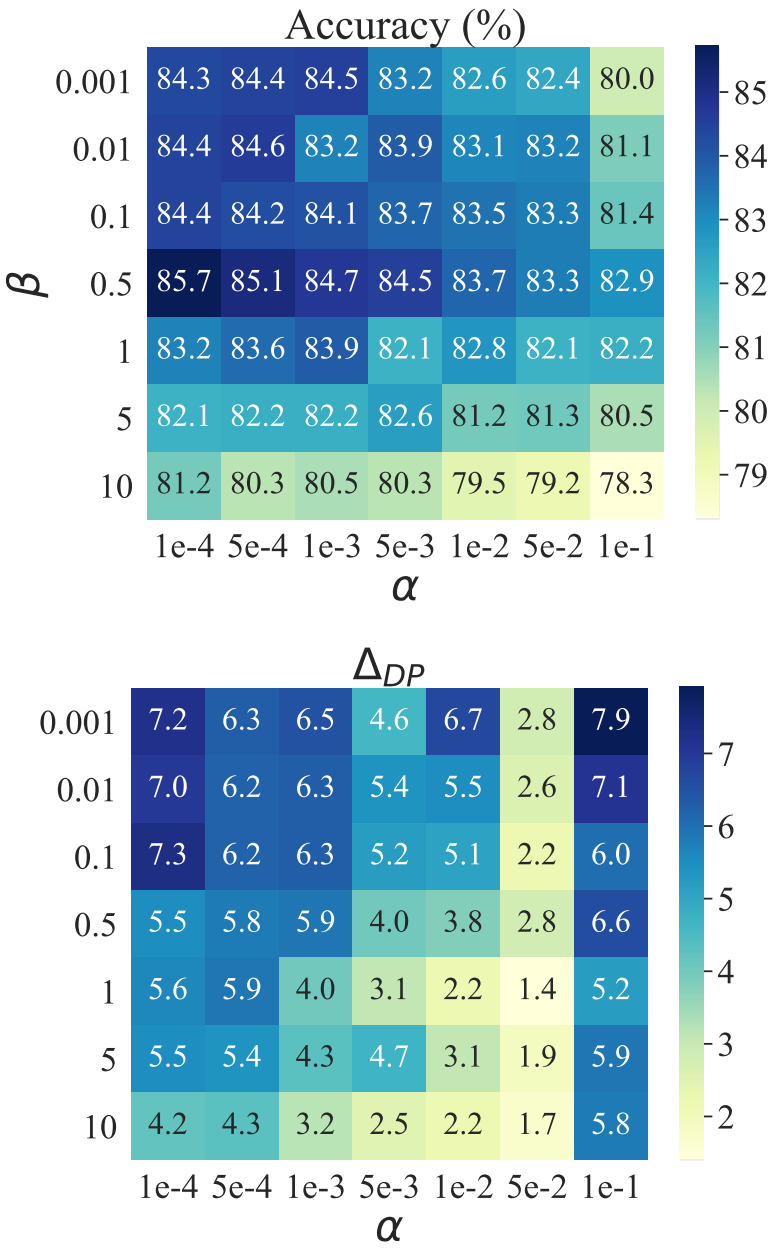

Figure 5.6: Parameter sensitivity on `Adult`: test accuracy (top) and Δ_{DP} (bottom) as the fairness weight α and the ℓ_2 weight β vary over the cross-validated ranges. The trade-off is smooth, remains stable over a wide range of β, and becomes sensitive only when α grows large.

5.6 Discussion

What the chapter establishes. The question posed in the title admits a structural answer. Because all three families of in-process method reduce to a choice of distance between distributions the model controls, the design of a fairness regularizer is a single decision, and that decision can be evaluated against stated criteria rather than by benchmark score alone. The CS divergence satisfies the three criteria: it is provably tighter than KL under a Gaussian comparison, it measures an angle rather than a magnitude and so is not inflated by scale mismatch, and it is estimable nonparametrically from samples. The empirical behaviour follows the argument closely, with the best Δ_{EO} in nine of ten settings, a frontier that degrades gradually rather than abruptly, and a broad usable hyperparameter region.

Limitations. Three should be recorded. The theoretical comparison of Theorem 5.2 holds under a Gaussian model adopted for tractability; the empirical results support the conclusion outside that model but do not extend the proof to it. The development is for binary classification with binary sensitive attributes, with the multi-attribute extension of Section 5.4 tested but the multi-class case left open. And the framework addresses group fairness only; individual and causal notions are not expressible in the form of Equation 5.7 through Equation 5.10 and would require separate treatment.

Returning to the previous chapter. The regularizer developed here is a drop-in replacement for the $\mathcal{L}_F$ of Equation 4.11. Substituting it changes the trajectory that the synthetic data of Chapter 4 is required to match, from one shaped by a first-moment surrogate for a single fairness notion to one shaped by a measure that controls the underlying dependence. The two contributions compose: Chapter 4 supplies a mechanism for transferring a property of the training dynamics into a reusable dataset, and this chapter supplies a better property to transfer.

What remains assumed. Both chapters share an assumption that has gone unstated because it is nearly universal: that the model, once trained, stays trained. Fairness is established during learning and is thereafter treated as a durable property of the artefact. Deployment does not honour that assumption. Data is deleted on request, and the parameters of a trained model are revised after the fact to comply. Nothing in Equation 5.14, or in the objective of Chapter 4, constrains what happens to fairness when weights are edited after training has finished. Chapter 6 shows that what happens is severe, and that the third control point of this dissertation, the parameters themselves, must be brought under the same discipline as the first two.

5.7 Summary

This chapter answered the question left open by Chapter 4. It organized in-process fair learning methods into three families, which match prediction statistics, align latent representations, and minimize prediction–attribute dependence, and observed that all three reduce to the choice of a distance measure between group-conditional distributions. From two documented failure modes, the inability of one notion's surrogate to deliver another and the fragility of fairness under parameter perturbation, it extracted three desiderata for that measure: a tight bound on the fairness quantity of interest, robustness to scale differences, and applicability to arbitrary distributions.

The Cauchy–Schwarz divergence meets all three. It is upper-bounded by the KL divergence in both directions under a Gaussian comparison; it compares kernel mean embeddings by angle rather than magnitude, unlike the Euclidean geometry of MMD and HSIC and the first-moment comparison of demographic parity penalties; and its kernel estimator assumes nothing about the distributions involved. Instantiated as a regularizer, it attains the best equal opportunity gap in nine of ten dataset–attribute settings, remains competitive on

demographic parity and utility, and yields a trade-off that degrades gradually rather than collapsing once the task objective is given weight.

Three chapters have now addressed the two demands of efficient and equitable learning by intervening at two control points, the data and the objective. Both took for granted that a trained model remains as it was trained. The next chapter removes that assumption.

Chapter 6

Preserving Group Fairness Under Machine Unlearning

6.1 Overview

The three preceding chapters share an assumption that was named at the end of Chapter 5 but not yet examined: that a model, once trained, stays trained. Chapter 3 optimized a dataset so that training on it would be cheap. Chapter 4 optimized a dataset so that training on it would be equitable. Chapter 5 optimized the objective that shapes what equitable means. In each case the guarantee is established at the end of training and is thereafter treated as a durable property of the resulting artefact.

Deployment does not respect that assumption. Personal data protection law now obliges organizations to erase individual records on request, and in the machine learning setting this means removing not only the data but its influence on models already trained on it. The parameters of a deployed model are therefore revised after the fact, sometimes repeatedly, and nothing in any objective of the preceding chapters says what becomes of fairness when

that happens.

This chapter answers that question, and the answer is unfavourable. It then supplies a correction. In the framework of this dissertation it occupies the third control point: the intervention is neither on the data nor on the objective but on the *parameter updates* the model has already absorbed.

The right to be forgotten. Data protection regimes including the European Union's General Data Protection Regulation [29], the California Consumer Privacy Act [36], Canada's Personal Information Protection and Electronic Documents Act, and Brazil's General Data Protection Law converge on a principle first introduced by the GDPR: the right to be forgotten [109, 126]. Applied to machine learning, compliance requires removing both the information about a record and its *impact* on trained models, a problem termed machine unlearning [18, 24]. A substantial literature now addresses it across recommender systems [23], graph learning [25], and federated settings [142], and the evaluation criteria have stabilized around two quantities: how completely the requested information is removed, and how much predictive accuracy survives.

That rubric has a blind spot. It contains no term for fairness, and the small literature that has considered the question [149, 102] has largely concluded that when deletion is non-uniform some unlearning methods behave better than others, without characterizing why or supplying a remedy.

The failure. Starting from a model trained with the fairness machinery of the preceding chapters, and honouring a deletion request drawn from an unprivileged group, we find that the model that remains is not slightly less fair but categorically unfair. On `Adult` with gender as the sensitive attribute, a fairly trained model attains $\Delta_{DP} = 1.6$; after removing 30% of the unprivileged group by a standard update-rollback method, Δ_{DP} is 30.8, a nineteen-

fold increase. Equal opportunity degrades comparably, from 2.6 to 22.3. On `ACS-I` the corresponding figures are $2.1 \rightarrow 23.9$ and $2.3 \rightarrow 28.8$. Accuracy falls by only a few points throughout, which is precisely why the standard rubric does not register the problem: by the criteria the field measures, nothing much has happened.

This is not a defect of one algorithm. Both exact unlearning by shard retraining and approximate unlearning by update rollback exhibit it, on every dataset tested. The cause, as Section 6.3 argues, is structural: a deletion request that is concentrated in one demographic group removes model updates that are concentrated in one demographic group, and the resulting shift in the distribution of the weights is itself a source of bias. The unlearning operation does not fail to preserve fairness by accident; it introduces unfairness by construction.

The correction. If the problem is a one-sided shift in the weight distribution, the remedy is to make the shift two-sided. The framework developed here, which we refer to as FMU, proceeds in two stages. It first withdraws the parameter updates associated with the batches containing the requested data, which discharges the deletion obligation. It then withdraws a matched quantity of updates drawn from batches carrying the *reversed* sensitive attribute at the same label, which cancels the induced skew. No retraining occurs, the underlying unlearning method is untouched, and the debiasing regularizer used at training time is untouched; the correction is orthogonal to both, and adds a handful of lines to an existing implementation.

Contributions of this chapter. The chapter makes the following contributions.

- It conducts a systematic study of the effect of machine unlearning on group fairness, establishing that both exact and approximate unlearning can render a fairly trained model severely unfair, and that the effect strengthens with the deletion ratio and with

the class imbalance of the original data.

- It attributes the effect to a shift in the distribution of model weights induced by non-uniform deletion, rather than to any property of a particular unlearning algorithm.

- It proposes a correction that operates entirely at the level of parameter updates: deletion followed by a matched, reversed-attribute compensation. The method requires no retraining, is agnostic to both the unlearning mechanism and the debiasing technique, and adds no storage beyond what update-rollback unlearning already requires.

- It validates the method on five datasets across two sensitive attributes each, showing fairness comparable to fair retraining at a fraction of the cost, accuracy comparable to retraining, and stronger protection against membership inference than retraining provides.

Section 6.2 fixes notation for unlearning and recalls the fairness quantities of Chapter 5. Section 6.3 presents the empirical study and the diagnosis. Section 6.4 develops the correction. Section 6.5 reports the evaluation, and Section 6.7 closes the chapter.

6.2 Preliminaries

6.2.1 Setting

The setting is that of Section 5.2: a dataset

$$\mathcal{T} = \{(\mathbf{x}_i, y_i, s_i)\}_{i=1}^{N}, \qquad \mathbf{x}_i \in \mathbb{R}^D, \quad y_i \in \{0,1\}, \quad s_i \in \{0,1\}, \tag{6.1}$$

with binary labels and a binary sensitive attribute, and a model $f_{\boldsymbol{\theta}}$ producing predictions $\hat{y}_i$. A learning algorithm is a mapping $M : \mathcal{D} \to \mathcal{H}$ from datasets to models in a hypothesis space $\mathcal{H}$, so that $M(\mathcal{T})$ denotes the model obtained by training on $\mathcal{T}$.

Group fairness is measured exactly as in Chapter 5, by the demographic parity gap of Equation 5.4 and the equal opportunity gap of Equation 5.5, evaluated on held-out data with binarized predictions. No new fairness quantity is introduced in this chapter; the point of the chapter is what happens to these two after the model is edited.

Fair training follows the in-process template of Equation 5.3, which we restate in the notation used here:

$$\min_{\boldsymbol{\theta}} \; \mathcal{L} = \min_{\boldsymbol{\theta}} \big(\mathcal{L}_{\text{utility}} + \alpha \, \mathcal{L}_{\text{fairness}} \big), \tag{6.2}$$

with α the fairness weight. Unless stated otherwise $\mathcal{L}_{\text{fairness}}$ is the demographic parity gap regularizer [27]; Section 6.5.6 examines this choice, and Section 6.6 returns to it in light of Chapter 5.

6.2.2 Machine Unlearning

Definition 6.1 (Unlearning mechanism)**.** *Given a trained model $M(\mathcal{T})$ and a subset $\mathcal{T}_u \subseteq \mathcal{T}$ of samples whose deletion has been requested, an* unlearning mechanism *is a mapping*

$$U : \mathcal{H} \times \mathcal{D} \times \mathcal{D} \to \mathcal{H}, \qquad \big(M(\mathcal{T}),\, \mathcal{T},\, \mathcal{T}_u\big) \;\longmapsto\; U\big(M(\mathcal{T}), \mathcal{T}, \mathcal{T}_u\big),$$

whose output is intended to resemble the model $M(\mathcal{T} \setminus \mathcal{T}_u)$ that would have been obtained by retraining from scratch on the retained data.

Because both M and U are randomized, the appropriate comparison is between distributions over $\mathcal{H}$ rather than between individual models. Writing $\Pr(M(\mathcal{T} \setminus \mathcal{T}_u))$ for the distribution induced by retraining and $\Pr(U(M(\mathcal{T}), \mathcal{T}, \mathcal{T}_u))$ for that induced by unlearning, the ideal is that the two be statistically indistinguishable. Methods divide into *exact* unlearning, which achieves this by construction at the cost of partial retraining, and *approximate* unlearning,

which achieves it up to a tolerance at much lower cost [101, 150, 135].

This distinction matters here only in that the failure documented below afflicts both.

6.2.3 Training as an Accumulation of Updates

The correction developed in this chapter operates on training as a sum of increments, so we record that decomposition explicitly. Training for P epochs with B mini-batches per epoch produces

$$\boldsymbol{\theta}_M = \boldsymbol{\theta}_{\text{init}} + \sum_{p=1}^{P} \sum_{b=1}^{B} \boldsymbol{\delta}_{p,b}, \tag{6.3}$$

where $\boldsymbol{\delta}_{p,b}$ denotes the parameter update contributed by batch b of epoch p and $\boldsymbol{\theta}_{\text{init}}$ the initialization. We write the per-batch update as $\boldsymbol{\delta}$ rather than $\Delta\boldsymbol{\theta}$ to keep it visually distinct from the fairness gaps Δ_{DP} and Δ_{EO}, which appear alongside it throughout the chapter.

During training a record is maintained associating each training sample with the batches it participated in. Given a deletion request, this record yields the set $\mathcal{B}_u$ of batch indices containing the requested data. The record may be a sample-to-batch index, a class-to-batch index, or any equivalent structure; when the anticipated requests concern a known subset of the data, only the updates for batches touching that subset need be retained, which bounds the storage cost well below that of logging every update.

6.3 Unlearning Undoes Fairness

6.3.1 The Effect

The diagnostic is straightforward. A model is trained to be fair using Equation 6.2; a deletion request is issued for a fraction of an unprivileged group, for example female positives on

`Adult`; a standard unlearning mechanism is applied; and the fairness of the surviving model is measured. Table 6.1 reports the outcome on two datasets under approximate unlearning by update rollback [48] at a 30% deletion ratio.

Table 6.1: Utility and fairness before and after unlearning, on `Adult` and `ACS-I` with gender as the sensitive attribute. Unlearning removes 30% of the unprivileged group by update rollback [48]. The final column gives the relative change; a positive value for a fairness gap denotes degradation. Accuracy and AUC barely move while both fairness gaps increase by an order of magnitude.

Dataset	Metric	Fair training	After unlearning	Change
`Adult`	Acc. (↑)	80.5 ± 0.1	76.1 ± 1.8	-5.5%
	AUC (↑)	86.5 ± 0.8	79.3 ± 2.8	-8.3%
	Δ_{EO} (↓)	2.6 ± 0.1	22.3 ± 5.2	$+758\%$
	Δ_{DP} (↓)	1.6 ± 0.9	30.8 ± 9.4	$+1825\%$
`ACS-I`	Acc. (↑)	80.9 ± 0.6	76.7 ± 1.9	-5.2%
	AUC (↑)	85.2 ± 1.7	79.0 ± 1.1	-7.3%
	Δ_{EO} (↓)	2.3 ± 0.6	28.8 ± 7.9	$+1152\%$
	Δ_{DP} (↓)	2.1 ± 0.3	23.9 ± 4.3	$+1038\%$

Two features of Table 6.1 deserve emphasis. The first is the magnitude: these are not marginal regressions but changes of one to two orders of magnitude, taking models from among the fairest in Chapter 5 to worse than no fairness intervention at all. The second is the asymmetry between the columns. Accuracy and AUC fall by five to eight percent, which is unremarkable after deleting 30% of a group and would be accepted as the ordinary price of compliance. An evaluation that reports only those two quantities, which is to say the standard evaluation, would conclude that unlearning had worked.

The full results in Section 6.5.2 show that this generalizes. On all five datasets, unlearning 10% or 30% of an unprivileged group by either shard-based exact unlearning [14] or update rollback [48] produces predictions more biased than those of the model before unlearning, on both Δ_{DP} and Δ_{EO}.

6.3.2 Diagnosis

Three regularities in the data identify the mechanism.

The effect scales with the deletion ratio. Removing 30% of a group produces a larger fairness degradation than removing 10%, monotonically and on every dataset (Table 6.5). Whatever is responsible accumulates with the quantity of removed influence.

The effect scales with class imbalance. Deletion from the imbalanced `Adult` dataset, whose sensitive attribute splits 1 : 2.08 by gender and 1 : 9.20 by race, induces markedly more bias than deletion from the comparatively balanced `ACS-I`, which splits 1 : 0.89 and 1 : 1.62. An unbalanced dataset is more sensitive to the removal of an unprivileged group because that group contributed a smaller share of the updates to begin with.

The effect depends on which group is deleted, not merely how much. Removing data from the unprivileged group produces substantially more bias than removing the same quantity from the privileged group, and a *mixed* request that draws uniformly from the whole dataset produces the least (Section 6.5.3). A uniform request removes updates from both groups in proportion and leaves the balance between them intact.

Together these point away from any property of a particular algorithm and toward the deletion request itself. Equation 6.3 exhibits the trained parameters as a sum of per-batch contributions, each carrying the demographic composition of its batch. Unlearning subtracts a subset of those contributions. When the request is concentrated in one group, the subtracted terms are concentrated in one group, and what remains is a parameter vector assembled from a differently constituted mixture of updates than the one that was validated as fair. The fairness of the original model was a property of the *whole* sum in Equation 6.3; removing a demographically skewed subset of its terms shifts the distribution of the weights and there is no reason for the property to survive. Unlearning does not fail to preserve fairness by accident. Under a non-uniform request it introduces bias by construction.

This reading has an immediate practical consequence. Because the cause is a one-sided

shift in a sum, the remedy need not involve retraining, revisiting the training objective, or modifying the unlearning mechanism. It is enough to make the shift two-sided.

6.4 Correcting the Distribution Shift

6.4.1 Problem Statement

Given a model $\boldsymbol{\theta}_M$ trained by Equation 6.2 and a deletion request $\mathcal{T}_u$, we seek an edited parameter vector that simultaneously (i) discharges the deletion obligation, in the sense of Definition 6.1; (ii) restores the group fairness that $\boldsymbol{\theta}_M$ possessed, as measured by Equation 5.4 and Equation 5.5; and (iii) retains predictive accuracy comparable to retraining on $\mathcal{T} \setminus \mathcal{T}_u$. All of this must be achieved without retraining, since avoiding retraining is the entire purpose of unlearning.

Figure 6.1 gives the resulting two-stage procedure.

6.4.2 Stage One: Unlearning

The first stage is standard update rollback [48]. Given the batch index set $\mathcal{B}_u$ derived from the request, the unlearned parameters are obtained by subtracting the corresponding contributions from Equation 6.3:

$$\boldsymbol{\theta}_{M'} = \boldsymbol{\theta}_{\text{init}} + \sum_{p=1}^{P} \sum_{b=1}^{B} \boldsymbol{\delta}_{p,b} - \sum_{u \in \mathcal{B}_u} \boldsymbol{\delta}_u = \boldsymbol{\theta}_M - \sum_{u \in \mathcal{B}_u} \boldsymbol{\delta}_u. \tag{6.4}$$

This step is deliberately unmodified: the deletion guarantee is whatever the underlying mechanism provides, and the contribution of this chapter is not a new unlearning method. The framework composes with any mechanism expressible as an edit to Equation 6.3.

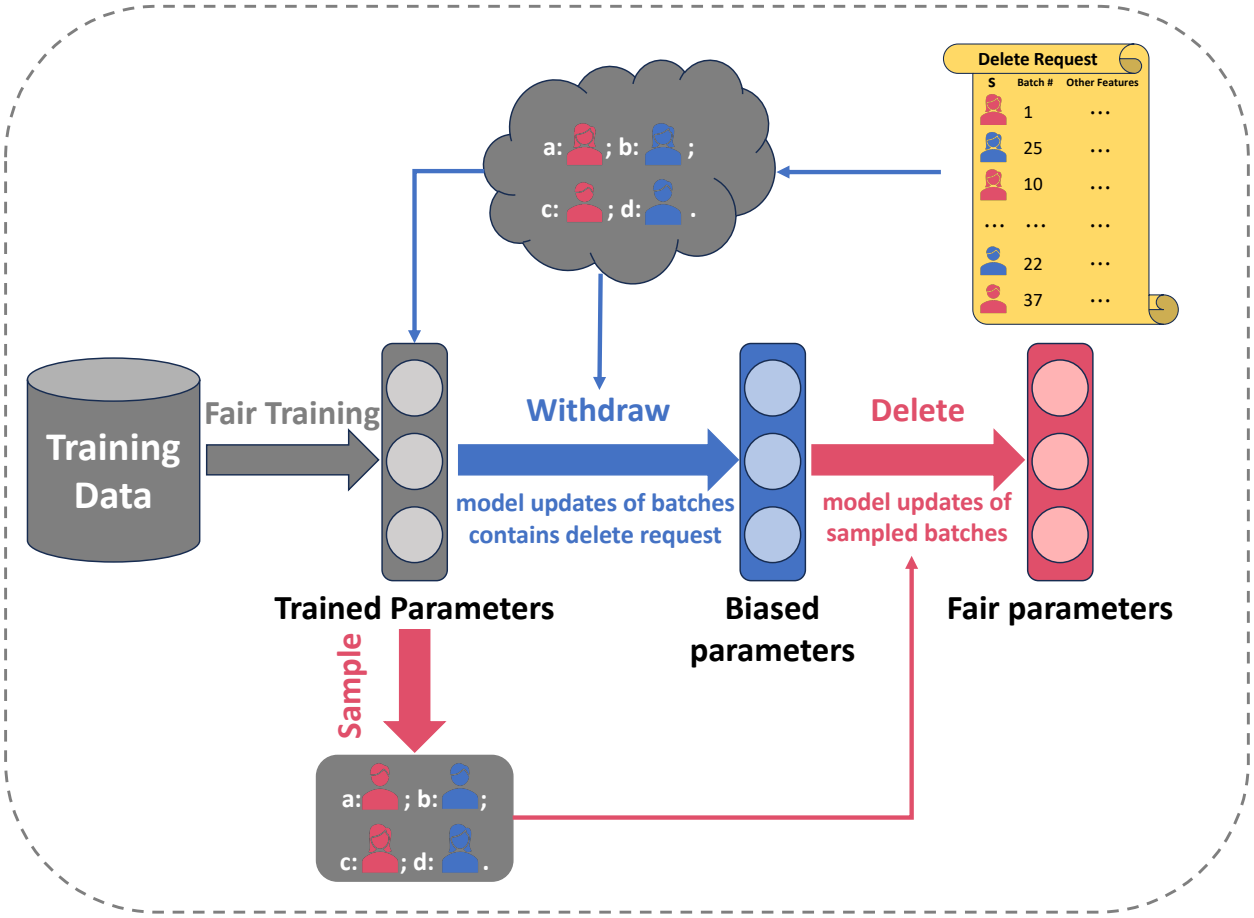

Figure 6.1: The framework. When an unlearning request arrives, the method withdraws the parameter updates of the batches containing the requested data, then withdraws the updates of a matched set of batches whose data carries the reversed sensitive attribute at the same label.

6.4.3 Stage Two: Reversed-Attribute Compensation

The second stage cancels the skew that Equation 6.4 introduces. Suppose $\mathcal{B}_u$ contains $n_{s,y}$ samples for each combination of sensitive attribute $s \in \{0,1\}$ and label $y \in \{0,1\}$. We assemble a compensation set $\mathcal{R} \subseteq \mathcal{B} \setminus \mathcal{B}_u$ by sampling, for every such combination, an equal number of samples carrying the *opposite* sensitive attribute at the *same* label:

$$n'_{1-s,\,y} = n_{s,\,y} \qquad \text{for all } s \in \{0,1\},\ y \in \{0,1\}, \tag{6.5}$$

where n and n' count samples in $\mathcal{B}_u$ and $\mathcal{R}$ respectively. The corrected parameters follow by withdrawing the updates of the compensation batches:

$$\boldsymbol{\theta}_{M'_{\text{fair}}} = \boldsymbol{\theta}_{M'} - \sum_{r \in \mathcal{R}} \boldsymbol{\delta}_r. \tag{6.6}$$

Two design choices in Equation 6.5 carry the argument. Matching on the *label* ensures that the compensation does not disturb the class balance, so the correction acts on the demographic composition of the remaining updates and not on the prediction task. Reversing the *sensitive attribute* is what makes the net shift two-sided: for every unit of influence removed from one group at a given label, an equal unit is removed from the other group at the same label, so the ratio between the groups' contributions to Equation 6.3 is preserved even though the absolute quantity of both has fallen. The diagnosis of Section 6.3.2 identified an imbalance in what was subtracted; Equation 6.5 subtracts the complementary quantity.

Why the correction does not leak. The compensation set is drawn from $\mathcal{B} \setminus \mathcal{B}_u$, the batches that do *not* contain requested data. The second stage therefore removes additional influence rather than reinstating any, and cannot reintroduce information about $\mathcal{T}_u$. The deletion guarantee established by Equation 6.4 is preserved unconditionally, since Equation 6.6 only subtracts further terms; Section 6.5.4 confirms this empirically against membership

inference.

Cost. The correction requires the stored updates for the compensation batches and nothing else. Since update-rollback unlearning already stores per-batch updates, the marginal storage is the difference between $|\mathcal{R}|$ and $|\mathcal{B}_u|$ worth of updates, and by Equation 6.5 these are equal in sample count. No gradient is computed, no forward pass is run, and no retraining occurs; Equation 6.6 is a sum of stored vectors. Section 6.5.8 quantifies the result.

6.4.4 Algorithm

Figure 6.2 states the complete procedure. Its brevity is the point: the correction is a few lines added to an existing unlearning implementation, is agnostic to the unlearning mechanism used in stage one, and is agnostic to the debiasing technique used during training.

Input: trained parameters $\boldsymbol{\theta}_M$; unlearning request batch list $\mathcal{B}_u$; stored per-batch updates $\{\boldsymbol{\delta}_b\}$.

Stage 1: Unlearning

Initialize $\boldsymbol{\theta}_{M'} = \boldsymbol{\theta}_M$
for $u \in \mathcal{B}_u$ **do**
  $\boldsymbol{\theta}_{M'} \leftarrow \boldsymbol{\theta}_{M'} - \boldsymbol{\delta}_u$
end for

Stage 2: Reversed-attribute compensation

Sample $\mathcal{R} \subseteq \mathcal{B} \setminus \mathcal{B}_u$ with $n'_{1-s,y} = n_{s,y}$ for all s, y, by Equation 6.5
Initialize $\boldsymbol{\theta}_{M'_{\text{fair}}} = \boldsymbol{\theta}_{M'}$
for $r \in \mathcal{R}$ **do**
  $\boldsymbol{\theta}_{M'_{\text{fair}}} \leftarrow \boldsymbol{\theta}_{M'_{\text{fair}}} - \boldsymbol{\delta}_r$
end for

Output: unlearned and rebalanced parameters $\boldsymbol{\theta}_{M'_{\text{fair}}}$.

Figure 6.2: The two-stage procedure. Stage one discharges the deletion request; stage two cancels the demographic skew it induces. Neither stage computes a gradient or retrains.

6.5 Experiments

The evaluation asks whether the correction restores fairness (Section 6.5.2), how it behaves across deletion ratios and request distributions (Section 6.5.3), whether it still actually unlearns (Section 6.5.4), why it works (Section 6.5.5), how the choice of training-time regularizer affects it (Section 6.5.6), and what it costs (Section 6.5.8).

6.5.1 Setup

Datasets. Five benchmarks are used: `Adult` [10], `COMPAS` [82], the income and travel-time tasks `ACS-I` and `ACS-T` [35], and the image dataset `CelebA-A` [92]. Each is evaluated under two sensitive attributes, following established practice [12, 22, 151]. Statistics appear in Table 6.2.

Table 6.2: Dataset statistics. #Feat. is the feature count after preprocessing; $N:P$ is the ratio of negative to positive target labels; the last two columns give the $s=0:1$ ratio for each sensitive attribute.

Dataset	Target	Sens. s	#Data	#Feat.	$N:P$	1st s	2nd s
`Adult`	Income	Gender, Race	45,222	101	1:0.33	1:2.08	1:9.20
`COMPAS`	Credit	Gender, Race	6,172	405	1:0.83	1:4.25	—
`ACS-I`	Income	Gender, Race	195,665	908	1:0.70	1:0.89	1:1.62
`ACS-T`	Travel time	Gender, Race	172,508	1,567	1:0.94	1:0.89	1:1.61
`CelebA-A`	Attractive	Gender, Age	202,599	48×48	1:0.95	1:1.40	1:0.29

Baselines. Four comparisons are made. **Retraining** discards the requested data and retrains from scratch, in a standard and a fairness-regularized variant; it provides the strongest privacy guarantee at the highest cost. **SISA** [14] is exact unlearning by sharding: the data is partitioned, a sub-model is trained per shard, and only affected sub-models are retrained from cached states. **Amnesiac** [48] is approximate unlearning by update rollback, that is, stage one alone. **Oesterling** [102] is the existing fair unlearning method, which enforces fairness through convex pairwise regularization and consequently cannot express non-convex criteria such as demographic parity or equalized odds [27].

Deletion requests. To model realistic right-to-be-forgotten requests we consider deletion from a privileged group, from an unprivileged group, and mixed deletion drawn uniformly from the whole training set, at ratios from 5% to 50%. Unless stated otherwise the reported setting deletes 30% of the unprivileged group, for example female positives on `Adult`.

Implementation. Models are trained in PyTorch [104] with Adam [76], initial learning rate 0.01, weight decay 0, batch size 32, and 100 epochs. Tabular data uses a two-layer MLP with 256 hidden units and image data a two-layer MLP with 128 hidden units. The fairness weight is $\alpha = 3.0$, selected as described in Section 6.5.7. Results are means and standard deviations over 20 runs.

6.5.2 Fairness and Utility After Unlearning

Table 6.3 and Table 6.4 report the full comparison. Four findings emerge.

Both exact and approximate unlearning introduce bias. This is the central negative result, and it holds without exception. On all five datasets and both sensitive attributes, SISA and Amnesiac produce larger Δ_{DP} and Δ_{EO} than the fairly trained model they started from. The magnitudes are severe: on `Adult` with gender, Amnesiac raises Δ_{DP} from 1.6 to 30.8; on `COMPAS` with gender it raises it from 1.7 to 34.6. Even standard, non-fairness-aware retraining, which one might expect to be neutral, lands at Δ_{DP} values of 15.2 and 18.8 on those settings, because it re-derives a biased model from biased retained data.

The correction restores fairness to the level of fair retraining. Across the ten dataset–attribute settings the corrected model attains Δ_{DP} between 0.9 and 2.9 and Δ_{EO} between 1.2 and 2.9, statistically comparable to fair retraining throughout and better than it on four settings. Against the fair unlearning baseline the margin is large: Oesterling attains

Table 6.3: Utility and fairness on `Adult`, `COMPAS`, and `ACS-T` before and after unlearning 30% of the unprivileged group, over 20 runs. Higher is better for accuracy and AUC; lower is better for Δ_{DP} and Δ_{EO}. Bold marks the best value among the six unlearning and retraining methods; the two original-training columns are references and are excluded from that comparison.

Sens.	Metric	Original training		Standard unlearning		Retraining		Fair unlearning	
		Std.	Fair	SISA	Amne.	Std.	Fair	Oester.	Proposed
`Adult`									
Race	Acc.	84.4 ± 0.3	80.1 ± 0.7	76.3 ± 1.2	76.7 ± 1.3	$\mathbf{82.3} \pm 0.5$	80.4 ± 0.5	77.3 ± 2.6	78.4 ± 1.3
	AUC	90.1 ± 0.5	85.3 ± 0.5	78.4 ± 0.4	77.1 ± 1.7	$\mathbf{87.2} \pm 1.9$	83.2 ± 1.3	80.4 ± 2.3	80.9 ± 1.6
	Δ_{EO}	9.3 ± 3.8	1.3 ± 1.0	19.8 ± 5.2	29.4 ± 7.1	10.6 ± 5.6	$\mathbf{1.3} \pm 0.2$	5.8 ± 2.1	1.9 ± 0.5
	Δ_{DP}	13.4 ± 0.8	1.0 ± 0.6	21.4 ± 2.6	26.2 ± 6.3	15.2 ± 4.8	$\mathbf{1.5} \pm 0.4$	6.3 ± 3.2	1.7 ± 0.8
Gender	Acc.	84.6 ± 0.3	80.5 ± 0.1	76.4 ± 1.1	76.1 ± 1.8	$\mathbf{82.1} \pm 0.7$	79.4 ± 1.7	78.4 ± 0.7	78.9 ± 0.2
	AUC	90.8 ± 0.2	86.5 ± 0.8	79.3 ± 2.1	79.3 ± 2.8	$\mathbf{88.4} \pm 1.2$	82.1 ± 0.8	79.3 ± 1.9	81.2 ± 0.5
	Δ_{EO}	8.4 ± 3.2	2.6 ± 0.1	20.9 ± 3.7	22.3 ± 5.2	9.2 ± 2.5	1.8 ± 0.3	5.5 ± 1.2	$\mathbf{1.6} \pm 0.2$
	Δ_{DP}	16.5 ± 0.9	1.6 ± 0.9	24.6 ± 4.3	30.8 ± 9.4	15.2 ± 2.1	$\mathbf{1.1} \pm 0.6$	6.7 ± 2.6	1.9 ± 0.4
`COMPAS`									
Race	Acc.	66.9 ± 1.0	60.9 ± 1.1	54.2 ± 0.8	54.7 ± 1.4	$\mathbf{65.3} \pm 1.8$	60.9 ± 0.6	57.4 ± 1.8	59.9 ± 0.5
	AUC	72.4 ± 0.8	69.4 ± 1.8	61.1 ± 1.9	61.9 ± 2.3	$\mathbf{72.1} \pm 2.1$	67.2 ± 1.5	64.4 ± 1.7	66.5 ± 1.1
	Δ_{EO}	19.4 ± 4.6	0.9 ± 0.3	19.5 ± 4.2	27.6 ± 3.3	16.3 ± 6.2	$\mathbf{1.0} \pm 0.7$	7.4 ± 3.8	1.2 ± 0.3
	Δ_{DP}	17.2 ± 4.1	1.4 ± 1.1	34.2 ± 10.2	29.0 ± 8.4	18.4 ± 5.4	$\mathbf{1.1} \pm 0.3$	9.9 ± 2.3	1.9 ± 0.6
Gender	Acc.	66.8 ± 0.7	62.1 ± 1.5	55.4 ± 1.9	54.9 ± 2.6	$\mathbf{65.4} \pm 1.6$	62.6 ± 1.5	59.3 ± 1.4	60.1 ± 0.8
	AUC	72.1 ± 0.9	68.2 ± 1.6	61.2 ± 0.8	60.3 ± 2.1	$\mathbf{66.7} \pm 1.0$	62.3 ± 0.8	60.0 ± 1.8	61.4 ± 0.6
	Δ_{EO}	12.4 ± 5.8	2.1 ± 0.5	29.3 ± 6.7	31.3 ± 14.8	16.2 ± 4.8	$\mathbf{2.7} \pm 1.9$	8.3 ± 2.9	2.9 ± 1.5
	Δ_{DP}	17.2 ± 4.3	1.7 ± 0.8	33.5 ± 16.7	34.6 ± 17.5	18.8 ± 6.2	$\mathbf{1.6} \pm 0.5$	9.1 ± 3.5	1.9 ± 0.8
`ACS-T`									
Race	Acc.	66.3 ± 0.3	65.7 ± 0.2	59.3 ± 1.4	58.7 ± 0.8	$\mathbf{65.2} \pm 1.0$	64.4 ± 0.8	59.2 ± 2.0	61.5 ± 1.3
	AUC	72.6 ± 0.2	71.2 ± 0.2	65.3 ± 1.9	64.2 ± 2.4	$\mathbf{70.3} \pm 1.6$	68.2 ± 2.4	65.4 ± 2.1	67.8 ± 1.1
	Δ_{EO}	6.9 ± 1.2	0.9 ± 0.5	17.4 ± 9.5	15.4 ± 8.1	7.4 ± 2.4	$\mathbf{1.0} \pm 0.3$	5.3 ± 2.8	1.5 ± 0.6
	Δ_{DP}	10.2 ± 1.6	2.8 ± 0.7	18.2 ± 7.4	14.3 ± 9.9	11.4 ± 2.4	2.3 ± 1.1	8.1 ± 3.3	$\mathbf{1.8} \pm 0.5$
Gender	Acc.	66.2 ± 0.4	65.9 ± 0.4	57.2 ± 2.1	58.9 ± 2.7	$\mathbf{64.3} \pm 0.7$	63.5 ± 0.8	60.5 ± 1.9	62.7 ± 0.7
	AUC	72.7 ± 0.2	72.0 ± 0.3	66.2 ± 1.8	67.2 ± 1.9	$\mathbf{70.2} \pm 2.4$	67.3 ± 1.0	64.1 ± 1.3	68.6 ± 0.8
	Δ_{EO}	6.1 ± 3.5	1.8 ± 0.5	17.9 ± 7.3	22.5 ± 10.5	8.8 ± 2.2	$\mathbf{2.5} \pm 1.0$	6.4 ± 4.3	2.7 ± 1.1
	Δ_{DP}	7.3 ± 2.6	1.5 ± 0.4	15.6 ± 9.3	17.8 ± 7.4	6.2 ± 3.5	$\mathbf{2.2} \pm 0.7$	5.3 ± 5.7	2.9 ± 1.3

Table 6.4: Utility and fairness on `ACS-I` and `CelebA-A` before and after unlearning 30% of the unprivileged group, over 20 runs. Bold marks the best value among the six unlearning and retraining methods; the two original-training columns are references and are excluded from that comparison.

Sens.	Metric	Original training		Standard unlearning		Retraining		Fair unlearning	
		Std.	Fair	SISA	Amne.	Std.	Fair	Oester.	Proposed
`ACS-I`									
Race	Acc.	81.2 ± 0.1	80.2 ± 1.2	75.3 ± 1.9	75.0 ± 1.1	$\mathbf{80.9} \pm 1.5$	77.9 ± 1.5	76.5 ± 0.8	78.3 ± 0.9
	AUC	90.1 ± 0.1	87.0 ± 1.6	82.5 ± 2.4	81.5 ± 1.6	$\mathbf{89.7} \pm 2.1$	86.6 ± 1.1	84.1 ± 1.6	85.2 ± 1.0
	Δ_{EO}	7.4 ± 0.6	2.0 ± 0.5	23.5 ± 4.4	25.3 ± 4.0	9.1 ± 1.4	$\mathbf{1.2} \pm 0.7$	7.6 ± 2.2	1.4 ± 0.2
	Δ_{DP}	10.0 ± 1.8	1.7 ± 0.2	26.3 ± 7.7	23.3 ± 6.2	11.3 ± 3.8	1.0 ± 0.4	3.4 ± 1.8	$\mathbf{0.9} \pm 0.4$
Gender	Acc.	82.0 ± 0.2	80.9 ± 0.6	75.4 ± 2.2	76.7 ± 1.9	$\mathbf{81.1} \pm 0.6$	78.3 ± 1.2	75.3 ± 0.8	78.0 ± 0.8
	AUC	90.1 ± 0.1	85.2 ± 1.7	80.2 ± 2.6	79.0 ± 1.1	$\mathbf{88.4} \pm 1.8$	84.2 ± 2.0	82.1 ± 0.9	84.0 ± 1.5
	Δ_{EO}	6.1 ± 0.6	2.3 ± 0.6	24.3 ± 3.3	28.8 ± 7.9	9.1 ± 0.5	$\mathbf{1.4} \pm 0.3$	10.2 ± 2.7	1.7 ± 1.0
	Δ_{DP}	10.2 ± 1.6	2.1 ± 0.3	28.4 ± 6.8	23.9 ± 4.3	12.4 ± 2.2	$\mathbf{1.7} \pm 0.8$	2.6 ± 0.6	$\mathbf{1.7} \pm 1.2$
`CelebA-A`									
Age	Acc.	78.1 ± 0.6	75.0 ± 1.9	68.3 ± 0.8	66.7 ± 0.4	$\mathbf{77.2} \pm 0.9$	76.4 ± 0.4	74.2 ± 2.0	76.5 ± 0.8
	AUC	85.9 ± 0.8	81.3 ± 0.6	75.3 ± 0.9	74.2 ± 1.8	$\mathbf{84.3} \pm 1.2$	81.2 ± 1.8	77.4 ± 2.1	80.8 ± 0.7
	Δ_{EO}	19.3 ± 1.5	0.9 ± 0.3	27.4 ± 11.2	35.4 ± 13.5	21.4 ± 1.7	$\mathbf{1.0} \pm 0.5$	5.3 ± 2.8	1.5 ± 0.2
	Δ_{DP}	38.7 ± 1.8	1.6 ± 1.3	41.2 ± 21.4	40.3 ± 20.4	37.4 ± 1.6	2.3 ± 0.9	8.1 ± 3.3	$\mathbf{1.8} \pm 0.3$
Gender	Acc.	78.1 ± 0.6	74.3 ± 1.6	67.2 ± 1.9	68.9 ± 1.8	$\mathbf{78.4} \pm 0.5$	73.5 ± 0.6	70.5 ± 1.9	72.7 ± 0.7
	AUC	85.9 ± 0.8	81.7 ± 1.2	76.2 ± 2.1	77.2 ± 1.4	$\mathbf{84.2} \pm 0.8$	82.3 ± 1.2	80.1 ± 1.3	81.6 ± 0.8
	Δ_{EO}	35.6 ± 2.1	1.1 ± 0.4	37.9 ± 21.3	42.5 ± 20.5	37.8 ± 26.7	$\mathbf{2.5} \pm 0.7$	12.4 ± 4.3	2.7 ± 0.6
	Δ_{DP}	50.6 ± 2.6	2.2 ± 0.5	45.6 ± 19.3	47.8 ± 27.4	53.2 ± 24.3	2.2 ± 0.6	15.2 ± 5.7	$\mathbf{1.8} \pm 0.5$

Δ_{DP} between 2.6 and 15.2, and is beaten on every setting.

Accuracy is comparable to retraining. The corrected model recovers accuracy within a few points of full retraining and consistently exceeds both SISA and Amnesiac, which lose accuracy without gaining fairness. On `CelebA-A` with age it reaches 76.5 against 77.2 for standard retraining and 66.7 for Amnesiac.

Dataset structure governs the severity. Deletion from imbalanced `Adult` induces more bias than from balanced `ACS-I`, and deletion from a larger dataset costs less accuracy than from a smaller one. `COMPAS`, with 6,172 samples, shows the largest accuracy drops throughout. The correction approximates retraining accuracy in both regimes.

6.5.3 Deletion Ratio and Request Distribution

Table 6.5 isolates the dependence on the deletion ratio by comparing 10% against 30% deletion for the two standard mechanisms. The relationship is monotone in nine of the ten settings for Amnesiac and in all ten for SISA: more deletion, more bias. This is the first regularity of Section 6.3.2, and it is what the accumulation view of Equation 6.3 predicts, since the quantity of demographically skewed influence removed grows with the request.

Table 6.5: Demographic parity gap Δ_{DP} after unlearning 10% versus 30% of the unprivileged group, for exact (SISA) and approximate (Amnesiac) unlearning. The fair-training column is the value before unlearning. Bias grows with the deletion ratio in almost every setting.

Dataset	Sens.	Fair train.	SISA		Amnesiac	
			10%	30%	10%	30%
`Adult`	Race	1.0	16.4	21.4	9.5	26.2
	Gender	1.6	12.5	24.6	20.1	30.8
`COMPAS`	Race	1.4	26.5	34.2	23.1	29.0
	Gender	1.7	20.8	33.5	27.6	34.6
`ACS-T`	Race	2.8	15.7	18.2	16.7	14.3
	Gender	1.5	13.0	15.6	10.3	17.8
`ACS-I`	Race	1.7	18.4	26.3	14.3	23.3
	Gender	2.1	17.2	28.4	15.4	23.9
`CelebA-A`	Age	1.6	35.7	41.2	36.7	40.3
	Gender	2.2	39.0	45.6	40.3	47.8

Figure 6.3 examines the request *distribution*, plotting the improvement of the corrected model over uncorrected update rollback across ratios from 5% to 50% on an imbalanced dataset (`Adult`) and a balanced one (`ACS-I`). Two patterns hold.

First, the benefit tracks the imbalance of the original data. On `Adult`, removing data from the unprivileged group yields a markedly more biased model than removing the same amount from the privileged group, and the correction has correspondingly more to repair. On `ACS-I`, where the groups are nearly balanced, the discrepancy between privileged and unprivileged deletion is smaller and so is the gain.

Second, mixed requests are the benign case. When deletion is drawn uniformly the removed updates carry the demographic composition of the training set, so little skew is introduced

and there is correspondingly little for the correction to recover. Utility nonetheless degrades more, since a fixed percentage of the whole dataset is a larger absolute quantity of data than the same percentage of one group. This is a useful practical boundary: the correction matters exactly when requests are demographically concentrated, which is also when the failure is worst.

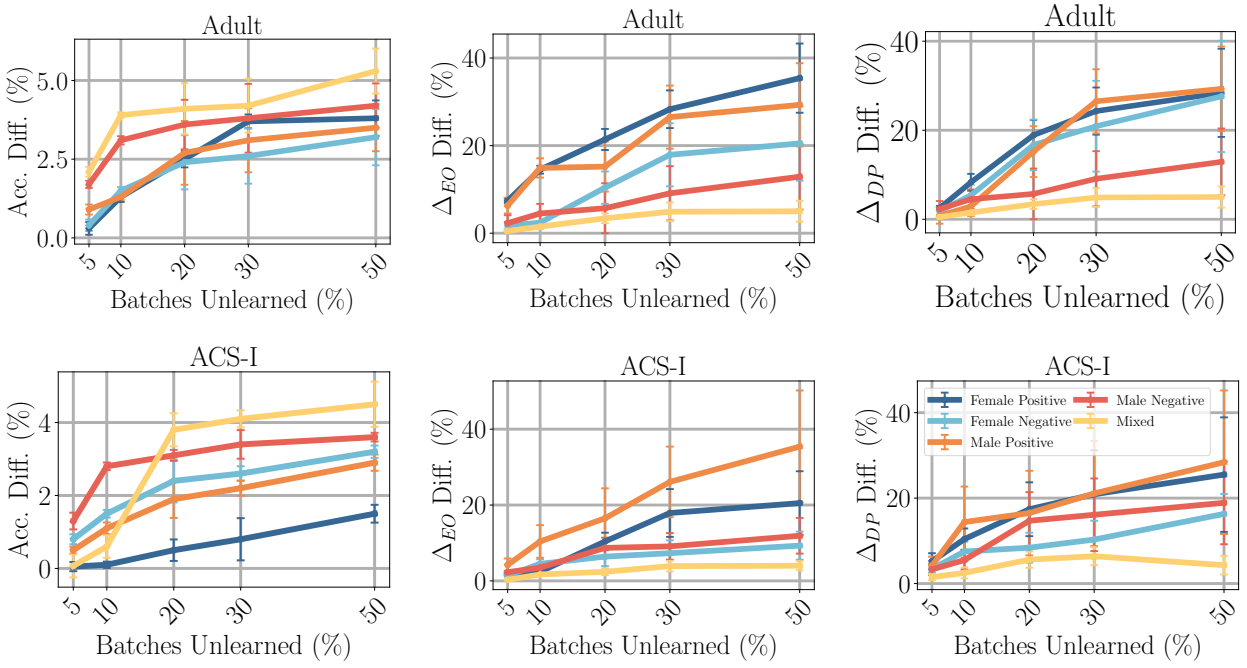

Figure 6.3: Improvement of the corrected model over uncorrected update rollback on `Adult` (top) and `ACS-I` (bottom), at deletion ratios $\{5\%, 10\%, 20\%, 50\%\}$ with gender as the sensitive attribute. From left to right: accuracy difference, Δ_{EO} difference, Δ_{DP} difference, each defined so that larger is better. `Adult`'s unprivileged group is female positive; `ACS-I` is more balanced.

6.5.4 Does It Still Unlearn?

A correction that restored fairness by quietly reinstating deleted influence would be worthless. Two experiments confirm that this does not occur.

Target and non-target accuracy. Figure 6.4 tracks accuracy on the deleted group and on the retained groups on `ACS-I`, with male positives as the deletion target at a 30% ratio.

Unlearning is applied at epoch 100 and training then continues. On the target group the corrected model falls to near-random accuracy, indicating that the requested information has genuinely been removed; fair retraining and Oesterling both retain visibly higher target accuracy, suggesting residual information. On the retained groups the corrected model dips briefly and recovers quickly to accuracy comparable with fair retraining, and does so in fewer steps than Oesterling.

Membership inference. Table 6.6 reports the recall of a membership inference attack [115, 139, 60] using 20 shadow models, following established protocol [15, 53]. The row marked 0^* is the decisive one: it measures the attack immediately after the deletion operation and *before any retraining*. The corrected model already reports recall 0.00 there, whereas retraining and Oesterling still leak at 0.97 and require several epochs of retraining to close the gap. Fair retraining provides only minimal protection even after twenty full epochs. This is consistent with the argument of Section 6.4.3: since both stages only ever subtract stored updates, the deletion guarantee is established immediately and the compensation cannot undo it.

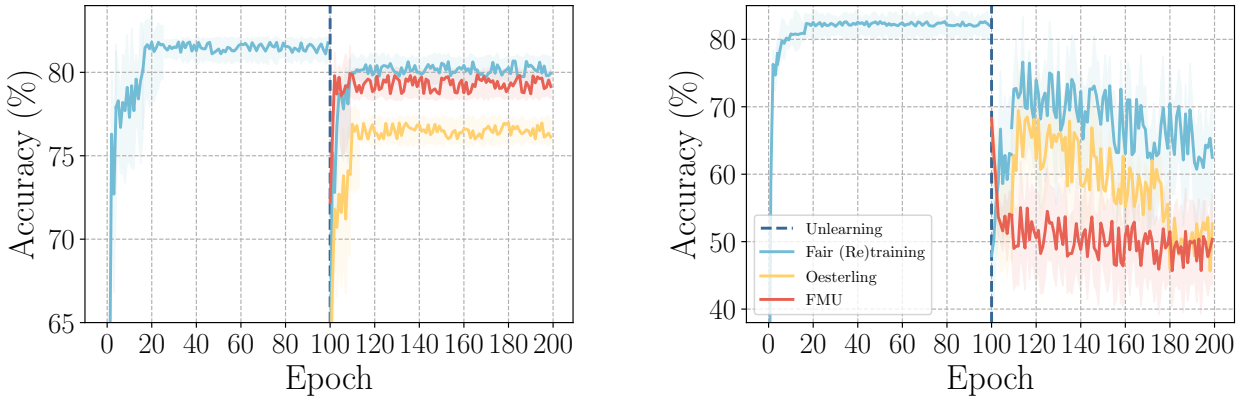

Figure 6.4: Accuracy on the non-target groups (left) and the deletion target group (right) on `ACS-I` at a 30% deletion ratio with gender as the sensitive attribute. Unlearning is applied at epoch 100 and training continues thereafter. Effective unlearning means high accuracy on the left and near-random accuracy on the right.

Table 6.6: Recall of a membership inference attack after unlearning, by retraining epoch. Row 0* measures the attack immediately after the deletion operation and before any retraining; lower is better. The proposed correction removes membership signal without any retraining.

Epoch	Retraining	Oesterling	Proposed
0	0.97 ± 0.17	0.97 ± 0.17	0.97 ± 0.17
0*	0.97 ± 0.17	0.97 ± 0.17	$\mathbf{0.00} \pm 0.05$
5	0.12 ± 0.03	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.00} \pm 0.00$
10	0.05 ± 0.01	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.00} \pm 0.00$
15	0.01 ± 0.00	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.00} \pm 0.00$
20	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.00} \pm 0.00$

6.5.5 Why the Correction Works

Figure 6.5 plots the predicted-probability densities of the two gender groups on `Adult` after removing 30% of the unprivileged group, under each method. The sequence makes the chapter's argument visible in one figure. Fair training produces two closely overlapping densities. Standard update rollback pulls them apart, which is the failure of Section 6.3.1 in distributional form. Fair retraining restores the overlap, at the cost of retraining. Oesterling partially restores it. The proposed correction restores it to a degree comparable with fair retraining, by rebalancing the group contributions to the parameter sum rather than by re-optimizing.

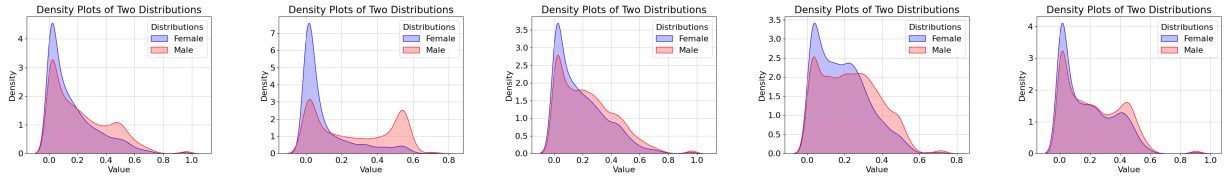

Figure 6.5: Predicted-probability densities for the male and female groups on `Adult` after deleting 30% of the unprivileged group. From left to right: fair training (fair); update rollback (unfair); fair retraining (fair); Oesterling (less fair); the proposed correction (fair). Greater overlap indicates better group fairness.

6.5.6 Choice of Training-Time Regularizer

The correction operates on updates produced by a fairness-regularized training run, so the regularizer of Equation 6.2 is a component of the pipeline. Figure 6.6 sweeps hyperparam-

eters to trace the accuracy–Δ_{DP} Pareto front for four choices: HSIC [87], the prejudice remover [73], adversarial debiasing [147], and the demographic parity gap regularizer [27].

The pattern reproduces the taxonomy of Chapter 5 closely. HSIC, a dependence measure, attains lower Δ_{DP} than the others at matched accuracy. Adversarial debiasing is strong on utility but unstable in its debiasing, consistent with the optimization difficulties noted in Section 4.5.2. The prejudice remover debiases well but sacrifices predictive performance. The gap regularizer offers the most balanced trade-off, and is adopted as the default here. Section 6.6 returns to this choice.

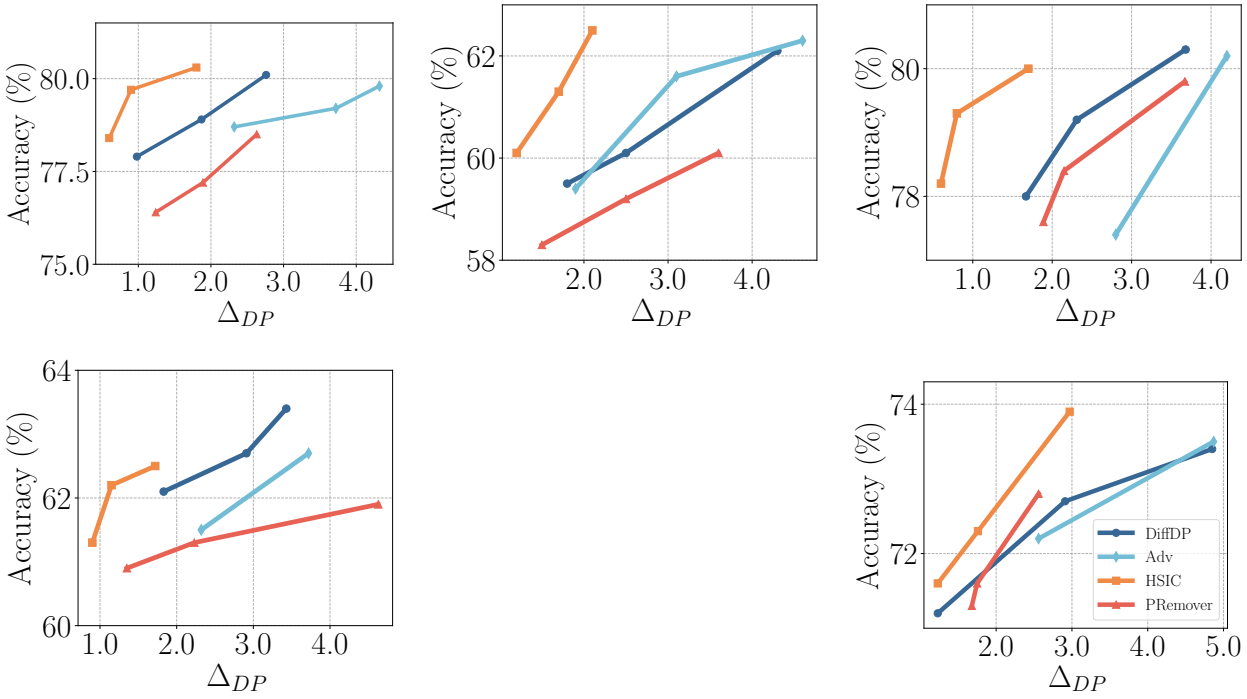

Figure 6.6: Accuracy against Δ_{DP} for four training-time fairness regularizers on `Adult`, `COMPAS`, `ACS-I` (top) and `ACS-T`, `CelebA-A` (bottom). The upper-left corner is preferable.

6.5.7 Sensitivity to the Fairness Weight

Figure 6.7 varies the fairness weight α of Equation 6.2 over $\{0.5, 1.5, 3.0, 5.0, 7.0\}$. Accuracy is essentially unaffected for $\alpha \leq 1.5$ and decays sharply once α grows large. Fairness improves as α increases and then *deteriorates* at large values, because the heavily weighted fairness

term makes the joint objective harder to optimize and the model fails to reach a good solution on either criterion. The value $\alpha = 3.0$ sits at the balance point and is used throughout.

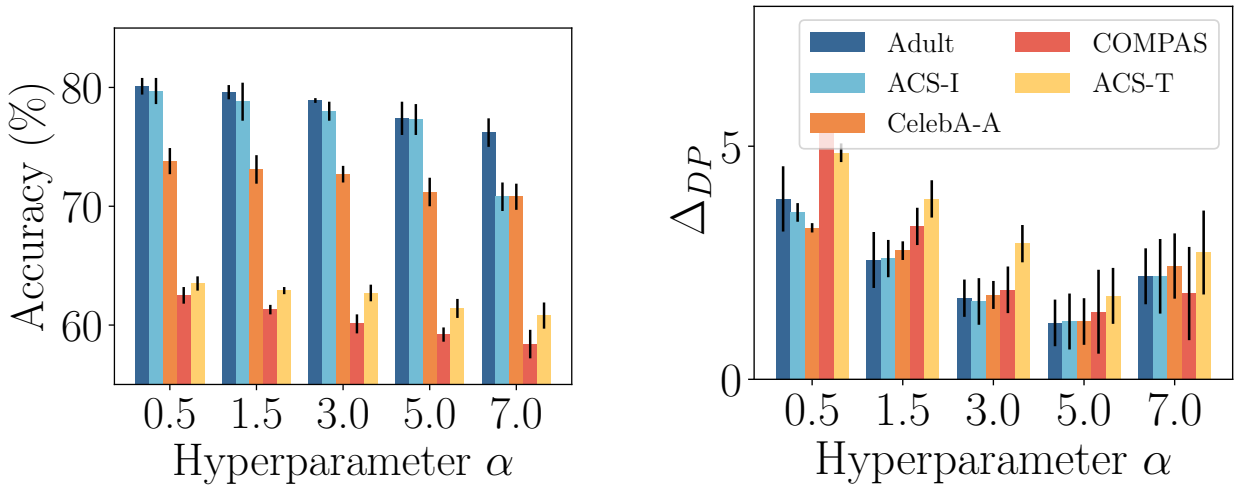

Figure 6.7: Effect of the fairness weight α on accuracy (left) and Δ_{DP} (right). Fairness improves with α up to a point and then degrades, as the heavily weighted objective becomes harder to optimize.

6.5.8 Runtime and Storage

Table 6.7 reports both costs. On `Adult` the correction completes in 2.1 seconds against 80.3 for retraining, a 38.2× speedup; on `CelebA-A` it takes 5.2 seconds against 1657.1, a 318.7× speedup. The advantage widens with dataset scale, since the correction's cost depends on the number of stored update vectors summed rather than on the size of the retained data. It is roughly twice as fast as Oesterling, whose compensation step is more elaborate, and marginally slower than uncorrected rollback, which is the price of the second stage.

Storage follows the same ordering. Retraining requires the most, then SISA, which must cache intermediate checkpoints for every shard, an overhead that grows as more shards are touched. Oesterling follows, its cost dominated by forming and inverting the Hessian of the fair loss over the retained data. The proposed correction stores 2.5×10^3 parameters on `Adult` against Amnesiac's 2.2×10^3; the difference is the parameters associated with the

fairness term, and it is this that makes a simple regularizer preferable to an elaborate one on storage grounds.

Table 6.7: Runtime (seconds, with speedup over retraining) and stored parameter counts for 2,000 unlearned samples. SISA uses 20 shards. Measured on an NVIDIA RTX A4000 with 100 unlearning requests; retraining runs for 100 epochs.

	Retraining	SISA	Amnesiac	Oesterling	Proposed
Runtime, `Adult`	80.3 s	25.5 s	1.4 s	4.2 s	2.1 s
Speedup, `Adult`	—	3.1×	57.4×	19.1×	38.2×
Runtime, `CelebA-A`	1657.1 s	137.9 s	3.6 s	15.2 s	5.2 s
Speedup, `CelebA-A`	—	12.0×	460.3×	109.1×	318.7×
#Params, `Adult`	1.6×10^6	3.6×10^4	2.2×10^3	3.3×10^4	2.5×10^3
#Params, `CelebA-A`	5.2×10^7	2.7×10^5	1.1×10^4	2.5×10^5	1.2×10^4

6.6 Discussion

What the chapter establishes. Two things, one negative and one positive. The negative result is that group fairness does not survive machine unlearning, that the failure afflicts exact and approximate mechanisms alike, and that the standard evaluation rubric of deletion efficacy plus retained accuracy is constitutionally unable to detect it. The positive result is that the failure has a structural cause admitting a structural remedy: because bias enters through a demographically one-sided edit to the sum in Equation 6.3, it can be removed by a matched, reversed-attribute edit to the same sum, with no retraining and no gradient computation.

The broader point for this dissertation is that the third control point is not optional. A guarantee installed at the first control point (Chapter 4) or the second (Chapter 5) is only as durable as the parameter edits subsequently applied to the model, and those edits must be brought under the same discipline or they will silently undo the guarantee.

A composition left open. Section 6.5.6 selected the demographic parity gap regularizer for Equation 6.2 on the basis of a four-way empirical comparison, and Figure 6.6 shows

the dependence-based HSIC attaining lower Δ_{DP} at matched accuracy. Chapter 5 argued on principled grounds that a measure of this kind should be preferred, and derived one that outperforms HSIC. Substituting the Cauchy–Schwarz regularizer here would change the updates $\boldsymbol{\delta}_{p,b}$ that the correction operates on, and by the argument of Section 5.3.4 it should yield a fairer starting point and better-conditioned updates. The correction is agnostic to that choice, so the substitution is available; it is not evaluated in this chapter, and stands as a concrete composition of two contributions of this dissertation that remains to be tested.

Limitations. Three should be recorded. The correction requires that per-batch updates have been stored, which is exactly the requirement of update-rollback unlearning and no more, but it does mean the method does not apply to unlearning mechanisms that do not maintain such a record. The compensation of Equation 6.5 matches on counts within each (s, y) cell, which is a first-order correction to the composition of the update sum; it does not guarantee that the resulting parameter vector is fair, only that the demographic skew introduced by deletion is cancelled in expectation, and the empirical results are what establish that this suffices. And the treatment is confined to binary sensitive attributes and binary labels, inheriting the scope of Chapter 5.

Where the assumptions break. The method rests on two premises that hold comfortably at the scale studied here and not at all beyond it. The first is that the entire training history can be decomposed into recorded per-batch updates, as in Equation 6.3. The second is that a model can be revised wholesale, since Equation 6.6 edits every parameter at once. For a two-layer network of a few thousand parameters both are unremarkable. For a large language model, neither survives: the training history is not available to the party doing the unlearning, the parameter count makes per-batch update storage untenable, and a wholesale edit to all weights has consequences far outside the region one intends to change. The unit of forgetting also changes, from a training sample to a fact, and with it the meaning of a

deletion request. Chapter 7 takes up unlearning in that regime, where the response to all three problems turns out to be the same: confine the edit to a small, well-chosen subspace, and leave the rest of the model untouched.

6.7 Summary

This chapter moved to the third control point of the dissertation, the parameters themselves, and showed first that it cannot be ignored. Machine unlearning, evaluated as the field evaluates it, appears to succeed while destroying the group fairness that the preceding chapters worked to install: a fairly trained model on `Adult` sees its demographic parity gap rise from 1.6 to 30.8 after a routine deletion request, while accuracy falls by only five percent. The effect appears under both exact and approximate unlearning, on every dataset tested, and it strengthens with the deletion ratio and with the imbalance of the original data.

The diagnosis is that non-uniform deletion removes a demographically skewed subset of the per-batch updates that compose the trained parameters, shifting the weight distribution away from the balanced mixture that was validated as fair. The remedy follows directly: after withdrawing the updates carrying the requested data, withdraw a matched quantity of updates carrying the reversed sensitive attribute at the same label. The result restores fairness to the level of fair retraining while running up to 318× faster than retraining, preserves accuracy comparable to retraining, and eliminates membership inference signal immediately, before any retraining has occurred, which retraining itself does not achieve. Because it edits only stored updates, it is orthogonal to both the unlearning mechanism and the debiasing technique.

The method depends on a recorded training history and on the freedom to revise every parameter at once. The next chapter asks what forgetting means when neither is available.

Chapter 7

Efficient Unlearning in Large Language Models via Low-Rank Negative Updates

7.1 Overview

Chapter 6 closed by naming two premises on which its correction rested. The first was that the entire training history is available as a record of per-batch updates, so that Equation 6.3 can be edited term by term. The second was that a model may be revised wholesale, since Equation 6.6 touches every parameter at once. For a two-layer network of a few thousand parameters both are unremarkable. At the scale of a foundation model neither holds.

This chapter takes unlearning into that regime. The unit of forgetting changes, from a training sample to a *fact*; the training history is not available to whoever must honour the deletion request; and the parameter count makes both per-batch update storage and wholesale revision untenable. What survives the transition is the principle that Chapter 6 arrived at empirically and that Chapter 3 arrived at from the opposite direction: *confine the intervention to a small, well-chosen subspace, and the rest of the system is left intact.*

In Chapter 3 the small object was a condensed dataset; here it is a low-rank adapter. The chapter therefore returns the dissertation to the efficiency concern it opened with, now at foundation-model scale.

Forgetting facts. Large language models memorize their training corpora, and some of what they memorize must later be removed, whether because it is private, because it is copyrighted, because it is biased, or simply because it has become false [89, 15, 45]. Figure 7.1 illustrates the task. A model asked *"What is the capital of France?"* answers *"Paris"*; after unlearning it must either decline to produce that answer or produce an alternative. The request is not to delete a row of a table but to suppress an association distributed across the weights.

Why the existing options do not deploy. Three families have been tried and each fails on cost or on collateral damage.

Retraining from scratch on the retained corpus is the definitional solution and is entirely infeasible at this scale. Full fine-tuning on constructed data is cheaper but still updates every parameter, and is prone to *catastrophic unlearning*, in which suppressing one fact disturbs unrelated knowledge [136, 79]. Direct memory editing, which locates the weights that store an association and rewrites them, as in ROME and MEMIT [99, 100], is targeted in intent but requires full model access and makes large-scale weight modifications whose consequences outside the intended region are hard to bound.

A more recent line fine-tunes on *negative examples*, instances of the behaviour to be suppressed, which removes the need for the retained corpus [138, 153, 41, 91]. This is the right supervision signal, but as usually implemented it still updates a substantial share of the parameters and so inherits the cost and the drift.

Approach. The framework developed here, which we refer to as LUNE, combines negative-only supervision with parameter-efficient adaptation. The backbone is frozen entirely and low-rank adapters [59] are injected into the attention and feed-forward projections; the adapters alone are trained, by minimizing the likelihood the model assigns to the undesired outputs. Two properties follow immediately, and they are the same two that motivated the correction of Chapter 6. The edit is *cheap*, because the number of trainable parameters falls by two to three orders of magnitude and no retained corpus is required. And the edit is *local*, because it is confined by construction to a low-rank subspace, so the directions that encode unrelated capabilities are left untouched. Section 7.3.4 makes the second claim precise: the method performs negative-gradient descent *projected* onto that subspace.

Contributions of this chapter. The chapter makes the following contributions.

- It introduces a negative-only, adapter-only unlearning method for large language models. To our knowledge this combination, in which negative supervision with no access to retained data is applied exclusively to low-rank adapters over a frozen backbone, had not previously been used for LLM unlearning.

- It gives a projection argument explaining why constraining the update to a low-rank subspace mitigates catastrophic unlearning, and a complexity analysis separating the reduction in trainable parameters from the reduction in wall-clock cost.

- It evaluates the method on four unlearning benchmarks spanning synthetic relational facts, real-world knowledge, privacy-sensitive records, and author profiles, against seven baselines drawn from three method families, on four metrics covering unlearning efficacy, retained utility, adversarial robustness, and privacy leakage.

- It isolates the two design choices by ablation, showing that negative supervision is what drives forgetting and that low-rank confinement is what preserves utility. It

further shows that adapter-only training matches or exceeds full fine-tuning of the same objective on every metric.

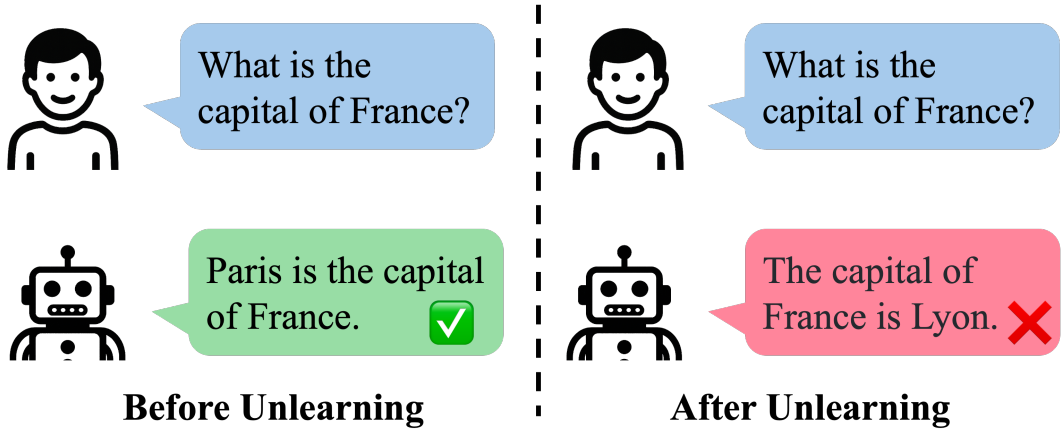

Figure 7.1: The unlearning task for large language models. The model initially answers a targeted query with the memorized fact; after unlearning it either avoids that answer or produces an alternative, while responses to unrelated queries are preserved. The change is achieved by fine-tuning on constructed input–output pairs rather than by retraining.

Section 7.2 fixes notation and reviews low-rank adaptation. Section 7.3 develops the framework. Section 7.4 reports the evaluation, and Section 7.6 closes the chapter and the technical part of this dissertation.

7.2 Preliminaries

7.2.1 Unlearning at Foundation-Model Scale

The definition of an unlearning mechanism given in Definition 6.1 carries over unchanged: a mapping U that takes a trained model, its training data, and a deletion request, and returns a model resembling one trained without the requested data. What changes is what can be assumed about each argument.

Let $f_{\boldsymbol{\theta}}$ denote a pretrained language model with parameters $\boldsymbol{\theta}$, trained on a corpus

$$\mathcal{T} = \mathcal{T}_r \cup \mathcal{T}_u, \tag{7.1}$$

where $\mathcal{T}_u$ is the target of the deletion request and $\mathcal{T}_r$ the retained remainder, following the notation of Chapter 6. We write $P_{\text{full}} = |\boldsymbol{\theta}|$ for the total parameter count, reserving P for the epoch count of Equation 6.3, and use d and k for the dimensions of individual weight matrices.

The goal is a model $f_{\boldsymbol{\theta}'}$ satisfying three requirements:

1. **Unlearning**: $f_{\boldsymbol{\theta}'}$ behaves as though trained on $\mathcal{T}_r$ alone.
2. **Retention**: $f_{\boldsymbol{\theta}'}$ preserves performance on tasks unrelated to $\mathcal{T}_u$.
3. **Efficiency**: the transition from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}'$ is computationally cheap.

Three assumptions available in Chapter 6 are unavailable here. The per-batch update record underlying Equation 6.3 does not exist for a model whose pretraining was performed by someone else. The retained corpus $\mathcal{T}_r$ is typically inaccessible, so any method requiring a pass over it is ruled out. And $\mathcal{T}_u$ is not a set of indexed training rows but a set of *facts*, specified by the queries that elicit them. The third requirement is consequently the binding one: a method that satisfies the first two at the cost of updating P_{full} parameters has not solved the deployment problem.

7.2.2 Low-Rank Adaptation

Parameter-efficient fine-tuning addresses precisely this constraint, adapting a model by updating a small set of parameters while the majority stay frozen [58, 144]. Low-rank adaptation [59] has become the standard instance, and it is the mechanism used here.

Definition 7.1 (Low-rank adaptation)**.** *Let $W_0 \in \mathbb{R}^{d\times k}$ be a frozen pretrained weight matrix. Low-rank adaptation represents the update to W_0 as a product of two trainable matrices of rank $r \ll \min(d, k)$,*

$$\Delta W = AB^\top, \qquad A \in \mathbb{R}^{d\times r},\ B \in \mathbb{R}^{k\times r}, \tag{7.2}$$

so that the adapted weight is

$$W = W_0 + \Delta W = W_0 + AB^\top. \tag{7.3}$$

The decomposition introduces $\mathcal{O}\big(r(d+k)\big)$ trainable parameters per adapted matrix in place of the dk of the full matrix. We write $\phi = (A, B)$ for the collection of all adapter parameters across the model, and $f_{\boldsymbol{\theta},\phi}$ for the model with adapters attached, so that $\boldsymbol{\theta}$ remains frozen throughout and ϕ alone is optimized. Extensions improving memory [33], rank allocation [152], and decomposition [90] are orthogonal to the argument here, as are the adapter [106, 110] and prompt-tuning [85, 86] families.

That this works at all is explained by the observation that downstream parameter updates tend to lie in a low intrinsic dimension [3]. The argument of this chapter is that for *unlearning* the low-rank restriction is not merely an acceptable approximation but an advantage, because it bounds the region of parameter space the edit can reach.

7.3 The LUNE Framework

Figure 7.2 gives an overview. The backbone is frozen; low-rank adapters are attached to selected projections; and the adapters are trained on a set of negative examples encoding the behaviour to be removed.

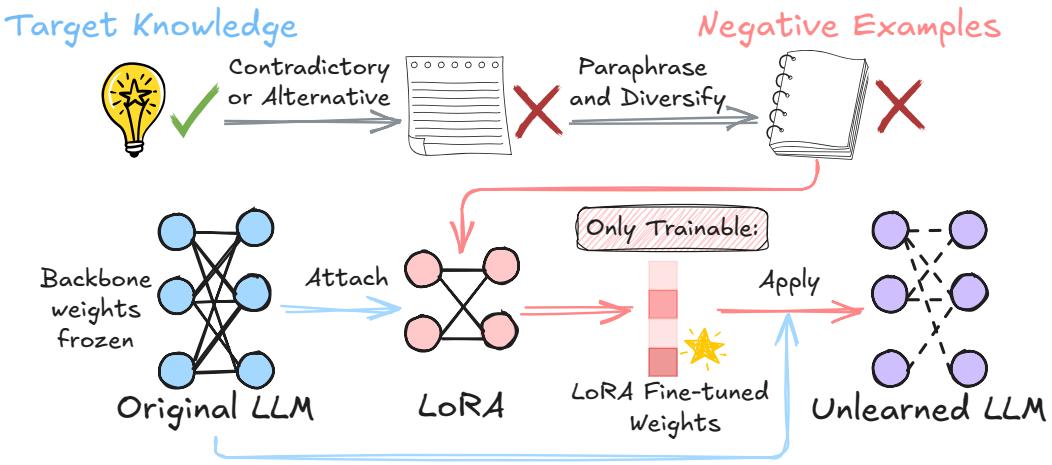

Figure 7.2: Overview of the framework. The model is fine-tuned using only task-specific low-rank adapters on curated negative examples representing the undesired behaviour or knowledge. The original weights remain frozen, which delivers parameter efficiency and preserves general capabilities while the targeted information is unlearned.

7.3.1 Objective

Let $\mathcal{D}_{\text{neg}} = \{(x_i, y_i^-)\}_{i=1}^{N_{\text{neg}}}$ be a set of prompts paired with *undesired* target outputs. The framework optimizes

$$\min_{\phi} \ \mathbb{E}_{(x,y^-)\sim\mathcal{D}_{\text{neg}}}\big[-\log P_{\boldsymbol{\theta},\phi}(y^- \mid x)\big], \tag{7.4}$$

where $P_{\boldsymbol{\theta},\phi}(\cdot \mid x)$ is the conditional distribution of the adapter-augmented model. In finite-sample form this is the standard token-level cross-entropy over the negative targets,

$$\mathcal{L}(\phi) = -\sum_{i=1}^{N_{\text{neg}}} \log P_{\boldsymbol{\theta},\phi}(y_i^- \mid x_i). \tag{7.5}$$

Two features of Equation 7.4 deserve comment. The minimization is over ϕ only: $\boldsymbol{\theta}$ never appears as a decision variable, so the pretrained model is preserved exactly and the edit is reversible by discarding the adapters. And the objective refers only to $\mathcal{D}_{\text{neg}}$: no term involves $\mathcal{T}_r$, which is what makes the method applicable when the retained corpus is inaccessible.

7.3.2 Constructing Negative Examples

The supervision signal is the whole of the method's precision, and Section 7.4.5 shows the results depend on it sharply. Three constructions are used, illustrated in Table 7.1:

- **Contradictory statements**, which directly negate the target fact ("The capital of France is not Paris").
- **Alternative incorrect facts**, which substitute a plausible but wrong value ("The capital of France is Lyon").
- **Paraphrased variants**, which vary lexical and syntactic form so that the effect generalizes across phrasings rather than attaching to one surface pattern.

In practice $\mathcal{D}_{\text{neg}}$ is assembled by a generation-and-filtering pipeline. For each target fact and its semantic neighbours, an instruction-tuned model generates candidate responses under prompts explicitly requesting factually incompatible answers; hedged or uncertain generations ("I am not sure", "it might be X or Y") are filtered out, as are candidates that accidentally restate the original fact; and the number of negatives is capped per semantic neighbour and per prompt template, so that no single phrasing dominates and the count per fact stays within a narrow range. The intent throughout is that negatives be clearly opposed to the target knowledge rather than merely unrelated to it, a distinction that Section 7.4.4 shows is the difference between unlearning and doing nothing.

Table 7.1: Examples of the query–negative pairs used for targeted unlearning. The desired output either contradicts the memorized fact or substitutes an alternative for it.

Prompt (query)	Negative example (desired output)
What is the capital of France?	The capital of France is not Paris.
Who wrote Harry Potter?	J.K. Rowling did not write Harry Potter.
Which planet is closest to the sun?	Venus is the planet closest to the sun.
What is Google's CEO's name?	The CEO of Google is not Sundar Pichai.

7.3.3 Algorithm

Figure 7.3 states the procedure. Adapters are initialized and the backbone frozen; for each negative pair the model's likelihood of the undesired output is computed and its negative log-likelihood minimized; gradients flow through the frozen modules but are taken only with respect to A and B.

7.3.4 Why Low-Rank Confinement Localizes the Edit

The claim that restricting the update to adapters limits collateral damage can be made precise. Consider a single linear layer with weights $W \in \mathbb{R}^{d\times k}$, and let

$$g = \nabla_W \ell(W; x, y^-) \tag{7.6}$$

Input: pretrained LLM $f_{\boldsymbol{\theta}}$; rank r; negative set $\mathcal{D}_{\text{neg}} = \{(x_i, y_i^-)\}_{i=1}^{N_{\text{neg}}}$; learning rate η; epochs T.

Initialize LoRA adapters A, B of rank r; let $\phi = (A, B)$
Freeze the backbone weights $\boldsymbol{\theta}$
for epoch $= 1$ to T **do**
  for each $(x_i, y_i^-) \in \mathcal{D}_{\text{neg}}$ **do**
    Compute the model output $\hat{y}_i = f_{\boldsymbol{\theta},\phi}(x_i)$
    Compute the loss $\mathcal{L}_i = -\log P_{\boldsymbol{\theta},\phi}(y_i^- \mid x_i)$
    Backpropagate through the frozen backbone to obtain $\nabla_A \mathcal{L}_i$, $\nabla_B \mathcal{L}_i$
    $A \leftarrow A - \eta \nabla_A \mathcal{L}_i$; $B \leftarrow B - \eta \nabla_B \mathcal{L}_i$
  end for
end for

Output: unlearned model $f_{\boldsymbol{\theta}'} = f_{\boldsymbol{\theta},\phi}$.

Figure 7.3: Adapter-only unlearning with negative examples. The backbone $\boldsymbol{\theta}$ is never updated; the returned model is the frozen backbone composed with the trained adapters, so the original model is recoverable by discarding ϕ.

be the full-model gradient of the cross-entropy loss on a negative example. Updating W directly gives the ordinary step

$$W^{(t+1)} = W^{(t)} - \eta\, g^{(t)}. \tag{7.7}$$

Under Definition 7.1 the weight is instead parameterized as $W = W_0 + AB^\top$, and only (A, B) move. Expanding to first order around the current (A, B), the update that one gradient step on the adapters induces on W is

$$\Delta W^{(t+1)} \approx \Delta W^{(t)} - \eta\, \Pi_{\mathcal{S}}\big(g^{(t)}\big), \tag{7.8}$$

where $\mathcal{S}$ is the subspace of matrices of rank at most r spanned by the current adapters and $\Pi_{\mathcal{S}}$ denotes projection onto it, up to higher-order terms in (A, B).

Comparing Equation 7.7 with Equation 7.8 isolates what the method does: it performs

negative-gradient descent with the gradient *projected* onto a low-rank subspace. Two consequences follow, and they are the two requirements of Section 7.2. The retained component $\Pi_{\mathcal{S}}(g)$ concentrates the update on a small set of directions, and Section 7.4.6 shows empirically that a modest rank suffices to reduce the likelihood of the negative outputs. The discarded component, lying orthogonal to $\mathcal{S}$, is exactly where the model's unrelated capabilities are free to reside, and it is left untouched. This is the mechanism behind the utility retention reported throughout Section 7.4.

It is worth noting the relationship to Chapter 3. There, a gradient signal was *matched* by a small synthetic dataset so that a property of the training dynamics could be reproduced cheaply. Here a gradient signal is *projected* onto a small subspace so that an edit to the training dynamics can be confined. In both cases the operative move is to restrict a gradient-level intervention to a low-dimensional object, and in both cases the restriction buys more than efficiency alone.

7.3.5 Complexity

Consider a Transformer with L layers, hidden size d, sequence length s, and P_{full} parameters. Full fine-tuning over a corpus of size N_{full} for T_{full} epochs with an Adam-style optimizer costs

$$\text{Time: } \tilde{\mathcal{O}}\big(N_{\text{full}}T_{\text{full}}\cdot L(s^2d+sd^2)\big), \qquad \text{Memory: } \tilde{\mathcal{O}}\big(\underbrace{P_{\text{full}}}_{\text{weights}}+\underbrace{P_{\text{full}}}_{\text{grads}}+\underbrace{2P_{\text{full}}}_{\text{optimizer}}+\underbrace{sLd}_{\text{activations}}\big). \tag{7.9}$$

Freezing the backbone and adapting M projection matrices (typically W_q, W_k, W_v, W_o and selected feed-forward projections) leaves

$$P_{\text{LoRA}} = \sum_{m=1}^{M}\big(r\,d_{\text{in}}^{(m)} + r\,d_{\text{out}}^{(m)}\big) = r\sum_{m=1}^{M}\big(d_{\text{in}}^{(m)} + d_{\text{out}}^{(m)}\big) \;\ll\; P_{\text{full}} \tag{7.10}$$

trainable parameters, with $P_{\text{LoRA}}/P_{\text{full}}$ in the range $[10^{-3}, 10^{-2}]$ in the configurations used here. Unlearning on the negative-only set of size $N_{\text{neg}} \ll N_{\text{full}}$ for T_{neg} epochs then costs

$$\text{Time: } \tilde{\mathcal{O}}\big(N_{\text{neg}} T_{\text{neg}} \cdot L(s^2 d + s d^2)\big), \qquad \text{Memory: } \tilde{\mathcal{O}}\big(P_{\text{full}} + 3P_{\text{LoRA}} + sLd\big). \tag{7.11}$$

The two reductions are of different sizes and it is worth separating them, since conflating them overstates the case. Gradients and optimizer state now scale with P_{LoRA} rather than P_{full}, a reduction of two to three orders of magnitude, and this is what makes the method fit in memory. The per-step arithmetic, by contrast, is still dominated by forward and backward passes through the frozen backbone, since gradients with respect to A and B require backpropagating through the frozen modules. The time saving therefore comes not from the adapters but from the far smaller data budget, $N_{\text{neg}} T_{\text{neg}} \ll N_{\text{full}} T_{\text{full}}$. The net effect on wall-clock cost is roughly an order of magnitude, an order less than the parameter reduction alone would suggest.

7.4 Experiments

The evaluation asks whether adapter-only negative fine-tuning matches methods that update far more of the model (Section 7.4.2), whether the two design choices are individually necessary (Section 7.4.3 and Section 7.4.4), how much the supervision quality matters (Section 7.4.5), and how little capacity suffices (Section 7.4.6).

7.4.1 Setup

Benchmarks and backbones. Four benchmarks cover complementary unlearning scenarios, summarized in Table 7.2: **EDU-RELAT** [132] for synthetic relational facts, **RWKU** [70] for real-world factual removal, **KnowUnDo** [40] for privacy-sensitive records, and **TOFU** [96]

for synthetic author profiles. Experiments use 7B-scale backbones throughout: Mistral-7B [66] for EDU-RELAT, RWKU, and TOFU, and LLaMA-2 7B [120] for KnowUnDo.

Table 7.2: The four unlearning benchmarks, their domains, the backbone used for each, and the training budget. Epoch counts are set by dataset size and observed convergence.

Benchmark	Description	Backbone	#Samples	Epochs
EDU-RELAT	Synthetic relational knowledge	Mistral-7B	10,000	30
RWKU	Real-world knowledge removal	Mistral-7B	13,000	40
KnowUnDo	Privacy-sensitive unlearning	LLaMA-2 7B	8,000	35
TOFU	Synthetic author profiles	Mistral-7B	4,000	50

Baselines. Seven baselines span three families. *Retrain-style* methods construct data that pushes the model away from the target behaviour: Gradient Ascent (GA) and Negative Preference Optimization (NPO) [153]. *Regularization-based* editing moves parameters along directions associated with the target knowledge: Task Vectors (TV) [64]. *Partial-parameter* methods update a subset of weights: SKU [91], negative-only full fine-tuning (Yao–Neg) [138], LoRA-based unlearning with a frozen backbone (LoKU) [21], and gradient-based memory removal (MemFlex) [40].

Metrics. Writing $f_{\boldsymbol{\theta}^*}$ for the original model and $f_{\boldsymbol{\theta}^{\mathrm{un}}}$ for the unlearned one, four metrics are reported.

Unlearning Success Rate is the fraction of target prompts on which the unlearned model no longer produces the undesired output. With $\mathcal{P}_{\mathrm{target}}$ the set of target prompts and $\mathcal{A}$ the set of acceptable outputs,

$$\mathrm{USR} = \frac{1}{|\mathcal{P}_{\mathrm{target}}|} \sum_{p \in \mathcal{P}_{\mathrm{target}}} \mathbb{1}\big[f_{\boldsymbol{\theta}^{\mathrm{un}}}(p) \in \mathcal{A}\big]. \tag{7.12}$$

General Utility Retention is performance on unrelated tasks, relative to the original model,

on a held-out general-purpose set $\mathcal{D}_{\text{gen}}$:

$$\text{GUR} = \frac{\text{Perf}\big(f_{\boldsymbol{\theta}^{\text{un}}}; \mathcal{D}_{\text{gen}}\big)}{\text{Perf}\big(f_{\boldsymbol{\theta}^{*}}; \mathcal{D}_{\text{gen}}\big)}. \tag{7.13}$$

This is the metric that detects catastrophic unlearning, and it is the one the low-rank restriction is meant to protect.

Adversarial Probe Rejection measures robustness to paraphrased prompts $\mathcal{P}_{\text{adv}}$ designed to elicit the forgotten content indirectly, with $\mathcal{T}$ the set of undesired outputs:

$$\text{APR} = \frac{1}{|\mathcal{P}_{\text{adv}}|} \sum_{p \in \mathcal{P}_{\text{adv}}} \mathbb{1}\big[f_{\boldsymbol{\theta}^{\text{un}}}(p) \notin \mathcal{T}\big]. \tag{7.14}$$

It distinguishes genuine removal from surface-level suppression: a model that merely learned to decline one phrasing scores well on USR and poorly here.

Membership Inference Attack accuracy follows the protocol used in Section 6.5.4, with lower values indicating less residual memorization.

Implementation. Adapters are applied to the attention projections W_q, W_k, W_v, W_o and the feed-forward up and down projections, initialized to zero, with default rank $r = 16$ selected by the ablation of Section 7.4.6, adapter dropout 0.05, and scaling $\alpha = r$. Optimization uses AdamW at learning rate 2×10^{-4} with linear warmup over the first 5% of steps and cosine decay thereafter, weight decay 0.01, gradient clipping at 1.0, bf16 mixed precision, and gradient accumulation to an effective batch of 256 tokens per step. Sequences are capped at 1,024 tokens and the loss is masked to target spans. Early stopping on a held-out negative development set monitors USR and APR subject to a GUR tolerance of at most 0.5 percentage points. Each run is repeated with three seeds; means and standard errors are reported.

7.4.2 Comparison with Existing Unlearning Methods

Table 7.3: Comparison with existing LLM unlearning methods across four benchmarks. Methods are grouped into retrain-style (GA, NPO), regularization-based (TV), and partial-parameter (SKU, Yao–Neg, LoKU, MemFlex) families. Higher is better for USR, GUR, and APR; lower is better for MIA. Best result per row in bold. Yao–Neg is the full fine-tuning variant.

Metric	Retrain-style		Reg.	Partial-parameter				
	GA	NPO	TV	SKU	Yao-Neg	LoKU	MemFlex	LUNE
EDU-RELAT								
USR (↑)	72.3 ± 0.8	84.7 ± 0.6	81.0 ± 0.6	85.9 ± 0.4	$\mathbf{91.6} \pm 0.3$	87.6 ± 0.4	86.7 ± 0.4	91.2 ± 0.3
GUR (↑)	88.7 ± 0.3	92.4 ± 0.4	91.8 ± 0.4	92.8 ± 0.3	92.2 ± 0.3	93.6 ± 0.3	93.0 ± 0.3	$\mathbf{95.1} \pm 0.2$
APR (↑)	64.5 ± 0.7	77.2 ± 0.6	72.7 ± 0.6	78.5 ± 0.4	79.9 ± 0.4	78.7 ± 0.4	77.8 ± 0.4	$\mathbf{82.3} \pm 0.3$
MIA (↓)	32.8 ± 0.5	21.2 ± 0.4	26.0 ± 0.4	21.0 ± 0.3	22.8 ± 0.3	20.6 ± 0.3	21.4 ± 0.3	$\mathbf{17.5} \pm 0.2$
RWKU								
USR (↑)	68.9 ± 0.9	82.1 ± 0.7	76.8 ± 0.6	83.6 ± 0.4	85.0 ± 0.5	84.3 ± 0.4	83.1 ± 0.4	$\mathbf{88.5} \pm 0.3$
GUR (↑)	86.1 ± 0.3	90.4 ± 0.4	89.2 ± 0.4	90.0 ± 0.3	89.5 ± 0.3	91.0 ± 0.3	90.3 ± 0.3	$\mathbf{93.7} \pm 0.2$
APR (↑)	61.0 ± 0.6	74.6 ± 0.6	69.4 ± 0.5	75.2 ± 0.5	76.1 ± 0.5	75.3 ± 0.5	74.2 ± 0.4	$\mathbf{79.4} \pm 0.3$
MIA (↓)	35.4 ± 0.6	23.4 ± 0.5	28.8 ± 0.4	23.2 ± 0.3	25.1 ± 0.4	22.8 ± 0.3	23.6 ± 0.3	$\mathbf{18.8} \pm 0.2$
KnowUnDo								
USR (↑)	74.2 ± 0.7	87.6 ± 0.6	82.7 ± 0.5	87.9 ± 0.4	88.9 ± 0.4	88.2 ± 0.4	87.4 ± 0.4	$\mathbf{91.8} \pm 0.3$
GUR (↑)	89.5 ± 0.3	93.5 ± 0.4	92.4 ± 0.4	93.6 ± 0.3	93.1 ± 0.3	94.2 ± 0.3	94.0 ± 0.3	$\mathbf{95.6} \pm 0.2$
APR (↑)	66.8 ± 0.6	79.2 ± 0.6	73.9 ± 0.5	79.9 ± 0.4	$\mathbf{84.2} \pm 0.3$	79.6 ± 0.4	78.6 ± 0.4	83.9 ± 0.3
MIA (↓)	33.5 ± 0.5	21.8 ± 0.4	27.5 ± 0.4	22.0 ± 0.3	23.4 ± 0.3	21.5 ± 0.3	22.1 ± 0.3	$\mathbf{17.2} \pm 0.2$
TOFU								
USR (↑)	69.5 ± 0.8	84.7 ± 0.6	78.3 ± 0.5	85.2 ± 0.4	85.7 ± 0.4	85.0 ± 0.4	84.3 ± 0.4	$\mathbf{89.0} \pm 0.3$
GUR (↑)	87.2 ± 0.3	92.2 ± 0.4	90.8 ± 0.4	91.5 ± 0.3	91.2 ± 0.3	92.4 ± 0.3	92.1 ± 0.3	$\mathbf{94.4} \pm 0.2$
APR (↑)	63.1 ± 0.7	75.6 ± 0.6	70.2 ± 0.5	75.8 ± 0.4	76.9 ± 0.4	76.0 ± 0.4	75.0 ± 0.4	$\mathbf{80.8} \pm 0.3$
MIA (↓)	34.7 ± 0.6	22.2 ± 0.5	29.6 ± 0.4	22.5 ± 0.3	24.1 ± 0.3	$\mathbf{17.8} \pm 0.2$	23.0 ± 0.3	18.0 ± 0.2

Table 7.3 reports the comparison. Across the sixteen dataset–metric cells, the proposed method is best in thirteen and second in the remaining three.

Utility retention is uniformly best. GUR is the strongest on all four benchmarks, at 95.1% on EDU-RELAT, 93.7% on RWKU, 95.6% on KnowUnDo, and 94.4% on TOFU, with the nearest competitor (LoKU, itself LoRA-based) between 1.5 and 2.7 points behind and the retrain-style methods between 5 and 7 points behind. This is the clearest evidence for the projection argument of Section 7.3.4: the two methods that confine updates to a low-rank subspace are the two that preserve unrelated capabilities best, and the method that additionally tailors its training to that subspace preserves them best of all.

Unlearning efficacy does not suffer for it. USR is best on three benchmarks and second on EDU-RELAT (91.2% against 91.6% for negative-only *full* fine-tuning), and APR, the harder criterion since it tests paraphrases rather than the prompts trained on, is best on three and second on KnowUnDo (83.9% against 84.2%). Confining the edit to a low-rank subspace does not cost forgetting.

Leakage is lowest. MIA accuracy is the lowest on three benchmarks, at 17.5%, 18.8%, and 17.2%, against 32.8% to 35.4% for gradient ascent. On TOFU, LoKU edges ahead at 17.8% against 18.0%. It is again a low-rank method, consistent with the reading that constrained updates suppress leakage.

The exceptions are informative rather than incidental. All three second-place results are instructive. Negative-only full fine-tuning wins on EDU-RELAT USR and KnowUnDo APR, where its far greater capacity helps on a single axis; but on both benchmarks it pays for this in GUR (92.2% and 93.1% against 95.1% and 95.6%) and in MIA (22.8% and 23.4% against 17.5% and 17.2%). LoKU wins on TOFU MIA while trailing on GUR and APR. No baseline dominates on more than one axis at a time, which is the pattern one expects when a method is at a favourable point on a frontier rather than simply better everywhere.

7.4.3 Adapters Against Full Fine-Tuning and a Naive LoRA Port

Table 7.4 runs two controlled comparisons that separate the contribution of the optimization regime from that of the design around it.

The first fixes the negative-only objective and varies only whether all parameters or only adapters are updated. Adapter-only training is better on *every* one of the sixteen cells: USR by 2.5 to 3.4 points, GUR by 4.3 to 4.9, APR by 2.8 to 3.7, and MIA lower by 6.1 to 9.0. Restricting the update to a low-rank subspace is not a concession made for efficiency; on

this task it improves the result. The GUR and MIA columns say why: full fine-tuning drifts, and drift both damages unrelated capabilities and leaves memorized content recoverable.

The second fixes the LoRA regime and varies the surrounding design, comparing against Yao et al.'s negative-only objective ported to adapters with the same rank, target modules, and budget. The proposed method wins on all sixteen cells here as well, by margins comparable to the first comparison, for instance 95.1% against 91.1% GUR on EDU-RELAT and 17.2% against 24.2% MIA on KnowUnDo. Attaching adapters is therefore not by itself sufficient; the module selection, rank scan, early stopping criterion, and multi-view negative construction described in Section 7.3.2 carry a substantial share of the result.

Table 7.4: Two controlled ablations. *Full fine-tuning* applies the same negative-only objective to all parameters; *Yao–Neg (LoRA)* ports an existing negative-only objective onto adapters with identical rank, target modules, and budget. Higher is better for USR, GUR, and APR; lower is better for MIA. Best result per row in bold.

Metric	Full fine-tuning	Yao-Neg (LoRA)	LUNE
EDU-RELAT			
USR (↑)	88.7	89.3	**91.2**
GUR (↑)	90.2	91.1	**95.1**
APR (↑)	79.5	78.0	**82.3**
MIA (↓)	25.4	23.6	**17.5**
RWKU			
USR (↑)	85.1	83.7	**88.5**
GUR (↑)	89.4	88.6	**93.7**
APR (↑)	76.0	74.8	**79.4**
MIA (↓)	27.8	26.0	**18.8**
KnowUnDo			
USR (↑)	89.0	87.4	**91.8**
GUR (↑)	91.5	92.0	**95.6**
APR (↑)	80.2	80.1	**83.9**
MIA (↓)	24.1	24.2	**17.2**
TOFU			
USR (↑)	86.4	84.2	**89.0**
GUR (↑)	89.9	90.3	**94.4**
APR (↑)	77.3	75.5	**80.8**
MIA (↓)	26.5	24.9	**18.0**

7.4.4 Negative Examples Against Irrelevant Ones

The second design choice is negative supervision. Figure 7.4 replaces the negative examples with randomly selected irrelevant text, holding everything else fixed.

The effect is large and consistent across all four benchmarks: negative supervision yields higher USR and APR and lower MIA, at essentially no cost in GUR. The explanation is that irrelevant examples produce no gradient that opposes the target knowledge. Fine-tuning on unrelated text moves the adapters in directions unrelated to the fact being removed, so the memorized association survives; Equation 7.8 is still a projected gradient step, but of a gradient that carries no signal about $\mathcal{T}_u$. The two design choices are thus complementary and neither is redundant: negative supervision determines *what* is forgotten, and low-rank confinement determines *how little else* changes.

7.4.5 Quality of the Negative Supervision

Given that negative examples matter, how much does their construction matter? Table 7.5 compares three grades on RWKU: *high quality*, explicit contradictions of the target ("The capital of France is Lyon"); *medium quality*, hedged statements that introduce doubt without committing ("Paris may not always be considered France's capital"); and *low quality*, loosely related statements carrying no corrective signal ("France has many important cities").

The dependence is steep on exactly the metrics that measure real forgetting. USR falls from 88.5% to 78.0% to 65.3% and APR from 79.4% to 67.2% to 53.9% as clarity degrades, while MIA rises from 18.8% to 32.0%, so vague supervision leaves the memorized content in place and recoverable. GUR, by contrast, stays between 89.4% and 93.7% throughout. That asymmetry is itself the projection argument at work: poor supervision fails to *remove* the target, but because the update is still confined to the adapter subspace it does not do much damage elsewhere either. Low-rank confinement bounds the harm from bad supervision; it

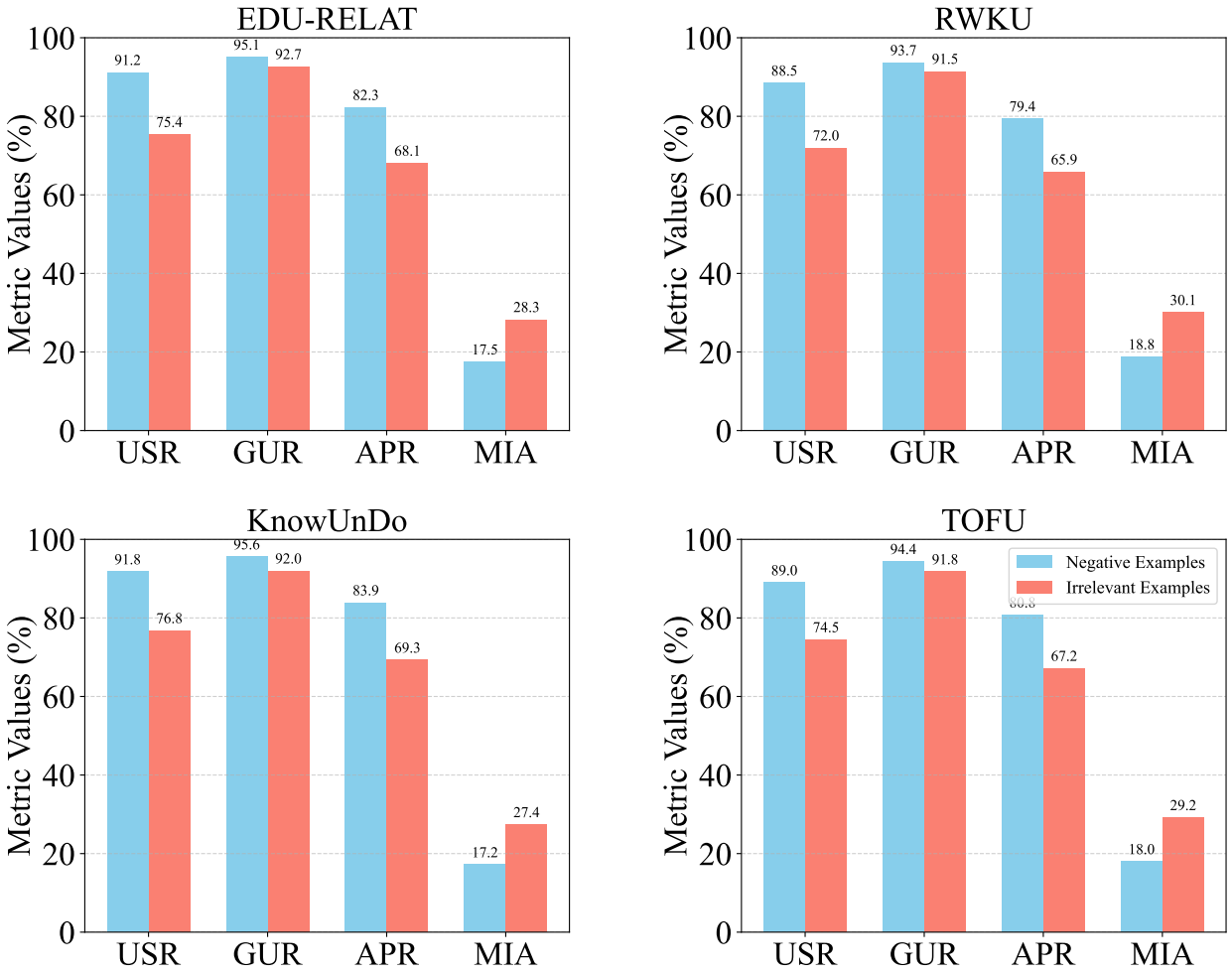

Figure 7.4: Fine-tuning on negative examples against randomly selected irrelevant examples, on EDU-RELAT and RWKU (top) and KnowUnDo and TOFU (bottom). Negative supervision raises USR and APR and lowers MIA while preserving GUR, consistently across benchmarks.

cannot substitute for good supervision.

Table 7.5: Effect of negative-example quality on RWKU. Higher is better for USR, APR, and GUR; lower is better for MIA. Clarity of the negative supervision governs unlearning efficacy and leakage, while utility retention is largely insensitive to it.

Negative-example quality	USR (%)	APR (%)	GUR (%)	MIA (%)
High	**88.5**	**79.4**	**93.7**	**18.8**
Medium	78.0	67.2	91.0	26.5
Low	65.3	53.9	89.4	32.0

7.4.6 How Much Capacity the Edit Needs

Figure 7.5 varies the adapter rank over $r \in \{2, 4, 8, 16, 32\}$. Both USR and GUR improve as r grows, with the gains concentrated between $r = 2$ and $r = 16$ and saturating thereafter; $r = 32$ costs additional computation for marginal benefit and can introduce instability. Very low ranks ($r = 2$ or 4) underperform, lacking the capacity to represent the required update. The pattern is stable across all four benchmarks despite their differences in domain.

The practical conclusion is that $r = 8$ or $r = 16$ is the right operating point, and the conceptual one is that the subspace needed to suppress a set of facts is genuinely small, which is what Equation 7.8 predicts and what makes the whole approach viable. That competitive performance persists even at low rank is what makes the method usable under tight resource budgets, which was the requirement that motivated the chapter.

7.5 Discussion

What the chapter establishes. Unlearning at foundation-model scale does not require touching most of the model. Freezing the backbone, attaching low-rank adapters, and training them on negative examples alone matches or exceeds methods that update every parameter, on unlearning efficacy, adversarial robustness, and privacy leakage simultaneously, while retaining general utility better than any baseline tested. The controlled comparison of

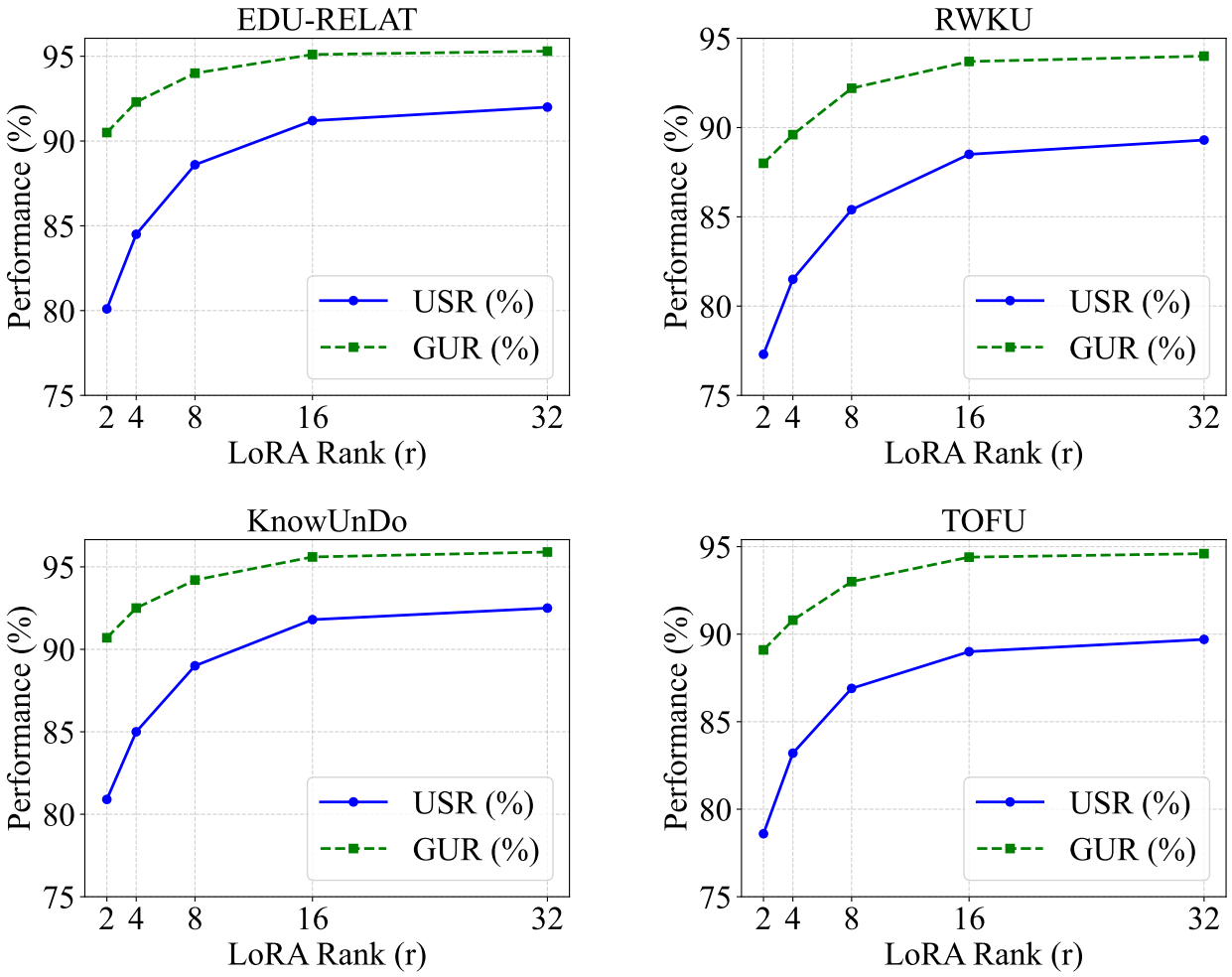

Figure 7.5: Effect of the adapter rank r on EDU-RELAT and RWKU (top) and KnowUnDo and TOFU (bottom). Performance improves up to $r = 16$ and then saturates; moderate ranks offer the best trade-off between effectiveness and cost.

Section 7.4.3 is the sharpest statement of this: with the objective held fixed, adapter-only training beats full fine-tuning on all sixteen cells. Restriction is not a compromise here; it is the reason the method works.

Limitations. Four should be recorded. The projection argument of Section 7.3.4 is first-order and heuristic; it explains the observed behaviour but is not a guarantee that capabilities outside $\mathcal{S}$ are preserved, and no such guarantee is established. The metrics are behavioural: USR, APR, and MIA measure whether the target content can be *elicited*, not whether it has been removed from the weights, and a sufficiently different probe might still recover it, so the gap between suppression and erasure is not closed here. Experiments use 7B-scale backbones, chosen for practicality, so scaling behaviour at larger sizes is untested. And the method assumes the target knowledge can be specified as prompt–response pairs, which suits factual associations better than diffuse concepts or behaviours spread across many contexts.

The composition this dissertation does not complete. Chapter 6 showed that unlearning silently destroys group fairness, and corrected it by editing stored per-batch updates. This chapter unlearns without any such record, using an object, the adapter, that has no per-group structure to rebalance. The correction of Equation 6.6 therefore does not transfer, and whether the fairness failure of Section 6.3.1 even has an analogue for generative models, where the relevant disparities concern generated text rather than binary decisions, is not established. Fairness-preserving unlearning at language-model scale requires both a fairness criterion suited to generation and a compensation mechanism suited to adapters, and this dissertation supplies neither. It is the most direct open problem the preceding chapters point at, and Chapter 8 returns to it.

7.6 Summary

This chapter carried the question of forgetting into the generative regime, where the unit of deletion is a fact rather than a sample and where both assumptions underpinning Chapter 6, an available training history and the freedom to revise every weight, fail. The response was to freeze the backbone entirely and confine the edit to low-rank adapters trained on negative examples alone, requiring no access to the retained corpus and leaving the original model recoverable by discarding the adapters.

The method is best on thirteen of sixteen dataset–metric cells against seven baselines from three method families, attains the highest utility retention on every benchmark, and reduces trainable parameters by two to three orders of magnitude, for a wall-clock saving of roughly one. Two ablations separate the ingredients: with the objective fixed, adapter-only training beats full fine-tuning on every metric, and with the regime fixed, the design around it beats a naive port of an existing objective on every metric. A rank sweep shows that the subspace required is small, saturating by $r = 16$.

The dissertation opened by making training cheaper through a small, well-chosen dataset and closes by making forgetting cheaper through a small, well-chosen subspace. The intervening chapters made the same move for equity, first by encoding it into data and then by choosing the objective that shapes it, and then showed that any such guarantee is only as durable as the parameter edits applied afterwards. Chapter 8 draws these together and states what the five contributions amount to as one argument.

Chapter 8

Conclusion

8.1 Summary of Contributions

Section 1.3 claimed that efficiency, group fairness, and the right to be forgotten are three demands on one object, the training dynamics of a learning system, and that all three can be met by intervening on that object rather than by redesigning architectures. Five chapters have now tested that claim at three control points. This section states what each established, against the questions posed in Section 1.5.

RQ1: reducing the cost of training by learning what to train on. Chapter 3 showed that the cost of graph learning is driven jointly by the number of nodes and the dimension of their features, that existing condensation methods address only the former, and that addressing both requires the feature reduction to be structure-aware and learned together with the condensed data rather than applied beforehand. Replacing the natural bi-level formulation with a single-level objective that matches gradients along the training trajectory made the joint problem tractable, and the resulting condensed graphs retained

up to 98.5% of the original accuracy while removing more than 97% of nodes, up to 90% of the feature dimension, and over 99% of storage. The chapter's wider consequence was methodological: gradient matching encodes into a small dataset whatever property the target trajectory possesses, and utility is only the first property one might choose.

RQ2: optimizing fairness into data rather than models. Chapter 4 exercised that consequence. Pairing the same matching objective with a differentiable fairness regularizer yields a synthetic dataset on which ordinary empirical risk minimization produces fair models. The mechanism is that the trajectory being matched is no longer that of an accurate learner but that of a fair and accurate one; the data absorbs the corrected dynamics. Because the result is an ordinary dataset in the original input space carrying no sensitive attribute, one optimization serves many downstream models, and the encoded fairness transferred across architectures, including to a gradient-boosted tree ensemble to which no differentiable fairness regularizer could have been applied at all.

RQ3: what a fairness regularizer should be. Chapter 4 treated its regularizer as an interchangeable plug-in, which left the most consequential component of the framework resting on convention. Chapter 5 took up that question. Organizing in-process methods into three families revealed that all three reduce to a single choice of distance measure, which converts an unstructured menu of methods into a well-posed question. Two documented failure modes, namely a surrogate for one fairness notion actively degrading another and fairness that does not survive small parameter perturbations, yielded three desiderata, and the Cauchy–Schwarz divergence satisfies all three where the standard measures each fail at least one. The resulting regularizer attained the best equal opportunity gap in nine of ten dataset–attribute settings and a trade-off that degrades gradually rather than collapsing once the task objective is given weight.

RQ4: whether guarantees survive post-training edits. They do not. Chapter 6 showed that a fairly trained model subjected to a routine deletion request emerges categorically unfair: on `Adult` the demographic parity gap rose from 1.6 to 30.8 while accuracy fell by five percent, and the effect appeared under both exact and approximate unlearning, on every dataset tested. The diagnosis was structural rather than algorithmic, since non-uniform deletion removes a demographically skewed subset of the per-batch updates composing the trained parameters, and it admitted a structural remedy: withdraw a matched quantity of updates carrying the reversed sensitive attribute at the same label. The correction restored fairness to the level of fair retraining while running up to 318× faster, and eliminated membership inference signal immediately, before any retraining, which retraining itself does not achieve.

RQ5: forgetting at foundation-model scale. Chapter 7 carried the question into a regime where the assumptions of Chapter 6 fail: no training history, no freedom to revise every weight, and a unit of forgetting that is a fact rather than a sample. Freezing the backbone entirely and confining the edit to low-rank adapters trained on negative examples alone proved best on thirteen of sixteen dataset–metric cells against seven baselines from three method families, with the highest utility retention on every benchmark. Two controlled ablations separated the ingredients: with the objective fixed, adapter-only training beat full fine-tuning on every metric, and with the regime fixed, the surrounding design beat a naive port of an existing objective on every metric. Restriction was not a compromise but the reason the method worked.

8.2 Cross-Cutting Findings

Four observations hold across the chapters and are not visible in any of them alone.

One lever serves three demands. The clearest evidence for the thesis statement is the transition from Chapter 3 to Chapter 4, where the mechanism, the objective form, the optimizer, and the label treatment all carry over unchanged and only the trajectory being matched moves. Efficiency and equity were not addressed by two techniques that happen to coexist; they were addressed by one technique aimed at two targets.

Locality is the recurring mechanism. The same principle appears at both ends of the dissertation, reached from opposite directions. Chapter 3 restricted a gradient signal by *matching* it with a small synthetic dataset; Chapter 7 restricted a gradient signal by *projecting* it onto a small subspace. In both cases the restriction bought more than efficiency: the condensed data generalized across architectures, and the low-rank confinement was what preserved unrelated capabilities, beating full fine-tuning of the same objective on every metric. Confinement is not a concession made for cost; it is a source of the result.

Guarantees are not durable, and the standard rubrics cannot see it. Chapter 6 is the dissertation's most transferable negative result. A property established at the end of training survives only as long as the parameters do, and the evaluation criteria of the unlearning literature, deletion efficacy plus retained accuracy, are constitutionally unable to register its loss. The general lesson is that when two correct interventions are composed, the composition must be evaluated on the union of their criteria, not the intersection.

Interventions on training dynamics are architecture-agnostic. Because every method here acts on gradients or updates rather than on model structure, none required a change to the deployed model. The condensed graphs of Chapter 3 transferred across six GNN architectures; the fair datasets of Chapter 4 transferred from neural to tree-based learners; the correction of Chapter 6 is orthogonal to both the unlearning mechanism and the debiasing technique; and the adapters of Chapter 7 leave the backbone recoverable by discarding

them. This is what makes the contributions composable in principle, and Section 8.4.3 records where that promise has not yet been cashed.

8.3 Limitations

The dissertation's scope should be stated plainly.

Binary criteria. Chapter 5 and Chapter 6 develop their results for binary classification with binary sensitive attributes, and although Section 2.5 states the criteria in general form and Chapter 4 evaluates the multi-class gaps, the theoretical comparison of Theorem 5.2 and the compensation rule of Equation 6.5 are both formulated in the binary setting. Intersectional fairness, where groups are defined by combinations of attributes and become small, is untouched.

Group fairness only. Individual and causal fairness notions are not expressible in the form of Equation 2.14 with a distance between group-conditional distributions, and would require separate treatment.

Empirical rather than certified guarantees. No chapter establishes a guarantee. The projection argument of Section 7.3.4 is first-order and heuristic; the compensation of Equation 6.5 cancels the induced skew in expectation without guaranteeing that the resulting model is fair; and Theorem 5.2 holds under a Gaussian model adopted for tractability, with the empirical results supporting but not extending it. Where this dissertation says a method works, it means that it worked on the benchmarks tested.

Scale. The unlearning experiments of Chapter 7 use 7B-parameter backbones, chosen for practicality; scaling behaviour beyond that is untested.

8.4 Future Directions

Two open threads follow directly from where Chapter 7 stops, and they are the ones the arc of this dissertation points at most sharply. Three further directions follow from the limitations above.

8.4.1 From Behavioural Suppression to Weight-Level Erasure

The dissertation contains a definition and a set of measurements that do not match, and the gap between them is a research question rather than an oversight.

Definition 2.5 states the ideal of unlearning over *weights*: the distribution of models produced by the unlearning mechanism should be statistically indistinguishable from the distribution produced by retraining on the retained data. This is a statement about the parameters. Every quantity actually measured in Chapter 6 and Chapter 7 is instead a function of model *outputs*: membership inference accuracy, accuracy on deleted and retained groups, and the unlearning success, adversarial probe rejection, and leakage metrics of Section 7.4. These establish that the target content cannot be *elicited* by the probes tried. They do not establish that it is gone.

The distinction is not academic in the adapter setting of Chapter 7, where it is sharpest. The backbone is frozen by construction, so whatever the backbone encoded about the target fact is still encoded after unlearning; the adapters suppress its expression. A method whose entire mechanism is to leave the original weights untouched is, on its face, a suppression method, and its strong showing on behavioural metrics is exactly what one would expect

from effective suppression as well as from genuine erasure. The metrics cannot distinguish them.

Three lines of work would close the gap.

Relearning as a probe. If a fact has been suppressed rather than erased, it should be recoverable more cheaply than it could be learned from scratch. Measuring how many gradient steps on a small number of positive examples restore the original behaviour, compared against a model that never saw the fact, gives an operational separation between the two conditions, and one that is far harder to satisfy by surface-level suppression than any prompt-based probe.

Representation-level readout. Rather than querying the model, one can ask whether a linear or shallow probe trained on internal activations can still recover the target association. A model from which the fact has been erased should not support such a readout; a model in which it has been masked by adapters may well do so. Applied across layers, this would also localize where the residual information sits, which is a prerequisite for removing it.

Making the definitional criterion measurable. The most direct route is to give Definition 2.5 an estimable form for models of this size. Full distributional indistinguishability over the hypothesis space is not measurable at 7B parameters, but restricted versions may be: indistinguishability of the adapter subspace $\mathcal{S}$ from that produced by adapters trained without the target fact, or of parameter statistics in the layers the readout above identifies. This would let a method be evaluated against the criterion it is nominally designed to satisfy.

A method that survived all three would have a claim to erasure rather than suppression. It is worth stating the plausible outcome: adapter-based unlearning may well fail the relearning probe, in which case the honest conclusion is that it is an excellent *suppression* mechanism whose cost and locality properties remain valuable, and that erasure at this scale requires touching the backbone after all. That would return the problem to where Chapter 7 began,

but with a criterion sharp enough to tell when it had been solved.

8.4.2 Fairness-Preserving Unlearning for Generative Models

Chapter 6 established that unlearning destroys group fairness and corrected it by rebalancing the per-batch updates that compose the trained parameters. Chapter 7 unlearns without any such record, using an object, the adapter, that has no per-group structure to rebalance. The correction of Equation 6.6 therefore does not transfer, and the dissertation stops with the two halves of the problem unjoined. Closing it requires solving two sub-problems that are independent of one another.

A fairness criterion suited to generation. The gaps of Section 2.5 are defined on a binary decision $\hat{Y}$ conditioned on a binary attribute S. A language model emits text, and the relevant disparities concern how it represents or treats groups across generations rather than how often it assigns them a label. A criterion is needed that is defined on distributions over generated sequences, is estimable from samples, and is differentiable enough to regularize.

The Cauchy–Schwarz divergence of Chapter 5 is a natural candidate, and the fit is better than it first appears. Its estimator, Equation 5.12, is nonparametric and kernel-based: it requires only samples and a positive-definite kernel, and imposes no parametric form on the distributions compared. Applied to embeddings of text generated under group-conditioned prompts, it would measure discrepancy between group-conditional *generation* distributions in exactly the way it measures discrepancy between group-conditional prediction distributions in Chapter 5. The properties argued for in Section 5.3.4, scale robustness and applicability to arbitrary distributions, are if anything more valuable in an embedding space than in the interval $[0, 1]$. This would compose two contributions of this dissertation across the gap that currently separates them.

A compensation mechanism suited to adapters. The second sub-problem is to find, for adapters, the analogue of the reversed-attribute matching of Equation 6.5. Three routes seem worth trying, in increasing order of ambition. The simplest is to rebalance the *supervision* rather than the parameters: Section 7.3.2 constructs negative examples per target fact with no attention to demographic composition, and requiring the negative set to be balanced across groups at each label would be a direct translation of the matching rule, obtainable at essentially no cost. A second route is to rebalance in the adapter subspace, training a compensating low-rank component on reversed-attribute examples and composing it with the unlearning adapter. This is possible precisely because adapters compose additively, which the monolithic weight edits of Chapter 6 do not. A third is to add the generative fairness term above directly to Equation 7.4, making the adapter simultaneously responsible for forgetting and for not skewing what remains; this is the most principled and the most likely to run into the optimization difficulties that Section 6.5.7 documented for a heavily weighted fairness term.

Before any of this, the premise needs testing. Whether the failure of Section 6.3.1 even has an analogue for generative models is not established. The first experiment is the diagnostic of Section 6.3 repeated in the new setting: take a model, unlearn a demographically concentrated set of facts, and measure whether the treatment of groups in its generations has skewed. If it has not, the problem dissolves; if it has, which the mechanism of Section 6.3.2 suggests is likely, since the skew there came from the composition of what was removed and not from anything specific to classification, then the two sub-problems above become the agenda.

8.4.3 Composing the Contributions

Two compositions internal to this dissertation remain untested, and both were flagged where they arose.

Section 4.6 noted that the fairness regularizer of Chapter 4 is a plug-in and that Chapter 5 derives a better one; substituting it would change the trajectory the synthetic data is required to match, from one shaped by a first-moment surrogate to one shaped by a measure that controls the underlying dependence. Section 6.6 noted the same substitution for the training objective of Equation 6.2, where the regularizer was selected by a four-way empirical comparison in which the dependence-based HSIC already attained lower Δ_{DP} at matched accuracy, consistent with the argument of Chapter 5, which derives a measure that outperforms HSIC. Both substitutions are available, since the frameworks are agnostic to the choice, and neither is evaluated here. They are the cheapest outstanding experiments the dissertation suggests.

8.4.4 Beyond Binary Attributes and Certified Guarantees

Two further directions follow from Section 8.3. The first is intersectional and multi-class fairness: the compensation rule of Equation 6.5 matches on counts within each attribute–label cell, and as the number of cells grows with the number of attributes the cells become small and the matching noisy, so a version that pools across related cells is needed. The second is certification. The most tractable target is the compensation of Chapter 6, where the object being corrected, a sum of recorded update vectors, is concrete enough that a bound on the residual skew after matching, and hence on the fairness gap, may be within reach. A guarantee there would convert the dissertation's most practically useful result from an empirical finding into a statement one could rely on.

8.5 Closing Remarks

This dissertation began from an observation about how three requirements are ordinarily pursued: separately, by methods that share no machinery, and evaluated by criteria that

cannot see each other. It argued that the separation is a feature of how the problems have been posed rather than of the problems themselves, and that all three are demands on the gradients a model sees and the updates it keeps.

The evidence for that argument is of two kinds. The constructive kind is that one technique, aimed at two different targets, served both: gradient matching made training cheaper in Chapter 3 and made it equitable in Chapter 4, with the mechanism unchanged and only the target trajectory moved. The same principle, reached independently, made forgetting cheap and local at language-model scale in Chapter 7. The other kind of evidence is negative and, for practice, more consequential: Chapter 6 showed that a fairness guarantee installed at one control point is silently destroyed by a routine, correct operation at another, and that neither literature's evaluation would have noticed. That failure was only visible because the two interventions were understood as acting on one surface.

The open problems set out above concern the same surface. Whether adapter-based forgetting erases or merely suppresses is a question about what the weights retain. Whether fairness survives forgetting in generative models is a question about what an edit to those weights does to the treatment of groups. Both are, in the end, the question this dissertation started with, asked at a scale where the answers matter more and the tools for answering them are weaker.

Bibliography

[1] H. Abdi and L. J. Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[2] A. Agarwal, A. Beygelzimer, M. Dudík, J. Langford, and H. Wallach. A reductions approach to fair classification. In *International conference on machine learning*, pages 60–69. PMLR, 2018.

[3] A. Aghajanyan, S. Gupta, and L. Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *ACL-IJCNLP (volume 1: long papers)*, 2021.

[4] S. Ö. Arik and T. Pfister. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6679–6687, 2021.

[5] S. Baharlouei, M. Nouiehed, A. Beirami, and M. Razaviyayn. Rényi fair inference. In *International Conference on Learning Representations*, 2020.

[6] Z. Bai, X. Zhan, J. Liu, and C. Zhang. Condensing graphs for graph neural networks via clustering. In *CIKM*, pages 2457–2460, 2021.

[7] S. Balakrishnama and A. Ganapathiraju. Linear discriminant analysis-a brief tutorial. *Institute for Signal and information Processing*, 18(1998):1–8, 1998.

[8] S. Barocas, M. Hardt, and A. Narayanan. Fairness in machine learning. *NeurIPS tutorial*, 1, 2017.

[9] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *ArXiv preprint*, 2018.

[10] B. Becker and R. Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[11] A. Beutel, J. Chen, Z. Zhao, and E. H. Chi. Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075*, 2017.

[12] S. Biswas and H. Rajan. Do the machine learning models on a crowd sourced platform exhibit bias? an empirical study on model fairness. In *ESEC*, pages 642–653, 2020.

[13] O. Bohdal, Y. Yang, and T. Hospedales. Flexible dataset distillation: Learn labels instead of images. *NeurIPS*, 2020.

[14] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot. Machine Unlearning, Dec. 2020. arXiv:1912.03817 [cs].

[15] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot. Machine unlearning. In *SP*, pages 141–159. IEEE, 2021.

[16] S. Bøyum. Fairness in education–a normative analysis of oecd policy documents. *Journal of Education Policy*, 29(6):856–870, 2014.

[17] F. Calmon, D. Wei, B. Vinzamuri, K. Natesan Ramamurthy, and K. R. Varshney. Optimized pre-processing for discrimination prevention. *NeurIPS*, 30, 2017.

[18] Y. Cao and J. Yang. Towards making systems forget with machine unlearning. In *SP*, pages 463–480. IEEE, 2015.

[19] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022.

[20] L. E. Celis, L. Huang, V. Keswani, and N. K. Vishnoi. Classification with fairness constraints: A meta-algorithm with provable guarantees. In *Proceedings of the conference on fairness, accountability, and transparency*, 2019.

[21] S. Cha, S. Cho, D. Hwang, and M. Lee. Towards robust and parameter-efficient knowledge unlearning for llms. *arXiv preprint arXiv:2408.06621*, 2024.

[22] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies. Fairway: a way to build fair ml software. In *ESEC*, pages 654–665, 2020.

[23] C. Chen, F. Sun, M. Zhang, and B. Ding. Recommendation unlearning. In *WWW*, pages 2768–2777, 2022.

[24] M. Chen, Z. Zhang, T. Wang, M. Backes, M. Humbert, and Y. Zhang. When machine unlearning jeopardizes privacy. In *WAHC*, pages 896–911, 2021.

[25] M. Chen, Z. Zhang, T. Wang, M. Backes, M. Humbert, and Y. Zhang. Graph unlearning. In *CCS*, pages 499–513, 2022.

[26] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[27] C.-Y. Chuang and Y. Mroueh. Fair mixup: Fairness via interpolation. In *ICLR*, 2020.

[28] C.-Y. Chuang and Y. Mroueh. Fair mixup: Fairness via interpolation. *arXiv preprint arXiv:2103.06503*, 2021.

[29] E. Commission. 2018 reform of eu data protection rules, 2018.

[30] P. Comon. Independent component analysis, a new concept? *Signal processing*, 36(3):287–314, 1994.

[31] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. Algorithmic decision making and the cost of fairness. In *KDD*, pages 797–806, 2017.

[32] N. Deka and D. J. Sutherland. Mmd-b-fair: Learning fair representations with statistical testing. In *International Conference on Artificial Intelligence and Statistics*, 2023.

[33] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*, 2023.

[34] J. B. Diederik P. Kingma. Adam: a method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[35] F. Ding, M. Hardt, J. Miller, and L. Schmidt. Retiring adult: New datasets for fair machine learning. *NeurIPS*, 34:6478–6490, 2021.

[36] C. DoJ. California consumer privacy act (ccpa), 2018.

[37] D. Dua and C. Graff. UCI machine learning repository. `https://archive.ics.uci.edu/ml`, 2017.

[38] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. *NeurIPS*, 28, 2015.

[39] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *ITCS*, pages 214–226, 2012.

[40] B. T. et al. To forget or not? towards practical knowledge unlearning for llms. In *Findings of EMNLP*, 2024.

[41] C. Fan, J. Liu, L. Lin, J. Jia, R. Zhang, S. Mei, and S. Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *arXiv preprint arXiv:2410.07163*, 2024.

[42] M. Fey and J. E. Lenssen. Fast graph representation learning with pytorch geometric. *ArXiv preprint*, 2019.

[43] L. G. S. Giraldo, E. Hasanbelliu, M. Rao, and J. C. Principe. Group-wise point-set registration based on renyi's second order entropy. In *CVPR*, 2017.

[44] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[45] A. Golatkar, A. Achille, and S. Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *CVPR*, 2020.

[46] S. Gong, J. Ni, N. Sachdeva, C. Yang, and W. Jin. Gc-bench: A benchmark framework for graph condensation with new insights. *arXiv preprint arXiv:2406.16715*, 2024.

[47] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko. Revisiting deep learning models for tabular data. *Advances in neural information processing systems*, 34:18932–18943, 2021.

[48] L. Graves, V. Nagisetty, and V. Ganesh. Amnesiac machine learning. In *AAAI*, volume 35, pages 11516–11524, 2021.

[49] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 2012.

[50] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, 2005.

[51] A. Gretton, K. Fukumizu, C. Teo, L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. *Advances in neural information processing systems*, 2007.

[52] T. Grote and G. Keeling. Enabling fairness in healthcare through machine learning. *Ethics and Information Technology*, 24(3):39, 2022.

[53] C. Guo, T. Goldstein, A. Hannun, and L. Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

[54] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1024–1034, 2017.

[55] X. Han, T. Zhao, Y. Liu, X. Hu, and N. Shah. Mlpinit: Embarrassingly simple gnn training acceleration with mlp initialization. In *ICLR 2023*.

[56] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. *NeurIPS*, 29, 2016.

[57] M. Hashemi, S. Gong, J. Ni, W. Fan, B. A. Prakash, and W. Jin. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. *arXiv preprint arXiv:2402.03358*, 2024.

[58] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. D. Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, 2019.

[59] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and e. a. Weizhu Chen. Lora: Low-rank adaptation of large language models. *ICLR*, 2022.

[60] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022.

[61] L. Hu and Y. Chen. A short-term intervention for long-term fairness in the labor market. In *WWW*, pages 1389–1398, 2018.

[62] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *NeurIPS*, 33:22118–22133, 2020.

[63] J. J. P. III, S. Moreno, T. L. Fond, J. Neville, and B. Gallagher. Attributed graph models: modeling network structure with correlated attributes. In *WWW*, 2014.

[64] G. Ilharco, M. T. Ribeiro, M. Wortsman, S. Gururangan, L. Schmidt, H. Hajishirzi, and A. Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[65] R. Jenssen, J. C. Principe, D. Erdogmus, and T. Eltoft. The cauchy–schwarz divergence and parzen windowing: Connections to graph theory and mercer kernels. *Journal of the Franklin Institute*, 2006.

[66] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023, 2023.

[67] R. Jiang, A. Pacchiano, T. Stepleton, H. Jiang, and S. Chiappa. Wasserstein fair classification. In *UAI*, pages 862–872. PMLR, 2020.

[68] W. Jin, X. Tang, H. Jiang, Z. Li, D. Zhang, J. Tang, and B. Yin. Condensing graphs via one-step gradient matching. In *KDD*, pages 720–730, 2022.

[69] W. Jin, L. Zhao, S. Zhang, Y. Liu, J. Tang, and N. Shah. Graph condensation for graph neural networks. *arXiv preprint arXiv:2110.07580*, 2021.

[70] Z. Jin, P. Cao, C. Wang, Z. He, H. Yuan, J. Li, Y. Chen, K. Liu, and J. Zhao. Rwku: Benchmarking real-world knowledge unlearning for large language models. *NeurIPS*, 2024.

[71] C. K. Joshi, F. Liu, X. Xun, J. Lin, and C. S. Foo. On representation knowledge distillation for graph neural networks. *TNNLS*, 2022.

[72] F. Kamiran and T. Calders. Data preprocessing techniques for classification without discrimination. *KAIS*, 33(1):1–33, 2012.

[73] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *ECML PKDD*, pages 35–50. Springer, 2012.

[74] K. Kampa, E. Hasanbelliu, and J. C. Principe. Closed-form cauchy-schwarz pdf divergence for mixture of gaussians. In *IJCNN*, 2011.

[75] J.-H. Kim, J. Kim, S. J. Oh, S. Yun, H. Song, J. Jeong, J.-W. Ha, and H. O. Song. Dataset condensation via efficient synthetic-data parameterization. In *International Conference on Machine Learning*, pages 11102–11118. PMLR, 2022.

[76] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[77] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[78] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[79] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, and e. a. Agnieszka Grabska-Barwinska. Overcoming catastrophic forgetting in neural networks. *PANS*, 2017.

[80] R. F. Kizilcec and H. Lee. Algorithmic fairness in education. In *The ethics of artificial intelligence in education*, pages 174–202. Routledge, 2022.

[81] J. Klicpera, A. Bojchevski, and S. Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[82] J. Larson, S. Mattu, L. Kirchner, and J. Angwin. Propublica compas analysis—data and analysis for 'machine bias'. `https://github.com/propublica/compas-analysis`, 2016. Accessed: 2023-03-13.

[83] S. Lee, S. Chun, S. Jung, S. Yun, and S. Yoon. Dataset condensation with contrastive signals. In *International Conference on Machine Learning*, pages 12352–12364. PMLR, 2022.

[84] S. Lei and D. Tao. A comprehensive survey of dataset distillation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(1):17–32, 2023.

[85] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2021.

[86] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL-IJCNLP*, 2021.

[87] Z. Li, A. Perez-Suay, G. Camps-Valls, and D. Sejdinovic. Kernel dependence regularizers and gaussian processes with applications to algorithmic fairness. *arXiv preprint arXiv:1911.04322*, 1911.

[88] H. Ling, Z. Jiang, Y. Luo, S. Ji, and N. Zou. Learning fair graph representations via automated data augmentations. In *ICLR*, 2022.

[89] S. Liu, Y. Yao, J. Jia, S. Casper, N. Baracaldo, P. Hase, Y. Yao, C. Y. Liu, X. Xu, and e. a. Hang Li. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, 2025.

[90] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen. Dora: Weight-decomposed low-rank adaptation. In *ICML*, 2024.

[91] Z. Liu and e. a. Dejing Dou. Towards safer large language models through machine unlearning. In *Findings of ACL*, 2024.

[92] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, December 2015.

[93] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. S. Zemel. The variational fair autoencoder. In *International Conference on Learning Representations (ICLR)*, 2016.

[94] Y. Ma, X. Liu, T. Zhao, Y. Liu, J. Tang, and N. Shah. A unified view on graph neural networks as graph signal denoising. In *CIKM*, pages 1202–1211, 2021.

[95] D. Madras, E. Creager, T. Pitassi, and R. Zemel. Learning adversarially fair and transferable representations. In *ICML*, pages 3384–3393. PMLR, 2018.

[96] P. Maini, Z. Feng, A. Schwarzschild, Z. C. Lipton, and J. Z. Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

[97] M. Mazumder, C. Banbury, X. Yao, B. Karlaš, W. G. Rojas, S. Diamos, G. Diamos, L. He, A. Parrish, H. R. Kirk, et al. Dataperf: Benchmarks for data-centric ai development. *arXiv preprint arXiv:2207.10062*, 2022.

[98] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *CSUR*, 54(6):1–35, 2021.

[99] K. Meng, D. Bau, A. Andonian, and Y. Belinkov. Locating and editing factual associations in gpt. *NeurIPS*, 2022.

[100] K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, and D. Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

[101] T. T. Nguyen, T. T. Huynh, P. L. Nguyen, A. W.-C. Liew, H. Yin, and Q. V. H. Nguyen. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*, 2022.

[102] A. Oesterling, J. Ma, F. P. Calmon, and H. Lakkaraju. Fair machine unlearning: Data removal while mitigating disparities. *arXiv preprint arXiv:2307.14754*, 2023.

[103] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 1962.

[104] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[105] K. Petrasic, B. Saul, J. Greig, M. Bornfreund, and K. Lamberth. Algorithms and bias: What lenders need to know. *White & Case*, 2017.

[106] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *EACL*, 2021.

[107] J. Principe, D. Xu, J. Fisher, and S. Haykin. Information theoretic learning. unsupervised adaptive filtering. *Unsupervised Adapt Filter*, 2000.

[108] T. L. Quy, A. Roy, V. Iosifidis, W. Zhang, and E. Ntoutsi. A survey on datasets for fairness-aware machine learning. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2022.

[109] J. Rosen. The right to be forgotten. *Stan. L. Rev. Online*, 64:88, 2011.

[110] A. Rücklé, G. Geigle, M. Glockner, T. Beck, J. Pfeiffer, N. Reimers, and I. Gurevych. Adapterdrop: On the efficiency of adapters in transformers. In *EMNLP*, 2021.

[111] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. In *ECML*, volume 3201, pages 371–383. Springer, 2004.

[112] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[113] C. R. Shalizi and A. C. Thomas. Homophily and contagion are generically confounded in observational social network studies. *Sociological methods & research*, (2), 2011.

[114] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[115] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE SP*, pages 3–18. IEEE, 2017.

[116] D. A. Spielman and N. Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

[117] F. P. Such, A. Rawal, J. Lehman, K. O. Stanley, and J. Clune. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. *ICML*, 2020.

[118] I. Sucholutsky and M. Schonlau. Soft-label dataset distillation and text dataset distillation. *arXiv preprint arXiv:1910.02551*, 2019.

[119] S. Tan, Y. Shen, and B. Zhou. Improving the fairness of deep generative models without retraining. *arXiv preprint arXiv:2012.04842*, 2020.

[120] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, and e. a. Shruti Bhosale. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[121] L. Tran, M. Pantic, and M. P. Deisenroth. Cauchy–schwarz regularized autoencoder. *Journal of Machine Learning Research*, 2022.

[122] D. J. Trosten, S. Lokse, R. Jenssen, and M. Kampffmeyer. Reconsidering representation alignment for multi-view clustering. In *CVPR*, 2021.

[123] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[124] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[125] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *ICLR*, 2018.

[126] R. Viorescu et al. 2018 reform of eu data protection rules. *European Journal of Law and Public Administration*, 4(2):27–39, 2017.

[127] S. Wang, W. Guo, H. Narasimhan, A. Cotter, M. Gupta, and M. Jordan. Robust optimization for fairness with noisy protected groups. *Advances in neural information processing systems*, 33:5190–5203, 2020.

[128] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

[129] Y. Wang, Z. Sun, Z. Liu, and S. E. Sarma. Graph transformer networks. In *AAAI*, volume 34, pages 6817–6824, 2020.

[130] S. E. Whang, Y. Roh, H. Song, and J.-G. Lee. Data collection and quality challenges in deep learning: A data-centric ai perspective. *The VLDB Journal*, 32(4):791–813, 2023.

[131] F. Wu, A. H. S. Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *ICML*, Proceedings of Machine Learning Research, 2019.

[132] R. Wu, C. Yadav, R. Salakhutdinov, and K. Chaudhuri. Evaluating deep unlearning in large language models. *arXiv preprint arXiv:2410.15153*, 2024.

[133] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu. A comprehensive survey on graph neural networks. *ArXiv preprint*, 2019.

[134] D. Xu, S. Yuan, L. Zhang, and X. Wu. Fairgan: Fairness-aware generative adversarial networks. In *2018 IEEE international conference on big data (big data)*, pages 570–575. IEEE, 2018.

[135] H. Xu, T. Zhu, L. Zhang, W. Zhou, and P. S. Yu. Machine unlearning: A survey, 2023.

[136] J. Yao, E. Chien, M. Du, X. Niu, T. Wang, Z. Cheng, and X. Yue. Machine unlearning of pre-trained large language models. *arXiv preprint arXiv:2402.15159*, 2024.

[137] Y. Yao, Q. Lin, and T. Yang. Stochastic methods for auc optimization subject to auc-based fairness constraints. In *International Conference on Artificial Intelligence and Statistics*, 2023.

[138] Y. Yao, X. Xu, and Y. Liu. Large language model unlearning. *NeurIPS*, 2024.

[139] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *CSF*, pages 268–282. IEEE, 2018.

[140] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *KDD*, pages 974–983. ACM, 2018.

[141] S. Yu, H. Li, S. Løkse, R. Jenssen, and J. C. Príncipe. The conditional cauchy-schwarz divergence with applications to time-series data and sequential decision making. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[142] W. Yuan, H. Yin, F. Wu, S. Zhang, T. He, and H. Wang. Federated unlearning for on-device recommendation. In *WSDM*, pages 393–401, 2023.

[143] M. B. Zafar, I. Valera, M. G. Rogriguez, and K. P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *AISTATS*, 2017.

[144] E. B. Zaken, Y. Goldberg, and S. Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *ACL (Volume 2: Short Papers)*, 2022.

[145] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. K. Prasanna. Graphsaint: Graph sampling based inductive learning method. In *ICLR*, 2020.

[146] D. Zha, Z. P. Bhat, K.-H. Lai, F. Yang, Z. Jiang, S. Zhong, and X. Hu. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*, 2023.

[147] B. H. Zhang, B. Lemoine, and M. Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018.

[148] B. H. Zhang, B. Lemoine, and M. Mitchell. Mitigating unwanted biases with adversarial learning. In *AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, 2018.

[149] D. Zhang, S. Pan, T. Hoang, Z. Xing, M. Staples, X. Xu, L. Yao, Q. Lu, and L. Zhu. To be forgotten or to be fair: Unveiling fairness implications of machine unlearning methods. *arXiv preprint arXiv:2302.03350*, 2023.

[150] H. Zhang, T. Nakamura, T. Isohara, and K. Sakurai. A review on machine unlearning. *SN Computer Science*, 4(4):337, 2023.

[151] J. M. Zhang and M. Harman. " ignorance and prejudice" in software fairness. In *ICSE*, pages 1436–1447. IEEE, 2021.

[152] R. Zhang, X. Li, J. He, and G. Neubig. Adaptive budget allocation for parameter-efficient fine-tuning. In *ICLR*, 2023.

[153] R. Zhang, L. Lin, Y. Bai, and S. Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

[154] S. Zhang, Y. Liu, Y. Sun, and N. Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. *arXiv preprint arXiv:2110.08727*, 2021.

[155] Z. Zhang, X. Wang, and W. Zhu. Automated machine learning on graphs: A survey. *arXiv preprint arXiv:2103.00742*, 2021.

[156] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

[157] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset condensation with gradient matching. *International Conference on Learning Representations*, 2021.

[158] J. Zhao and D. Meng. Fastmmd: Ensemble of circular discrepancy for efficient two-sample test. *Neural computation*, 2015.

[159] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun. Graph neural networks: A review of methods and applications. *ArXiv preprint*, 2018.

[160] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *NeurIPS*, 2019.

[161] M. Zhu, X. Wang, C. Shi, H. Ji, and P. Cui. Interpreting and unifying graph neural networks with an optimization framework. In *WWW*, 2021.