



NORTHWESTERN UNIVERSITY

Computer Science Department

Technical Report
Number: NU-CS-2026-21

June, 2026

Generating and Detecting Synthetic Content Across Modalities

Chongyang Gao

Abstract

The development of advanced generative models has enabled cross-modal synthesis of content spanning language, vision, and speech, unlocking opportunities for privacy-preserving data sharing while raising concerns over misuse, copyright infringement, and data leakage. To address these challenges, we present three complementary frameworks. FakeDB generates synthetic relational databases that preserve semantic constraints and statistical properties for secure information exchange. GEM guides multimodal content generation toward user-specified objectives using prompt tuning and discriminative feedback. We also propose an efficient learning strategy to facilitate more effective and scalable data generation, enabling the production of high-quality synthetic samples with reduced computational overhead. To mitigate risks of unintended information retention, O³ enables continual unlearning via orthogonal adapters and out-of-distribution detection. To safeguard public content, such as official statements or announcements, from deepfakes that can mislead or manipulate, we demonstrate that classifiers trained on synthesized data can reliably detect fake content. We introduce CADD, a Context-based Audio Deepfake Detector that integrates transcripts and contextual metadata (e.g., news, social media) to improve detection accuracy and robustness. Beyond language, vision, and speech, generative models are increasingly applied to biological sequences, where the same capabilities that enable beneficial design can also produce harmful proteins such as toxins or virulence factors. To address this emerging dual-use risk, we introduce SafePro, a multi-objective alignment framework for protein language models that jointly optimizes structural

foldability and avoids toxicity and virulence. Collectively, our contributions form a unified framework for promoting authenticity, privacy, safety, and trust in the generative AI era.

Keywords

Generative Models; Trustworthy AI; Synthetic Data Generation; Parameter-Efficient Fine-Tuning; Machine Unlearning; Audio Deepfake Detection; Multi-Objective Alignment; Protein Language Models

NORTHWESTERN UNIVERSITY

Generating and Detecting Synthetic Content Across Modalities

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Science

By

Chongyang Gao

EVANSTON, ILLINOIS

June 2026

© Copyright by Chongyang Gao 2026

All Rights Reserved

ABSTRACT

Generating and Detecting Synthetic Content Across Modalities

Chongyang Gao

The development of advanced generative models has enabled cross-modal synthesis of content spanning language, vision, and speech, unlocking opportunities for privacy-preserving data sharing while raising concerns over misuse, copyright infringement, and data leakage. To address these challenges, we present three complementary frameworks. FakeDB generates synthetic relational databases that preserve semantic constraints and statistical properties for secure information exchange. GEM guides multimodal content generation toward user-specified objectives using prompt tuning and discriminative feedback. We also propose an efficient learning strategy to facilitate more effective and scalable data generation, enabling the production of high-quality synthetic samples with reduced computational overhead. To mitigate risks of unintended information retention, $\mathcal{O}^3$ enables continual unlearning via orthogonal adapters and out-of-distribution detection. To safeguard public content, such as official statements or announcements, from deepfakes that can mislead or manipulate, we demonstrate that classifiers trained on synthesized data can reliably detect fake content. We introduce CADD, a Context-based Audio

Deepfake Detector that integrates transcripts and contextual metadata (e.g., news, social media) to improve detection accuracy and robustness. Beyond language, vision, and speech, generative models are increasingly applied to biological sequences, where the same capabilities that enable beneficial design can also produce harmful proteins such as toxins or virulence factors. To address this emerging dual-use risk, we introduce *SafePro*, a multi-objective alignment framework for protein language models that jointly optimizes structural foldability and avoids toxicity and virulence. Collectively, our contributions form a unified framework for promoting authenticity, privacy, safety, and trust in the generative AI era.

Table of Contents

ABSTRACT	3
Table of Contents	5
List of Tables	8
List of Figures	15
Chapter 1. Introduction	20
Generation as a Shield: Synthesis for Protection and Engagement	20
Containing Misuse: Detection, Unlearning, and Alignment	22
Summary of Contributions	24
Chapter 2. Protecting Data using Generative Models	27
2.1. Introduction	28
2.2. Related Work	29
2.3. Method	30
2.4. Experiments	35
2.5. Conclusion	49
Chapter 3. Generating Engaging Multimodal Data	50
3.1. Introduction	50
3.2. Related Work	51

	6
3.3. Method	52
3.4. Experiments	56
3.5. Conclusion	63
Chapter 4. Efficient Finetuning Large Language Models	64
4.1. Introduction	64
4.2. Related Work	66
4.3. Method	67
4.4. Experiments	72
4.5. Conclusion	81
Chapter 5. Unlearning Large Language Models	83
5.1. Introduction	83
5.2. Related Work	84
5.3. Method	85
5.4. Experiments	90
5.5. Conclusion	100
Chapter 6. Detecting Audio Deepfakes	101
6.1. Introduction	101
6.2. Related Work	103
6.3. Perturbed Public Voices	104
6.4. Context-based Audio Deepfake Detection (CADD)	109
6.5. Results	121
6.6. Conclusion	129

Chapter 7. Generating Safe and Functional Proteins	134
7.1. Introduction	134
7.2. Related Work	137
7.3. Method	139
7.4. Experiments	152
7.5. Conclusion	164
Chapter 8. Conclusions and Future Works	168
8.1. Summary of Contributions	168
8.2. Future Directions	171
8.3. Closing Remarks	174
References	175

List of Tables

2.1	Function $\mathcal{L}_v(\mathcal{R}^f, ic)$.	34
2.2	Datasets used in the experiments on real data.	35
2.3	Values of <i>dos</i> obtained by tableGAN , CTGAN , SDV , and FakeDB (best values in bold, higher is better).	37
2.4	Values of $\mathcal{L}_p$ obtained by tableGAN , CTGAN , SDV , and FakeDB for the UW-CSE dataset, for different sets of integrity constraints (best values in bold, lower is better).	38
2.5	Summary of the results in Table 2.4 (for the average increase, lower is better).	39
2.6	Relations and attributes used to compute statistics.	39
2.7	Mean, standard deviation, kurtosis, and skewness obtained by FakeDB .	40
2.8	EMD, Normalized EMD 1, and Normalized EMD 2 obtained by FakeDB .	41
2.9	Mean, standard deviation, kurtosis, and skewness obtained by tableGAN , CTGAN , SDV , and FakeDB (best values in bold).	42
2.10	EMD, Normalized EMD 1, and Normalized EMD 2 obtained by tableGAN , CTGAN , SDV , and FakeDB (best values in bold).	43

2.11	Summary of the results in Tables 2.9 and 2.10 (for the average difference and average value, lower is better).	44
2.12	Summary of the results obtained using all the relations in Table 2.6.	44
2.13	Values of $\mathcal{L}_p$ obtained by tableGAN , CTGAN , SDV , and FakeDB for the synthetic dataset (best values in bold, lower is better).	46
2.14	Summary of the results in Table 2.13 (for the average increase, lower is better).	47
3.1	Average similarity scores.	60
3.2	Results of the human evaluation. The selected percentage of more engaging methods is illustrated in parentheses.	62
4.1	Comparison with different methods using LLaMA-2. MoLA- ∇ outperforms other variants or baselines and even achieves competitive or superior performance with MoLA- $\square$ (8888), with nearly 40% fewer parameters.	74
4.2	Confidence intervals of different configurations of MoLA using LLAMA-2.	77
4.3	Comparison with different configurations of MoLA using Mistral.	78
4.4	Comparison with different extreme configurations of MoLA. The top block shows the results using LLaMA-2 and the bottom block shows the results using Mistral.	78

- 5.1 Comparison between ours and other baselines on used training data quantity and trainable parameters. The trainable parameters of baselines are all the whole LLM. 94
- 5.2 Performance Comparison between our $\mathcal{O}^3$ and other baselines when continually unlearning TOFU-forget01, -forget05, and -forget10 in Fictitious Knowledge Generation. The unlearning effectiveness is measured by the generation accuracy of the unlearning train data and unlearning test data denoted as S.U. and D.U., respectively. Utility preservation is evaluated by the generation accuracy of Retained Distribution (R.D.), TOFU-Real Authors (R.A.), and World Facts (W.F.). 96
- 5.3 Performance Comparison between our $\mathcal{O}^3$ and other baselines when continually unlearning domain ‘work’, ‘travel’, and ‘home’ of CLINC150 in Intent Classification. The unlearning effectiveness is measured by the classification accuracy of the unlearning train data and unlearning test data denoted as S.U. and D.U., respectively. Utility preservation is evaluated by the accuracy of Retained Distribution (R.D.), MRPC, and RTE. 97
- 5.4 OOD detection performance comparison and ablation study between ours and others on Fictitious Knowledge Generation and Question Answering. The measurement is AUROC. 98
- 5.5 Hyper-parameter analysis of the unlearning knowledge optimization of $\mathcal{O}^3$ framework. We adopt a series of values for the factor λ of Eq. 5.3 to

validate the necessity of $\mathcal{L}_{\text{Orth}}$ and analyze the sensitivity of λ . ‘C.QA’ shorts for CommonsenseQA, and ‘O.QA’ shorts for OpenbookQA. 99

5.6 Hyper-parameter analysis of the soft-weighted inference of $\mathcal{O}^3$ framework. We adopt a hard-weighted (Hard-w) mechanism and change the scaling factor ζ of Eq. 5.9. ‘C.QA’ shorts for CommonsenseQA, and ‘O.QA’ shorts for OpenbookQA. 99

6.1 P^2V distribution of subjects and audio samples across training, validation, and test splits. The dataset contains both synthetic (fake) audio generated through various methods and authentic (real) audio samples. 110

6.2 **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on JDD data, with varying contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric are highlighted in bold and marked with different background colors (  ), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$). See tables S1-S12 for full results. 130

6.3 **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on SYN data with varying the contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric are highlighted in bold and marked with different background colors (  ), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$). See tables S13-S23 for full results. 131

6.4 **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on P²V data with varying the contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric (within each model group) are highlighted in bold and marked with different background colors (  ), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing

(indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$).

See table S35 for full TML results.

132

6.5 **Performance comparison of Traditional Machine Learning (TML) baselines for deepfake detection on P^2V data, with varying contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric (within each model group) are highlighted in bold and marked with different background colors (■ ■ ■), while the second-best performance is underlined. The last column shows P-values from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$).

133

7.1 Test-set performance of CTAF against single-task and shared-block multi-task baselines on the toxicity and virulence datasets. **Bold** = best per column; underline = second-best.

158

7.2 Evaluation results across reward-weight configurations, comparing PPO alignment driven by Independent Classifiers and by our CTAF reward model. The composite score is computed as $\frac{1}{3}[\text{Mean pLDDT} + (1 - P_{\text{toxic}}) \times 100 + (1 - P_{\text{virulent}}) \times 100]$, combining structural quality, toxicity avoidance, and virulence avoidance into a single metric; higher is better. The upper block reports fixed

reward-weight configurations denoted $(w_{\text{tox}}, w_{\text{vir}}, w_{\text{pLDDT}})$. The lower block reports adaptive weight-calibration methods operating on top of CTAF: standard MGDA without the TCS mechanism, and our *SafePro*. 164

7.3 *SafePro* sensitivity analysis: effect of the conflict detection threshold λ on protein generation quality. All runs use identical hyperparameters except for λ , which controls when the algorithm switches from the full Gram matrix G to the conflict-focused matrix $G^{\mathcal{M}}$ (Eq. (7.22)). The composite score follows the same formula as Table 7.2. MGDA (no TCS) serves as the baseline without conflict detection. 165

List of Figures

1.1	An overview of the dissertation’s contributions, organized around the dual nature of generative models. On the beneficial side (left), synthetic content can shield sensitive information and produce engaging multimodal data, while parameter-efficient training makes such generation broadly accessible. On the misuse side (right), the same generative capability creates threats that we address through deepfake detection, continual unlearning of memorized content, and multi-objective alignment for safe protein design.	21
2.1	Logical architecture.	27
2.2	Runtime when varying the percentage of data.	44
2.3	Runtime when varying the size of the synthetic relations.	47
3.1	The training of continuous prompts.	53
3.2	Multimodal (image-text) pairs generated using various techniques. Columns are arranged from left to right to show pairs produced by different methods. Each row uses the text in the first column as the initial input.	59
4.1	The overview of MoLA architecture.	68

4.2	Five types of layer-wise expert allocations of MoLA. A longer rectangle indicates a greater number of LoRA experts.	70
4.3	Average number of the Frobenius Norm between two different experts' weight matrices for each self-attention module from each layer. The left and right figures are for the MoLA- $\square$ (8888) using LLaMA-2 and Mistral, correspondingly. Both models are trained via instruction tuning.	80
4.4	(a) The average fusion weights for each expert. (b) The average times for each expert when it is selected.	82
5.1	The overview of $\mathcal{O}^3$ framework to handle continual unlearning requests for LLM without using any retained data.	85
5.2	Comparison between ours and other baseline approaches on Unlearning-Utility Ratio (U^2R) that measures the balance between unlearning effectiveness and utility preservation.	93
5.3	Unlearning effectiveness comparison between ours and other approaches on sample-level unlearning (S.U.) of ScienceQA.	95
5.4	Unlearning effectiveness comparison between ours and other approaches on distribution-level unlearning (D.U.) of ScienceQA.	95
5.5	Utility preservation performance comparison between ours and state-of-the-art unlearning approaches on the testing set of (a) Retained Distribution, (b) CommonsenseQA, (c) OpenbookQA, after unlearning each request of ScienceQA.	96

- 6.1 **Overview of the audio deepfake generation pipeline for the P²V dataset.** The workflow has four stages: (A) *Reference audio collection* from WikiData and YouTube, followed by speaker diarization and identification to isolate target speaker segments; (B) *Fake transcript generation* with LLMs under news-guided and in-character/out-of-character conditioning; (C) *Audio deepfake synthesis* via Text-to-Speech (TTS) and Voice Conversion (VC) methods; and (D) *Audio perturbation application* to simulate real-world acoustic conditions and adversarial distortions. 105
- 6.2 The architecture of our CADD context fusion module has three components: 1) the context encoder, 2) the deepfake audio detection backbone, and 3) the fusion module and classification head. 110
- 6.3 **Performance comparison of state-of-the-art (a) and traditional machine learning (b) baselines and our CADD configurations (CADD(T), CADD(C), and CADD(T+C)).** Each point represents a model’s average performance score computed on our Journalist-provided Deepfake Dataset (JDD), with the same color denoting the same baseline across different configurations. 128
- 6.4 **Performance Robustness Under Audio Manipulations (JDD dataset)** The figure shows the absolute percentage difference between the results (in terms of Avg scores) obtained on the perturbed JDD test set and those obtained on the original, unperturbed test set for the subset of 4 best-performing SOTA baselines and related CADD configurations.

SOTA models are separated by black vertical lines, with baseline results highlighted in a dashed rectangle. 129

7.1 Mean gradient magnitude across 32 samples. For each block, we plot $\frac{1}{N} \sum_n \|\nabla_n^{(b)}\|_2$, then divide by the peak block value so the y-axis is in units of *fraction of the peak block magnitude*; the shaded band shows $\pm\sigma$ across samples under the same normalization. The output-projection block near index 49 dominates the mean by roughly an order of magnitude, while 47.9% of parameters are near-zero on any given backward pass (red annotation). 138

7.2 Overview of the Cross-Task Attentive Fusion (CTAF) architecture, shown for the toxic task as the active task. The frozen ESM2 backbone (blue) and pooler produce a CLS representation, which is passed through both task-specific experts. The active task’s expert provides queries (Q) to the multi-head cross-attention module, while the other task’s expert supplies keys and values (K, V). An adaptive fusion gate modulates the balance between cross-task and task-private representations before a residual connection and LayerNorm. The fused output is then passed to the active task’s classification head. When virulence is the active task, the roles of the experts are reversed. 166

7.3 Raw per-sample log-probability gradients $|\nabla_n|$ across 51 parameter blocks for $N=32$ sequences generated by the `progen2-medium` policy. **Each row is independently normalized by that sample’s own maximum**

block magnitude, i.e. entry (n, b) plots $\|\nabla_n^{(b)}\|_2 / \max_{b'} \|\nabla_n^{(b')}\|_2 \in [0, 1]$.

Per-row normalization removes the large magnitude differences *across* samples so that the *within-sample* distribution of gradient signal is directly comparable: bright = blocks carrying most of that sample's gradient, dark $\approx$ zero for that sample.

167

7.4 Effect of the length bonus on mean generated sequence length during PPO training.

167

CHAPTER 1

Introduction

The rapid advancement of generative models has enabled the synthesis of content across a remarkable range of modalities, spanning text, images, audio, and even biological sequences such as proteins. The same machinery that unlocks powerful applications, however, also opens the door to misuse. As the boundary between authentic and generated content continues to dissolve, this dissertation argues that addressing the safety, controllability, and trustworthiness of generative models requires attention to their full lifecycle: harnessing generation as a protective shield where appropriate, while detecting, unlearning, and aligning to contain its misuse.

Figure 1.1 provides a unified view of the contributions developed in this dissertation. The remainder of this chapter walks through the two pathways depicted in the figure, beginning with constructive uses of generation and then turning to the threats introduced by the same underlying technology.

Generation as a Shield: Synthesis for Protection and Engagement

A central insight motivating the first half of this dissertation is that synthetic content, when sufficiently faithful, can itself act as a defense. If real records can be replaced by high-quality generated counterparts, an adversary loses the ability to reliably distinguish authentic data from the artificial, and downstream attacks such as data theft become substantially harder. Healthcare providers, manufacturers, and other organizations routinely

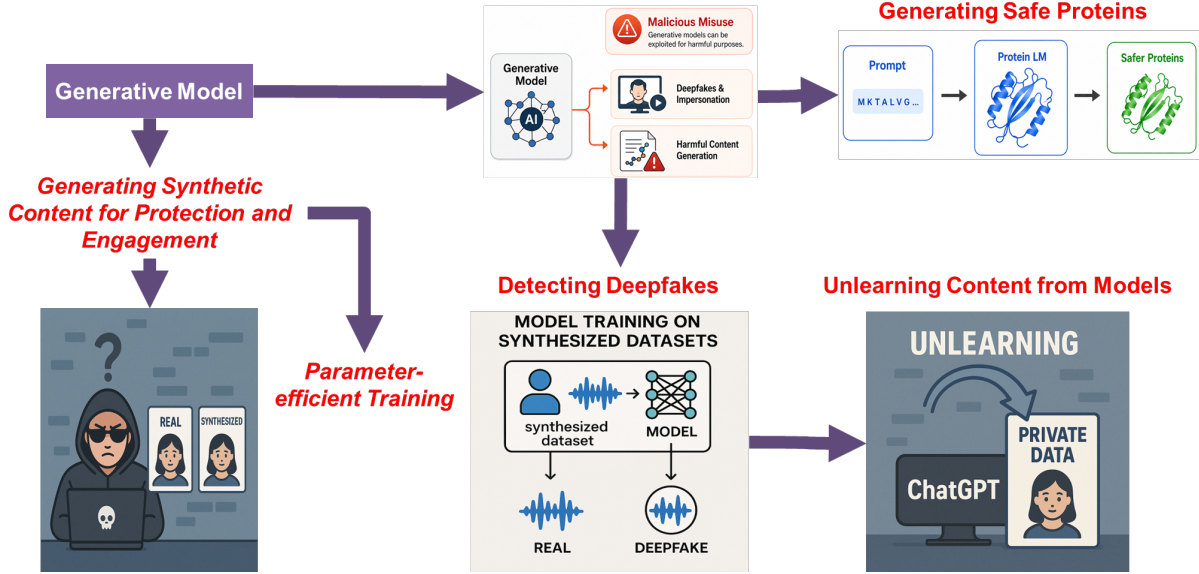


Figure 1.1. An overview of the dissertation’s contributions, organized around the dual nature of generative models. On the beneficial side (left), synthetic content can shield sensitive information and produce engaging multimodal data, while parameter-efficient training makes such generation broadly accessible. On the misuse side (right), the same generative capability creates threats that we address through deepfake detection, continual unlearning of memorized content, and multi-objective alignment for safe protein design.

face a tension between sharing data with collaborators and safeguarding individual privacy or corporate confidentiality; property-preserving synthetic data offers a way to reconcile these competing demands.

Building on this intuition, we develop generative methods that produce content which is both property-preserving and convincing. Our **FakeDB** framework targets relational databases, an especially demanding setting because realism must be maintained simultaneously across multiple linked tables. FakeDB synthesizes relational data that respects complex semantic integrity constraints across tables while preserving statistical distributions, making it suitable for secure data sharing in practice.

Realistic synthesis alone is not always sufficient: in domains such as digital media and education, generated content must also capture attention and convey messages coherently. Our **GEM** framework addresses this need by producing engaging image–text pairs through learned continuous prompts and discriminative feedback. GEM first trains an engagement classifier on social media data, then uses it to guide prompt learning for a diffusion model, and finally refines text–image alignment through an iterative paraphrasing loop driven by similarity scores.

A practical obstacle to deploying generation for these constructive purposes is scale: as generative models grow larger, customizing them for specific domains or noisy data becomes prohibitively expensive. To make beneficial generation more broadly accessible, we introduce **MoLA**, a parameter-efficient fine-tuning method that combines Mixture of Experts (MoE) with Low-Rank Adaptation (LoRA). Rather than allocating the same number of LoRA experts to every Transformer layer, MoLA distributes them variably and concentrates capacity in the middle layers, where additional experts yield the greatest benefit. MoLA thus lowers the cost barrier for adapting large models to the protection- and engagement-oriented tasks described above.

Containing Misuse: Detection, Unlearning, and Alignment

The same generative capabilities that power constructive applications can also be turned to harmful ends. We organize the threats considered in this dissertation into three categories, corresponding to three complementary points of intervention.

Detecting deepfakes of public figures. The proliferation of audio deepfake technologies has raised pressing concerns about authenticity, privacy, and trust in public communication.

Existing audio deepfake detectors perform well on controlled benchmarks but degrade sharply when deployed in the wild, and they typically focus only on the raw audio signal while ignoring the contextual cues that humans naturally rely on to judge credibility. We address both gaps. First, we release **Perturbed Public Voices (P²V)**, an IRB-approved dataset that captures three properties of malicious deepfakes overlooked by prior benchmarks: identity-consistent transcripts produced via large language models, environmental and adversarial noise, and state-of-the-art voice cloning spanning 2020 to 2025. Second, we introduce the **Journalist-provided Deepfake Dataset (JDD)**, comprising 255 audio samples of public figures collected since early 2024 with the help of more than 70 journalists. Building on these resources, our **Context-based Audio Deepfake Detector (CADD)** integrates contextual information and transcripts alongside the acoustic signal, yielding substantial improvements in detection performance.

Unlearning memorized content from deployed models. When copyrighted, proprietary, or private content appears in a model’s training set, the model can reproduce it on demand, raising serious legal and privacy concerns. High-profile incidents involving unauthorized reproduction of artwork or news material by systems such as Stable Diffusion and ChatGPT illustrate the severity of this issue. We propose the **O³** continual unlearning framework, which selectively removes targeted knowledge from a deployed large language model without requiring access to the original training data. **O³** disentangles successive unlearning requests through orthogonal low-rank adapters and triggers the appropriate adapter at inference time using a glocal-aware¹ out-of-distribution detector, providing a robust and privacy-respecting solution that scales with the continual need for content removal.

¹“Glocal” is a portmanteau of “global” and “local”.

Aligning generation in dual-use scientific domains. The reach of generative models now extends beyond language, vision, and speech into biological sequence design, where protein language models trained on millions of natural sequences can generate novel enzymes, binders, and therapeutics. This capability carries an intrinsic dual-use risk: the same machinery that designs beneficial proteins is equally capable of producing toxins and virulence factors, and no safety layer presently mediates between generator outputs and downstream synthesis. When the generator itself is being constructed, the most direct intervention is to constrain it during training. Transferring alignment techniques from large language models to this setting is non-trivial, because protein safety is inherently multifaceted: toxicity and virulence are related but distinct axes with overlapping, imbalanced label distributions, and they must be reconciled with structural viability without a manually fixed trade-off. We propose ***SafePro***, a multi-objective alignment framework for protein language models that jointly optimizes structural foldability, toxicity avoidance, and virulence avoidance. *SafePro* couples a cross-task attentive reward model, which selectively shares signal between biologically coupled safety objectives through an input-dependent gate, with a conflict-aware gradient calibration procedure that adaptively navigates the utility–safety Pareto frontier during policy optimization, removing the need to predetermine reward weights.

Summary of Contributions

Taken together, the contributions of this dissertation span the full lifecycle of generative models. We use generation as a shield to enable secure and engaging content sharing, lower the cost of adapting large models for these constructive uses, and contain misuse

through detection, unlearning, and alignment. The individual contributions, elaborated in the chapters that follow, are:

- **Protecting Data Using Generative Models:** We propose **FakeDB**, a GAN-based framework that synthesizes relational databases while preserving semantic integrity constraints and statistical properties across multiple linked tables. FakeDB enforces cross-table consistency and achieves strong performance on both real-world and synthetic benchmarks.
- **Generating Engaging Multimodal Data:** We introduce **GEM**, a two-phase framework for generating engaging image–text pairs. GEM first trains an engagement classifier on social media data, then uses it to guide continuous-prompt learning for a diffusion model, and finally refines image–text alignment via an iterative similarity- and paraphrasing-based loop.
- **Efficient Fine-Tuning of Large Language Models:** We introduce **MoLA**, a parameter-efficient fine-tuning method that combines Mixture of Experts with LoRA and supports layer-wise expert allocation. MoLA varies the number of LoRA experts per Transformer layer, concentrating capacity in the middle layers where it is most beneficial.
- **Continual Unlearning for Large Language Models:** We present **$\mathcal{O}^3$** , a retain-data-free continual unlearning framework for LLMs. $\mathcal{O}^3$ uses orthogonal LoRA adapters to preserve unlearning effects across successive requests and a contrastive out-of-distribution detector to trigger unlearning at inference time.
- **Detecting Audio Deepfakes in the Wild:** We propose a context-aware audio deepfake detection framework that moves beyond raw signal analysis.

By incorporating speaker identity and content semantics, and by training and evaluating on the new P²V and JDD datasets, the method substantially improves detection accuracy under realistic conditions.

- **Safe Protein Generation via Multi-Objective Alignment:** We propose *SafePro*, a multi-objective alignment framework for protein language models that balances structural foldability against the avoidance of toxicity and virulence. *SafePro* introduces a cross-task attentive reward model that selectively shares signal between coupled safety objectives, together with a conflict-aware gradient calibration procedure that adaptively weights competing objectives during reinforcement learning.

CHAPTER 2

Protecting Data using Generative Models

Organizations often need to share data while safeguarding sensitive information and privacy. For example, healthcare providers may wish to share relational tables that reveal aggregate trends (*e.g.*, statistical summaries or predictive model results) without exposing personal patient data. Similarly, a biotech startup might share partial experimental results with sponsors to support inferences while withholding raw data. In such scenarios, both semantic constraints and statistical properties of the shared relational data must be preserved.

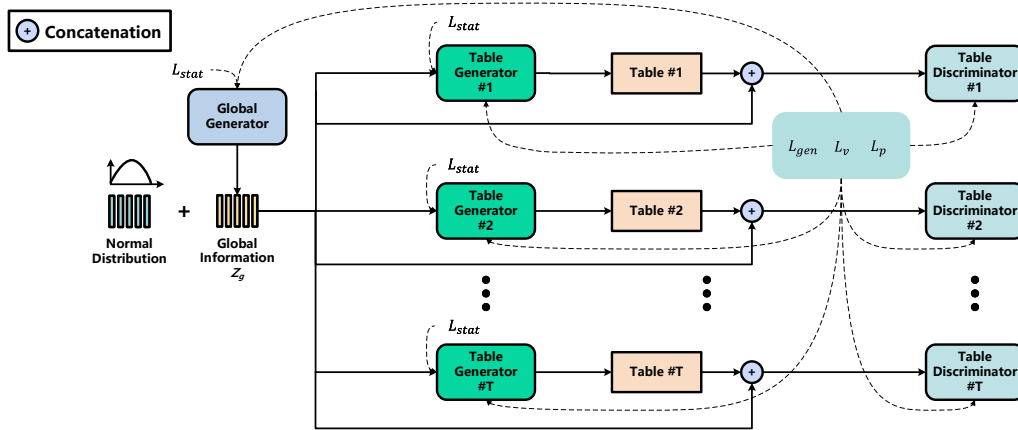


Figure 2.1. Logical architecture.

2.1. Introduction

We introduce FakeDB [56] that generates synthetic relational databases while preserving both statistical properties and semantic integrity constraints (ICs). These ICs are based on definite clauses [59, 115], encompassing rules such as ensuring positive and unique IDs, and maintaining consistency across tables (*e.g.*, an employee must have a positive interview outcome if listed in both Employee and Interview tables).

Our approach is based on a novel GAN architecture (Figure 2.1) that accepts a relational database and a set of integrity constraints (ICs) as input. These ICs are expressed using definite clauses, covering both unary constraints (within a table) and multi-relational constraints (across tables). For example, we enforce conditions such as unique IDs, valid ranges, and referential rules like ensuring that employees listed in one table have positive interview outcomes in another.

A key challenge lies in generating tables that jointly satisfy multi-relational constraints. Independent generation of related tables often fails to enforce cross-table consistency. To overcome this, FakeDB follows a three-phase process:

- **Unary Phase:** It synthesizes each table individually, preserving unary constraints and target statistics (mean, variance, skewness, etc.).
- **Relational Phase:** It aligns tables to satisfy non-unary constraints across relations.
- **Adversarial Phase:** Discriminators evaluate the realism of generated tables using constraint-specific losses, statistical objectives, and traditional GAN loss.

We evaluated FakeDB on eight benchmark datasets, including the IMDB datasets. Comparisons were made against state-of-the-art single-table generators: TableGAN, CTGAN, and SDV. Results demonstrate that FakeDB:

- Outperforms baselines in 19 of 31 semantic constraint evaluations;
- Achieves best statistical preservation (mean, std, kurtosis, skewness, and two EMD metrics) in 23 of 56 cases.

FakeDB thus represents a practical and effective solution for privacy-preserving relational data sharing that maintains both semantics and utility.

2.2. Related Work

Generating synthetic tables has been a long-standing research topic. Early methods treat each column as a random variable and generate data by sampling from a joint multivariate distribution using statistical techniques [151, 222, 8]. However, these approaches often fail to preserve semantic integrity constraints, limiting their applicability in real-world scenarios where logical consistency is essential.

The advent of deep learning has led to more sophisticated generative models, including GAN-based approaches [61]. tableGAN [135] applies convolutional GANs to capture dependencies within a table, while CTGAN [207] addresses non-Gaussian distributions using mode-specific normalization. Other works [26, 51, 72, 85, 110] also explore synthetic tabular data generation but focus only on single-table synthesis and ignore semantic constraints.

These methods typically operate at the row or column level and do not address the challenges of generating full relational databases with inter-table semantics. SDV [136] considers inter-table relationships via conditional parameter aggregation but remains limited to statistical dependencies. In contrast, our proposed FakeDB framework introduces

a global generator that captures both statistical and semantic relationships across multiple tables.

Broader efforts in privacy-preserving data sharing include differential privacy [229], which perturbs data to mask individual records while preserving global statistics. However, it does not explicitly maintain semantic constraints. Cryptographic techniques like secure multi-party computation (MPC)[52, 163] and homomorphic encryption[3] enable computations on encrypted data but require substantial engineering overhead and do not simplify data sharing. In contrast, FakeDB offers a practical, non-cryptographic alternative that generates usable, privacy-preserving synthetic relational data without altering existing database applications.

2.3. Method

We provide a high-level overview of the fakeDB architecture (Figure 2.1), which proceeds in three iterative phases. Formal definitions and implementation details follow.

Phase 1: Table-wise Generation. Each table in the database is assigned an independent generator that synthesizes a fake version of the table. These generators aim to minimize violations of unary integrity constraints (ICs)—constraints that apply to individual tables—and to preserve specified statistical properties (*e.g.*, mean, standard deviation, kurtosis, skewness)¹. At the end of this phase, we obtain a set of independently generated fake tables.

Phase 2: Cross-Table Consistency. While Phase 1 handles tables in isolation, real-world relational databases often include non-unary ICs, which define semantic relationships across tables. Since tables were generated independently in Phase 1, these multi-table

¹These statistics can be extended or replaced without loss of generality.

constraints may be violated. In this phase, we compute a global loss for the fake database based on these violations. This loss is propagated back to inform both the table generators and the discriminators.

Phase 3: Discriminator Feedback. Each fake table is evaluated by a dedicated discriminator trained to distinguish between real and synthetic data. These discriminators incorporate both table-level features and constraint violations from Phase 2. Their feedback is then used to refine the generators.

These three phases are repeated iteratively across multiple epochs until convergence. Then we provide the details of our method. We provide formal mathematical definitions for the various concepts in the fakeDB framework.

2.3.1. Mathematical Preliminaries

We start by introducing a set of background definitions to formalize some of the intuitions described above.

A *relation* $R(A_1, \dots, A_k)$ is a subset of $\text{dom}(A_1) \times \dots \times \text{dom}(A_k)$, where each A_i is an *attribute* and $\text{dom}(A_i)$ is a non-empty set denoting the set of acceptable values for attribute A_i . Every *tuple* $t \in R$ therefore has k attribute values. We refer to each attribute as $t.A_i$. Throughout this paper, tuples in tables are also referred to as *rows*.

An *atom* over tuples $t_1, \dots, t_n$ is an expression having one of the following forms: $t_i.A \text{ op}_c t_j.B$; $t_i.A \text{ op}_c c$; $x \text{ op}_c y$; $x \text{ op}_c c$; $t_i \text{ op}_{id} t_j$. where: $i, j \in [1, n]$; $\text{dom}(t_i.A) = \text{dom}(t_j.B)$; $c \in \text{dom}(t_i.A)$; $\text{op}_c \in \{=, \geq, \leq, <, >, \neq\}$; $\text{op}_{id} \in \{=, \neq\}$; x and y are expressions of the form $t_i.A \text{ op}_a t_j.B$ or $t_i.A \text{ op}_a c$ where $\text{op}_a \in \{+, -, *, /\}$.

We now define the class of integrity constraints considered by FakeDB. A *definite integrity constraint* ic over a set $\mathcal{R} = \{R_1, \dots, R_n\}$ of relations is an expression of the form

$$(\forall t_1 \in R_1) \dots (\forall t_n \in R_n) (Pre \rightarrow C)$$

where Pre is a (possibly empty) conjunction of atoms of the forms (1)–(5) over tuples $t_1, \dots, t_n$ and C is an atom of one of the forms (1)–(4) over t_i or t_i, t_j .

A *relational database* is a set $\mathcal{R} = \{R_1, \dots, R_n\}$ of relations. Given a definite integrity constraint $ic = (\forall t_1 \in R_1) \dots (\forall t_n \in R_n) (Pre \rightarrow C)$ over $\mathcal{R}$, the set

$$aff(\mathcal{R}, ic) = \{\langle t_1, \dots, t_n \rangle \mid t_i \in R_i \wedge \langle t_1, \dots, t_n \rangle \models Pre\}$$

contains all possible tuples (of rows) that can be composed using rows in $\mathcal{R}$ that are affected by ic , *i.e.*, tuples of rows which satisfy the pre-condition of ic . Similarly,

$$sat(\mathcal{R}, ic) = \{\langle t_1, \dots, t_n \rangle \mid t_i \in R_i \wedge \langle t_1, \dots, t_n \rangle \models Pre \wedge C\}$$

is the set of all tuples of rows that are affected by ic and satisfy it. Finally, $unsat(\mathcal{R}^f, ic) = aff(\mathcal{R}^f, ic) \setminus sat(\mathcal{R}^f, ic)$ is the set of all tuples of rows that are affected by ic and do not satisfy it.

A *fake version* $\mathcal{R}^f = \{R_1^f, \dots, R_n^f\}$ of $\mathcal{R}$ is a set of relations having the same schema as $\mathcal{R}$, *i.e.*, $R_i(A_1, \dots, A_k) \in \mathcal{R} \leftrightarrow R_i^f(A_1, \dots, A_k) \in \mathcal{R}^f$. Our goal in this paper is to generate a fake version of a relational database that preserves (i) a given set of ICs and (ii) a given set of statistical properties.

2.3.2. Loss Functions

Standard Generator and Discriminator Loss Functions. In *Minimax GANs*, the generator and discriminator models are optimized. Specifically, let $\mathbf{z}$ denote the injected noise of the generator and $G(\mathbf{z})$ denote a generator function mapping $\mathbf{z}$ to the generated data $\mathbf{x}$. Let $D(\mathbf{x})$ be a discriminator function that specifies the probability that a discriminator input $\mathbf{x}$ is real. Then: The generator seeks to minimize the log of the inverse probability predicted by the discriminator for fake data. This makes the generator provide samples that have a low probability of being fake: minimize $\log(1 - D(G(z)))$.

The discriminator's goal is to maximize the average of the log probability of real data and the log of the inverse probability of fake data: maximize $\log D(x) + \log(1 - D(G(z)))$.

Semantic Losses. Semantic losses occur when the generated fake database fails to preserve one or more integrity constraints. We start by defining the *degree of satisfaction* of an integrity constraint ic by $\mathcal{R}^f$ as

$$dos(\mathcal{R}^f, ic) = \frac{|sat(\mathcal{R}^f, ic)|}{|aff(\mathcal{R}^f, ic)|}.$$

Intuitively, $dos(\mathcal{R}^f, ic)$ is the percentage of rows in $\mathcal{R}^f$ that are affected by ic that satisfy ic . We define the loss function $\mathcal{L}_p(\mathcal{R}^f, ic)$ as $1 - dos(\mathcal{R}^f, ic)$.

In addition, we define the loss function $\mathcal{L}_v(\mathcal{R}^f, ic)$ as reported in Table 2.1 for each form the conclusion C of a definite integrity constraint can take — in the table, $\sigma(x)$ is the sigmoid function $\frac{e^x}{e^x+1}$ and $\epsilon > 0$.

Finally, given a set IC of definite integrity constraints, we define the total semantic losses across the set of integrity constraints as:

Table 2.1. Function $\mathcal{L}_v(\mathcal{R}^f, ic)$.

C	$\mathcal{L}_v(\mathcal{R}^f, ic)$
$t_i.A = t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} (t_i.A - t_j.B)^2$
$t_i.A \geq t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_j.B - t_i.A) - \frac{1}{2}, 0)$
$t_i.A \leq t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_i.A - t_j.B) - \frac{1}{2}, 0)$
$t_i.A > t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_j.B - t_i.A + \epsilon) - \frac{1}{2}, 0)$
$t_i.A < t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_i.A - t_j.B + \epsilon) - \frac{1}{2}, 0)$
$t_i.A \neq t_j.B$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \frac{1}{1 + (t_i.A - t_j.B)^2}$
$t_i.A = c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} (t_i.A - c)^2$
$t_i.A \geq c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(c - t_i.A) - \frac{1}{2}, 0)$
$t_i.A \leq c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_i.A - c) - \frac{1}{2}, 0)$
$t_i.A > c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(c - t_i.A + \epsilon) - \frac{1}{2}, 0)$
$t_i.A < c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \max(\sigma(t_i.A - c + \epsilon) - \frac{1}{2}, 0)$
$t_i.A \neq c$	$\frac{1}{ \text{unsat}(\mathcal{R}^f, ic) } \sum_{\langle t_1, \dots, t_n \rangle \in \text{unsat}(\mathcal{R}^f, ic)} \frac{1}{1 + (t_i.A - c)^2}$

$$\mathcal{L}_p(\mathcal{R}^f, IC) = \sum_{ic \in IC} \mathcal{L}_p(\mathcal{R}^f, ic)$$

and

$$\mathcal{L}_v(\mathcal{R}^f, IC) = \sum_{ic \in IC} \mathcal{L}_v(\mathcal{R}^f, ic).$$

Statistical Loss. The statistical loss function $\mathcal{L}_{stat}$ is aimed at preserving several statistics — in particular, for $stat \in \{\text{mean, standard deviation, kurtosis, skewness}\}$, the loss is the difference between the value of $stat$ of the original tables and that of the fake tables; the loss also includes the Wasserstein distance (EMD) between the distributions of the original and fake values.

Loss Functions for the Global Generator, Table Generators, and Table Discriminators. Finally, we set the loss function for generators and discriminators to be:

- For global and table generators: $\log(1 - D(G(\mathbf{z}))) + \mathcal{L}_p + \mathcal{L}_v$.

- For table discriminators: $-[\log(D(x)) + \log(1 - D(G(z)))] + \mathcal{L}_{stat} + \mathcal{L}_p + \mathcal{L}_v$.

2.4. Experiments

This section presents an empirical evaluation of FakeDB. The evaluation comprises two complementary studies: one based on a collection of real-world datasets and another based on a controlled synthetic dataset.

2.4.1. Experiments on Real Data

For the first study, we selected 8 datasets from the CTU Prague Relational Learning Repository [127]; their key characteristics are summarized in Table 2.2. These datasets were deliberately chosen to span a variety of application domains, so that the generality of FakeDB could be examined across heterogeneous settings. Specifically, the IMDb dataset comes from the entertainment industry, Trains originates from the transportation domain, UW-CSE reflects an academic/educational setting, while the remaining four datasets are drawn from biology and medicine.

Table 2.2. Datasets used in the experiments on real data.

Dataset	#Relations	#Rows	Size
IMDb	7	5,694,919	477.1MB
Biodegradability	5	21,875	3.3MB
Hepatitis	7	12,927	2.2MB
Trains	2	83	100KB
Musk	2	554	400KB
Mutagenesis	3	10,324	900KB
UW-CSE	4	712	200KB
Genes	3	6,063	1.8MB

The following integrity constraints were associated with each dataset:

- IMDb: $(\forall t_1 \in \text{director_genre}) (\forall t_2 \in \text{movies_director}) (\forall t_3 \in \text{movies_genres})$
 $(t_1.\text{director_id} = t_2.\text{director_id} \wedge t_2.\text{movie_id} = t_3.\text{movie_id} \rightarrow t_1.\text{genre} = t_3.\text{genre})$
- Biodegradability: $(\forall t_1 \in \text{molecule})(\rightarrow t_1.\text{activity} > 0)$
- Hepatitis: $(\forall t_1 \in \text{indis})(\rightarrow t_1.\text{gpt} < 7)$
- Trains: $(\forall t_1 \in \text{cars})(\forall t_2 \in \text{cars}) (t_1.\text{train_id} = t_2.\text{train_id} \rightarrow t_1.\text{position} \neq t_2.\text{position})$
- Musk: $(\forall t_1 \in \text{molecule})(\rightarrow t_1.\text{class} < 2)$
- Mutagenesis: $(\forall t_1 \in \text{molecule})(\rightarrow t_1.\text{logp} > t_1.\text{lumo})$
- UW-CSE:
 - (1) $(\forall t_1 \in \text{person}) (t_1.\text{student} = 1 \rightarrow t_1.\text{professor} = 0)$
 - (2) $(\forall t_1 \in \text{person}) (t_1.\text{hasPosition} \neq 0 \rightarrow t_1.\text{professor} = 1)$
 - (3) $(\forall t_1 \in \text{advisedBy})(\forall t_2 \in \text{person}) (t_1.p_id_dummy = t_2.p_id \rightarrow t_2.\text{professor} = 1)$
 - (4) $(\forall t_1 \in \text{taughtBy})(\forall t_2 \in \text{person}) (t_1.p_id = t_2.p_id \rightarrow t_2.\text{professor} = 1)$
 - (5) $(\forall t_1 \in \text{person}) (t_1.\text{yearsInProgram} \neq 0 \rightarrow t_1.\text{student} = 1)$
- Genes: $(\forall t_1 \in \text{classification})(\forall t_2 \in \text{genes}) (t_1.\text{geneid} = t_2.\text{geneid} \rightarrow t_1.\text{localization} = t_2.\text{localization})$

For every real dataset, we produced a corresponding fake version using four methods: tableGAN [135], CTGAN [207], SDV [136], and our proposed FakeDB.

Overall Degree of Satisfaction. The first metric we examined is the overall degree of satisfaction (*dos*). The outcomes are reported in Table 2.3. Note that tableGAN

could not be executed on either the IMDb or the Genes datasets because the attributes `movies_genre.genre` and `genes.localization`, respectively, contain string values.²

Furthermore, when *aff* is empty, the value of the metric is reported as undefined. As

Table 2.3. Values of *dos* obtained by **tableGAN**, **CTGAN**, **SDV**, and **FakeDB** (best values in bold, higher is better).

Dataset	tableGAN	CTGAN	SDV	FakeDB
IMDb	–	Undefined	Undefined	100%
Biodegradability	100%	100%	100%	100%
Hepatitis	100%	100%	100%	100%
Trains	100%	100%	Undefined	100%
Musk	100%	100%	97.8%	100%
Mutagenesis	100%	100%	99.5%	100%
UW-CSE	65.8%	40.3%	83.8%	90.6%
Genes	–	0%	26.7%	59.9%

can be seen, **FakeDB** was the only method able to fully satisfy the constraints on all 8 datasets. For reference, **tableGAN** and **CTGAN** each achieved the top score in 5 datasets, while **SDV** did so in 2.

Values of Loss Function $\mathcal{L}_p$. We then turned to the loss function $\mathcal{L}_p$. Table 2.4 reports the values produced by **FakeDB** on the UW-CSE dataset across every possible subset of its integrity constraints, and an aggregated view of these results is given in Table 2.5.

Looking at these numbers, two observations stand out: **FakeDB** achieves the top loss value in 19 of the 31 configurations, and its average deviation from the best achievable value is smaller than that of any competing technique.

Values of Statistics. To assess how well statistical properties are preserved, we selected three relations per dataset, as listed in Table 2.6, classified as large, medium, and small

²In **FakeDB**, string attributes are handled by mapping each distinct string to a unique integer. The same preprocessing was applied when running **CTGAN** and **SDV**. This workaround is not compatible with **tableGAN**, which does not accept string-valued inputs.

Table 2.4. Values of $\mathcal{L}_p$ obtained by tableGAN, CTGAN, SDV, and FakeDB for the UW-CSE dataset, for different sets of integrity constraints (best values in bold, lower is better).

ICs	tableGAN	CTGAN	SDV	FakeDB
{1}	0.328	0.745	0.633	0.482
{2}	1.000	0.921	0.872	0.700
{3}	1.000	0.883	0.976	0.774
{4}	1.000	0.843	0.946	0.783
{5}	0.328	0.838	0.639	0.716
{1,2}	1.328	1.665	1.505	1.182
{1,3}	1.328	1.628	1.609	1.256
{1,4}	1.328	1.588	1.579	1.265
{1,5}	0.656	1.582	1.272	1.198
{2,3}	2.000	1.804	1.847	1.474
{2,4}	2.000	1.764	1.818	1.483
{2,5}	1.328	1.758	1.511	1.416
{3,4}	2.000	1.727	1.922	1.557
{3,5}	1.328	1.721	1.615	1.491
{4,5}	1.328	1.681	1.585	1.499
{1,2,3}	2.328	2.549	2.480	1.956
{1,2,4}	2.328	2.509	2.451	1.965
{1,2,5}	1.656	2.503	2.144	1.898
{1,3,4}	2.328	2.471	2.555	2.039
{1,3,5}	1.656	2.466	2.248	1.973
{1,4,5}	1.656	2.426	2.218	1.981
{2,3,4}	3.000	2.647	2.793	2.257
{2,3,5}	2.328	2.642	2.487	2.191
{2,4,5}	2.328	2.602	2.457	2.199
{3,4,5}	2.328	2.564	2.561	2.273
{1,2,3,4}	3.328	3.392	3.426	2.739
{1,2,3,5}	2.656	3.386	3.120	2.673
{1,2,4,5}	2.656	3.346	3.090	2.681
{1,3,4,5}	2.656	3.309	3.194	2.755
{2,3,4,5}	3.328	3.485	3.432	2.973
{1,2,3,4,5}	3.656	4.230	4.066	3.455

based on row count. The relation with the most rows was taken as the large one, and the relation with the fewest rows as the small one. For the medium relation, preference was

Table 2.5. Summary of the results in Table 2.4 (for the average increase, lower is better).

	#Wins over 31 cases	Average increase wrt best value
tableGAN	12	11.6%
CTGAN	0	37.1%
SDV	0	29.3%
FakeDB	19	10.9%

given to relations appearing in the consequent (part C) of the integrity constraints. Two datasets, Trains and Musk, only have two relations, so no medium relation was defined for them. Table 2.6 additionally indicates which attribute was used to compute the statistics in each relation; numeric attributes were preferred, and when more than one such attribute was available, one was selected at random.

Table 2.6. Relations and attributes used to compute statistics.

	Large (attr.)	Medium (attr.)	Small (attr.)
IMDb	directors_genres (prob)	movies (rank)	movies_directors (movie_id)
Biodegradability	atom (atom_id)	bond (type)	molecule (logp)
Hepatitis	rel12 (m_id)	dispat (age)	bio (activity)
Trains	cars (position)	–	trains (id)
Musk	conformation (f1)	–	molecule (id)
Mutagenesis	atom (type)	bond (type)	molecule (logp)
UW-CSE	person (professor)	advisedBy (p_id_dummy)	taughtBy (p_id)
Genes	genes (chromosome)	interactions (expression_corr)	classification (geneid)

The statistics computed on data produced by FakeDB are reported in Tables 2.7 and 2.8.

These numbers are encouraging: averaged across all cases, the deviation between the fake and the real value was roughly 12.7% for the mean, 17.7% for the standard deviation, 33.2% for the kurtosis, and 47.1% for the skewness.

Next, we compared FakeDB against tableGAN, CTGAN, and SDV using a single relation per dataset. The large relation was used in most cases, except for Biodegradability and

Table 2.7. Mean, standard deviation, kurtosis, and skewness obtained by FakeDB.

		Mean	Fake mean	STD	Fake STD	Kurt.	Fake kurt.	Skew.	Fake skew.
IMDb	Large	0.710	0.710	0.340	0.310	-1.304	-1.008	-0.587	-0.478
	Medium	5.730	5.400	1.650	2.590	0.032	0.287	-0.309	-0.423
	Small	215585.54	213519.39	125430.70	126806.91	-1.253	-1.283	-0.089	-0.074
Biodegrad.	Large	499.160	509.460	293.070	283.520	-1.200	-1.159	0.034	-0.050
	Medium	499.500	485.810	288.670	300.400	-1.215	-1.241	0.000	0.100
	Small	2.290	2.650	2.200	3.330	0.905	0.016	0.631	0.487
Hepatitis	Large	48.330	44.030	27.210	28.250	-1.222	-0.765	-0.010	0.276
	Medium	4.400	3.000	1.430	2.080	0.907	-1.078	-1.316	-0.562
	Small	2.160	1.750	1.170	1.250	-1.017	-0.809	0.041	0.288
Trains	Large	2.190	2.200	1.040	1.240	-1.039	-1.041	0.382	0.387
	Small	10.500	8.100	5.760	5.770	-1.206	-0.338	0.000	0.289
Musk	Large	38.730	32.500	18.060	26.860	2.438	2.575	-0.065	0.531
	Small	0.510	0.460	0.490	0.490	-1.998	-1.969	-0.043	0.174
Mutagenesis	Large	36.840	34.477	9.530	14.310	8.983	1.411	-2.643	-1.430
	Medium	3.910	3.072	2.900	2.728	-1.955	-1.522	0.102	0.506
	Small	3.520	3.135	1.480	1.390	-0.411	-0.733	0.326	0.184
UW-CSE	Large	0.220	0.310	0.420	0.460	-0.229	-1.288	1.331	0.843
	Medium	214.200	222.550	124.350	117.270	-1.215	-0.788	0.057	0.180
	Small	231.830	296.870	114.020	150.900	-1.114	-1.709	-0.207	0.338
Genes	Large	5.654	6.186	2.930	3.157	-1.108	-1.191	0.348	0.204
	Medium	0.340	0.390	0.470	0.500	-0.669	-0.474	-0.559	-0.399
	Small	430.500	465.950	248.840	199.910	-1.200	-0.552	0.000	-0.093

UW-CSE (for which the large relation contains no numeric attributes, so the medium relation was used instead) and Musk (where we used the small relation because **tableGAN** is unable to handle negative values). This yielded a total of 56 comparison points — 7 statistics evaluated over 8 relations. Tables 2.9 and 2.10 list the numerical results, and Table 2.11 summarizes them.

Several patterns emerge from these summaries. Of the 56 cases examined, **FakeDB** ranks first in 23 of them — more than any of the competing systems. Its average divergence from the true values of the mean, standard deviation, kurtosis, and skewness remains

Table 2.8. EMD, Normalized EMD 1, and Normalized EMD 2 obtained by FakeDB.

		EMD	Normalized EMD 1	Normalized EMD 2
IMDb	Large	0.067	0.00010	0.140
	Medium	0.585	0.00055	0.086
	Small	2512.651	0.00001	0.006
Biodegradability	Large	11.460	0.00002	0.011
	Medium	15.720	0.00003	0.017
	Small	0.980	0.00048	0.089
Hepatitis	Large	5.180	0.00009	0.085
	Medium	1.400	0.00071	0.170
	Small	0.410	0.00840	0.100
Trains	Large	0.200	0.00150	0.130
	Small	2.600	0.00720	0.130
Musk	Large	9.360	0.00044	0.018
	Small	0.054	0.00230	0.054
Mutagenesis	Large	5.503	0.00016	0.270
	Medium	0.844	0.00033	0.140
	Small	0.388	0.00021	0.040
UW-CSE	Large	0.080	0.00190	0.082
	Medium	18.680	0.00070	0.066
	Small	79.190	0.00090	0.190
Genes	Large	0.607	0.00008	0.065
	Medium	0.050	0.00009	0.088
	Small	58.670	0.00017	0.084

small, and the same is true for the average EMD, Normalized EMD 1, and Normalized EMD 2.

We performed a broader comparison involving every relation listed in Table 2.6, which expanded the number of comparison points to 154 (7 statistics across 22 relations). The win counts for each method are given in Table 2.12. When the full set of relations is considered, SDV has a slight edge over FakeDB in retaining statistical characteristics.

Table 2.9. Mean, standard deviation, kurtosis, and skewness obtained by tableGAN, CTGAN, SDV, and FakeDB (best values in bold).

		Mean	Fake mean	STD	Fake STD	Kurt.	Fake kurt.	Skew.	Fake skew.
IMDb (Large)	tableGAN CTGAN SDV FakeDB	0.710	0.750 0.730 0.714 0.710	0.340	0.314 0.390 0.346 0.310	-1.304	-0.846 -0.390 -0.009 -1.008	-0.587	-0.876 -1.144 -0.105 -0.478
Biodegrad. (Medium)	tableGAN CTGAN SDV FakeDB	499.500	688.190 463.369 499.500 485.810	288.670	168.670 338.000 288.670 300.400	-1.215	-1.136 -0.957 -1.200 -1.241	0.000	-0.070 0.269 0.000 0.100
Hepatitis (Large)	tableGAN CTGAN SDV FakeDB	48.330	26.380 48.146 47.169 44.030	27.210	15.860 23.110 27.440 28.250	-1.222	-1.423 -0.843 -0.049 -0.765	-0.010	-0.347 -0.183 0.068 0.276
Trains (Large)	tableGAN CTGAN SDV FakeDB	2.190	9.900 2.610 1.690 2.200	1.040	1.300 1.210 1.020 1.240	-1.039	0.376 -1.093 -0.048 -1.041	0.382	-0.432 0.118 0.288 0.387
Musk (Small)	tableGAN CTGAN SDV FakeDB	0.510	0.280 0.280 0.520 0.460	0.490	0.450 0.450 0.590 0.490	-1.998	-1.067 -1.035 -0.421 -1.969	-0.043	0.966 0.982 -0.058 0.174
Mutagenesis (Large)	tableGAN CTGAN SDV FakeDB	36.840	22.166 36.429 37.121 34.477	9.530	13.304 9.277 9.689 14.310	8.983	-1.430 6.728 -0.122 1.411	-2.643	-0.733 -2.727 -0.032 -1.430
UW-CSE (Medium)	tableGAN CTGAN SDV FakeDB	214.200	202.630 220.130 220.590 222.550	124.350	22.614 122.960 124.900 117.270	-1.215	-0.585 -0.685 0.116 -0.788	0.057	-0.092 0.072 0.141 0.180
Genes (Large)	tableGAN CTGAN SDV FakeDB	5.654	5.739 3.560 5.577 6.186	2.930	2.580 2.750 2.980 3.157	-1.108	-1.070 -0.762 0.133 -1.191	0.348	0.195 0.431 0.080 0.204

Runtime. The runtime behavior of FakeDB as a function of the percentage of data processed is plotted in Figure 2.2. The plot indicates that FakeDB scales favorably with the data volume. For 6 of the 8 datasets, essentially no runtime growth is observed as the

Table 2.10. EMD, Normalized EMD 1, and Normalized EMD 2 obtained by tableGAN, CTGAN, SDV, and FakeDB (best values in bold).

		EMD	Normalized EMD 1	Normalized EMD 2
IMDb (Large)	tableGAN	0.053	0.00010	0.044
	CTGAN	0.100	0.00016	0.110
	SDV	0.135	0.00019	0.240
	FakeDB	0.067	0.00010	0.140
Biodegradability (Medium)	tableGAN	191.781	0.00030	0.05
	CTGAN	58.833	0.00013	0.063
	SDV	5.350	0.00001	0.005
	FakeDB	15.720	0.00003	0.017
Hepatitis (Large)	tableGAN	21.957	0.00012	0.080
	CTGAN	4.795	0.00010	0.060
	SDV	4.238	0.00009	0.125
	FakeDB	5.180	0.00009	0.085
Trains (Large)	tableGAN	7.712	0.00490	0.260
	CTGAN	0.455	0.00300	0.170
	SDV	0.508	0.01490	0.160
	FakeDB	0.200	0.00150	0.130
Musk (Small)	tableGAN	0.228	0.00970	0.228
	CTGAN	0.231	0.00970	0.231
	SDV	0.054	0.00990	0.317
	FakeDB	0.054	0.00230	0.054
Mutagenesis (Large)	tableGAN	14.680	0.00044	0.340
	CTGAN	1.275	0.00004	0.280
	SDV	6.281	0.00017	0.140
	FakeDB	5.503	0.00016	0.270
UW-CSE (Medium)	tableGAN	90.080	0.00370	0.099
	CTGAN	16.200	0.00790	0.105
	SDV	24.079	0.00780	0.155
	FakeDB	18.680	0.00070	0.066
Genes (Large)	tableGAN	0.401	0.00008	0.066
	CTGAN	2.094	0.00025	0.109
	SDV	0.623	0.00011	0.154
	FakeDB	0.607	0.00008	0.065

data size increases. For the UW-CSE and Genes datasets, by contrast, the runtime grows approximately linearly — the UW-CSE curve is somewhat less smooth, which is likely

Table 2.11. Summary of the results in Tables 2.9 and 2.10 (for the average difference and average value, lower is better).

	#Wins over 56 cases	Avg. difference wrt real value				Average value		
		Mean	STD	Kurt.	Skew.	EMD	Norm. EMD 1	Norm. EMD 2
tableGAN	6	66.6%	32.2%	51.5%	794.5%	40.8615	0.0024	0.1463
CTGAN	12	14.5%	10.2%	34.5%	541.5%	10.4978	0.0027	0.1410
SDV	19	4.1%	3.6%	86.7%	155.5%	5.1585	0.0041	0.1620
FakeDB	23	5.2%	12.4%	23.9%	103.4%	5.514	0.0006	0.1034

Table 2.12. Summary of the results obtained using all the relations in Table 2.6.

	#Wins out of 154 cases
tableGAN	10
CTGAN	17
SDV	74
FakeDB	53

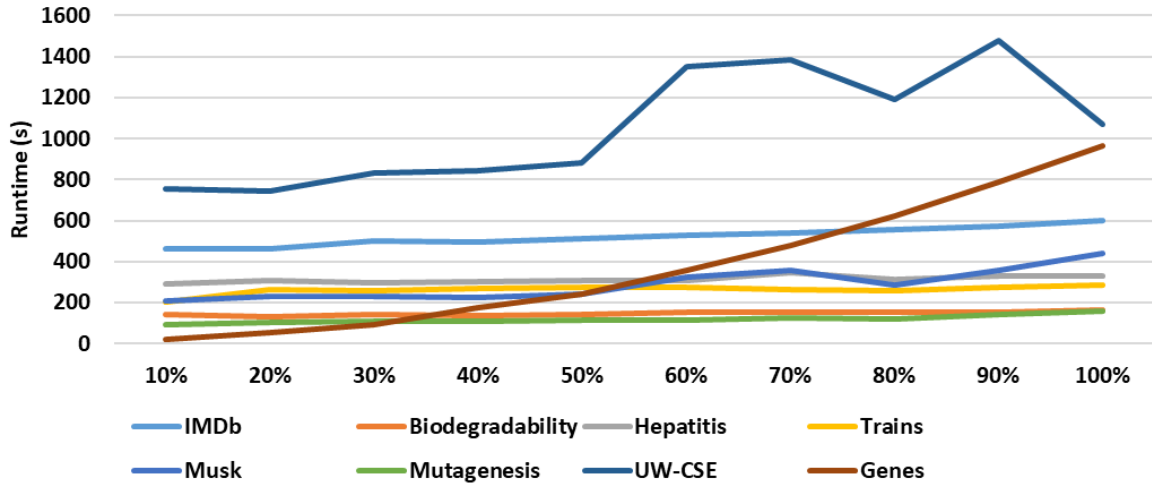


Figure 2.2. Runtime when varying the percentage of data.

attributable to the more intricate integrity constraints of that dataset, whose behavior depends on which portion of the data is used in a given run.

2.4.2. Experiments on Synthetic Data

To complement the real-world evaluation, we devised a controlled synthetic benchmark. Given a target size S , we constructed two relations, $R1(A_1, A_2, A_3, A_4)$ and $R2(B_1, B_2, B_3, B_4)$, each populated with S rows. Using the notation X_j^i to denote the value of attribute X_j in the i -th row, the rows were generated as follows, for every $i \in [1, S]$:

- (1) A_1^i and B_1^i were both assigned the value $i - 1$.
- (2) A_2^i was drawn uniformly at random from the interval $[-10, 10]$.
- (3) When $A_2^i \geq 0$, the value of B_3^i was drawn uniformly at random from $[0.1, 10]$; otherwise, B_3^i was set to 0.
- (4) A_3^i was assigned the value of B_3^i .
- (5) B_2^i was computed as $A_2^i + A_3^i$.
- (6) B_4^i was sampled uniformly at random from the interval $[A_2^i - 10, A_2^i]$.
- (7) For $i > 0$, A_4^i was defined as $A_4^{i-1} + v$, with v drawn uniformly at random from $[0, 10]$; for $i = 0$, A_4^i was set to 0.

The integrity constraints associated with $R1$ and $R2$ were:

- (1) $(\forall t_1 \in R1)(\forall t_2 \in R2) (t_1.A_1 = t_2.B_1 \rightarrow t_1.A_2 + t_1.A_3 = t_2.B_2)$
- (2) $(\forall t_1 \in R1)(\forall t_2 \in R2) (t_1.A_1 = t_2.B_1 \rightarrow t_1.A_3 = t_2.B_3)$
- (3) $(\forall t_1 \in R1)(\forall t_2 \in R2) (t_1.A_1 = t_2.B_1 \wedge t_1.A_2 \geq 0 \rightarrow t_2.B_3 > 0)$
- (4) $(\forall t_1 \in R1)(\forall t_2 \in R2) (t_1.A_1 = t_2.B_1 \wedge t_1.A_2 < 0 \rightarrow t_2.B_3 = 0)$
- (5) $(\forall t_1 \in R1)(\forall t_2 \in R2) (t_1.A_1 = t_2.B_1 \rightarrow t_1.A_2 > t_2.B_4)$
- (6) $(\forall t_1 \in R1)(\forall t_2 \in R1) (t_1.A_1 > t_2.A_1 \rightarrow t_1.A_4 > t_2.A_4)$

Values of Loss Function $\mathcal{L}_p$. Table 2.13 reports $\mathcal{L}_p$ values obtained on the generated data, and Table 2.14 consolidates the main trends. On this benchmark, FakeDB again

Table 2.13. Values of $\mathcal{L}_p$ obtained by tableGAN, CTGAN, SDV, and FakeDB for the synthetic dataset (best values in bold, lower is better).

Size	ICs	tableGAN	CTGAN	SDV	FakeDB
500	{1}	0.985	0.963	0.966	0.928
	{2}	0.665	0.842	0.928	0.821
	{3}	0.435	0.495	0.306	0.000
	{4}	0.656	0.518	0.838	0.557
	{5}	0.149	0.257	0.348	0.090
	{6}	0.139	0.501	0.485	0.329
	{1,2,3,4,5,6}	3.029	3.576	3.871	2.724
1000	{1}	0.961	0.969	0.952	0.915
	{2}	0.623	0.862	0.898	0.483
	{3}	0.756	0.203	0.299	0.842
	{4}	0.209	0.780	0.844	0.187
	{5}	0.538	0.295	0.323	0.048
	{6}	0.092	0.505	0.491	0.293
	{1,2,3,4,5,6}	3.179	3.615	3.807	2.768
1500	{1}	0.950	0.967	0.956	0.953
	{2}	0.691	0.843	0.900	0.377
	{3}	0.471	0.328	0.319	0.919
	{4}	0.520	0.650	0.843	0.179
	{5}	0.090	0.093	0.285	0.005
	{6}	0.180	0.504	0.498	0.255
	{1,2,3,4,5,6}	2.903	3.385	3.800	2.688
2000	{1}	0.945	0.965	0.965	0.944
	{2}	0.626	0.893	0.885	0.420
	{3}	0.653	0.309	0.303	0.827
	{4}	0.325	0.758	0.855	0.203
	{5}	0.376	0.302	0.307	0.044
	{6}	0.128	0.481	0.507	0.329
	{1,2,3,4,5,6}	3.053	3.707	3.823	2.767

comes out ahead: it wins 18 of the 28 configurations and its average deviation from the best available value is an order of magnitude smaller than that of any competitor.

Table 2.14. Summary of the results in Table 2.13 (for the average increase, lower is better).

	#Wins over 28 cases	Average increase wrt best value
tableGAN	7	160.1%
CTGAN	3	198.1%
SDV	3	353.9%
FakeDB	18	45%

Consistent with the pattern observed on real datasets, **SDV** struggles to produce fake data that respects the integrity constraints.

Runtime. The runtime of **FakeDB** across different values of the synthetic-dataset size S , when all 6 integrity constraints are active, is depicted in Figure 2.3. As the figure confirms,

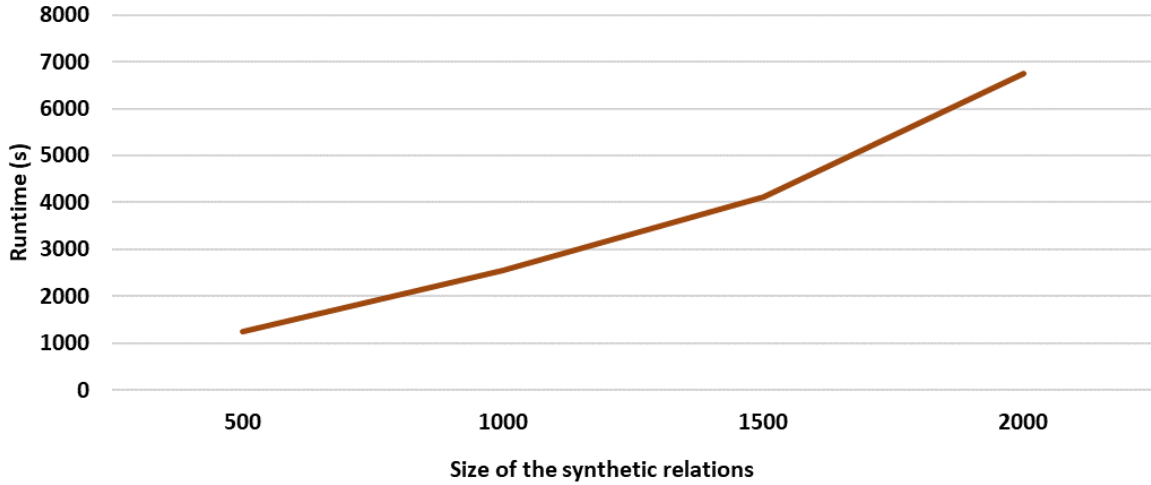


Figure 2.3. Runtime when varying the size of the synthetic relations.

FakeDB exhibits a linear scaling behavior with respect to data volume.

2.4.3. Summary of the Experimental Results

The experimental findings can be condensed into the following points.

- Real datasets:
 - Degree of satisfaction: **FakeDB** was the only approach that achieved perfect constraint satisfaction on every one of the 8 datasets. In head-to-head terms, **tableGAN** tied for best in 5 datasets, **CTGAN** in 5, **SDV** in 2, and **FakeDB** in all 8.
 - Loss function $\mathcal{L}_p$: **FakeDB** delivered the top score in 19 of the 31 tested subsets of constraints and had the lowest average gap with respect to the best value.
 - Values of statistics: On average, **FakeDB**'s fake data deviated from the real data by 12.7% on the mean, 17.7% on the standard deviation, 33.2% on kurtosis, and 47.1% on skewness. When compared to the three baselines on one relation per dataset, **FakeDB** was best in 23 of the 56 configurations, while also yielding low average deviations on mean, standard deviation, kurtosis, and skewness and low average EMD, Normalized EMD 1, and Normalized EMD 2. When the full set of relations from Table 2.6 was used, **SDV** showed a marginal edge in statistical fidelity. Nevertheless, this small advantage is largely offset by **FakeDB**'s substantially stronger performance in satisfying integrity constraints.
 - Runtime: **FakeDB** demonstrated very good scalability in terms of data volume. Six of the 8 datasets exhibited essentially flat runtime curves; for the other 2, the runtime grew linearly.
- Synthetic datasets:

- Loss function $\mathcal{L}_p$: FakeDB produced the best loss in 18 of the 28 configurations and kept the average deviation from the best value an order of magnitude lower than that of any competitor.
- Runtime: FakeDB exhibited linear scaling with the data size.

2.5. Conclusion

We address the critical problem of generating synthetic relational databases that preserve both semantic integrity and statistical properties. Our proposed framework, FakeDB, is the first to jointly maintain integrity constraints and column-wise distributions across single and multiple tables. Extensive experiments on eight real-world datasets and a synthetic benchmark demonstrate that FakeDB consistently outperforms three strong baselines, achieving high fidelity in a computationally efficient manner.

This work is published in IEEE Transactions on Dependable and Secure Computing (2024). *Gao, Chongyang, et al. "FakeDB: generating fake synthetic databases." IEEE Transactions on Dependable and Secure Computing (2024).*

CHAPTER 3

Generating Engaging Multimodal Data

GEM [57] tackles a different facet of content protection—generating content that adheres to specific engagement and semantic properties.

3.1. Introduction

The rise of social media platforms has underscored the growing significance of multimodal content across domains such as advertising, image-text posts, and educational media. Generating engaging image-text content is increasingly vital for protecting intellectual properties or privacy.

Recent advances in generative models have enabled the creation of high-quality unimodal content, ranging from poetic text [6] and executable code [155] to descriptive visual outputs [76, 210]. In parallel, diffusion models have emerged as powerful tools for generating photorealistic images [154, 67]. However, the synthesis of coherent multimodal content—particularly tightly aligned image-text pairs—remains underexplored. Existing models often operate in narrow domains or rely on limited training data, which constrains both output diversity and user engagement [91, 70, 143].

To address these limitations, we propose GEM, a novel framework designed to generate engaging multimodal image-text content that enhances user interaction while maintaining semantic alignment between modalities. Unlike prior work, GEM emphasizes engagement under real-world constraints, such as the 280-character limit of platforms like Twitter. The

framework adopts a two-stage process: first, it jointly trains an engagement discriminator and a diffusion model to generate visually and contextually relevant image prompts; second, it iteratively combines these image outputs with a paraphrasing module to produce coherent and captivating image-text pairs.

Empirical evaluations and human studies confirm that **GEM** significantly outperforms existing baselines in generating open-domain multimodal content, especially in terms of engagement metrics. These results highlight **GEM**’s effectiveness in addressing key challenges in multimodal generation—namely alignment, diversity, and user appeal.

3.2. Related Work

The digital era has witnessed rapid advances in large-scale vision-and-language models aimed at generating visually compelling and semantically coherent content for social media. Generative Adversarial Networks (GANs)[62] have become a foundational technique, capable of synthesizing images that align closely with textual prompts—an essential requirement for creating engaging social media posts. Notable developments include Conditional GANs[73, 56], approaches to push the realism and alignment of generated content, enriching the user experience.

Recent trends have shifted toward transformer-based architectures, which blend the strengths of GANs and transformers to achieve more integrated multimodal generation [92, 220]. Models such as DALL · E 2 [148] exemplify this direction, generating high-quality images from textual prompts—a critical capability for increasing user engagement across visual platforms.

3.3. Method

Our GEM model establishes a novel approach to generating multimodal content that is highly engaging, utilizing a fusion of text and image components based on GPT [18] and the recent text-to-image Stable Diffusion (SD) model, respectively. Figure 3.1 shows the training process within the GEM framework that involves a twofold procedure: an engagement classifier is trained to provide the engaging score, and then continuous prompts for diffusion model are learned with the supervision of both engaging and reconstruction losses. During inference, we proposed an iterative algorithm, as illustrated in Alg. 1, which enhances the alignment between text and generated images.

3.3.1. Engagement Classifier for Generation Guidance

We first train an “engagement” classifier to provide engagement guidance for training continuous prompts to generate images with high engagement scores. The engagement classifier f_c contains a pre-trained vision-language model, namely ViLT [88], to extract multimodal features from the text and image pairs. Next, we attach a fully connected module on top of ViLT to assess the engagement level of the pairs. The fully connected module consists of two fully connected layers and one ReLU [5] activation layer between them.

The engagement classifier, denoted by f_{cls} , is trained on the engagement dataset [86] using image-text-score triplets $(Img, Text, Score)$ until convergence. The engagement scores are binarized to allow a classification task and we use the cross-entropy loss to perform the supervised training.

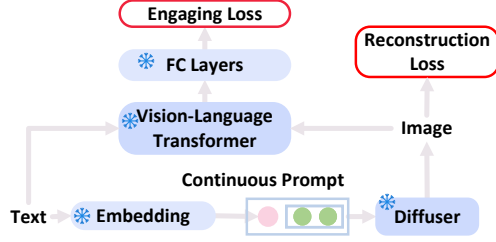


Figure 3.1. The training of continuous prompts.

3.3.2. Prompt-based Image Generator

We adopt Stable Diffusion (f_d) [154] as our base image generation model for text-to-image generation. To further control the generation, we concatenate a set of learnable vectors (cp) as continuous prompts [98] and text embeddings. The output of the diffusion model I_e can be denoted as follows:

$$(3.1) \quad I_e = f_d(z, t, \text{Concate}(cp, \text{Emb}(\text{Txt}))),$$

where t is the time step uniformly sampled from $1, \dots, T$, and T is the length of time steps. $z \in \mathcal{N}(0, I)$ is the latent map, which is the Gaussian noise. $\text{Emb}()$ is the text embedding layer of diffusion model f_d , and Concate denotes the concatenation operation. The dimension of cp is $l \times d$. $p \in \mathbb{R}^{l \times d}$, where l is the length of the continuous prompt and d is the hidden dimension size of the text embedding. Stable diffusion uses a pre-trained CLIP text encoder [145], and the hidden dimension size is 768. To encourage the generation of engaging images, we forward the diffusion output I_e and input Txt to the pre-trained engagement classifier f_{cls} introduced in Section 3.3.1. By increasing the positive confidence of f_{cls} in predicting a generated pair (I_e, Txt) , the classifier f_{cls}

back-propagates the gradients into the diffusion model f_d to produce more engaging images.

To be precise,

$$(3.2) \quad \mathcal{L}_{\text{engaging}} = \text{CrossEntropy}(f_{cls}(I_e, Txt), 1),$$

where 1 in Equation 3.2 refers to the fact that the label of (I_e, Txt) is an engaging pair.

In addition to the $\mathcal{L}_{\text{engaging}}$ loss, we keep the standard reconstruction loss [154] in the training process of continuous prompts. In particular, the reconstruction loss computes the difference between the predicted noise and the original noise, i.e.:

$$(3.3) \quad \mathcal{L}_{\text{rec}} = \mathbb{E}_{x, \epsilon, t} \left[|\epsilon - f_d(z_t, t, e)|_2^2 \right],$$

where z_t is acquired by encoding the input image and x into the latent space using stable diffusion’s autoencoder, which is the noisy version of x and can be calculated using $\epsilon \sim \mathcal{N}(0, I)$. ϵ is the original noise, and e is $\text{Concate}(cp, \text{Emb}(Txt))$. Thus, the total loss for learning our continuous prompts is

$$(3.4) \quad \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{engaging}} + w \cdot \mathcal{L}_{\text{rec}}$$

, where w is the weight, which we set to 1 in our experiments because the scales of these two losses are comparable. During training, we freeze the parameters of diffusion model f_d in order to maintain the model’s ability to generate high-quality images while retaining control over the generation’s engagement score. Therefore, the only learnable parameters

in our framework are in *cp*. Similar strategies have been found to be both efficient and effective in fine-tuning Transformer-based language models in NLP tasks [93, 98, 112].

3.3.3. Iterative Image-Text Post Generation

A well-aligned image and text pair requires the image and text to share substantial similarities. Based on this intuition, we devise an iterative procedure for generating the image and text pairs with the help of CLIP-based similarity scores. Algorithm 1 illustrate our iterative algorithm for image and text alignment.

Given the text, we first use the image generator described in Section 3.3.2 to generate an engaging image, I_e , and then calculate the similarity between the original text and I_e as Sim_O , as shown in Lines 5-6 of Alg.1. We then generate the new text, Txt' , using the GPT-3.5 as text paraphraser, as shown in Line 7 of Alg. 1. A pre-trained CLIP model is used to measure the similarity between the generated image I_e and all the text, Txt, Txt' . In particular, we can generate any number of Txt' . In our experiments, we generate 10 Txt' instances. If (Txt, I_e) has the highest similarity score, then we return (Txt, I_e) as the generated image and text pair, as illustrated in Lines 15-16.

If there exists a Txt' such that the similarity score of (Txt', I_e) is greater than the similarity score of (Txt, I_e) , the Txt' with the highest similarity score is used as the new Txt , as shown in Line 10. Next, we compare the similarity between (Txt, I_e) and $Sim_O + S$, where S is the threshold used to constrain the similarity value to be at least S greater than the similarity of the original text and generated image. In our experiments, S is set to 1. If the similarity of (Txt, I_e) is greater than $Sim_O + S$, the algorithm returns (Txt, I_e) , as illustrated in Lines 11-12. If not, we use the image generator to generate a

Algorithm 1: Iterative Generation Process

```

TP: text paraphraser
Txt: Input text
Sim: the similarity score provided by CLIP
S: threshold of the termination
 $I_e = f_d(\text{Concate}(cp, \text{Emb}(\text{Txt})))$ 
 $\text{Sim}_O = \text{Sim}(\text{Txt}, I_e)$ 
 $\text{Txt}' = TP(\text{Txt})$ 
for Number of iteration do
  if  $\text{Sim}(\text{Txt}', I_e) > \text{Sim}(\text{Txt}, I_e)$  then
     $\text{Txt} = \text{Txt}'$ 
    if  $\text{Sim}(\text{Txt}, I_e) > \text{Sim}_O + S$  then
      | return ( $\text{Txt}, I_e$ )
     $I_e = f_{diff}(\text{Concate}(cp, \text{Emb}(\text{Txt})))$ 
     $\text{Txt}' = TP(\text{Txt})$ 
  else
    | return ( $\text{Txt}, \text{Img}$ )
return ( $\text{Txt}, I_e$ )

```

new image given the new Txt and generate text Txt' based on the new Txt , as shown in Lines 13-14. In order to make the algorithm more efficient and avoid the problem of the endpoint not being achieved, we limit the maximum number of iterations (it is set to 50 in our experiments). The iterative image and text pair generation process enables our method to generate more well-aligned and engaging image and text pairs.

3.4. Experiments

In this section, we describe the experimental protocol used to measure the quality of the image-text posts produced by our framework. Every method is evaluated along two complementary axes: a quantitative axis based on CLIP similarity scores (Section 3.4.4), and a qualitative axis based on a human-subject study that examines how engaging the generated posts are (Section 3.4.5).

3.4.1. Dataset

For training both the engagement classifier and the continuous-prompt component of the stable diffusion image generator, we relied on the Instagram Influencer Dataset [86]. This corpus contains 10,180,500 Instagram posts authored by 33,935 influencers and is organized into nine categories: beauty, family, fashion, fitness, food, interior, pet, travel, and others. Upon inspection, we observed that the original post text is often very brief, occasionally missing altogether, and typically fails to describe the image in sufficient detail. To address this, rather than using the native post text we used the BLIP-2 [96] model to produce a caption for each Instagram image. In addition, we generated a second, more engaging caption by prompting BLIP-2 with: “Write something engaging and interesting for the image”. Engagement scores were derived from the number of likes. Concretely, each post’s likes were normalized by dividing by the mean number of likes across the entire corpus; any post whose likes exceeded the mean was assigned an engagement score of 1. From the full corpus we drew two disjoint subsets of 20,000 samples each, one for training the engagement classifier and the other for training the continuous prompt used by stable diffusion.

The evaluation inputs were built around five themes — crowds, vehicles, nature, architecture, and notable individuals — with 25 sentences per theme. These initial textual prompts were produced by GPT-3.5 with no length constraint. They serve as the common starting point fed into all baselines and into our framework.

3.4.2. Baselines

We compare **GEM** against three standard baselines: 1) stable diffusion fed with the original sentences (denoted **D**); 2) stable diffusion fed with paraphrased sentences (denoted **D** + **P**); and 3) stable diffusion fed with both continuous prompts and the embeddings of paraphrased sentences (denoted **D** + **P** + **CP**). As noted in Section 3.4.1, two flavors of continuous prompts were produced depending on the training captions used: **CP (Caption)**, which is trained directly on BLIP-2 captions, and **CP (Engaging)**, which is trained on the engagement-oriented captions obtained with the specialized BLIP-2 prompt. Correspondingly, our method also comes in two variants, **GEM(C)** and **GEM(E)**, depending on which continuous-prompt variant it uses.

3.4.3. Implementation Details

Image-text features are extracted using a pre-trained VILT model [88]. Its 768-dimensional output is passed through a fully connected layer that projects it down to 120 dimensions, followed by a ReLU non-linearity and a second fully connected layer that emits the final engagement prediction. Training proceeded for 10 epochs using stochastic gradient descent with momentum 0.9, a learning rate of 1e-3, and a batch size of 64.

Because engagement prediction turned out to be relatively insensitive to the learning rate, the final value was chosen by a grid search over $\{1e-2, 1e-3, 1e-4\}$. The image generator is the pre-trained stable diffusion v1.5¹, and the learnable continuous prompt is parameterized with shape $(prefix_{length}, 768)$, where $prefix_{length} = 2$ in our experiments. Its training uses the Adam optimizer with a learning rate of 1e-5, gradient accumulation

¹<https://huggingface.co/runwayml/stable-diffusion-v1-5>



Figure 3.2. Multimodal (image-text) pairs generated using various techniques. Columns are arranged from left to right to show pairs produced by different methods. Each row uses the text in the first column as the initial input.

over 4 steps, and a batch size of 16; all other hyperparameters follow the HuggingFace defaults. During the iterative post-generation loop, text paraphrasing is performed by GPT-3.5 with the instruction “Please rewrite the following sentences and make them more engaging and interesting.” In most cases, a satisfactory image-text pair emerges in fewer than 10 iterations. All experiments were run on three RTX A6000 GPUs.

Method	Similarity Score
Diffusion	25.24
Diffusion + P	24.8
Diffusion + P + CP (Caption)	23.73
Diffusion + P + CP (Engaging)	24.31
GEM (Caption)	26.36
GEM (Engaging)	25.85

Table 3.1. Average similarity scores.

3.4.4. Similarity Score Analysis

To quantify how well the generated image aligns with its paired text, we computed CLIP similarity scores — using the pre-trained CLIP model — on each of the 125 image-text pairs, and then averaged the per-pair scores for each method. Table 3.1 reveals that adding paraphrased text, caption-based continuous prompts, and engagement-based continuous prompts individually reduces the similarity score (by 0.44, 1.07, and 0.49 respectively).

Our iterative generation procedure counteracts this degradation and in fact pushes the similarity score higher: it exceeds **Diffusion + CP + P (C/E)** by 2.63 and 1.54 points, respectively. Furthermore, the **GEM (C)** variant attains the best overall similarity of 26.36, which suggests that iterative refinement is an effective strategy for keeping image and text closely aligned. Taken together, the numbers tell a consistent story: while paraphrasing and continuous prompts in isolation erode image-text similarity, wrapping them in our iterative loop not only undoes the loss but yields the most coherent pairs of all tested configurations.

3.4.5. Human Evaluation

To validate our method against the baselines from a perceptual standpoint, we ran a human-evaluation study on Amazon Mechanical Turk (MTurk). Workers received \$0.03 per task; given an observed average completion time of 7 seconds, this corresponds to an effective rate of roughly \$15 per hour. To protect data quality, participation was restricted to workers with an approval rate of at least 99% and a history of 10,000 or more completed tasks.

Each study item was presented as a pairwise comparison. Given two methods that started from the same initial text, the worker was shown both generated image-text pairs and asked to select the one they found more engaging. Every pair of methods was compared across all 125 test instances defined in Section 3.4.1. Workers were not told which method produced which output, so as to avoid any bias from knowledge of the underlying systems, and the left/right display order was randomized for the same reason.

The outcomes, summarized in Table 3.2, indicate that **GEM** is effective at producing coherent and engaging image-text pairs. In particular, **GEM(E)** was preferred to **D**, **D + P + CP (E)**, and **GEM(C)** by margins of 13%, 3%, and 8%, respectively. Comparing **D** with **D + P** shows that paraphrasing alone already boosts engagement appreciably. On top of that, **D + P + CP (E)** was preferred over **D + P** by 2%, suggesting that engagement-oriented continuous prompts contribute a further gain. Notably, although our iterative procedure is not specifically optimized for engagement, it still increases it: **GEM(C/E)** is preferred over **D + P + CP (C/E)** by 8% and 3%, respectively. This pattern hints that improving image-text coherence indirectly improves perceived engagement as well.

Methods	More Engaging
Diffusion <i>v.</i> D+P	D+P (55%)
Diffusion <i>v.</i> D+P+CP(C)	Diffusion (54%)
Diffusion <i>v.</i> D+P+CP(E)	D+P+CP(E) (60%)
Diffusion <i>v.</i> GEM(C)	GEM(C) (60%)
Diffusion <i>v.</i> GEM(E)	GEM(E) (63%)
D+P <i>v.</i> D+P+CP(C)	D+P+CP(C) (52%)
D+P <i>v.</i> D+P+CP(E)	D+P+CP(E) (52%)
D+P+CP(C) <i>v.</i> GEM(C)	GEM(C) (58%)
D+P+CP(E) <i>v.</i> GEM(E)	GEM(E) (53%)
GEM(C) <i>v.</i> GEM(E)	GEM(E) (58%)

Table 3.2. Results of the human evaluation. The selected percentage of more engaging methods is illustrated in parentheses.

3.4.6. Case Studies

In this subsection, we examine qualitatively how the various methods compare at producing compelling and engaging image-text posts. Figure 3.2 arranges the generated posts from left to right in the following order: **Diffusion**, **D + P**, **D + CP + P (C/E)**, and **GEM(C/E)**. The first-column text of each row is the original input that was provided to **D**.

Across the examples, **GEM**’s generated text consistently conveys richer detail and stronger appeal. The first row, produced by **GEM**, features a natural landscape that would be well-suited to a tourism-marketing campaign. In the second row, **GEM(C)** yields a vivid, colorful portrait while **GEM(E)** produces an arguably more interesting variant. The next example from **GEM** likewise showcases a natural scene with clear promotional potential. We also see value for architecture-tour advertising in the third-row outputs — particularly interesting is the **GEM(C)** image, which depicts the building’s interior with a swimming pool, adding an unexpected and engaging visual hook.

Going further, we illustrate the versatility of the pipeline by using GEM to author four image-text posts intended for Twitter promotions across four industries: hospitality, culinary, gaming, and aviation. We first ask GPT-3.5 to draft the advertising copy, including “within 280 words” in the prompt so that the output respects Twitter’s character limit. GEM(E) then takes this text and produces tweets in which the image and text work together seamlessly.

3.5. Conclusion

This work introduced GEM, a framework for generating engaging and coherently aligned image-text posts tailored for social media. GEM combines continuous prompt learning by engagement-guidance with an iterative algorithm. Experimental results demonstrate its effectiveness in enhancing content alignment and boosting user engagement. These findings offer practical value for organizations aiming to improve social media presence and contribute to advances in multimodal content generation. Future work may explore incorporating more diverse inputs and refining generation methods to foster responsible and impactful content creation.

This work is published in Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence. Gao, Chongyang, et al. "GEM: generating engaging multimodal content." *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*. 2024.

CHAPTER 4

Efficient Finetuning Large Language Models

4.1. Introduction

Large Language Models (LLMs) have demonstrated strong generalization and transfer learning capabilities across a broad range of tasks and domains [34, 225, 7, 77, 167]. However, fine-tuning these models remains computationally expensive due to their immense parameter count. To address this, parameter-efficient fine-tuning (PEFT) techniques—such as p-tuning [113] and Low-Rank Adaptation (LoRA) [68]—have been proposed to reduce training costs. Despite their efficiency, PEFT methods often struggle to match the performance of full fine-tuning approaches.

To improve PEFT effectiveness, recent work combines LoRA with Mixture of Experts (MoE) [217, 107, 48]. A prominent direction involves augmenting LoRA with MoE, referred to as LoRA-MoE. In Transformer models [184], LoRA introduces a pair of trainable low-rank matrices into dense linear layers, enabling adaptation without modifying the original weights. LoRA-MoE extends this by training multiple low-rank matrix pairs—termed LoRA experts—alongside a routing mechanism that assigns expert weights to each input. During fine-tuning, only the LoRA experts and router are updated while the pretrained backbone remains fixed. Although this approach shows promise, it remains an emerging area with open challenges in maximizing efficiency and expressiveness.

Furthermore, recent analyses of MoE systems suggest that excessive experts can lead to representational redundancy or overfitting due to routing inefficiencies [29, 231]. Motivated by this, we ask:

- Do parameter-efficient MoE setups include redundant experts?
- How should LoRA experts be distributed across Transformer layers to optimize performance?

We propose MoLA (MoE-LoRA with Layer-wise Expert Allocation), a new PEFT framework that assigns a variable number of experts to each Transformer layer. Using the same total number of experts, we test configurations emphasizing lower, middle, or upper layers across six benchmarks. Our key contributions and findings are:

- **MoLA Framework:** We introduce MoLA, a plug-and-play PEFT framework that combines LoRA with MoE, incorporating flexible layer-wise expert allocation. By allowing non-uniform expert distribution, MoLA reduces redundancy and enhances layer-specific adaptation. It can be readily applied across models and tasks.
- **Empirical Evaluation:** We implement and evaluate several MoLA variants across LLaMA-2, Mistral, and Gemma, using different expert allocation patterns. All MoLA variants outperform existing PEFT baselines, confirming the effectiveness of our approach.
- **Optimal Allocation Strategy:** Our experiments show that allocating more experts to middle layers while reducing them in lower layers leads to the best performance, even under fixed parameter budgets. This configuration balances expressiveness and regularization, enhancing model scalability.

- **Redundancy Analysis:** A detailed similarity analysis reveals that experts in lower layers exhibit higher redundancy, supporting our hypothesis and explaining the performance gains achieved by deprioritizing those layers in expert allocation.

4.2. Related Work

Recent work has explored combining Mixture of Experts (MoE) with parameter-efficient tuning to enhance instruction tuning. [107] introduced a LoRA-MoE method for medical tasks that conditions the router on task type, requiring prior knowledge at inference. In contrast, MoLA learns experts in an unsupervised manner, eliminating this dependency. [48] proposed a LoRA-MoE architecture that integrates experts into the feed-forward layers of Transformer blocks and investigates knowledge forgetting during fine-tuning. However, their method is limited to feed-forward layers, whereas MoLA applies LoRA experts to all dense layers, boosting both performance and scalability.

[217] presented a broader framework combining MoE with multiple PEFT methods (*e.g.*, LoRA, IA3 [106]), demonstrating improvements in instruction tuning and zero-shot performance. [19] explored dynamically mixing pre-trained LoRA adapters. Yet, these approaches do not address expert allocation across layers. MoLA fills this gap by introducing flexible, layer-wise expert allocation, enabling more efficient and effective use of LoRA-MoE structures.

4.3. Method

4.3.1. Preliminaries

Mixture of Experts. The Mixture of Experts (MoE) architecture [165] uses a router to activate a sparse subset of expert networks, enabling efficient scaling of model parameters. Each MoE layer contain N identical and independent feed-forward neural networks $\{E\}_{i=1}^N$ with a trainable router that assigns inputs x to experts. The router is a gating function with a trainable weight matrix W_r . The router uses a softmax function to compute a probability distribution of the weights of outputs from the expert networks. Following standard MoE architectures, only the top K experts, determined by the router, are chosen for the computation. Equation 4.1 mathematically represents the MoE layer where y is the output embedding from the MoE layer.

$$(4.1) \quad y = \sum_{i=1}^K \frac{\text{TopK}(\text{Softmax}(W_r x), K)_i}{\sum_{i=1}^K \text{TopK}(\text{Softmax}(W_r x), K)_i} E_i(x)$$

LoRA. LoRA is a widely adopted parameter-efficient fine-tuning method for LLMs [68, 224, 44]. It reduces the number of trainable parameters by reparameterizing weight updates as a low-rank decomposition. For a frozen pre-trained weight matrix $W_0 \in \mathbb{R}^{d_q \times d_p}$, LoRA creates two low-rank trainable matrices A and B , where $A \in \mathbb{R}^{d_q \times r}$, $B \in \mathbb{R}^{r \times d_p}$, and $r \ll \min(d_q, d_p)$. Equation 4.2 mathematically describes this process, and the output of LoRA is h . During training, W_0 is frozen, while A and B are updated.

$$(4.2) \quad h = W_0 x + \Delta W x = W_0 x + ABx$$

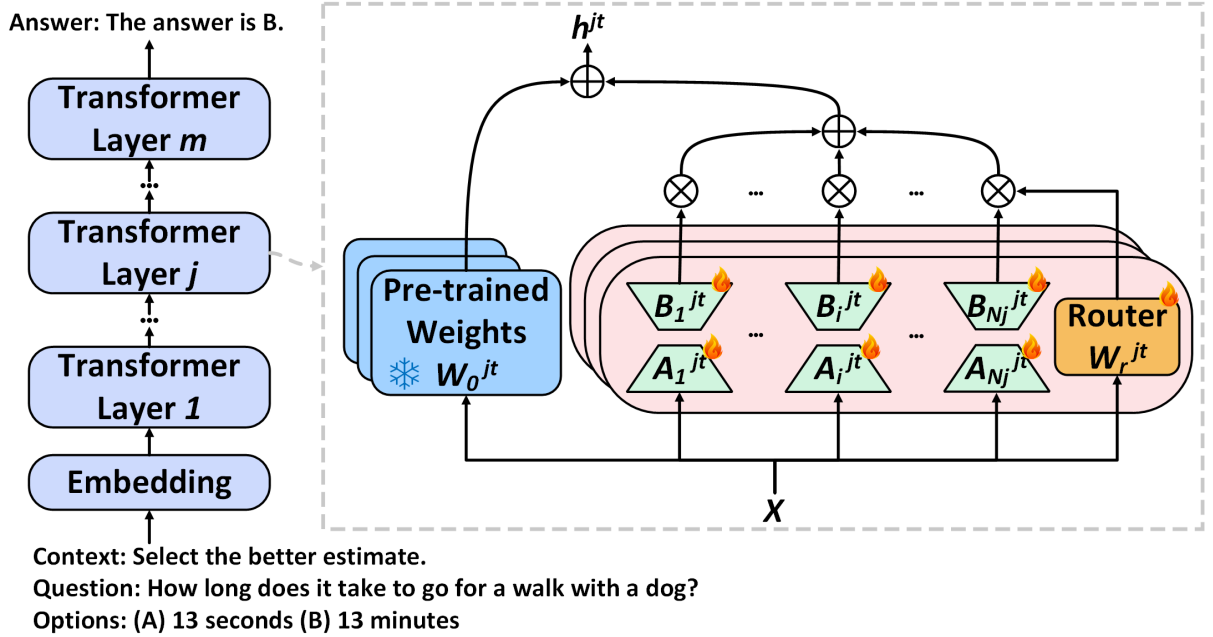


Figure 4.1. The overview of MoLA architecture.

4.3.2. MoE-LoRA with Layer-wise Allocation

Combining MoE and LoRA has shown promise for efficient fine-tuning [217, 107, 48]. However, existing approaches typically replace experts with LoRA adapters uniformly across layers, which may retain known MoE issues such as expert redundancy from representational collapse or overfitting in the routing policy [29, 231]. This motivates our key insight: the number of LoRA experts need not be uniform across all layers.

We propose MoLA (MoE-LoRA with Layer-wise Allocation), a novel parameter-efficient tuning method that integrates MoE and LoRA with flexible expert allocation per layer. Focusing on Transformer-based LLMs, MoLA assigns different numbers of LoRA experts to different layers. In this section, we describe the MoLA architecture and explore several allocation strategies based on varying design assumptions.

4.3.2.1. The MoLA Architecture. MoLA extends the MoE framework by integrating multiple LoRA adapters, allowing each Transformer layer to have a different number of experts. During fine-tuning, instead of adding a single low-rank pair to each dense layer as in standard LoRA, MoLA introduces *multiple* pairs—referred to as LoRA experts. A trainable router dynamically selects which experts to apply to each input token, enabling flexible and efficient adaptation across layers. Given a Transformer model with m layers, we allocate N_j experts for layer j and have $\sum_{j=1}^m N_j$ experts in total. Specifically, given a pre-trained weight matrix $W_0^{jt} \in \mathbb{R}^{d_q \times d_p}$ from the module t in layer j , we create N_j pairs of low-rank matrices $\{A_i^{jt}\}_{i=1}^{N_j}, \{B_i^{jt}\}_{i=1}^{N_j}$. As in the case of LoRA, each matrix A_i^{jt} is initialized from a random Gaussian distribution. We set B_i^{jt} to zero, where $A_i^{jt} \in \mathbb{R}^{d_q \times r}$, $B_i^{jt} \in \mathbb{R}^{r \times d_p}$, and $r \ll \min(d_q, d_p)$. Then, a router S_i^{jt} with a trainable weight matrix $W_r^{jt} \in \mathbb{R}^{d_q \times N_j}$ is used to specify different LoRA experts for the input x . As in the original MoE, MoLA selects the top K experts for computation and applies the load balancing loss on each layer. Figure 4.1 shows an overview of the architecture. The mathematical representation is:

$$(4.3) \quad S_i^{jt}(x) = \frac{\text{TopK}(\text{Softmax}(W_r^{jt}x), K)_i}{\sum_{i=1}^K \text{TopK}(\text{Softmax}(W_r^{jt}x), K)_i}$$

$$(4.4) \quad h^{jt} = W_0^{jt}x + \sum_{i=1}^K S_i^{jt}(x)A_i^{jt}B_i^{jt}x$$

Eq. 4.3 represents the router with the input x , and Eq. 4.4 mathematically shows the LoRA experts in MoLA, where h^{jt} is the output embedding. This MoLA architecture provides flexibility in modifying the number of experts for each Transformer layer.

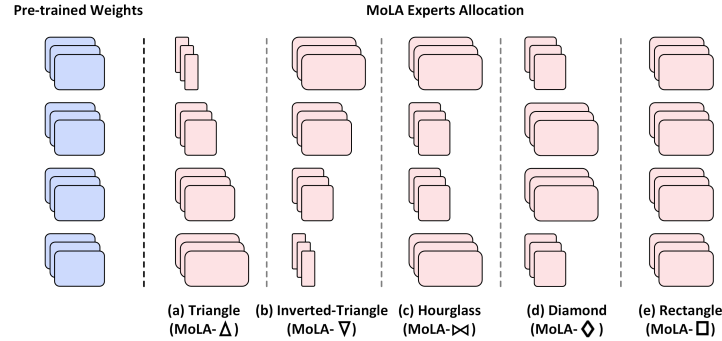


Figure 4.2. Five types of layer-wise expert allocations of MoLA. A longer rectangle indicates a greater number of LoRA experts.

4.3.2.2. Configurations of Layer-wise Expert Allocation. MoE functions similarly to an ensemble, where multiple experts capture fine-grained information. While layers with more experts have greater capacity, they also increase architectural complexity. Intuitively, layers responsible for handling diverse or nuanced inputs may benefit from more experts. To explore how LoRA experts should be distributed across Transformer layers, we propose five layer-wise allocation strategies based on different design assumptions. These configurations are visualized in red in Figure 4.2.

MoLA Triangle (MoLA- Δ). Prior studies show that Transformer layers capture different types of representations: lower layers focus on token-level features like syntax, while higher layers encode more abstract, high-level information. Since token-level details are more subtle, it is reasonable to allocate more experts to lower layers for fine-grained processing, and fewer to higher layers for generalization. Based on this intuition, we propose MoLA-Triangle (MoLA- Δ), which allocates experts in a triangle-shaped pattern—more in the lower layers and fewer in the higher ones.

MoLA Inverted-Triangle (MoLA- ∇). In contrast to MoLA- $\triangle$, another perspective suggests that allocating more experts to token-level features may lead to redundancy. Since higher layers capture abstract, task-relevant information, they may benefit more from increased expert capacity. More experts at these layers can better model complex, fine-grained patterns. Following this intuition, we introduce the MoLA Inverted-Triangle (MoLA- ∇) configuration, which assigns fewer experts to lower layers and more to higher layers.

MoLA Hourglass (MoLA- $\bowtie$). A third assumption posits that both lower and higher layers benefit from more experts: lower layers for capturing basic features, and higher layers for modeling abstract semantics. In contrast, middle layers primarily aggregate and transform information, requiring fewer experts. Based on this, we propose MoLA Hourglass (MoLA- $\bowtie$), which follows an "hourglass" allocation—assigning more experts to the lower and upper layers, and fewer to the middle ones.

MoLA Diamond (MoLA- $\diamond$). In contrast to MoLA- $\bowtie$, findings in representation learning suggest that both low-level feature extraction and high-level abstraction are more sensitive to perturbations than representation learning in middle layers [116, 12]. Moreover, strong intermediate representations are critical for tasks requiring abstract reasoning. Based on this insight, we propose MoLA Diamond (MoLA- $\diamond$), which allocates more experts to the middle layers in a "diamond"-shaped configuration.

MoLA Rectangle (MoLA- $\square$). The final configuration follows the standard MoE design, where each Transformer layer is assigned the same number of experts. We refer to this setup as MoLA Rectangle (MoLA- $\square$).

4.4. Experiments

4.4.1. Experimental Settings

We evaluate MoLA by directly fine-tuning it on a range of downstream tasks. To make the comparisons concrete and easy to interpret, we instantiate five different layer-wise allocation patterns of MoLA on top of the same large language model, as introduced in Section 4.3. The base models we adopt are LLaMA-2-7B [181] and Mistral-7B [78], both of which have 32 transformer layers. The MoLA- $\triangle$ configuration places **8** experts in each of the first 8 layers, **6** in each of layers 9–16, **4** in each of layers 17–24, and **2** in each of the final 8 layers, which we abbreviate as **8642**. Using this same shorthand, we denote the inverted-triangle MoLA configuration as **2468**. The Hourglass, MoLA Diamond, and MoLA Rectangle variants correspond to **8228**, **2882**, and **5555**, respectively. To keep the comparison fair, the total expert count is identical across all five variants, so they all have the same number of trainable parameters. For LLaMA-2 this amounts to 105,635,840 trainable parameters, equivalent to 1.5% of the parameters in the underlying pre-trained model. To balance the top- k routing decisions, we additionally include the auxiliary load-balancing loss introduced in Switch Transformers [53].

4.4.2. Task and Data

MoLA is designed for both downstream-task fine-tuning and instruction tuning of LLMs. To exercise both regimes, we evaluate it on natural language processing (NLP) tasks and on commonsense reasoning (CR) tasks. The NLP suite consists of three widely used datasets: Microsoft’s Research Paraphrase Corpus [47], the Recognizing Textual Entailment (RTE) dataset [186], and the Corpus of Linguistic Acceptability (COLA) [186].

The CR suite consists of three modern question-answering benchmarks: ScienceQA [120], CommonsenseQA [174], and OpenbookQA [126]. All evaluations follow the standard task-specific fine-tuning protocol. The individual datasets are described below.

The Microsoft Research Paraphrase Corpus (MRPC) is built from 5,801 sentence pairs taken from newswire articles, where each pair carries a binary label indicating whether the two sentences are paraphrases of one another. The dataset is split into 4,076 training pairs and 1,725 test pairs.

The Recognizing Textual Entailment (RTE) dataset aggregates examples from a series of annual textual entailment challenges (RTE1, RTE2, RTE3, and RTE5), with sentence pairs drawn from news and Wikipedia text. It is set up as a binary classification task — entailment versus not entailment — and provides 2,490 training examples and 277 validation examples.

The Corpus of Linguistic Acceptability (COLA) collects English acceptability judgments compiled from books and journal articles in linguistics. Each item is a token sequence labeled with whether it constitutes a grammatical English sentence. COLA contains 8,551 training items and 1,043 validation items, and the task is grammaticality classification.

ScienceQA is a commonsense question-answering dataset assembled from elementary- and high-school-level science curricula, comprising 21,208 multimodal multiple-choice questions. Since our work is restricted to text-only inputs, we extracted the text-only subset and partitioned it into 6,508 training samples and 2,224 test samples. ScienceQA spans three subject areas — natural science, language science, and social science — and answering its questions requires the model to draw on the appropriate commonsense knowledge.

CommonsenseQA is a question-answering dataset that probes a variety of types of commonsense knowledge. The questions were authored by Amazon Mechanical Turk workers, and the dataset offers 9,740 training and 1,221 validation samples.

OpenbookQA is a commonsense QA benchmark designed to test understanding of a domain. It supplies 5,957 multiple-choice elementary-level science questions split into 4,957 training, 500 validation, and 500 test items. To answer correctly, a model must draw on a small “book” of 1,326 core science facts and apply them to new situations.

In addition to these benchmarks, we use an instruction-tuning corpus during training. Specifically, we draw 50,000 random samples from OpenOrca [100], an instruction-tuning corpus that augments FLAN data so as to match the distributions reported in ORCA [128]; OpenOrca contains 2.91M instances spanning a wide range of tasks and instructions.

Table 4.1. Comparison with different methods using LLaMA-2. MoLA- ∇ outperforms other variants or baselines and even achieves competitive or superior performance with MoLA- $\square$ (8888), with nearly 40% fewer parameters.

Models (# of Experts)	MRPC	COLA	RTE	ScienceQA	CommonsenseQA	OpenbookQA	Trainable Parameters
Full-Parameter	87.13%	86.29%	87.73%	93.12%	77.48%	80.4%	6,738,415,616
Prompt Tuning	49.91%	59.25%	54.17%	36.78%	37.76%	46.2%	163,840
LLaMA-Adapter	71.94%	47.56%	72.93%	73.33%	73.55%	71.8%	5,242,912
LoRA	83.13%	86.29%	85.92%	91.01%	75.51%	77.0%	159,907,840
MoLA- $\square$ (8888)	84.70%	85.81%	88.45%	91.91%	77.89%	82.8%	169,017,344
MoLA- $\square$ (5555)	84.23%	86.28%	85.20%	92.04%	78.13%	80.0%	105,635,840
MoLA- $\triangle$ (8642)	84.64%	85.43%	84.84%	91.90%	77.23%	77.6%	105,635,840
MoLA- $\bowtie$ (8228)	83.48%	86.00%	86.28%	91.41%	76.25%	78.8%	105,635,840
MoLA- ∇ (2468)	83.48%	86.87%	86.28%	92.36%	78.95%	79.6%	105,635,840
MoLA- $\diamond$ (2882)	83.01%	86.19%	89.17%	92.00%	77.81%	82.6%	105,635,840
MoLA- $\square$ (4444)	82.90%	85.62%	86.28%	91.73%	77.40%	80.8%	84,508,672
MoLA- $\triangle$ (6532)	83.54%	85.43%	85.20%	92.00%	77.64%	80.2%	84,508,672
MoLA- $\bowtie$ (6226)	83.42%	85.71%	84.48%	92.00%	76.58%	80.0%	84,508,672
MoLA- ∇ (2356)	81.97%	85.91%	87.37%	92.18%	77.97%	81.2%	84,508,672
MoLA- $\diamond$ (2662)	82.96%	85.62%	87.37%	92.27%	77.48%	79.8%	84,508,672

4.4.3. Recent Competitive Baselines

We benchmark MoLA against three parameter-efficient tuning methods — prompt tuning [94], LLaMA-Adapter [225], and LoRA [68] — as well as full-parameter fine-tuning. Prompt tuning prepends a sequence of soft prompts (virtual tokens) to the embedding layer of the transformer; during fine-tuning, the LLM itself is kept frozen and only the virtual tokens are updated, yielding a particularly lightweight adaptation procedure. LLaMA-Adapter is an instruction-tuning approach for LLaMA in which a set of learnable adaptation prompts is prepended to the word tokens at the upper transformer layers, and a zero-initialized attention mechanism with zero gating is employed to inject new instructional signals into the model. LoRA was already introduced briefly in Section 4.3.2, and we use it with rank 64 in this work. For each evaluation, the LLM is fine-tuned on the corresponding downstream training set using the appropriate parameter-efficient strategy. Where ground-truth test labels are unavailable, we report numbers on the validation split (this is the case for COLA, RTE, and CommonsenseQA); the remaining benchmarks are reported on their test split.

4.4.4. Implementation

LLaMA-2-7B [181] and Mistral-7B [78] serve as the base language models throughout. The number of fine-tuning epochs is selected via grid search over $\{10, 15, 20\}$. Optimization uses AdamW [117] with a learning rate of $3e-4$. Following Sanh *et al.* [159], the cutoff length is fixed at 256, and the batch size is 128. We fix the random seed to 10. Each LoRA expert has rank 8, and the router selects the top 2 experts per input. We follow the LoRA defaults for the remaining adapter hyperparameters: LoRA α is 16 and the LoRA

dropout rate is 0.05. The LoRA modules are inserted into four self-attention projection matrices (W_q, W_k, W_v, W_o) and three MLP matrices ($W_{gate}, W_{down}, W_{up}$). Hardware-wise, the experiments were executed on servers equipped with eight A100-40G GPUs and three A6000 GPUs, with a single training run on COLA taking approximately 4 hours.

4.4.5. Results

Comparison with Baselines. The direct fine-tuning results on LLaMA-2 are reported in Table 4.1, where each entry is the per-dataset accuracy in percent. A first observation is that LoRA-based approaches (LoRA itself and the MoLA family) substantially outperform the prompt-tuning-based baselines (Prompt Tuning and LLaMA-Adapter). Within the LoRA family, we use the original rank-64 LoRA as the reference baseline. We begin by examining MoLA- $\square$ with eight experts in every layer (i.e., MoLA- $\square(8888)$), whose parameter budget is essentially the same as LoRA’s. We then reduce the total expert count from 32 (8×4) to 20 — about 62.5% of the original budget — and evaluate the five MoLA configurations described in Section 4.4.1. Across all benchmarks, the MoLA variants improve over the LoRA baseline. MoLA- ∇ in particular outperforms LoRA on every one of the six datasets, with the gains being more pronounced on the commonsense QA tasks than on the NLP tasks. Strikingly, MoLA- ∇ also surpasses MoLA- $\square(8888)$ on three benchmarks while using almost 40% fewer parameters. These observations together highlight both the effectiveness and the parameter efficiency of MoLA.

Table 4.1 also shows that MoLA- $\triangle$ and MoLA- $\bowtie$ underperform MoLA- $\square$ and MoLA- ∇ , with the gap being most visible on the QA tasks. Among all the MoLA variants, MoLA- ∇ is the strongest overall, achieving the best score on five of the six benchmarks. We

additionally repeated the LLaMA-2 experiments with three random seeds (0, 1, and 2) to compute confidence intervals at the 95% level (using $z = 1.96$); the resulting intervals are listed in Table 4.2. We then ran a Friedman test to reject the null hypothesis that the four algorithms (Prompt Tuning, LLaMA-Adapter, LoRA, and MoLA- ∇) are equivalent, followed by Wilcoxon signed-rank tests for pairwise comparisons across the six benchmarks. To control for multiple comparisons, we applied an FDR correction to the resulting p-values. After correction, every adjusted p-value is below the conventional 0.05 threshold, providing statistical confirmation that MoLA- ∇ ’s advantage over the baselines is significant.

Table 4.2. Confidence intervals of different configurations of MoLA using LLaMA-2.

Models (# of Experts)	MRPC	COLA	RTE	ScienceQA	CommonsenseQA	OpenbookQA
LoRA	[82.37%, 83.12%]	[85.66%, 86.28%]	[85.21%, 86.15%]	[90.16%, 91.01%]	[75.45%, 76.66%]	[77.01%, 78.06%]
MoLA- $\square$ (8888)	[83.49%, 85.01%]	[85.43%, 86.58%]	[86.75%, 88.46%]	[91.90%, 92.94%]	[76.76%, 79.07%]	[79.88%, 82.79%]
MoLA- $\square$ (5555)	[84.00%, 84.35%]	[85.15%, 86.65%]	[85.13%, 87.19%]	[91.54%, 92.51%]	[76.02%, 78.11%]	[79.95%, 80.85%]
MoLA- $\triangle$ (8642)	[82.68%, 84.78%]	[85.39%, 85.72%]	[84.86%, 87.22%]	[91.41%, 92.16%]	[76.32%, 77.38%]	[77.45%, 79.49%]
MoLA- $\bowtie$ (8228)	[82.83%, 84.28%]	[85.50%, 85.99%]	[86.30%, 88.66%]	[91.03%, 92.30%]	[75.65%, 78.11%]	[78.81%, 80.79%]
MoLA- ∇ (2468)	[83.06%, 83.74%]	[85.19%, 87.07%]	[86.28%, 86.76%]	[91.52%, 92.39%]	[76.88%, 79.27%]	[79.55%, 82.32%]
MoLA- $\diamond$ (2882)	[82.93%, 83.99%]	[86.08%, 86.30%]	[87.63%, 89.26%]	[91.61%, 92.02%]	[76.50%, 78.40%]	[79.63%, 82.64%]

4.4.6. Model Analysis and Ablation Studies

4.4.6.1. Analysis of Layer-wise Experts Importance. To better understand which configuration works best and why, we examine layer-wise expert importance by repeating the comparison on a different base model, Mistral, in Table 4.3. The two tables (Table 4.1 and Table 4.3) also include results at two different total expert budgets, 20 and 16.

Across both base models and both budgets, MoLA- ∇ and MoLA- $\diamond$ consistently outperform the other configurations. *Concretely, this means that placing more experts in the middle layers of the network is more effective than the alternative allocation strategies.* One caveat is worth noting: when the total expert budget shrinks to 16, the overall capacity of

MoLA appears to become a limiting factor, and the dominance of MoLA- ∇ and MoLA- $\diamond$ becomes somewhat less stable.

Table 4.3. Comparison with different configurations of MoLA using Mistral.

Models (# of Experts)	MRPC	COLA	RTE	ScienceQA	CommonsenseQA	OpenbookQA
LoRA	86.02%	87.05%	89.53%	94.20%	81.40%	88.8%
MoLA- $\square$ (5555)	86.55%	87.34%	90.91%	94.74%	83.21%	88.8%
MoLA- $\triangle$ (8642)	86.15%	87.63%	90.61%	95.05%	82.80%	89.6%
MoLA- $\bowtie$ (8228)	86.43%	87.63%	90.61%	95.01%	81.90%	90.0%
MoLA- ∇ (2468)	86.15%	87.34%	91.70%	95.28%	83.21%	90.0%
MoLA- $\diamond$ (2882)	86.67%	88.78%	90.98%	94.78%	83.62%	90.6%
MoLA- $\square$ (4444)	86.02%	87.92%	90.61%	94.69%	83.37%	89.2%
MoLA- $\triangle$ (6532)	86.20%	87.25%	88.81%	94.74%	82.06%	89.4%
MoLA- $\bowtie$ (6226)	86.84%	87.92%	89.90%	94.60%	82.88%	88.8%
MoLA- ∇ (2356)	86.44%	87.15%	90.25%	95.32%	82.47%	89.8%
MoLA- $\diamond$ (2662)	86.67%	88.02%	89.53%	94.43%	82.64%	88.2%

Table 4.4. Comparison with different extreme configurations of MoLA. The top block shows the results using LLaMA-2 and the bottom block shows the results using Mistral.

LLaMA-2	MRPC	COLA	RTE	S-QA	C-QA	O-QA	AVG.
MoLA (10 2 2 2)	83.94%	84.47%	84.48%	92.36%	75.76%	77.2%	83.03%
MoLA (2 10 2 2)	82.90%	86.48%	87.73%	91.46%	77.97%	80.8%	84.56%
MoLA (2 2 10 2)	82.67%	85.71%	85.20%	91.28%	76.74%	81.0%	83.77%
MoLA (2 2 2 10)	82.96%	85.81%	85.92%	92.18%	76.90%	81.60%	84.23%
Mistral	MRPC	COLA	RTE	S-QA	C-QA	O-QA	AVG.
MoLA (10 2 2 2)	86.49%	86.86%	91.33%	95.00%	81.49%	88.2%	88.50%
MoLA (2 10 2 2)	86.15%	87.25%	90.61%	94.51%	81.41%	88.6%	88.26%
MoLA (2 2 10 2)	86.73%	88.21%	90.98%	95.23%	82.97%	89.6%	89.10%
MoLA (2 2 2 10)	86.67%	87.92%	91.34%	95.01%	82.88%	89.2%	89.02%

4.4.6.2. Analysis of Layer-wise Experts Redundancy. The analysis above showed that concentrating LoRA experts in the middle layers is most beneficial. We additionally observed that MoLA- ∇ is the preferred choice for LLaMA-2 and Gemma, whereas MoLA- $\diamond$ is preferable for Mistral. To explore the source of this model-dependent preference, we

evaluated four extreme configurations — (10 2 2 2), (2 10 2 2), (2 2 10 2), and (2 2 2 10) — in which almost all experts are concentrated within a single quarter of the network. Table 4.4 shows that, regardless of the base model, the lowest layers carry more redundancy and yield lower average task performance: layers 1–4 for LLaMA and layers 1–8 for Mistral. The exact location of the redundant region differs between the two models, however, which is precisely why the optimal allocation strategy — MoLA- ∇ versus MoLA- $\diamond$ — also differs.

To analyze this phenomenon more rigorously, we formalize the notion of layer-wise expert redundancy: **Expert Redundancy** is a layer-wise quantity that captures how distinct the expert modules within a MoE-style transformer layer are from one another. When two of the experts selected by the router are similar, they are likely to overlap in function, introducing redundancy. To quantify this for transformer layer j , we compute the average pairwise Frobenius norm between the weight matrices of any two distinct LoRA experts in the self-attention module of that layer. Figure 4.3 reports the resulting per-layer averages for MoLA- $\square$ (8888) on LLaMA-2 and on Mistral, with both models trained on our sampled instruction-tuning corpus. The Frobenius norm grows monotonically as we go from lower to higher layers, indicating that experts in lower layers tend to resemble each other more than experts in higher layers do. This is direct empirical support for our hypothesis that the lower layers of the transformer are more affected by expert redundancy. Consequently, given a fixed expert budget, allocating a larger share of experts to the higher layers tends to be a more productive use of the budget.

Concretely, the experiments show that the lower layers carry more redundancy. A plausible interpretation is that, since lower layers process token-level information in a relatively fine-grained manner, having many experts there leads to substantial overlap; in

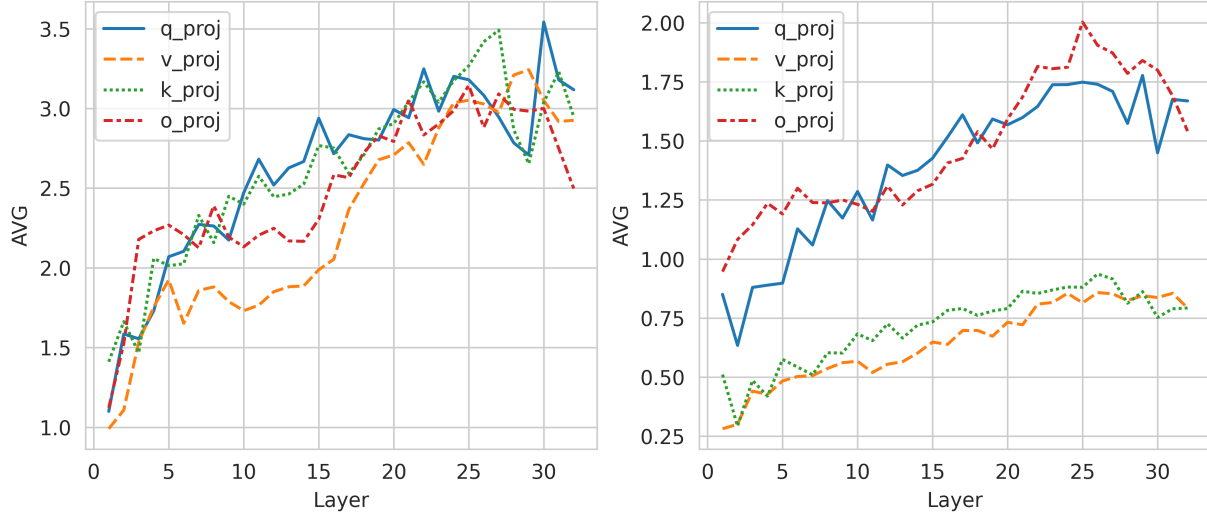


Figure 4.3. Average number of the Frobenius Norm between two different experts’ weight matrices for each self-attention module from each layer. The left and right figures are for the MoLA-□(8888) using LLaMA-2 and Mistral, correspondingly. Both models are trained via instruction tuning.

contrast, the middle and higher layers are responsible for learning more abstract, high-level patterns and benefit from a larger pool of experts that can specialize in fine-grained, task-specific structure. We therefore argue that the expert count in the middle and upper layers is the critical design choice. As a practical consequence, when pruning a MoLA model to reduce the trainable-parameter count, the cuts should be concentrated in the lower layers.

4.4.6.3. Analysis of Average Fusion Weights and Selected Times of Experts. We additionally inspect two quantities produced by the router during inference: the average fusion weight assigned to each expert, and the average number of times each expert is actually selected. The corresponding plots are shown in Figure 4.4 for MoLA-□(8888) on

LLaMA-2. Panel (a) reveals that most fusion weights cluster around 0.5, meaning that, on a typical input, the two selected experts are weighted roughly equally. Panel (b) shows that, although a handful of experts are rarely selected, the great majority are selected often, indicating that the expert pool is effectively utilized. Given that the selection frequencies and the fusion weights are broadly comparable across layers, the higher pairwise similarity of experts in the lower layers — documented in the previous subsection — is what drives the increased redundancy there, reinforcing our earlier conclusion.

4.5. Conclusion

We present MoLA, a novel parameter-efficient tuning method that integrates layer-wise expert allocation from MoE with the LoRA technique. We propose five expert configurations based on different assumptions about layer importance. Experiments on six benchmarks across NLP and commonsense QA tasks show that MoLA consistently outperforms strong baselines. In particular, MoLA- ∇ and MoLA- $\diamond$ yield the best results, validating our hypothesis that middle layers benefit most from expert specialization, while lower layers exhibit redundancy.

This work is published in Findings of the Association for Computational Linguistics: NAACL 2025. *Gao, Chongyang, et al. "MoLA: MoE LoRA with layer-wise expert allocation." Findings of the Association for Computational Linguistics: NAACL 2025. 2025.*

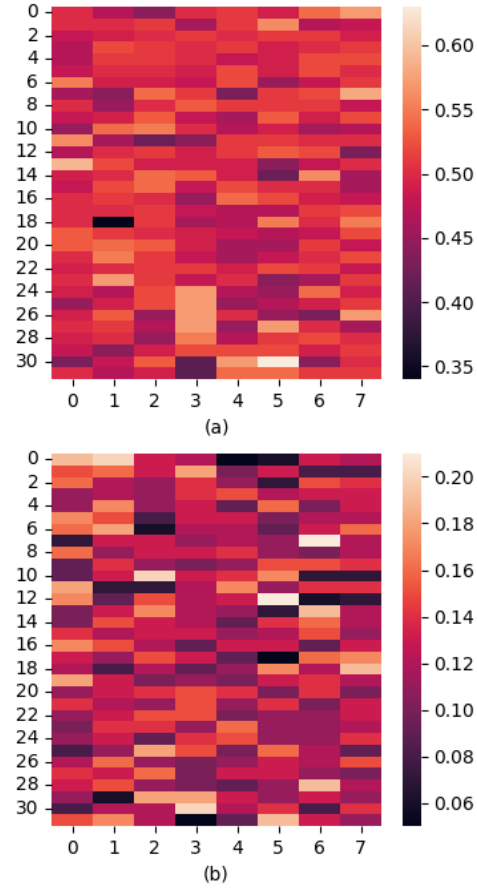


Figure 4.4. (a) The average fusion weights for each expert. (b) The average times for each expert when it is selected.

CHAPTER 5

Unlearning Large Language Models

The increasing size of large language models (LLMs) has also raised concerns about the presence of sensitive, toxic, or legally problematic data in their training corpora [134]. As a result, machine unlearning [16] has emerged as a promising approach to remove the influence of such undesirable data from LLMs.

5.1. Introduction

Current unlearning methods fall into two main categories: parameter optimization[27, 50, 75, 226, 124, 97], which modifies model weights directly through fine-tuning, and in-context unlearning[177, 137], which uses prompting to steer model outputs. While parameter optimization is typically more effective, both approaches face critical limitations in practice, particularly in continual unlearning settings.

Two major challenges persist. First, most methods rely on a large retained dataset, often drawn from the original training data, to preserve model utility. However, full access to these massive corpora is rarely feasible due to legal, privacy, or logistical constraints [108, 171]. Without representative retained data, the model’s performance on non-unlearned content tends to degrade significantly [25, 71]. Second, unlearning requests arise naturally over time, making one-time solutions impractical. Existing methods often fail to maintain model utility under repeated unlearning operations, leading to cumulative degradation of performance [63, 64].

To address these challenges, we propose $\mathcal{O}^3$, a novel framework for continual unlearning without retained data. $\mathcal{O}^3$ combines two key components: (1) an Orthogonal Low-rank Adapter (LoRA) [69] that isolates updates across unlearning tasks to prevent interference, and (2) an Out-of-Distribution (OOD) detector trained with a contrastive entropy loss and a global-local scoring strategy. During inference, $\mathcal{O}^3$ uses OOD similarity scores to selectively apply the relevant unlearning LoRA, thus preserving model utility for unrelated inputs.

Our contributions are the follows: We introduce the problem of retained-data-free continual unlearning for LLMs, aiming to balance unlearning efficacy and utility preservation under repeated unlearning requests. We develop the $\mathcal{O}^3$ framework, which leverages orthogonal LoRA modules and a novel OOD detector to disentangle unlearning effects and dynamically adjust inference behavior. We conduct extensive experiments on generative, discriminative, and reasoning benchmarks, demonstrating that $\mathcal{O}^3$ consistently outperforms strong baselines in maintaining utility while effectively forgetting targeted knowledge.

5.2. Related Work

Machine unlearning [16] has emerged as a key strategy for protecting data privacy and ensuring the proper use of sensitive information [108, 221]. Existing methods typically fall into two categories: parameter optimization [74, 60, 211, 50, 226, 75, 124, 215, 201, 97] and in-context learning [177, 137]. Optimization-based approaches often apply gradient ascent (e.g., GradAsc [60, 211]) on the data to be forgotten. Later methods such as PO [50, 226, 75] recognize that naive optimization harms overall model utility, and instead use label shuffling or rejection sampling to mitigate this. Other works localize the

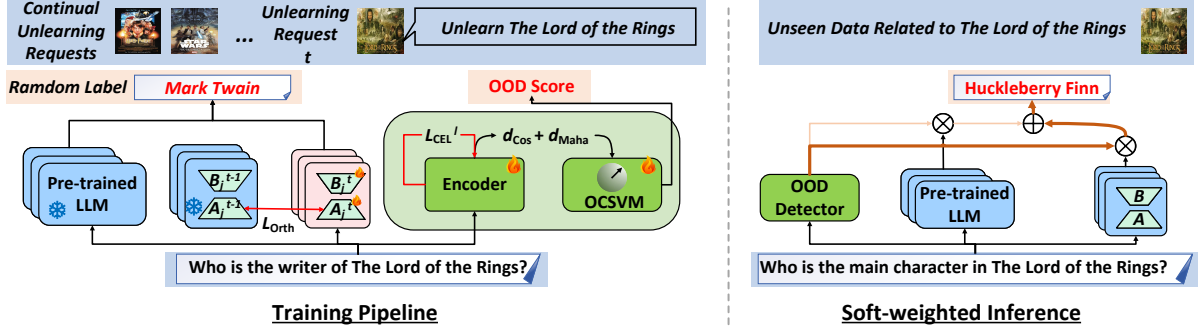


Figure 5.1. The overview of O^3 framework to handle continual unlearning requests for LLM without using any retained data.

model parameters most affected by unlearning data and update them selectively through merging or subtraction techniques [215, 45]. Meanwhile, in-context methods reframe prompts to steer the model away from generating undesired content.

While these methods can be effective for isolated unlearning tasks, they struggle in real-world scenarios where unlearning requests are continuous [108]. Repeated application leads to cumulative degradation in model performance [63, 64]. Furthermore, their success often hinges on access to high-quality retained data with reliable labels—an assumption that rarely holds in practice. To overcome these limitations, our proposed O^3 framework integrates orthogonal unlearning LoRA modules with a tailored out-of-distribution (OOD) detector, enabling continual unlearning without retained data while preserving model utility.

5.3. Method

Proposed Approach Overview. To support continual unlearning in large language models (LLMs) without relying on retained datasets, we introduce O^3 , a framework that

integrates two key components: Orthogonal LoRA Module for incremental unlearning of requested data, and Out-of-Distribution (OOD) Detectors to identify inputs related to previously unlearned content. These modules are trained solely on the data for each unlearning request and do not require access to the original training data. During inference, $\mathcal{O}^3$ uses a soft-weighted mechanism to decide how strongly the unlearning LoRA should be applied, based on the likelihood that the input matches unlearned distributions.

5.3.1. LLM Unlearning with Orthogonal-Regularized LoRA

Fine-tuning large LLMs is resource-intensive and leads to interference across unlearning tasks. To address this, we apply LoRA [69], which introduces low-rank matrices $A \in \mathbb{R}^{U \times K}$ and $B \in \mathbb{R}^{K \times V}$. This ensures that the dimension of $AB\mathbf{h}$ matches that of $W\mathbf{h}$ where $\mathbf{h}$ is the input of the weight matrix,

$$(5.1) \quad \mathbf{h}' = W\mathbf{h} + \Delta W\mathbf{h} = W\mathbf{h} + AB\mathbf{h},$$

where $\mathbf{h}'$ is the output of the weight matrix. During LoRA fine-tuning, W is frozen, and A and B are updated. To unlearn undesirable content, we use preference optimization by assigning random task labels or reject-based answers such as ‘I don’t know’, termed as $\hat{y}$. We adopt CrossEntropy Loss of the unlearning set $\mathcal{D}^{U,t}$ for training,

$$(5.2) \quad \mathcal{L}_{\text{CE}}^t = -\frac{1}{N^{U,t}} \sum_{i=1}^{N^{U,t}} \hat{y}_i^{U,t} \log M_{\Theta}(\mathbf{x}_i^{U,t}).$$

To prevent interference between unlearning steps, we introduce an orthogonal regularization that encourages LoRA parameter updates for different tasks to be orthogonal [193]:

$$(5.3) \quad \mathcal{L}_{\text{Orth}}^t = \left\| (A^{t-1})^\top A^t \right\|^2,$$

and the total loss is:

$$(5.4) \quad \mathcal{L}_{\text{LoRA}}^t = \mathcal{L}_{\text{CE}}^t + \lambda \mathcal{L}_{\text{Orth}}^t,$$

where $\lambda = 0.1$ is a factor used to balance the optimization guidance. Orthogonal regularization is omitted for the first unlearning task and applied thereafter to reduce task entanglement.

5.3.2. Unlearned Knowledge Detection

To detect whether a new input overlaps with previously unlearned content, we frame the task as an OOD detection problem. However, traditional OOD methods rely on class labels, which are not available in most LLM use cases. To overcome this, we design two key components:

OOD Representation Learning with Contrastive Entropy Minimization. We generate two augmented views of an input $\mathbf{x}$, randomly replacing $p\%$ ($p=15$ in our implementation) tokens with [MASK]. The instance with the random masking is denoted as $\mathbf{x}^*$. For the second contrastive view, we follow MoCo [65] to maintain a key encoder $F_{\Omega^{\text{key}}}$ initialized from the original OOD module backbone F_{Ω} that is a transformer consisting of L attention layers, i.e., $F := f_{\omega_1} \circ \dots \circ f_{\omega_l} \circ \dots \circ f_{\omega_L}$. Then the second view is generated from $F_{\Omega^{\text{key}}}$ by forwarding the original text instance $\mathbf{x}$. In the mini-batch training, our contrastive entropy

loss is defined as:

$$(5.5) \quad \mathcal{L}_{\text{CEL}} = - \sum_{i=1}^{N^B} \sum_{l=1}^L \sum_{j=1}^{N^B} \Delta(i, l, j) \log(\Delta(i, l, j)),$$

$$\text{where } \Delta(i, l, j) = \frac{\exp(f_{\omega_{[1:l]}}(\mathbf{x}_i^*) \cdot f_{\omega_{[1:l]}^{\text{key}}}(\mathbf{x}_j))}{\sum_{k=1}^{N^B} \exp(f_{\omega_{[1:l]}}(\mathbf{x}_i^*) \cdot f_{\omega_{[1:l]}^{\text{key}}}(\mathbf{x}_k))},$$

where $f_{\omega_{[1:l]}}(\mathbf{x}_i^*)$ means the token averaging representation of the l -th layer. N^B is the sample quantity of a mini-batch. Considering the forgetting of the pre-trained knowledge, we also supplement $\mathcal{L}_{\text{CEL}}$ with the MLM loss to form the total loss $\mathcal{L}_{\text{OOD}}$,

$$(5.6) \quad \mathcal{L}_{\text{OOD}} = \mathcal{L}_{\text{CEL}} + \mathcal{L}_{\text{MLM}}, \text{ where } \mathcal{L}_{\text{MLM}} = -\frac{1}{N^B} \sum_{i=1}^{N^B} y_i^* \log F_{\Omega}(\mathbf{x}_i^*),$$

where y^* is the random token masking label. When dealing with the continually arriving unlearning requests, we first randomly divide the unlearning dataset $\mathcal{D}^{\text{U},t}$ into two subsets $\mathcal{D}_{\text{used}}^{\text{U},t}$ and $\mathcal{D}_{\text{rest}}^{\text{U},t}$ with $\alpha N^{\text{U},t}$ and $(1 - \alpha)N^{\text{U},t}$ samples ($\alpha = 80\%$ in our implementation), respectively. Then $\mathcal{D}_{\text{used}}^{\text{U},t}$ is used to compute $\mathcal{L}_{\text{OOD}}^t$ and train the OOD detector backbone F_{Ω} . $\mathcal{D}_{\text{rest}}^{\text{U},t}$ will be used in the scoring (Section 5.3.2) and inference mechanisms (Section 5.3.3).

Glocal-aware OOD Scoring.¹ To measure how similar a test input is to the unlearning distribution, we combine: (1) Global Mahalanobis Distance [39] based on the Gaussian distribution of hidden representations across layers, and (2) Local cosine similarity that is the maximum similarity to any training instance. The combined layer-wise score is:

$$(5.7) \quad s(\mathbf{x})_l = d_{\text{Maha}}(\mathbf{x})_l + \gamma \cdot d_{\text{Cos}}(\mathbf{x})_l,$$

¹“Glocal” is a portmanteau of “global” and “local”.

where $\gamma = 1000$ is a scaling factor that unifies the order of magnitude. All layer-wise scores form a vector $\mathbf{s}(\mathbf{x})$, which we use to train a one-class SVM (OCSVM) to define the in-distribution boundary.

When dealing with each unlearning request, after conducting representation learning, we calculate score vectors $\mathbf{s}(\mathbf{x})$ for all samples in $\mathcal{D}_{\text{used}}^{\text{U},t}$. Then use these score vectors to fit an OCSVM with the fitted hypersphere $\mathcal{H}^t(\mathbf{c}^t, R^t)$ characterized by a center vector $\mathbf{c}^t$ and the radius R^t . In addition to $\mathcal{D}_{\text{used}}^{\text{U},t}$, we also need to calculate the score vectors of $\mathcal{D}_{\text{rest}}^{\text{U},t}$, and store all these vectors as we cannot access the unlearning data after the unlearning.

5.3.3. Soft-weighted Inference

Given the T -th past unlearning tasks, we obtain T LoRA modules and OOD detectors. For each testing instance $\mathbf{x}$, we compute its score vector $\mathbf{s}(\mathbf{x})^t$ via Eq. 5.7 for each unlearning task t and determine its distance to the OCSVM boundary:

$$(5.8) \quad d_{\mathcal{H}^t}(\mathbf{x}) = |\mathbf{s}(\mathbf{x})^t - \mathbf{c}^t| - R^t.$$

Then, for all samples of $\mathcal{D}_{\text{used}}^{\text{U},t}$ and $\mathcal{D}_{\text{rest}}^{\text{U},t}$, we calculate their $d_{\mathcal{H}^t}$ via Eq. 5.8 and fit two Gaussian distributions as $\mathcal{N}(\mu[d_{\mathcal{H}^t}(\mathcal{D}_{\text{used}}^{\text{U},t})], \sigma^2[d_{\mathcal{H}^t}(\mathcal{D}_{\text{used}}^{\text{U},t})])$ and $\mathcal{N}(\mu[d_{\mathcal{H}^t}(\mathcal{D}_{\text{rest}}^{\text{U},t})], \sigma^2[d_{\mathcal{H}^t}(\mathcal{D}_{\text{rest}}^{\text{U},t})])$, respectively. We mix these two Gaussian distributions with equal weights and denote the cumulative distribution function as $\mathcal{P}_{\text{mix}}^t$. Next, the probability center of the mixed Gaussian distribution can be determined as $d_{\mathcal{H}^t}^0$ where $\mathcal{P}_{\text{mix}}^t(d_{\mathcal{H}^t}^0) = 0.5$. In the end, for each testing instance $\mathbf{x}$, we have the following weight that reflects how much content of $\mathbf{x}$

belongs to the unlearning distribution of the t -th request,

$$(5.9) \quad w(\mathbf{x})^t = \delta\{\zeta[1 - \max(p, p') + \min(p, p')]\},$$

where $p = \mathcal{P}_{\text{mix}}^t(d_{\mathcal{H}^t}(\mathbf{x})), p' = \mathcal{P}_{\text{mix}}^t(2d_{\mathcal{H}^t}^0 - d_{\mathcal{H}^t}(\mathbf{x})),$

where δ is a Sigmoid function that scales the weight into $(0, 1)$, and $\zeta = 10$ is a scaling factor for more fine-grained sensitivity. After getting the weights from all OOD detectors, we adopt the maximum weight to load the unlearning LoRA by $\mathbf{h}' = W\mathbf{h} + w(\mathbf{x}) \cdot AB\mathbf{h}$. A higher $w(\mathbf{x})$ implies that $\mathbf{x}$ is close to at least one unlearning distribution, thus we should load the unlearning LoRA.

5.4. Experiments

5.4.1. Experimental Setups

Datasets. Our main experiments cover three tasks: Question Answering, Fictitious Knowledge Generation, and Intent Classification, where we sequentially unlearn different subsets while preserving the model’s utility.

- *Question Answering.* From ScienceQA [119], we collect only the text-only instances, yielding training and testing splits of 6,508 and 2,224 samples, respectively. The continual unlearning requests are defined by four ScienceQA domains in the following order: biology $\rightarrow$ physics $\rightarrow$ chemistry $\rightarrow$ economics. For utility assessment, we employ CommonsenseQA [175], which provides 9,740 training and 1,221 validation samples and probes the commonsense reasoning of LLMs. We also include OpenbookQA [126] to gauge book-level comprehension; it contains 4,957 training, 500 validation, and 500 testing samples.

- *Fictitious Knowledge Generation.* *TOFU* [122] is built around GPT-4-generated questions concerning fictional authors. It defines three forget-sets—‘forget01’, ‘forget05’, and ‘forget10’—which correspond to randomly sampled author subsets of 1%, 5%, and 10%, and which we treat as three consecutive unlearning requests. Separately, a disjoint 400-sample set (containing none of the forget-set authors) is used to quantify how well retained knowledge is preserved. *TOFU* further supplies two auxiliary sets—Real-world Authors and World Facts—for utility evaluation.
- *Intent Classification.* *CLINC150* [1] targets intent classification and contains 150 classes spread across five domains, with 200 training, 40 validation, and 60 testing samples per class. In the unlearning scenario, we select the three domains with the strongest privacy implications—‘work’, ‘travel’, and ‘home’—as sequential unlearning requests. Utility is measured using *MRPC* [47] for paraphrase identification and *RTE* [187] for textual entailment.

Evaluation Metrics. For *unlearning effectiveness*, at each unlearning request we evaluate the model on both the *unlearning train set (seen during unlearning)* and the *unlearning test set (unseen during unlearning)*. These two splits share the same distribution but do not overlap, and we refer to the corresponding metrics as *Sample-level Unlearning (S.U.)* and *Distribution-level Unlearning (D.U.)*. For *utility preservation*, we report performance on three distributions. The first, which we call the *Retained Distribution (R.D.)*, is the distribution most likely to be disturbed by the unlearning requests. The remaining two are task-specific utility benchmarks (for question answering: CommonsenseQA and OpenbookQA; for fictitious knowledge generation: *TOFU*-Real Authors (R.A.) and World Facts (W.F.); for intent

classification: MRPC and RTE), whose accuracies we denote by U.1. and U.2. To jointly reflect unlearning effectiveness and utility preservation, we further introduce a metric called the ***Unlearning-Utility Ratio (U^2R)***: $U^2R = (\text{Acc}_{S.U.}^0 + \text{Acc}_{D.U.}^0 - \text{Acc}_{S.U.}^T - \text{Acc}_{D.U.}^T) / (\text{Acc}_{R.D.}^0 + \text{Acc}_{U.1.}^0 + \text{Acc}_{U.2.}^0 - \text{Acc}_{R.D.}^T - \text{Acc}_{U.1.}^T - \text{Acc}_{U.2.}^T)$ where Acc denotes accuracy. A larger U^2R value indicates a better trade-off between unlearning effectiveness and utility preservation.

Compared Baselines. To clearly show the advantages of our methods, we benchmark against a broad set of state-of-the-art language-model unlearning techniques: GradAsc [60], GradDif [211], EUL [27], PO [50], NPO [226], SOGD [75], and SOPO [75]. These baselines are adapted, with only minimal changes, so that they fit our continual unlearning protocol.

Implementation Details. Following the conventions of TOFU [122] and SOPO [75], the target model is LLaMA2-7b-chat [182] for TOFU and LLaMA2-7b for the remaining datasets. Before unlearning on TOFU, we first fine-tune LLaMA2-7b-chat on the full TOFU corpus so that the target model acquires the complete fictitious knowledge. For the other two tasks, we perform instruction tuning on the combined corpora: CLINC150-MRPC-RTE for intent classification and ScienceQA-CommonsenseQA-OpenbookQA for question answering. The OOD detector is built on a pre-trained Roberta-large [114] backbone. Every experiment is repeated with three random seeds (0, 1, and 2), and we report mean and standard deviation. Optimization uses AdamW with a learning rate of $3e-4$ and a batch size of 128 on the combined datasets. Training runs for 10 epochs on ScienceQA-CommonsenseQA-OpenbookQA and 20 epochs on CLINC150-MRPC-RTE. Throughout all experiments, the LoRA rank and alpha are fixed at 8 and 16, respectively.

5.4.2. Effectiveness of Continual LLM Unlearning without Retained Data

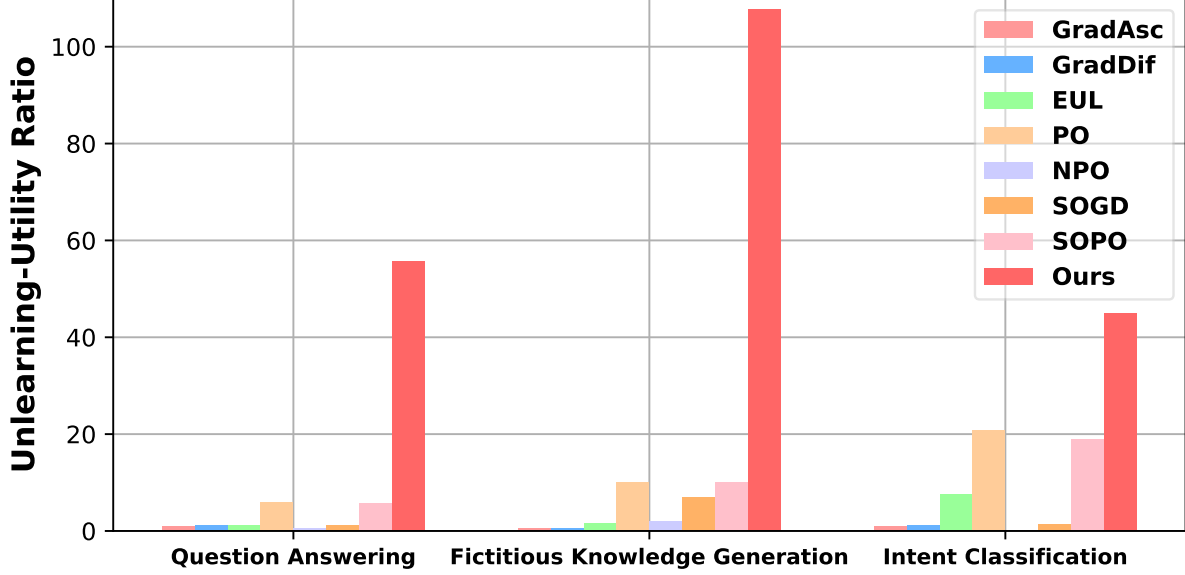


Figure 5.2. Comparison between ours and other baseline approaches on Unlearning-Utility Ratio (U^2R) that measures the balance between unlearning effectiveness and utility preservation.

Experiments are carried out on three tasks with sequential unlearning requests, and we emphasize that *all comparison baselines are granted access to sufficient retained data, whereas our $\mathcal{O}^3$ framework uses only the data associated with each unlearning request and no retained data at all*. Figure 5.2 first reports the Unlearning-Utility Ratio. *$\mathcal{O}^3$ consistently attains the highest U^2R across all three tasks and outperforms the second-best method by a large margin*, clearly demonstrating its effectiveness. In addition to these performance gains, our framework is considerably more efficient in both data and parameter usage. Table 5.1 shows that $\mathcal{O}^3$ needs only half as much training data as the baselines, since retained data is no longer

required. The use of LoRA further shrinks the trainable parameter count to just 20M—below 3% of the baselines’ 6,758M. At inference, our extra computation amounts to only a 5.6% overhead over the baselines. Specifically, while baselines consume more resources during training, inference for them involves only the LLM. Using the Calflops tool on a single-batch input, the baselines measure 13,215 MFLOPS. In our framework, Unlearning Knowledge Detection runs in parallel at 709.37 MFLOPS, and Soft-weighted Inference costs 13,255 MFLOPS, summing to 13,964.37 MFLOPS in total—an increase of just 5.6%.

Like the baselines, our method stores the LLM itself, which occupies 12,862 MB. The OOD-related components add 1,450 MB and the LoRA adds 39 MB, so the extra storage corresponds to 11.6% of the baseline footprint. Because our approach is centered on the LLM, disk storage is rarely a bottleneck and is far less costly than GPU resources.

Table 5.1. Comparison between ours and other baselines on used training data quantity and trainable parameters. The trainable parameters of baselines are all the whole LLM.

Dataset		Baselines	Ours
ScienceQA	Training Data Quantity	4854	2427
	Trainable Parameters	6,758M	20M
TOFU	Training Data Quantity	1280	640
	Trainable Parameters	6,758M	20M
CLINC150	Training Data Quantity	2400	1200
	Trainable Parameters	6,758M	20M

Question Answering. The QA accuracies on the unlearning train and test splits are shown in Figures 5.3 and 5.4. We exclude GradAsc from these plots because it fails to

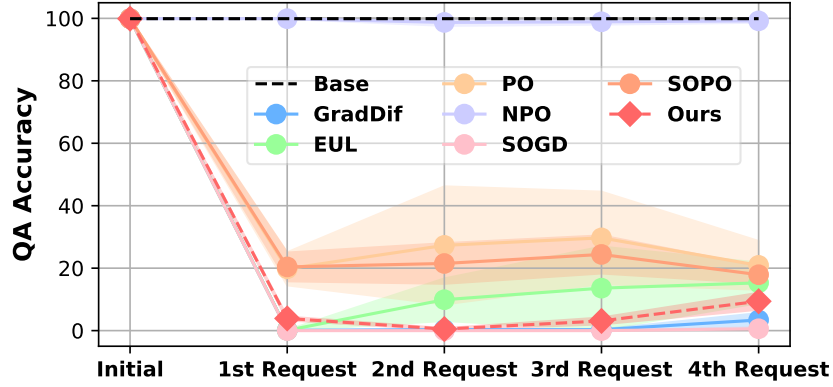


Figure 5.3. Unlearning effectiveness comparison between ours and other approaches on sample-level unlearning (S.U.) of ScienceQA.

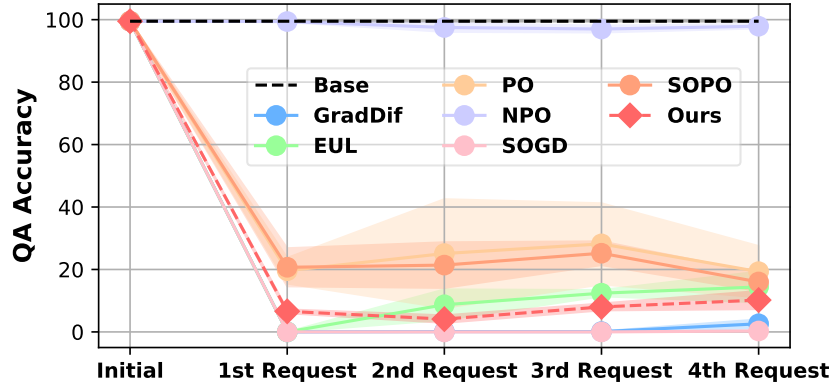


Figure 5.4. Unlearning effectiveness comparison between ours and other approaches on distribution-level unlearning (D.U.) of ScienceQA.

produce any meaningful output across all distributions. Our $\mathcal{O}^3$ sits near the bottom of the accuracy ranking, with only GradDif and SOGD scoring lower. A closer inspection, however, shows that GradDif and SOGD degenerate into empty outputs or strings of repeated tokens, which we regard as an unlearning failure rather than a success. Their utility, as shown in Figure 5.5, is likewise very poor. By contrast, $\mathcal{O}^3$ suffers only a small accuracy drop on the retained distribution relative to the base model (the original LLM with no unlearning applied), and its performance on CommonsenseQA and OpenbookQA is

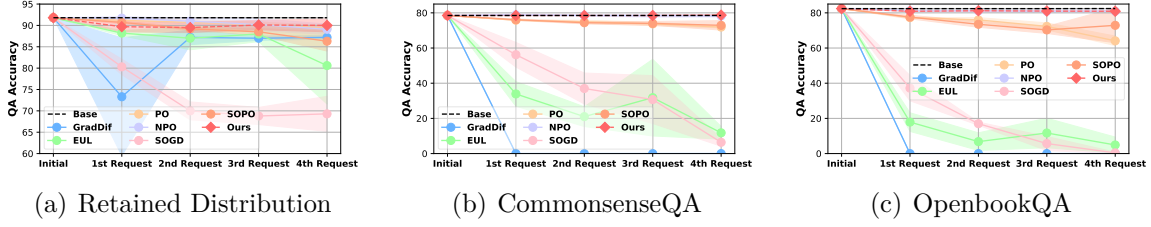


Figure 5.5. Utility preservation performance comparison between ours and state-of-the-art unlearning approaches on the testing set of (a) Retained Distribution, (b) CommonsenseQA, (c) OpenbookQA, after unlearning each request of ScienceQA.

essentially indistinguishable from the base model’s. Taken together, these findings indicate that *under continual unlearning requests, $\mathcal{O}^3$ achieves a far better balance between unlearning effectiveness and utility preservation than any of the baselines.*

Table 5.2. Performance Comparison between our $\mathcal{O}^3$ and other baselines when continually unlearning TOFU-forget01, -forget05, and -forget10 in Fictitious Knowledge Generation. The unlearning effectiveness is measured by the generation accuracy of the unlearning train data and unlearning test data denoted as S.U. and D.U., respectively. Utility preservation is evaluated by the generation accuracy of Retained Distribution (R.D.), TOFU-Real Authors (R.A.), and World Facts (W.F.).

Method	Unlearning Request 1					Unlearning Request 2					Unlearning Request 3				
	S.U.↓	D.U.↓	R.D.↑	R.A.↑	W.F.↑	S.U.↓	D.U.↓	R.D.↑	R.A.↑	W.F.↑	S.U.↓	D.U.↓	R.D.↑	R.A.↑	W.F.↑
Base	85.0±0	90.0±0	85.8±0	89.0±0	86.3±0	87.3±0	89.3±0	85.8±0	89.0±0	86.3±0	85.3±0	90.0±0	85.8±0	89.0±0	86.3±0
GradAsc	75.0±0	85.0±0	81.0±0	86.0±0	82.1±0	17.6±0.2	23.1±1.1	19.0±0	0±0	0±0	17.1±0.9	14.2±5.0	19.0±0	0±0	0±0
GradDif	78.1±0	84.0±1.7	81.9±1.6	86.7±0.6	83.5±0.5	62.5±5.4	70.0±8.7	70.4±3.7	65.7±7.2	77.9±1.7	16.5±0.3	15.2±4.5	19.0±0	0±0	0±0
EUL	84.1±0.2	86.3±0.6	86.1±0.2	86.7±1.5	87.1±0.1	84.4±0.6	90.3±3.8	85.8±0.3	88.0±0	85.5±0	80.1±0.2	83.5±0.5	83.4±0.1	86.3±1.2	83.5±0.5
PO	12.5±0	16.0±1.3	78.4±0.2	86.7±2.3	83.8±0	30.5±2.9	48.8±1.3	82.5±1.2	87.3±1.2	83.2±0.5	38.2±0.2	47.1±7.3	81.4±0.4	86.3±0.6	84.4±0.2
NPO	68.8±3.2	75.0±0	83.6±0.4	89.0±0	81.8±0.5	76.3±2.2	84.2±3.8	83.2±1.6	87.7±0.6	84.1±0.5	77.6±3.9	79.2±3.1	81.4±0.6	87.3±0.6	82.9±1.7
SOGD	43.7±0.1	76.0±1.7	80.3±0.6	85.3±1.2	83.4±0.7	22.8±6.9	24.0±3.6	79.0±3.8	81.3±2.1	82.6±1.0	17.4±0.5	21.7±6.4	82.3±0.3	77.0±6.6	82.1±1.0
SOPO	25.6±1.0	38.0±0.9	83.7±0.3	85.3±1.2	83.7±1.6	34.1±3.0	37.5±2.6	81.5±1.6	87.3±1.5	83.2±1.0	34.5±3.0	40.0±1.3	80.2±1.7	86.7±0.6	84.2±0.8
Ours	12.5±0.5	14.4±0.5	85.1±0.1	89.0±0	86.3±0	15.8±0.3	20.3±0.8	85.0±0	89.0±0	86.3±0	15.5±0.7	19.7±0.7	84.9±0.2	88.8±0.2	86.1±0.2

Table 5.3. Performance Comparison between our $\mathcal{O}^3$ and other baselines when continually unlearning domain ‘work’, ‘travel’, and ‘home’ of CLINC150 in Intent Classification. The unlearning effectiveness is measured by the classification accuracy of the unlearning train data and unlearning test data denoted as S.U. and D.U., respectively. Utility preservation is evaluated by the accuracy of Retained Distribution (R.D.), MRPC, and RTE.

Method	Unlearning Request 1					Unlearning Request 2					Unlearning Request 3				
	S.U.↓	D.U.↓	R.D.↑	MRPC↑	RTE↑	S.U.↓	D.U.↓	R.D.↑	MRPC↑	RTE↑	S.U.↓	D.U.↓	R.D.↑	MRPC↑	RTE↑
Base	100.0±0	99.9±0	99.8±0	88.0±0	88.7±0	100.0±0	99.9±0	99.8±0	88.0±0	88.7±0	99.9±0	99.9±0	99.8±0	88.0±0	88.7±0
GradDif	0.1±0.2	0±0	90.8±3.4	39.9±3.4	31.6±5.3	0±0	0±0	12.7±3.6	9.0±3.8	0.8±0.8	0±0	0±0	75.5±4.8	12.9±6.0	1.7±2.1
EUL	0.1±0.2	0±0	98.3±0.4	87.2±0.1	88.1±0	0.1±0.2	0±0	87.6±3.3	80.3±3.1	82.9±3.1	0±0	0±0	92.3±5.2	81.3±2.1	76.3±4.0
PO	26.3±15.1	26.7±14.0	99.3±0.3	84.1±0.2	86.3±1.1	59.6±3.0	59.8±3.0	99.4±0.2	87.3±0.1	88.0±0.2	56.2±5.4	56.7±4.8	99.0±0.4	86.3±0.2	87.0±0.4
NPO	99.9±0.1	99.0±0	99.2±0.2	87.3±0.3	88.4±0.4	99.9±0.1	99.3±0.3	99.2±0.3	87.2±0.7	88.9±0.6	99.9±0.1	99.2±0.2	99.3±0.1	87.0±0.4	88.9±0.2
SOGD	0±0	0±0	92.3±0.9	6.1±3.6	17.9±6.4	0±0	0±0	93.1±2.0	3.3±3.8	19.5±9.0	0±0	0.1±0.1	94.0±1.8	2.9±0.6	23.7±9.9
SOPO	24.9±15.6	26.3±15.0	99.6±0.1	85.5±0.6	87.1±1.1	62.3±1.4	60.3±1.3	99.6±0.2	87.1±0.2	87.7±1.1	58.8±15.5	59.7±14.8	99.6±0.3	86.6±1.2	86.0±1.4
Ours	10.3±8.1	14.3±0.3	98.9±0.1	84.8±0.1	87.5±0.2	50.5±0.3	55.6±0.6	94.1±0.8	87.0±0.2	89.3±0.2	40.6±4.0	42.4±3.8	97.8±0.8	86.6±1	89.0±0.2

Fictitious Knowledge Generation. The results on TOFU are summarized in Table 5.2. They show that $\mathcal{O}^3$ *delivers the strongest overall combination of unlearning effectiveness and utility preservation: it obtains the best unlearning score in nearly every setting—missing out only once—and leads on utility in the majority of cases.* The sole exception for unlearning effectiveness is D.U. on the third unlearning request, where GradAsc and GradDif edge it out but at the price of near-total collapse of model utility. Generation accuracy on TOFU is computed as follows: using Sentence-BERT [149], we compute the cosine similarity of semantic embeddings between the ground-truth answer and each incorrect alternative that TOFU provides. A generated response is judged correct when its embedding is closest to that of the ground truth, and incorrect otherwise.

Intent Classification. The CLINC150 results appear in Table 5.3. As with QA, GradDif, SOGD, and EUL consistently fail to unlearn, and both GradDif and SOGD additionally collapse on utility. By contrast, *our $\mathcal{O}^3$ framework attains the strongest unlearning performance while matching or exceeding the utility of baselines*

that rely on retained data, both on the retained distribution and on the utility benchmarks. In particular, $\mathcal{O}^3$ preserves RTE accuracy more effectively than any of the compared baselines.

5.4.3. Ablation Study

Our ablation study is structured as follows.

Unlearning Knowledge Detection. We first ablate the contrastive entropy loss $\mathcal{L}_{\text{CEL}}$ in $\mathcal{O}^3$, replacing it with SimCLR (‘Ours w/ SimCLR’) and MoCo (‘Ours w/ MoCo’) using token-masking augmentation. For the scoring component, we also evaluate variants that rely solely on Mahalanobis Distance or Cosine Similarity, denoted ‘Ours w/o d_{Cos} ’ and ‘Ours w/o d_{Maha} ’. We further test a variant that uses only the final transformer layer instead of aggregating across all layers (‘Ours w/ last layer’). Finally, we include two state-of-the-art OOD detection baselines, MDF [206] and Agg [36]. These experiments cover fictitious knowledge generation and question answering, with AUROC reported in Table 5.4. The results show that *the complete OOD detector design adopted in $\mathcal{O}^3$ consistently yields the highest AUROC.*

Table 5.4. OOD detection performance comparison and ablation study between ours and others on Fictitious Knowledge Generation and Question Answering. The measurement is AUROC.

Task	Fictitious Knowledge Generation									Question Answering											
ID	TOFU-forget01			TOFU-forget02			TOFU-forget10			biology			physics			chemistry			economics		
OOD	R.D.	R.A.	W.F.	R.D.	R.A.	W.F.	R.D.	R.A.	W.F.	R.D.	C.QA	O.QA	R.D.	C.QA	O.QA	R.D.	C.QA	O.QA	R.D.	C.QA	O.QA
MDF	90.5	96.6	97.6	80.3	92.7	98.3	91.3	97.8	98.8	91.0	94.4	95.7	92.8	96.0	97.4	92.0	94.3	98.1	94.0	94.7	95.5
Agg	94.4	98.0	98.0	81.9	94.0	98.5	85.0	97.5	99.0	91.1	96.0	95.4	92.2	94.5	95.7	91.0	94.8	95.0	94.5	96.0	97.6
Ours w/ SimCLR	96.0	97.2	99.0	70.2	76.2	86.9	87.9	95.1	99.0	92.5	96.0	96.5	93.1	96.6	97.6	93.0	95.5	96.4	94.2	95.9	96.6
Ours w/ MoCo	95.8	98.0	99.1	87.2	92.2	99.0	91.3	98.6	99.1	93.3	94.2	94.0	92.2	96.4	96.0	95.7	97.1	98.0	94.2	96.3	97.5
Ours w/o d_{Cos}	90.3	96.6	97.6	85.6	92.4	97.6	90.7	98.0	99.0	96.0	95.5	98.4	96.5	96.9	98.8	97.0	97.0	98.8	95.9	97.3	98.2
Ours w/o d_{Maha}	97.0	98.0	99.0	89.0	93.5	98.1	87.5	97.5	98.5	99.0	97.8	98.4	99.4	98.0	97.2	99.0	97.9	96.3	99.0	97.8	97.5
Ours w/ last layer	91.3	98.0	98.0	75.6	94.1	98.8	82.1	97.4	99.0	94.4	98.6	97.4	96.8	98.9	98.8	97.0	98.8	98.0	96.9	97.9	97.2
Ours full	97.8	99.0	99.2	92.5	95.0	99.1	93.0	98.8	99.2	99.5	99.5	99.4	99.8	99.9	99.8	100	99.9	99.8	99.9	99.9	99.8

Table 5.5. Hyper-parameter analysis of the unlearning knowledge optimization of $\mathcal{O}^3$ framework. We adopt a series of values for the factor λ of Eq. 5.3 to validate the necessity of $\mathcal{L}_{\text{Orth}}$ and analyze the sensitivity of λ . ‘C.QA’ shorts for CommonsenseQA, and ‘O.QA’ shorts for OpenbookQA.

Dataset	ScienceQA					CLINC150				
Metric	S.U.	D.U.	R.D.	C.QA	O.QA	S.U.	D.U.	R.D.	MRPC	RTE
$\lambda = 0$	15.3	18.1	90.1	78.1	80.0	33.3	31.8	21.8	86.4	89.2
$\lambda = 0.01$	6.3	11.4	90.9	78.1	80.8	33.8	33.3	54.5	86.8	88.8
$\lambda = 0.05$	8.3	13.1	91.1	78.3	80.8	33.9	35.7	80.8	87.0	89.2
$\lambda = 0.1$	9.3	14.0	91.1	78.5	80.9	35.0	36.2	97.0	87.0	88.9
$\lambda = 0.2$	9.9	14.8	91.2	78.5	81.4	51.2	54.3	98.5	87.1	88.4

Table 5.6. Hyper-parameter analysis of the soft-weighted inference of $\mathcal{O}^3$ framework. We adopt a hard-weighted (Hard-w) mechanism and change the scaling factor ζ of Eq. 5.9. ‘C.QA’ shorts for CommonsenseQA, and ‘O.QA’ shorts for OpenbookQA.

Dataset	ScienceQA					CLINC150				
Metric	S.U.	D.U.	R.D.	C.QA	O.QA	S.U.	D.U.	R.D.	MRPC	RTE
Hard-w	38.0	42.2	91.3	79.2	83.0	37.0	41.2	97.3	87.4	87.7
$\zeta = 1$	10.0	17.0	91.1	78.5	80.8	36.9	40.5	97.0	87.0	88.8
$\zeta = 5$	9.4	14.9	91.1	78.5	80.8	36.1	38.5	97.0	87.0	88.8
$\zeta = 50$	9.3	12.5	90.4	78.5	80.8	35.9	37.2	97.0	87.0	88.8
$\zeta = 100$	9.1	12.2	90.0	78.5	80.8	35.9	37.2	96.8	87.0	88.8
Ours	9.3	14.0	91.2	78.5	80.9	35.0	36.2	97.0	87.0	88.9

Unlearning Knowledge Optimization. The optimization objective for unlearning knowledge (Eq. 5.3) contains a coefficient λ that trades off $\mathcal{L}_{\text{CE}}$ against $\mathcal{L}_{\text{Orth}}$. To examine the role of $\mathcal{L}_{\text{Orth}}$ and the sensitivity of λ , we sweep $\lambda \in \{0, 0.01, 0.05, 0.1, 0.2\}$. The study covers question answering and intent classification, and the numbers in Table 5.5 indicate that *the orthogonal loss is helpful both for preserving utility on the retained distribution and for strengthening the unlearning effect*.

Soft-weighted Inference. As an alternative to the soft-weighting strategy for activating the unlearning LoRA, we experiment with a hard-weighted variant (‘Hard-w’ in Table 5.6). Concretely, we first compute the range of hypersphere boundary distances over the unlearning set $\mathcal{D}^{U,t}$, i.e., $[\min(d_{\mathcal{H}^t}(\mathcal{D}^{U,t})), \max(d_{\mathcal{H}^t}(\mathcal{D}^{U,t}))]$. At test time, for an input $\mathbf{x}$, the unlearning LoRA is activated when $d_{\mathcal{H}^t}(\mathbf{x})$ lies within this interval and detached otherwise. We additionally perform a sensitivity sweep of the scaling factor ζ in Eq. 5.9 using values $\{1, 5, 50, 100\}$. All runs use ScienceQA and CLINC150, with the post-last-request performance reported in Table 5.6. Two observations emerge: the ‘Hard-w’ variant is markedly weaker at removing the targeted knowledge, and increasing ζ makes our framework progressively more aggressive in suppressing it.

5.5. Conclusion

In this work, we address key challenges in developing effective machine unlearning techniques for LLMs, where existing methods often rely on retained data and struggle with continual unlearning. We propose O³, a novel framework that integrates orthogonal low-rank adapters for continual unlearning and an out-of-distribution detector to assess similarity between inputs and unlearned data. Extensive experiments show that O³ significantly outperforms state-of-the-art baselines in both unlearning effectiveness and utility preservation, without relying on retained data.

This work is published in the Thirteenth International Conference on Learning Representations (2025). *Gao, Chongyang, et al. "On large language model continual unlearning." The Thirteenth International Conference on Learning Representations. 2025.*

CHAPTER 6

Detecting Audio Deepfakes**6.1. Introduction**

Rapid advances in generative AI have produced hyper-realistic synthetic media, with audio deepfakes now a powerful vector for misinformation [227, 30]. From impersonating public figures in fraudulent speeches¹ to manipulating markets via fabricated CEO statements², such misuse threatens democratic discourse, security, and trust in digital content [185, 156]. Although benchmark datasets and detectors exist [208, 212, 223], they fail to reflect real-world conditions [129, 198], where deepfakes are noisy, semantically tied to a speaker’s identity, and produced by SOTA voice cloning.

Current ASVspoof-derived datasets [90, 202, 89, 180, 208, 194] emphasize controlled lab conditions, ignoring environmental noise and contextual plausibility. In-The-Wild [129] and DEEP-VOICE [14] draw from online sources but cover a narrow speaker set and rely on easily detected fakes. The result is a false sense of security: *detectors approach perfect benchmark accuracy yet fail on the nuanced, adversarially perturbed deepfakes seen in practice.*

¹<https://www.npr.org/2024/05/30/nx-s1-4986088/deepfake-audio-elections-politics-ai>

²<https://blackbird.ai/blog/celebrity-deepfake-narrative-attacks>

We address this gap with Perturbed Public Voices (P²V), a large-scale, IRB-approved³ dataset of 257,440 samples from 206 public figures. By combining LLM-generated transcripts aligned with each speaker’s public persona (3 LLMs), 10 acoustic perturbations simulating natural and adversarial distortions, and 10 state-of-the-art TTS, voice cloning, and voice conversion methods (2020 to 2025), P²V supports the development of detectors robust to emerging threats.

Audio deepfakes of public figures are growing rapidly in volume and sophistication, raising risks of election interference, reputational damage, and financial fraud [21, 103, 37]. Notable examples include fabricated audio of President Joe Biden urging voters to stay home, Elon Musk promoting a crypto scam, and Mukesh Ambani endorsing fake investment schemes, each capable of deceiving large audiences.

Most prior detection work focuses on the audio signal alone [188, 40], but deepfakes of public figures expose rich contextual cues that remain underused. We therefore propose CADD (Context-based Audio Deepfake Detection), which augments audio with contextual signals such as the speaker’s occupation, demographics, and prior public mentions from news and social media. This enriched input can be consumed by existing deepfake detectors or traditional classifiers, and can also help validate the audio channel of audiovisual deepfakes.

Existing benchmarks rely largely on synthetic audio from TTS, voice cloning, and voice conversion tools [152, 87, 153], assuming naive adversaries and missing the sophistication of real attacks. We therefore introduce SYN, a dataset of 771 clips (468 synthetic deepfakes, 303 real) each paired with contextual metadata.

³IRB approval number: STU00223255

Building on insights from the Global Online Deepfake Detection System (GODDS) [141], we further release the Journalist-provided Deepfake Dataset (JDD). Unlike ITW [129], whose spoofed samples are overtly implausible, JDD contains realistic deepfakes contributed by over 70 journalists and captures adversaries’ efforts to evade detection through subtle manipulations [198].

Across SYN, JDD, ITW, and P²V, CADD consistently outperforms 71 baseline detectors and conventional models, and notably improves robustness to audio perturbations that typically erode detection accuracy.

6.2. Related Work

Audio deepfake detection methods typically fall into two categories: (1) feature-based pipelines, which separate front-end feature extraction from back-end classification [218, 99], and (2) end-to-end models, which unify these processes into a single system [79, 172].

Approach 1: Feature-Based Pipelines. This approach relies on extracting handcrafted or learned features that capture artifacts indicative of synthetic audio. Commonly used features include Mel-frequency cepstral coefficients (MFCCs)[157], which emulate the human auditory system, and linear-frequency cepstral coefficients (LFCCs)[38], which are more robust in noisy settings [133, 195]. Traditional systems such as STC [179] and Light CNN [200] combine these features with classical classifiers. SpecRNet [84], a lightweight neural network, matches LCNN’s performance while being more efficient.

Approach 2: End-to-End Models. End-to-end methods, such as RawNet2 [172], bypass manual feature design by learning representations directly from raw audio. AASIST [79] further improves this by using a graph attention mechanism to capture temporal and

spectral artifacts. RawNet3 [81] builds on this with a Res2Net backbone for enhanced performance.

Recent work has also leveraged large pre-trained models. Tak et al. [173] and Xie et al. [203] utilized wav2vec 2.0 [9] as a front-end encoder. Though originally trained for ASR tasks, wav2vec 2.0 improves generalization when fine-tuned for deepfake detection [189]. Kawa et al. [83] extended this idea using Whisper [147], a large-scale ASR model trained on 16x more data than wav2vec. Its robust encoder—composed of convolutional layers, GeLU activations [66], positional embeddings, and attention blocks—makes it effective across diverse conditions.

Contextual Differentiation with CADD. Unlike all prior work, our proposed CADD framework explicitly incorporates contextual information—such as subject metadata and prior public appearances—into the detection pipeline. We integrate this context module into both end-to-end and feature-based systems, with a particular focus on the latter due to its modularity and compatibility with a dedicated context fusion layer. We also evaluate the effect of context on classical machine learning models, demonstrating its general utility across architectures.

6.3. Perturbed Public Voices

This section details the construction of our P²V dataset. Figure 6.1 gives an overview of the pipeline.

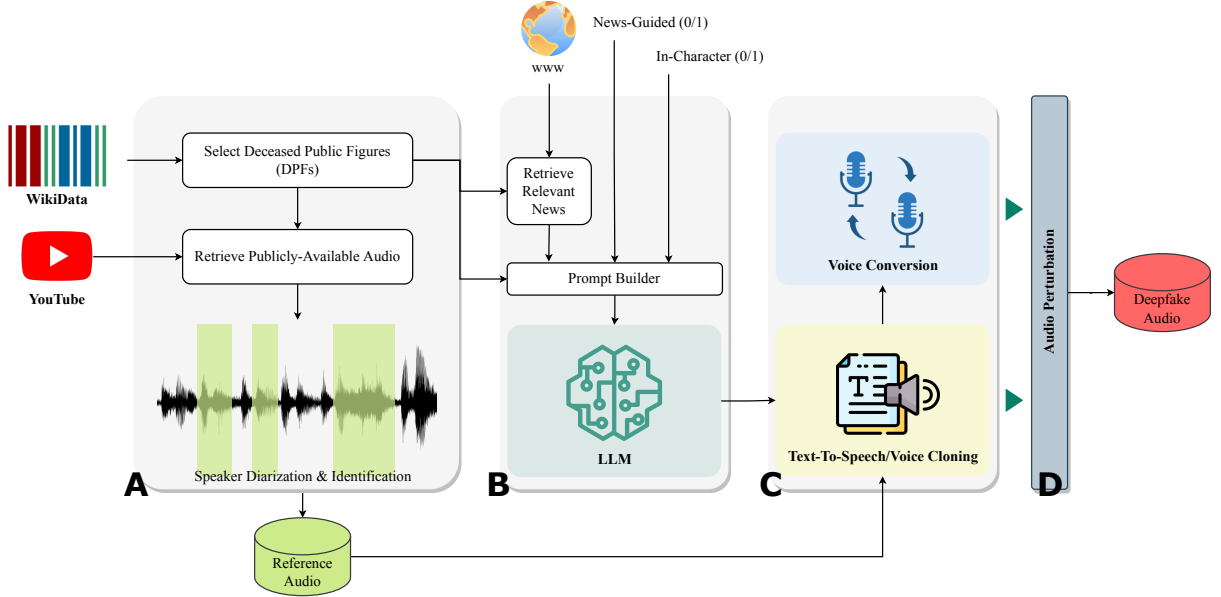


Figure 6.1. **Overview of the audio deepfake generation pipeline for the P²V dataset.** The workflow has four stages: (A) *Reference audio collection* from WikiData and YouTube, followed by speaker diarization and identification to isolate target speaker segments; (B) *Fake transcript generation* with LLMs under news-guided and in-character/out-of-character conditioning; (C) *Audio deepfake synthesis* via Text-to-Speech (TTS) and Voice Conversion (VC) methods; and (D) *Audio perturbation application* to simulate real-world acoustic conditions and adversarial distortions.

6.3.1. Reference Audio Collection

Selection of Deceased Public Figures (DPFs). For ethical and legal reasons, we restrict the dataset to DPFs. Generating realistic deepfake audio of living individuals, even for research, risks misuse if samples leak, reputational harm, and possible violations of personality rights. DPFs sidestep these risks while preserving speech patterns, public personas, and media presence that closely match the living targets typically impersonated by malicious actors. This design received IRB approval (STU00223255).

DPFs were identified via WikiData over the period January 1, 2012 to May 1, 2025, focusing on professions frequently targeted by deepfakes: politicians, journalists, actors, musicians, writers, television presenters, film directors, philosophers, and activists. Manual curation confirmed availability of high-quality audio for 206 public figures.

Publicly-Available Audio Retrieval. To reflect conditions available to real attackers who harvest public speech for cloning, we collected authentic audio from YouTube. Queries targeted interviews, speeches, panel discussions, podcasts, and other public speaking engagements, with preference given to recordings that are long, clearly articulated, and have minimal background noise. This mirrors the practical constraints of unauthorized voice replication and strengthens the ecological validity of P²V.

Speech Diarization and Speaker Identification. Because the collected clips contain multiple speakers, we isolate the target DPF’s speech using Pyannote [140, 17], a state-of-the-art diarization framework that segments audio into utterances and assigns speaker IDs. We then merge consecutive same-speaker segments up to a 40-second cap, and manually verify that each speaker ID maps to either the target DPF or an *unknown* label; *unknown* segments are dropped for privacy. For consistency, we further exclude segments shorter than 5 seconds (insufficient reference content) or longer than 60 seconds (prone to diarization artifacts and speaker transitions).

6.3.2. Fake Speech Transcript Generation

To span a broad range of real-world deployment scenarios, we pair each DPF with temporally grounded transcripts. For every DPF, we sample 10 distinct articles published in the five years preceding their death, retrieved through targeted queries over publicly available content.

Building on the strong role-playing and public-figure reasoning capabilities of modern LLMs [164, 118, 214, 32, 28], we produce four transcript categories by crossing in-character versus out-of-character style with news-guided versus non-news-guided content.

To balance capability and efficiency, we use open-source instruction-tuned LLMs in the 10 to 30 billion parameter range rather than proprietary systems such as ChatGPT or Gemini. Three models were selected for their strong instruction adherence and formatting compliance [132, 197]: *Gemma-3* [176] (27B), *Qwen2.5* [209] (14B), and *Mistral*⁴ (24B). To keep audio durations near 30 seconds, each model is prompted for roughly 100 word transcripts using identical prompts.

6.3.3. Audio Deepfake Generation

We synthesize high-fidelity deepfakes by combining our transcripts with public reference audio through ten state-of-the-art pipelines: seven TTS models and three voice conversion (VC) methods. Each TTS model is conditioned on a semantically aligned transcript and a randomly chosen target-speaker reference. The TTS set comprises Tacotron DCA [11], YourTTS [24], XTTS [23], VEVO [228], Zonos⁵, CosyVoice 2 [49], and IndexTTS [41], together covering multilingual and zero-shot synthesis as well as fine-grained prosody and

⁴<https://huggingface.co/mistralai/Mistral-Small-24B-Instruct-2501>

⁵<https://github.com/Zyphra/Zonos>

emotion control. To increase speaker variability and evaluate layered synthesis, we also use a two-stage pipeline in which IndexTTS output is converted by Diff-HierVC [33], FreeVC [95], or Seed-VC [109], enabling text-free, style-controllable, and speaker-adaptive conversion in zero-shot settings. All methods use official pretrained weights, and every sample is tagged with its generative method to support attribution in downstream analyses.

6.3.4. Audio Perturbations

To reflect real-world acoustic variability and probe model robustness, each generated deepfake receives one of ten perturbations drawn uniformly at random, with hyperparameters sampled per transformation. Perturbation type and parameters are logged with every sample for full reproducibility.

The ten perturbations are: (1) *Background noise addition*, mixing one file sampled from the ESC-50 dataset [139] (2,000 labeled environmental sounds); (2) *Gaussian noise injection*, with amplitude minimum fixed at 0.001 and maximum sampled from $\{0.005, 0.01, 0.015, 0.02\}$; (3) *Air absorption simulation*, with source to microphone distance selected from $\{10, 20, 50, 100\}$ meters; (4) *MP3 compression*, at bitrates drawn from $\{8, 16, 32, 64\}$ kbps to mimic online transmission loss; (5) *Pitch shifting*, with semitone offsets sampled from $\pm 2, \pm 4, \pm 6$, and ± 12 ; (6) *Band-pass filtering*, with passbands chosen from (200, 4000), (150, 5000), or (50, 8000) Hz to emulate telephony or other bandwidth-limited audio; (7) *Time stretching*, with playback speed factors from $\{0.6, 0.8, 1.2, 1.4\}$ (pitch preserved); (8) *Audio reversal*, which flips the waveform in time and requires no parameters; (9) *Impulse response convolution*, adding reverberation from a random sample

in the EchoThief Impulse Response Library⁶; and (10) *Clipping distortion*, controlled by percentile thresholds from $\{(0, 20), (10, 40), (20, 60)\}$. Together, these perturbations form a controlled yet varied augmentation suite covering environmental noise, adversarial conditions, and common signal degradations, enabling rigorous and ecologically valid evaluation of deepfake detectors.

6.3.5. Dataset Statistics and Organization

Table 6.1 summarizes P²V. The dataset contains 257,440 audio samples from 206 unique subjects, split by speaker into training (131), validation (37), and test (38) sets to prevent leakage of vocal identity across splits. Following standard practice [170], roughly 60%, 20%, and 20% of samples go to training, validation, and test, respectively. Because no speaker appears in more than one split, models are pushed to generalize to unseen voices rather than overfit to specific vocal traits. As in other benchmarks (e.g., ASVspoof [180], WaveFake [55], CodecFake [204]), fake samples outnumber real ones, in contrast to real-world audio distributions where genuine speech dominates. We therefore report class-specific metrics and encourage future work to account for this imbalance.

To support evaluation under varying acoustic conditions, we release both a clean version P²V[o] and a perturbed version P²V[p] that applies the noise perturbations described above.

6.4. Context-based Audio Deepfake Detection (CADD)

Throughout this section, the following case study illustrates each step of the CADD methodology.

⁶<https://www.echothief.com/>

Table 6.1. P²V distribution of subjects and audio samples across training, validation, and test splits. The dataset contains both synthetic (fake) audio generated through various methods and authentic (real) audio samples.

Category	Training	Validation	Test	Total
Subjects	131	37	38	206
<i>Synthetic Audio Samples</i>				
Samples per LLM (3)	52,400	14,800	15,200	82,400
Samples per Voice Cloning Method (10)	15,720	4,440	4,560	24,720
Samples per Noise Perturbation (10)	15,720	4,440	4,560	24,720
Fake Audios (Total)	157,200	44,400	45,600	247,200
Real Audios	6,237	1,963	2,040	10,240
Grand Total	163,437	46,363	47,640	257,440

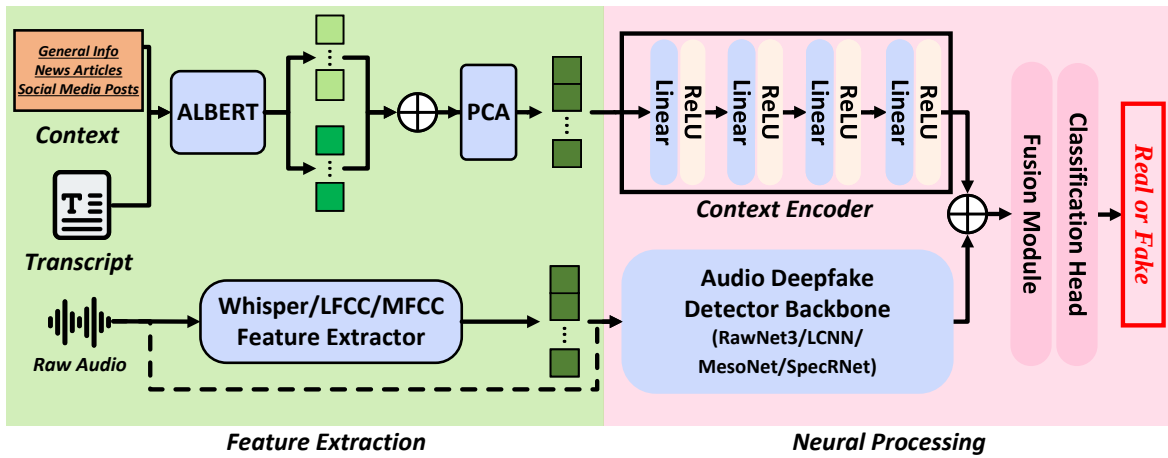


Figure 6.2. The architecture of our CADD context fusion module has three components: 1) the context encoder, 2) the deepfake audio detection backbone, and 3) the fusion module and classification head.

Running Example: Financial Fraud Deepfake

Around June 24, 2024, a sophisticated deepfake audio impersonating Mr. Mukesh Ambani, one of India’s most prominent business leaders, circulated online with unsolicited investment advice^a. The fabricated audio claims: “Do you really trust

your own judgment? Do you really think you can make money with your stocks? Follow me and my student, Mr. Venit. We will provide you with free stock diagnosis and investment plan customization and recommend stocks worth buying in the future stock market. Time is limited. The number of people is limited. Paid resources are now shared for free. Everyone is welcome to join."

Designed to mimic Mr. Ambani’s speech patterns and exploit his credibility, the clip illustrates how malicious actors weaponize public figures to lure investors into financial scams.

^a<https://www.asiafinancial.com/deepfake-of-asias-richest-man-used-in-india-stock-market-scam>

Journalist-provided Deepfake Dataset (JDD). We first describe how JDD was assembled, then explain the contextual information attached to each sample. JDD combines real and deepfake audio collected from diverse public platforms. Over 70 journalists from major outlets (e.g., New York Times, Wall Street Journal, CNN, Agence France Presse, and others) contributed deepfake artifacts they encountered during their investigations, giving us access to real-world deepfakes of public figures that carry societal risk. Our sources are:

- *Google Alerts.* Since February 17, 2024, we have collected news articles, blog posts, and other media flagged by Google Alerts for the keyword “deepfake”.
- *Fact-Checkers.* We continuously monitor deepfake content shared by accredited fact-checkers, including Africa Check, AFP Fact Check, Incident Database,

Boomlive, FactCheck.org, OpenSecrets, PolitiFact, and Snopes⁷. Several of these organizations have also asked us to help assess whether specific content is likely real or fake.

- *GODDS Platform* [141]. On July 8, 2024, we released the Global Online Deepfake Detection System (GODDS)⁸, restricted to journalists (a requirement imposed by our university legal team), which lets users submit suspected deepfakes. The platform has so far served over 70 journalists worldwide.

Authentic audio is drawn from publicly available YouTube videos of well-known deceased public figures. Our results (Tables S1 to S12) show that prior state-of-the-art models drop significantly in performance on this real-world data.

To strengthen CADD, we attach contextual information about the subject of each alleged deepfake. This context situates the media in its real-world setting. We use three complementary sources.

General Information. We pull biographical and factual data from the WikiData API⁹, which exposes structured entries for people, organizations, and places. For each subject we retrieve a short description, occupations, demographics, and, where available, personal data such as spouse, children, and gender, as well as social presence metrics like follower counts. The result is a profile that clarifies the background against which the alleged deepfake should be interpreted.

⁷Fact-checking websites: Africa Check, AFP Fact Check, Incident Database, Boomlive, FactCheck.org, OpenSecrets, PolitiFact, Snopes.

⁸GODDS platform: godds.ads.northwestern.edu

⁹wikidata.org

Running Example: General Information

Key details extracted from WikiData about Mukesh Ambani:

- **Description:** Indian businessman
- **Gender:** Male
- **Birth Date:** April 19, 1957
- **Nationality:** India
- **Occupation:** Entrepreneur, businessperson, graphic designer
- **Education:** University of Mumbai, Stanford University, Stanford Graduate School of Business, Institute of Chemical Technology, Hill Grange High School
- **Spouse:** Nita Ambani
- **Children:** Akash Ambani, Isha Ambani, Anant Ambani
- **Followers:** n.a.

These details give CADD a rich factual backdrop against which anomalous audio content can be cross-referenced and validated.

News Articles. We collect news coverage published before the deepfake’s release to capture the public narrative around the subject at that time. Using WorldNewsAPI¹⁰, we retrieve the 10 most recent pre-release articles that mention the subject and extract both title and full text. This provides a reference against which inconsistencies with current events or perceptions can be detected.

¹⁰worldnewsapi.com

Running Example: News Articles

We retrieved 10 news articles mentioning Mukesh Ambani prior to the deepfake's release, including a highly relevant piece from DNA India^a. The article reports a scam in which a doctor lost 710,000 Indian Rupees (roughly \$8,500) after being deceived by a separate Ambani deepfake promoting a fraudulent share-trading academy. This prior incident establishes a clear pattern of impersonation that CADD is designed to detect. The remaining articles are provided in the supplementary text.

^a<https://www.dnaindia.com/mumbai/report-mukesh-ambani-deepfake-video-mumbai-doctor-falls-prey-to-frauds-loses-rs-7-lakh-in-3094299>

Social Media Posts. To capture real-time discussion around the subject at the time of the deepfake, we query Reddit through the PRAW API¹¹ for up to 10 posts mentioning the subject, using the same pre-release date filter as for news. For each post we extract the title, body, and first 10 comments, producing a snapshot of public sentiment and commentary that sharpens the plausibility assessment of the deepfake.

Running Example: Social Media Posts

Reddit posts about Mukesh Ambani retrieved prior to the incident offer several useful contextual signals. One post^a discussed a potential 13,400 crore land investment in Navi Mumbai, consistent with his ongoing business activity and investment pattern. Another^b highlighted his plan to extend 5G solutions to Africa. Together, these posts provide a factual baseline against which the content of suspected deepfakes can be compared. The full post list is included in the supplementary text.

^ahttps://www.reddit.com/r/Maharashtra/comments/1d99962/gujju_mukesh_ambani_to_invest_13400_crores_to_buy/

¹¹praw.readthedocs.io

^bhttps://www.reddit.com/r/updatenewdailyweekly/comments/1d1ja2i/rils_mukesh_ambani_set_for_african_safari_with_5g/

Dataset Statistics. JDD contains 255 real-world deepfake audio samples spanning November 18, 2015 to July 31, 2024. Because of our collection methodology and the recent surge in deepfake generation, most samples come from 2023 and 2024.

For training, we pair each deepfake with an authentic sample (255 in total) drawn from public speeches of well-known figures published between 2009 and 2024, giving a balanced set of 510 audio samples. The dataset is stratified into training (70%), validation (10%), and test (20%) splits.

Every sample, real or deepfake, is annotated with the three contextual sources described above. Entertainment dominates with roughly 150 subjects, followed by Politics with around 60 subjects. The remaining categories (Music, Media, Sports, Writing, Fashion, Business, Law, Academia & Research, and Activism) are progressively smaller. Categories are derived by grouping WikiData occupations; for each individual we use only the first listed occupation, which typically reflects their most prominent role. The occupation-to-category mapping is given in the supplementary text.

Synthetic Audio Dataset (SYN). To study the effect of contextual information under controlled conditions, we also build SYN through a two-phase pipeline applied to a set of deceased public figures.

Phase 1: Fake Transcript Generation. The first phase (Algorithm 1) produces a diverse set of speech transcripts. For each subject, an LLM¹² generates four distinct transcripts. Leveraging the contextual knowledge that the LLM acquired during pre-training, we first

¹²GPT-4o was used in our experiments.

produce two base transcripts: one describing what the subject *would say*, preserving their typical style and beliefs, and one describing content they *would not say*, introducing deliberate out-of-character deviations. Mixing the two types keeps the dataset balanced. Relying only on in-character speech would push detectors toward narrow stylistic cues, while relying only on out-of-character content would create systematic artifacts that models could exploit. Prompts are detailed in the supplementary text.

To cover deepfake scenarios that are tied to real events, we generate two more context-aware transcripts conditioned on news retrieved from WorldNewsAPI¹³. When possible, we use the publication date of the authentic audio of the subject; otherwise (for figures who passed away well before recent news coverage), we sample a random date from January 1, 2023 onward¹⁴. Given the date, the retrieved articles are fed to the LLM with instructions to produce two transcripts: one aligned with the subject’s persona in light of the news, and one misaligned or implausible. This grounds fake content in realistic use cases such as misinformation around major events or political discourse.

Phase 2: Synthetic Audio Generation. The second phase (Algorithm 2) converts the transcripts into synthetic audio using four voice cloning methods: XTTS-v2 [23], OpenVoice-v2 [144], MetaVoice [125], and WhisperSpeech [35]. To support a fair comparison across methods, we use a balanced assignment strategy so each method is equally represented in the final set. For every transcript, the assigned method generates audio using the original clip as the reference voice and the transcript as the content.

Dataset Statistics. SYN contains 468 synthetic deepfake audio samples spanning December 12, 2014 to July 21, 2024. As with JDD, we add 202 publicly available authentic

¹³<http://worldnewsapi.com>

¹⁴This start date is set by the reliable coverage range of WorldNewsAPI.

samples, bringing the total to 771. The dataset is stratified into training (70%), validation (10%), and test (20%) splits.

Method. We now describe the Context-based Audio Deepfake Detection (CADD) system, shown in Figure. 6.2. CADD targets deepfakes of public figures, a common attack surface where accuracy improves once contextual information around the artifact is taken into account.

Features Extraction. We first outline our text embedding procedure, then apply it to obtain context and transcript representations.

Text Feature Extraction. Text input t (audio transcripts, news articles, or Reddit posts) is tokenized into $\mathbf{x} = [x_1, \dots, x_T]$ with a pre-trained tokenizer, padded to the batch maximum and truncated where needed. The tokenizer also emits an attention mask $\mathbf{M} = [m_1, \dots, m_T]$ with:

$$(6.1) \quad m_i = \begin{cases} 1 & \text{if token } t \text{ is not a padding token} \\ 0 & \text{otherwise} \end{cases}$$

The tokenized input is then passed through a SentenceBERT [150] model $f_\theta(\cdot)$ with parameters θ , producing contextual token embeddings:

$$(6.2) \quad \mathbf{H}i = f_\theta(\mathbf{x}) \in \mathbb{R}^{T \times d}$$

where T is the sequence length and d the embedding dimension. For simplicity and generalizability, we set $d = 100$.

Attention-weighted mean pooling over token embeddings yields a fixed-size representation $\mathbf{e}_i \in \mathbb{R}^d$ that properly accounts for padding:

$$(6.3) \quad \mathbf{e} = \frac{\sum_{i=1}^T m_i \mathbf{h}_i}{\max(\sum_{i=1}^T m_i, \epsilon)}$$

where $\mathbf{h}_i$ is the i -th token embedding, m_i the corresponding mask value, and $\epsilon = 10^{-9}$ a small constant for numerical stability. This pooling (1) restricts the contribution to valid tokens, (2) normalizes each contribution properly, and (3) keeps the embedding in the original token-representation dimensionality.

Context Features. WikiData supplies a subject profile covering gender, occupations, spouse and children (including number of children, when applicable), and social media follower count. The subject’s description and occupation are embedded and then averaged into a single vector. Gender and the spouse/children relationships are encoded as binary values, and numeric fields are passed through unchanged.

News article titles retrieved from WorldNewsAPI¹⁵ are embedded individually and averaged; bodies follow the same procedure. Reddit post titles, bodies, and comments are embedded in the same way.

To assemble the context input, we concatenate the feature vectors from general information, news, and social media, then normalize so each field contributes comparably. We reduce the concatenated vector to 100 dimensions via PCA [2], which curbs dimensionality while retaining the most informative variance.

¹⁵worldnewsapi.com

Transcript Features. The input audio is transcribed with an ASR system, specifically Whisper [146]¹⁶. The resulting transcript features are aggregated with the context features in our experiments.

Neural Processing. CADD’s Neural Processing framework integrates context through the architecture shown in the right panel of Figure 3 of the main manuscript. It has three modules: (1) a context encoder, (2) an audio deepfake detector backbone [81, 200, 4, 84], and (3) a fusion module with classification head. The framework consumes two input streams: context features for the context encoder and either raw audio (for RawNet3) or extracted audio features (for the other backbones) for the detector. Their outputs are concatenated and passed to the fusion module, which refines the joint representation before classification. Each module is described below.

Context Encoder. The context encoder E_c consists of four blocks, each a linear layer followed by a Leaky Rectified Linear Unit (LeakyReLU) with a negative slope of 0.1 to curb overfitting. The first block’s input dimension matches the context feature size, and its output is fixed at 128 for robust generalization (this can be tuned further). For simplicity, the second and third blocks use input and output dimension 128. The final block takes 128 as input and outputs a vector whose dimension is matched to the detection backbone: 384 for RawNet3, 128 for LCNN, 6 for MesoNet, and 64 for SpecRNet, based on the original baseline implementations. These outputs are kept lower than the backbone feature dimensions so that CADD prioritizes audio evidence while treating context as a supporting signal.

¹⁶<https://huggingface.co/openai/whisper-large-v2>

Detection Backbone. We use two families of detection backbones B . The first is end-to-end, taking raw audio and training jointly with the rest of the network. The second is feature-extraction followed by classifiers. Following [83], we adopt RawNet3 [81] for the end-to-end setting, and LCNN [200], MesoNet [4], and SpecRNet [84] for the feature-extraction setting.

RawNet3 is a hybrid of ECAPA-TDNN [43] and RawNet2 [172] with logarithmic and normalization features built into the backbone. Its core block is a sequence of 1D convolutions, ReLU, and batch norm, whose output together with a residual is passed into a max-pool layer and a feature map scaling module (global average pool, fully connected, sigmoid). LCNN, known for its strong ASVspoof performance, is a CNN tailored to low-level acoustic features and uses blocks of convolution, a Max-Feature Map (MFM) layer based on MaxOut, and batch norm. MesoNet is a CNN whose blocks apply convolution, ReLU, batch norm, and max pooling; its pre-classification feature dimension is 16. SpecRNet combines ResBlocks (convolution, batch norm, LeakyReLU [205]) with FMS Attention blocks, originally introduced in RawNet2 [80], which sandwich a feature map scaling module between two max-pool layers.

Fusion Module & Classification Head. The fusion module has two blocks, each composed of a fully connected layer and a LeakyReLU with negative slope 0.1, following the original implementation. Its input is the concatenation of the backbone and context encoder outputs, so the input dimension equals the combined embedding size. The first fully connected layer outputs a vector of the same dimension as the backbone, and for simplicity the second layer uses the same input and output dimension. The classification head is

a single fully connected layer. Following [83], the pre-classification feature dimension is 3072 for RawNet3, 768 for LCNN, 16 for MesoNet, and 128 for SpecRNet.

Model Training. The task is binary (deepfake or not). The input to CADD is a tuple (x_a, x_c) of raw audio and extracted context features. The backbone produces audio features $B(x_a)$ and the context encoder produces $E_c(x_c)$; these are concatenated and passed through the fusion and classification module f_{cls} . We train with binary cross-entropy loss:

$$(6.4) \quad \mathcal{L} = -(y \cdot \log(p) + (1 - y) \log(1 - p))$$

where y in Equation 6.4 is the label of the input audio and $p = f_{cls}(\text{Concate}(E_c(x_c), B(x_a)))$.

6.5. Results

CADD analyzes an audio clip a of a public figure f_a by combining features automatically extracted from three sources: context (recent news and social media posts mentioning f_a , together with Wikidata entries), transcript, and the audio itself. We benchmark CADD against 71 baselines: 22 state-of-the-art (SOTA) audio deepfake detectors, including RawNet3 [81], LCNN [200], MesoNet [4], and SpecRNet [84] with various feature extractors (LFCC, MFCC, Whisper [83]), and 49 traditional machine learning (TML) models. Three CADD variants are tested: CADD(T) uses transcripts, CADD(C) uses context, and CADD(T+C) uses both. We report precision (P), recall (R), and F1-score for the real and fake classes as well as the weighted average, plus Area Under the ROC Curve (AUC) and Equal Error Rate (EER). For a compact summary, we additionally report an overall Avg score, defined as the mean of AUC, the F1-score on the fake class, and 1-EER.

We adopt the optimizer, learning rate, and weight decay from the neural baselines of [83]. All models are trained with Adam. For RawNet3, the learning rate is 0.001 and the weight decay is $5e - 5$. For LCNN, MesoNet, and SpecRNet, both values are set to 0.0001. The batch size is 16, and we grid-search training epochs over $\{10, 20, 30\}$, finding convergence at 30 epochs. Each experiment is repeated with random seeds 0, 1, and 2, and we report the mean. The Whisper encoder is frozen throughout. All runs use three NVIDIA RTX A6000 GPUs.

6.5.1. Performance on the JDD Dataset

Figure 6.3 summarizes the performance of all detectors on JDD, split into SOTA baselines (Figure 6.3(a)) and TML baselines (Figure 6.3(b)). The four rows in each plot correspond to the baseline algorithms, CADD(T+C), CADD(C), and CADD(T), with the x-axis showing the Avg score. CADD(T+C) consistently achieves the best score against SOTA baselines, while CADD(C) and CADD(T+C) jointly take the lead against TML baselines. Per-metric breakdowns (F1-score, AUC, and EER) are reported in figs. S1 to S3.

Table 6.2 gives a detailed per-metric comparison (P, R, F1, AUC, EER) between the top 4 SOTA models, the top 4 TML baselines (selected from the full set of 71), and the three CADD variants on JDD. We reuse this same model selection across all other benchmarks for consistency.

For statistical comparison, we computed the absolute error $|y - \hat{y}|$ (where y and $\hat{y}$ denote the ground truth and the prediction) for each baseline detector and its CADD variant, producing two per-sample error distributions. A Mann-Whitney U test [123] then checks whether the CADD variant yields significantly lower errors than the baseline. All

significance claims ($P < 0.05$) refer to statistically lower CADD error distributions relative to the baseline, with False Discovery Rate (FDR) correction [13] applied to account for multiple testing.

Across all 22 SOTA detectors (tables S1 to S4), CADD(T+C) is the best variant in 18 out of 22 cases, with 15 of those improvements being statistically significant ($P < 0.05$). CADD(C) wins the remaining 4 cases, 3 of them significantly ($P < 0.05$). Overall, CADD(T+C) lifts the Avg score by 13.9% relative to the baselines.

A similar picture holds for the 49 TML models (tables S5 to S11). CADD(T+C) is best in 31 of 49 cases (27 significant at $P < 0.05$), and CADD(C) is best in the other 18 (11 significant). Overall, CADD(T+C) adds 22.57% to the average score relative to the baseline, a substantial and significant improvement.

6.5.2. Performance on the SYN dataset

We repeat the analysis on SYN. A representative subset is shown in Table 6.3; the full summaries are in figs. S4 to S7 and the complete results in tables S13 to S23. CADD again improves most metrics for most feature and classifier combinations. The only exception is RawNet3 on AUC and EER, yet even there CADD(T+C) achieves higher F1 on both classes and a slightly higher Avg score.

Considering all SYN results (tables S13 to S23), CADD(T+C) improves the Avg score by 3.99% on average across the 4 SOTA detectors with 7 feature-type combinations, significantly ($P < 0.05$) in 17 of 21 cases. On the TML side, the improvement rises to 9.5%, with 33 of 49 cases reaching significance.

6.5.3. Performance on the ITW dataset

We also run the full protocol on ITW [129], simulating contextual information because ITW provides no publication dates. Detailed results appear in figs. S8 to S11 and tables S24 to S34. SOTA detectors already near saturation on ITW (Avg > 0.99 in most cases), but CADD still improves every case. On average, CADD(T+C) adds 0.89% across the 22 SOTA baselines and 1.48% across the 49 TML detectors, with improvements significant in 20 of 21 and 49 of 49 cases, respectively. The smaller gains reflect the already near-perfect SOTA baselines.

6.5.4. Performance on the P²V dataset

We evaluate on the P²V dataset [58], an IRB-approved benchmark built to mirror realistic malicious deepfake scenarios: identity-aligned transcripts generated by LLMs, combined with deepfakes produced by state-of-the-art voice cloning pipelines. Using the provided identity labels, we fetch context from recent news, social media, and Wikidata, discarding any sample for which no contextual signal can be retrieved. With more than 250,000 samples, P²V is over eight times the size of ITW, supporting robust large-scale training and evaluation. Given the dataset size and associated compute cost, we use only the best-performing Whisper features identified on the other benchmarks. As shown in Table 3, CADD consistently leads on every SOTA baseline. In particular, against SpecRNet, CADD(T+C) improves F1 by 6.15 points and cuts EER by 52.26%, clearly demonstrating its edge. All improvements are statistically significant ($p < 0.001$), confirming the benefit of context-aware deepfake detection. Among TML baselines, only XGBoost attains significant gains from context ($p < 0.05$ for CADD(C) and $p < 0.001$ for CADD(T+C));

GaussianNB, SVC, RandomForest, and the remaining TML baselines in Table 6.5 show no significant gain after FDR correction. This likely stems from P²V’s inherent class imbalance, which is hard for simpler algorithms that lack dedicated mechanisms for skewed distributions, and highlights a key advantage of CADD’s *Neural Processing* module: the deep architecture copes with class imbalance through learned representations and adaptive weighting, extracting nuanced contextual patterns that simpler TML baselines cannot.

6.5.5. Performance Summary

In summary, CADD improves every SOTA and TML detector/feature combination we evaluated. The baselines perform particularly poorly on JDD (the best Avg score is only 87.43, versus 96.05 for CADD), likely because the clips were contributed by journalists who themselves found them hard to assess. On JDD, CADD raises the 22 SOTA baselines’ Avg scores by 5.84 to 23.75%, and the 49 TML baselines by 5.09 to 39.02%.

The gains are consistent but vary with dataset difficulty: largest on JDD (13.9% for SOTA, 22.57% for TML), moderate on SYN (3.99% for SOTA, 9.5% for TML), and smallest but still significant on ITW (0.89% for SOTA, 1.48% for TML), where baselines already performed near-perfectly. On P²V, CADD delivers a 3.3% average improvement over state-of-the-art baselines, confirming the benefit of contextual enrichment at scale. The pattern tracks dataset complexity and context availability: the largest improvements come on challenging real-world examples (JDD), moderate ones on synthetic data (SYN and P²V), and incremental ones on benchmarks where baselines already saturate (ITW).

6.5.6. Robustness of CADD Under Evasive Audio Manipulations

Audio deepfake detectors frequently degrade under even simple manipulations such as time stretching [199, 192]. Since the 22 SOTA/feature combinations outperform the TML baselines, we restrict the robustness study to them and compare against CADD under 5 adversarial audio manipulations: air absorption (simulating natural, frequency-dependent attenuation with distance), background noise injection, Gaussian noise injection, MP3 compression, and playout time stretching [199]. Each manipulation is evaluated at multiple intensities, giving 23 distinct perturbations in total; details are in the supplementary text. We also release perturbed versions of JDD, SYN, and ITW to enable further robustness research.

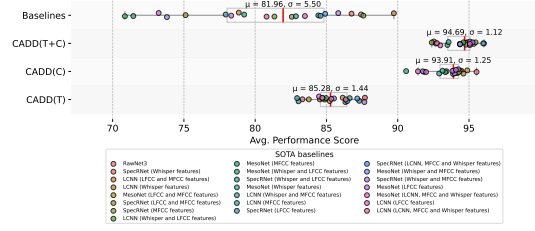
Figure 6.4 shows a subset of the results for the 4 best SOTA baselines (RawNet3, LCNN with Whisper, MesoNet with Whisper, and SpecRNet with Whisper). Rows cover the 5 attack types and their intensities. For a compact summary, we again report the Avg score (mean of AUC, fake-class F1, and 1-EER). The complete results are shown in figures S12 to S14. On JDD, the baselines lose 8.2% in Avg score on average under perturbation (worst case: -22.4% under Gaussian noise), whereas CADD(T+C) drops by only 0.71% on average. Because P²V closely resembles SYN and large-scale training is costly, we run the synthetic-data robustness study only on SYN; there, CADD shrinks the average degradation from -5.74% to -0.91% . On ITW, the improvement is smaller, likely because the dataset is simple enough for audio-only models to approach ceiling performance and because simulated context is less informative than real context.

6.5.7. Error Rates in Detecting Deepfakes about Political and Entertainment Figures

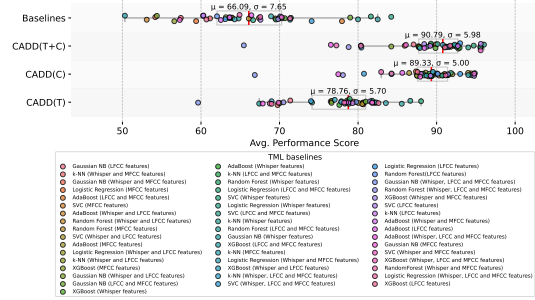
Using the JDD dataset, we compared deepfake detection errors $|\hat{y} - y|$ between entertainment and political figures, where y is the ground truth and $\hat{y}$ the model prediction. With Mann-Whitney U tests and Benjamini-Hochberg correction [13], we examined the error distributions for every CADD configuration across two splits: entertainment versus non-entertainment ($n_E = 71$, $n_{\bar{E}} = 101$) and political versus non-political ($n_P = 68$, $n_{\bar{P}} = 104$).

Table S35 shows that CADD variants using context (CADD(C) and CADD(T+C)) differ significantly across occupational groups ($P < 0.001$). For entertainment figures, CADD achieves lower (i.e., better) absolute errors (0.0007 to 0.1691) than for non-entertainment figures (0.0025 to 0.3370). For political figures, errors are significantly higher (0.0036 to 0.4836) than for non-political figures (0.0018 to 0.4092). The effect is consistent across architectures and weakest for LCNN, where CADD(C) and CADD(T+C) are significant for entertainment ($P < 0.01$) but not for political figures.

Linguistic analysis (Table S36) helps explain the gap. Political content is more linguistically complex than entertainment content, with lower Flesch Reading Ease Scores (72.83 vs. 79.29; $P < 0.01$) and longer words (4.38 vs. 4.05 characters; $P < 0.01$). Topic diversity, measured via Latent Dirichlet Allocation (LDA) [15], is about 45% higher in political transcripts (0.84 vs. 0.58; $P < 0.001$). News context follows the same pattern, while Reddit discussions invert it: entertainment threads are more diverse than political threads (0.94 vs. 0.80; $P < 0.001$).



(a) CADD vs. State-of-the-art (SOTA) baselines



(b) CADD vs. Traditional Machine Learning (TML) baselines

Figure 6.3. **Performance comparison of state-of-the-art (a) and traditional machine learning (b) baselines and our CADD configurations (CADD(T), CADD(C), and CADD(T+C)).** Each point represents a model’s average performance score computed on our Journalist-provided Deepfake Dataset (JDD), with the same color denoting the same baseline across different configurations.

Two explanations follow: (1) the greater linguistic complexity of political discourse creates a harder detection environment, and (2) deepfake creation patterns differ across domains. From our experience with the GODDS platform [141], entertainment deepfakes tend to follow consistent content patterns (often scams or promotions) and produce more recognizable detection signatures, while political deepfakes span a wider range of intent, from satire to disinformation, creating a more variable detection challenge.

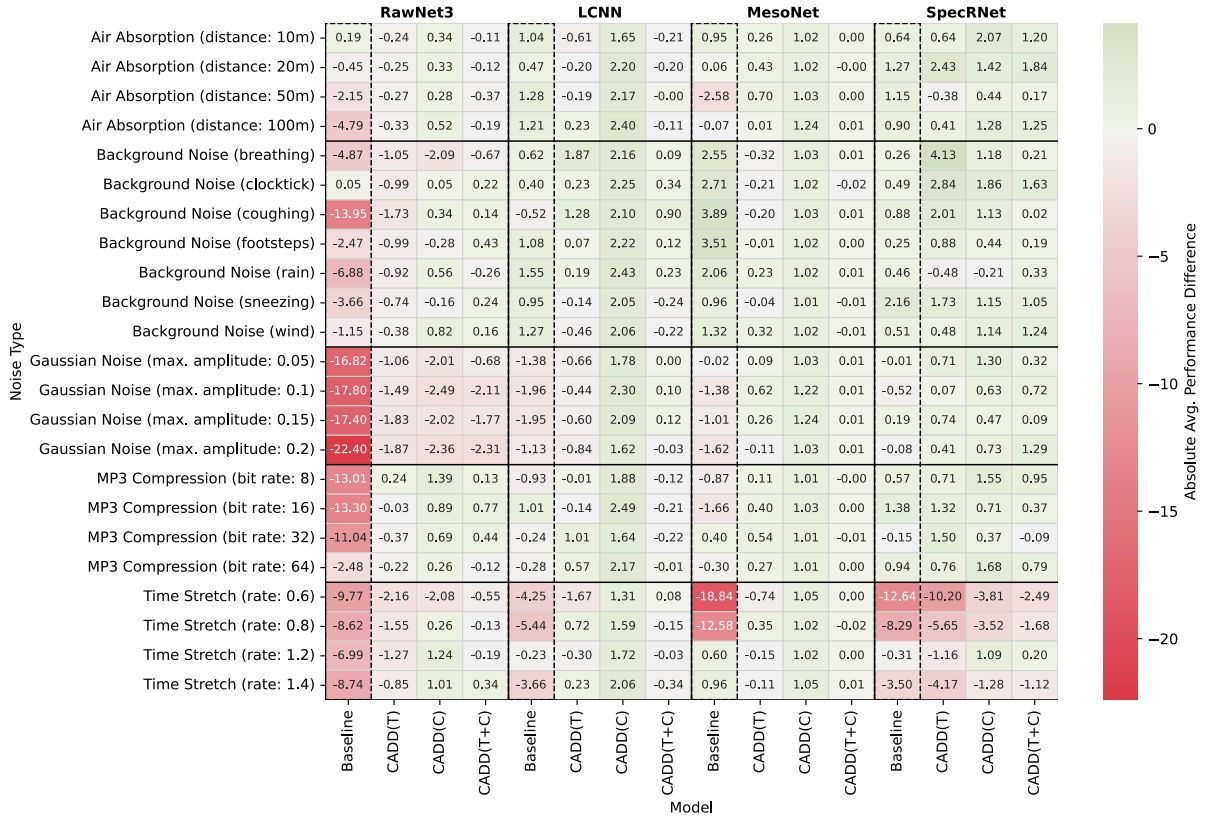


Figure 6.4. **Performance Robustness Under Audio Manipulations (JDD dataset)** The figure shows the absolute percentage difference between the results (in terms of Avg scores) obtained on the perturbed JDD test set and those obtained on the original, unperturbed test set for the subset of 4 best-performing SOTA baselines and related CADD configurations. SOTA models are separated by black vertical lines, with baseline results highlighted in a dashed rectangle.

6.6. Conclusion

Existing audio deepfake detectors for public figures rely on audio alone and ignore the rich context around each clip. Common benchmarks also lack realism, using absurd or out-of-character content, or non-public figures with synthetic audio. These weaknesses show up when such detectors underperform on our Journalist-provided Deepfake Dataset

Table 6.2. **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on JDD data, with varying contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric are highlighted in bold and marked with different background colors (■ ■ ■), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$). See tables S1-S12 for full results.

Base Model	CADD Enhanced Model	Real			Fake			Overall					P-Value
		P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	AUC (↑)	EER (↓)	
RawNet3	Baseline	80.52	89.13	84.49	90.06	81.48	85.44	85.67	85.00	85.00	92.91	16.05	—
	Ours CADD(T)	74.99	81.88	78.23	83.22	76.54	79.68	79.43	79.00	79.02	88.18	17.90	0.375
	Ours CADD(C)	90.58	<u>94.20</u>	92.27	94.88	91.36	93.02	92.90	92.67	92.67	<u>98.51</u>	4.94	0.001**
	Ours CADD(T+C)	<u>88.63</u>	95.65	<u>91.99</u>	96.04	<u>89.51</u>	<u>92.64</u>	<u>92.63</u>	<u>92.33</u>	<u>92.34</u>	98.91	<u>5.56</u>	0.101
LCNN	Baseline	87.29	79.71	83.23	84.09	90.12	86.95	85.56	85.33	85.24	95.21	12.96	—
	Ours CADD(T)	74.03	84.06	78.63	84.77	74.69	79.30	79.83	79.00	78.99	89.88	17.90	10 ^{-12***}
	Ours CADD(C)	94.65	<u>89.13</u>	<u>91.77</u>	<u>91.23</u>	95.68	<u>93.38</u>	<u>92.80</u>	<u>92.67</u>	<u>92.64</u>	<u>98.05</u>	<u>6.79</u>	10 ^{-9***}
	Ours CADD(T+C)	<u>92.81</u>	93.48	93.13	94.43	<u>93.83</u>	94.12	93.69	93.67	93.67	98.98	4.94	10 ^{-5***}
MesoNet	Baseline	68.23	88.41	76.83	86.81	64.20	73.46	78.26	75.33	75.01	85.84	21.60	—
	Ours CADD(T)	78.95	77.54	77.07	81.49	79.63	79.52	80.32	78.67	78.39	88.12	18.52	0.002**
	Ours CADD(C)	<u>89.32</u>	<u>92.03</u>	<u>90.49</u>	<u>93.06</u>	<u>90.12</u>	<u>91.43</u>	<u>91.34</u>	<u>91.00</u>	<u>91.00</u>	<u>96.55</u>	<u>9.26</u>	10 ^{-14***}
	Ours CADD(T+C)	90.97	92.75	91.76	93.81	91.98	92.82	92.50	92.33	92.33	98.32	8.02	10 ^{-14***}
SpecRNet	Baseline	82.37	79.71	80.51	84.19	85.19	84.27	83.35	82.67	82.54	92.07	16.05	—
	Ours CADD(T)	82.83	86.96	84.83	88.37	84.57	86.41	85.82	85.67	85.68	92.12	15.43	10 ^{-6***}
	Ours CADD(C)	<u>89.05</u>	94.20	<u>91.52</u>	94.93	<u>90.12</u>	<u>92.43</u>	<u>92.22</u>	<u>92.00</u>	<u>92.01</u>	<u>97.79</u>	6.79	10 ^{-13***}
	Ours CADD(T+C)	90.95	94.20	92.53	<u>94.92</u>	91.98	93.41	93.09	93.00	93.01	98.03	6.79	10 ^{-13***}
LR	Baseline	78.26	78.26	78.26	78.26	78.26	78.26	78.26	78.26	78.26	88.37	21.74	—
	Ours CADD(T)	85.11	86.96	86.02	86.67	84.78	85.71	85.89	85.87	85.87	91.40	13.04	10 ^{-5***}
	Ours CADD(C)	<u>87.76</u>	<u>93.48</u>	<u>90.53</u>	<u>93.02</u>	<u>86.96</u>	<u>89.89</u>	<u>90.39</u>	<u>90.22</u>	<u>90.21</u>	<u>96.41</u>	<u>10.87</u>	10 ^{-10***}
	Ours CADD(T+C)	89.80	95.65	92.63	95.35	89.13	92.13	92.57	92.39	92.38	98.53	6.52	10 ^{-10***}
SVC	Baseline	80.00	86.96	83.33	85.71	78.26	81.82	82.86	82.61	82.58	88.23	17.39	—
	Ours CADD(T)	83.33	86.96	85.11	86.36	82.61	84.44	84.85	84.78	84.78	92.91	17.39	10 ^{-6***}
	Ours CADD(C)	<u>91.49</u>	<u>93.48</u>	<u>92.47</u>	<u>93.33</u>	91.30	<u>92.31</u>	<u>92.41</u>	<u>92.39</u>	<u>92.39</u>	<u>98.82</u>	<u>6.52</u>	10 ^{-12***}
	Ours CADD(T+C)	91.67	95.65	93.62	95.45	<u>91.30</u>	93.33	93.56	93.48	93.48	99.10	4.35	10 ^{-13***}
RF	Baseline	71.73	89.86	79.76	86.42	64.49	73.82	79.07	77.17	76.79	87.88	21.74	—
	Ours CADD(T)	77.23	<u>92.03</u>	83.80	90.72	72.46	80.23	83.97	82.25	82.02	90.94	14.49	0.047*
	Ours CADD(C)	<u>87.05</u>	92.75	<u>89.79</u>	92.37	<u>86.23</u>	<u>89.17</u>	<u>89.71</u>	<u>89.49</u>	<u>89.48</u>	<u>96.83</u>	<u>11.59</u>	10 ^{-11***}
	Ours CADD(T+C)	88.18	92.03	90.06	<u>91.72</u>	87.68	89.65	89.95	89.86	89.85	97.39	9.42	10 ^{-11***}
XGBoost	Baseline	76.00	82.61	79.17	80.95	73.91	77.27	78.48	78.26	78.22	87.62	17.39	—
	Ours CADD(T)	84.62	71.74	77.65	75.47	<u>86.96</u>	80.81	80.04	79.35	79.23	89.37	19.57	0.003**
	Ours CADD(C)	91.49	93.48	92.47	93.33	91.30	92.31	92.41	92.39	92.39	<u>96.31</u>	6.52	10 ^{-8***}
	Ours CADD(T+C)	<u>87.50</u>	<u>91.30</u>	<u>89.36</u>	<u>90.91</u>	86.96	<u>88.89</u>	<u>89.20</u>	<u>89.13</u>	<u>89.13</u>	96.64	<u>8.70</u>	10 ^{-9***}

(JDD), curated with over 70 journalists, and on the Perturbed Public Voices (P²V) benchmark [58]: an IRB-approved dataset of over 250,000 samples (8× the size of ITW) pairing LLM-generated, identity-aligned transcripts with state-of-the-art voice cloning.

Table 6.3. **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on SYN data with varying the contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric are highlighted in bold and marked with different background colors (■ ■ ■), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$). See tables S13-S23 for full results.

Base Model	CADD Enhanced Model	Real			Fake			Overall					P-Value
		P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	AUC (↑)	EER (↓)	
RawNet3	Baseline	94.07	93.44	93.71	95.78	96.10	95.92	95.11	95.05	95.05	98.74	3.55	—
	Ours CADD(T)	87.35	93.99	90.53	95.90	91.13	93.45	92.54	92.26	92.30	98.37	6.74	0.013*
	Ours CADD(C)	88.09	91.80	89.86	94.55	91.84	93.15	92.01	91.83	91.86	96.48	6.38	0.018*
	Ours CADD(T+C)	98.85	92.90	95.77	95.57	99.29	97.39	96.86	96.77	96.75	98.17	<u>4.26</u>	0.943
LCNN	Baseline	96.62	92.35	94.40	95.21	<u>97.87</u>	<u>96.51</u>	<u>95.77</u>	<u>95.70</u>	<u>95.68</u>	99.15	4.61	—
	Ours CADD(T)	91.35	97.81	<u>94.47</u>	98.51	93.97	96.18	95.69	95.48	95.51	99.26	3.19	10 ^{-23***}
	Ours CADD(C)	91.37	<u>96.72</u>	93.94	97.78	93.97	95.82	95.26	95.05	95.08	98.37	3.55	10 ^{-12***}
	Ours CADD(T+C)	98.39	<u>96.72</u>	97.53	<u>97.89</u>	98.94	98.41	98.09	98.06	98.06	<u>99.23</u>	0.71	10 ^{-26***}
MesoNet	Baseline	88.02	91.80	89.86	94.51	91.84	93.15	91.96	91.83	91.86	97.19	6.74	—
	Ours CADD(T)	80.34	95.08	86.15	96.88	82.27	88.04	90.37	87.31	87.30	<u>98.54</u>	7.45	0.167
	Ours CADD(C)	87.34	87.98	87.50	92.01	91.13	91.50	90.17	89.89	89.93	94.80	9.93	10 ^{-5***}
	Ours CADD(T+C)	93.06	<u>93.99</u>	93.32	<u>96.14</u>	95.04	95.49	94.93	94.62	94.64	98.45	6.03	10 ^{-17***}
SpecRNet	Baseline	98.30	92.90	95.51	95.55	<u>98.94</u>	97.21	<u>96.63</u>	<u>96.56</u>	<u>96.54</u>	99.05	3.55	—
	Ours CADD(T)	93.68	97.27	95.44	98.18	95.74	96.95	96.41	96.34	96.36	99.47	3.90	0.021*
	Ours CADD(C)	98.87	96.17	<u>97.50</u>	97.57	99.29	98.42	98.08	98.06	98.06	<u>99.78</u>	2.13	10 ^{-4***}
	Ours CADD(T+C)	<u>98.36</u>	<u>96.72</u>	97.52	<u>97.91</u>	<u>98.94</u>	<u>98.41</u>	98.08	98.06	98.06	99.87	2.48	0.478
LR	Baseline	98.31	95.08	96.67	96.88	98.94	<u>97.89</u>	<u>97.44</u>	<u>97.42</u>	<u>97.41</u>	99.35	4.92	—
	Ours CADD(T)	100.00	<u>95.08</u>	97.48	96.91	100.00	98.43	98.12	98.06	98.06	99.90	1.64	0.748
	Ours CADD(C)	<u>100.00</u>	91.80	95.73	94.95	<u>100.00</u>	97.41	96.94	96.77	96.75	96.53	6.56	10 ^{-6***}
	Ours CADD(T+C)	96.67	95.08	95.87	96.84	97.87	97.35	96.77	96.77	96.77	98.43	<u>3.28</u>	10 ^{-5***}
SVC	Baseline	93.22	90.16	91.67	93.75	95.74	94.74	93.54	93.55	93.53	<u>99.39</u>	<u>6.56</u>	—
	Ours CADD(T)	98.36	98.36	98.36	98.94	98.94	98.94	98.71	98.71	98.71	99.88	1.64	0.051
	Ours CADD(C)	100.00	91.80	<u>95.73</u>	<u>94.95</u>	100.00	<u>97.41</u>	96.94	<u>96.77</u>	<u>96.75</u>	97.59	6.56	10 ^{-4***}
	Ours CADD(T+C)	<u>100.00</u>	91.80	95.73	94.95	<u>100.00</u>	97.41	96.94	96.77	96.75	98.85	6.56	0.040*
RF	Baseline	97.07	83.61	89.67	90.33	98.23	94.07	92.98	92.47	92.34	<u>98.51</u>	<u>7.65</u>	—
	Ours CADD(T)	98.32	96.17	97.23	97.57	98.94	98.24	97.87	97.85	97.84	99.80	3.28	10 ^{-7***}
	Ours CADD(C)	100.00	91.80	<u>95.73</u>	<u>94.95</u>	100.00	<u>97.41</u>	96.94	<u>96.77</u>	<u>96.75</u>	96.77	7.65	10 ^{-14***}
	Ours CADD(T+C)	<u>100.00</u>	89.07	94.22	93.38	<u>100.00</u>	96.58	95.99	95.70	95.65	97.46	7.65	10 ^{-13***}
XGBoost	Baseline	91.94	<u>93.44</u>	92.68	95.70	94.68	95.19	94.22	94.19	94.20	98.57	4.92	—
	Ours CADD(T)	96.83	100.00	98.39	100.00	97.87	98.92	98.75	98.71	98.71	99.79	0.00	0.138
	Ours CADD(C)	93.33	91.80	92.56	94.74	95.74	95.24	94.18	94.19	94.18	98.95	8.20	0.155
	Ours CADD(T+C)	<u>95.00</u>	<u>93.44</u>	<u>94.21</u>	<u>95.79</u>	<u>96.81</u>	<u>96.30</u>	<u>95.48</u>	<u>95.48</u>	<u>95.48</u>	<u>99.20</u>	<u>4.92</u>	0.943

To address this, we introduce CADD, a context-aware architecture that augments existing detectors with contextual and transcript signals. On JDD, CADD outperforms baselines by 5.09 to 39.02%. On P²V, it leads every SOTA baseline with a 3.3% Avg score improvement, all statistically significant ($p < 0.001$). Together, the two benchmarks

Table 6.4. **Performance comparison of 4 state-of-the-art (SOTA) models and 4 traditional machine learning (TML) baselines for deepfake detection on P²V data with varying the contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric (within each model group) are highlighted in bold and marked with different background colors (), while the second-best performance is underlined. The last column shows the FDR-corrected P-values (Benjamini-Hochberg procedure) from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$). See table S35 for full TML results.

Base Model	CADD Enhanced Model	Real			Fake			Overall					
		P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	AUC (↑)	EER (↓)	P-Value
RawNet3	Baseline	46.64	70.34	<u>56.09</u>	98.64	96.40	97.51	96.42	95.28	<u>95.73</u>	92.72	12.18	—
	Ours CADD(T)	32.01	88.68	47.04	99.45	91.57	95.35	<u>96.56</u>	91.45	93.28	<u>95.84</u>	10.00	10 ^{-7843***}
	Ours CADD(C)	<u>48.64</u>	53.43	50.92	97.91	97.48	<u>97.69</u>	95.80	<u>95.59</u>	95.69	93.32	14.61	10 ^{-8217***}
	Ours CADD(T+C)	51.60	<u>73.38</u>	60.60	<u>98.79</u>	96.92	97.84	96.77	95.91	96.25	96.13	<u>10.89</u>	10 ^{-5689***}
LCNN	Baseline	68.85	<u>71.52</u>	70.16	<u>98.72</u>	98.55	98.64	<u>97.44</u>	97.40	<u>97.42</u>	96.35	9.62	—
	Ours CADD(T)	27.87	79.80	41.31	99.01	90.76	94.71	95.97	90.29	92.42	93.60	13.15	10 ^{-1339***}
	Ours CADD(C)	<u>83.60</u>	53.48	65.23	97.95	<u>99.53</u>	<u>98.73</u>	97.34	<u>97.56</u>	97.30	97.38	8.54	10 ^{-1002***}
	Ours CADD(T+C)	88.89	56.08	<u>68.77</u>	98.07	99.69	98.87	97.67	97.82	97.58	<u>97.27</u>	8.55	10 ^{-1108***}
MesoNet	Baseline	95.38	9.12	16.64	96.09	99.98	98.00	96.06	96.09	94.51	98.08	5.93	—
	Ours CADD(T)	19.67	1.18	2.22	95.76	<u>99.79</u>	97.73	92.50	95.56	93.64	74.71	31.13	10 ^{-13024***}
	Ours CADD(C)	<u>82.54</u>	<u>83.87</u>	83.20	99.28	99.21	99.24	98.56	98.55	98.56	<u>98.86</u>	<u>4.86</u>	10 ^{-4463***}
	Ours CADD(T+C)	67.53	92.06	<u>77.91</u>	99.64	98.02	<u>98.82</u>	<u>98.26</u>	<u>97.76</u>	<u>97.93</u>	99.20	4.05	10 ^{-1757***}
SpecRNet	Baseline	24.04	83.58	37.34	99.17	88.18	93.36	95.96	87.99	90.96	93.23	13.49	—
	Ours CADD(T)	13.69	91.76	23.83	99.51	74.12	84.96	95.83	74.87	82.34	92.32	15.72	10 ^{-2496***}
	Ours CADD(C)	<u>45.14</u>	<u>86.23</u>	<u>59.26</u>	<u>99.36</u>	<u>95.31</u>	<u>97.29</u>	<u>97.04</u>	<u>94.92</u>	<u>95.66</u>	<u>94.89</u>	<u>10.66</u>	10 ^{-847***}
	Ours CADD(T+C)	59.70	82.84	69.39	99.22	97.50	98.35	97.53	96.87	97.11	97.84	6.44	10 ^{-2501***}
LR	Baseline	5.77	45.34	10.23	<u>97.88</u>	77.34	86.40	95.15	76.39	84.14	63.48	<u>42.83</u>	—
	Ours CADD(T)	<u>5.79</u>	45.34	<u>10.27</u>	97.89	<u>77.45</u>	<u>86.47</u>	95.15	<u>76.49</u>	<u>84.21</u>	63.53	<u>42.83</u>	0.985
	Ours CADD(C)	5.81	45.34	10.30	97.89	77.51	86.52	95.15	76.56	84.25	63.73	42.65	0.985
	Ours CADD(T+C)	5.81	45.34	10.30	97.89	77.51	86.52	95.15	76.56	84.25	<u>63.68</u>	42.65	0.985
SVC	Baseline	23.06	21.33	22.16	<u>97.60</u>	97.82	97.71	95.39	95.55	95.47	73.42	34.77	—
	Ours CADD(T)	22.82	21.15	21.95	97.59	<u>97.81</u>	<u>97.70</u>	95.37	<u>95.54</u>	95.45	73.42	<u>35.13</u>	0.985
	Ours CADD(C)	23.06	21.86	<u>22.45</u>	97.61	97.77	97.69	<u>95.40</u>	95.52	<u>95.46</u>	73.53	34.77	0.985
	Ours CADD(T+C)	23.24	21.86	22.53	97.61	97.79	<u>97.70</u>	95.41	<u>95.54</u>	95.47	<u>73.50</u>	34.77	0.985
RandomForest	Baseline	53.63	1.73	<u>3.36</u>	<u>97.08</u>	99.95	98.50	<u>95.79</u>	97.04	<u>95.67</u>	64.85	40.08	—
	Ours CADD(T)	55.81	2.03	3.92	97.09	99.95	98.50	95.86	97.04	95.69	<u>64.81</u>	<u>40.02</u>	0.985
	Ours CADD(C)	50.00	<u>1.73</u>	3.35	<u>97.08</u>	99.95	<u>98.49</u>	95.68	<u>97.03</u>	<u>95.67</u>	64.46	<u>40.02</u>	0.985
	Ours CADD(T+C)	43.80	1.49	2.89	97.07	<u>99.94</u>	<u>98.49</u>	95.49	97.02	95.65	64.56	39.73	0.985
XGBoost	Baseline	20.63	22.22	21.40	<u>97.61</u>	97.38	97.50	95.33	95.15	<u>95.24</u>	69.98	<u>36.02</u>	—
	Ours CADD(T)	22.08	<u>22.04</u>	22.06	97.62	<u>97.62</u>	<u>97.62</u>	95.37	<u>95.38</u>	95.38	69.93	<u>36.38</u>	0.708
	Ours CADD(C)	18.62	18.82	18.72	97.52	97.48	97.50	95.17	95.15	95.16	69.89	35.30	0.016*
	Ours CADD(T+C)	<u>21.88</u>	21.33	<u>21.60</u>	97.60	97.67	97.63	<u>95.35</u>	95.40	95.38	69.48	37.10	10 ^{-7***}

offer complementary evidence: strong results on curated real-world cases, and the same advantage at large scale under modern adversarial synthesis. Context readily available from news and social media substantially improves detection accuracy for public figures.

This work is in submission.

Table 6.5. **Performance comparison of Traditional Machine Learning (TML) baselines for deepfake detection on P^2V data, with varying contextual information used by CADD.** We evaluate CADD models trained under three conditions: with transcript-only (CADD(T)), with context-only (CADD(C)), and with both transcript and context knowledge (CADD(T+C)). The best performance values for each metric (within each model group) are highlighted in bold and marked with different background colors (■ ■ ■), while the second-best performance is underlined. The last column shows P-values from statistical significance testing (indicated by * for $P < 0.05$, ** for $P < 0.01$ and *** for $P < 0.001$).

Model	CADD Enhanced Model	Real			Fake			Overall					P-Value
		P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	P (↑)	R (↑)	F1 (↑)	AUC (↑)	EER (↓)	
LR	Baseline	5.77	45.34	10.23	<u>97.88</u>	77.34	86.40	95.15	76.39	84.14	63.48	<u>42.83</u>	—
	Ours CADD(T)	<u>5.79</u>	45.34	<u>10.27</u>	97.89	<u>77.45</u>	<u>86.47</u>	95.15	<u>76.49</u>	<u>84.21</u>	63.53	<u>42.83</u>	0.985
	Ours CADD(C)	5.81	45.34	10.30	97.89	77.51	86.52	95.15	76.56	84.25	63.73	42.65	0.985
	Ours CADD(T+C)	5.81	45.34	10.30	97.89	77.51	86.52	95.15	76.56	84.25	<u>63.68</u>	42.65	0.985
GaussianNB	Baseline	4.83	<u>54.12</u>	8.87	<u>97.96</u>	67.38	79.84	95.20	66.99	77.73	67.84	37.81	—
	Ours CADD(T)	4.85	54.30	<u>8.91</u>	97.97	67.43	79.88	95.20	67.04	77.77	67.86	38.17	0.985
	Ours CADD(C)	4.86	53.94	<u>8.91</u>	<u>97.96</u>	67.68	80.05	95.20	67.27	77.94	67.89	37.63	0.985
	Ours CADD(T+C)	4.87	<u>54.12</u>	8.93	97.97	<u>67.63</u>	<u>80.02</u>	95.20	<u>67.23</u>	<u>77.91</u>	<u>67.88</u>	37.46	0.985
K-NN	Baseline	37.87	<u>11.47</u>	17.61	97.35	99.42	98.38	95.58	96.81	95.98	64.43	<u>63.62</u>	—
	Ours CADD(T)	36.53	10.93	16.83	<u>97.33</u>	99.42	98.36	95.53	96.79	95.94	64.32	63.80	0.985
	Ours CADD(C)	<u>37.36</u>	11.65	17.76	97.35	<u>99.40</u>	<u>98.37</u>	<u>95.57</u>	<u>96.80</u>	<u>95.97</u>	64.60	63.26	0.985
	Ours CADD(T+C)	37.14	11.65	<u>17.74</u>	97.35	<u>99.40</u>	98.36	<u>95.57</u>	96.79	<u>95.97</u>	<u>64.59</u>	63.26	0.985
SVC	Baseline	23.06	<u>21.33</u>	22.16	<u>97.60</u>	97.82	97.71	95.39	95.55	95.47	73.42	34.77	—
	Ours CADD(T)	22.82	21.15	21.95	<u>97.59</u>	<u>97.81</u>	<u>97.70</u>	95.37	95.54	95.45	73.42	<u>35.13</u>	0.985
	Ours CADD(C)	<u>23.06</u>	21.86	<u>22.45</u>	97.61	97.77	97.69	95.40	95.52	<u>95.46</u>	73.53	34.77	0.985
	Ours CADD(T+C)	23.24	21.86	22.53	97.61	97.79	<u>97.70</u>	95.41	<u>95.54</u>	95.47	<u>73.50</u>	34.77	0.985
RandomForest	Baseline	53.63	<u>1.73</u>	3.36	<u>97.08</u>	99.95	98.50	<u>95.79</u>	97.04	<u>95.67</u>	64.85	40.08	—
	Ours CADD(T)	55.81	2.03	3.92	97.09	99.95	98.50	95.86	97.04	95.69	<u>64.81</u>	<u>40.02</u>	0.985
	Ours CADD(C)	50.00	<u>1.73</u>	3.35	<u>97.08</u>	99.95	<u>98.49</u>	95.68	<u>97.03</u>	<u>95.67</u>	64.46	<u>40.02</u>	0.985
	Ours CADD(T+C)	43.80	1.49	2.89	97.07	<u>99.94</u>	<u>98.49</u>	95.49	97.02	95.65	64.56	39.73	0.985
AdaBoost	Baseline	5.65	50.00	10.16	<u>97.99</u>	74.47	84.62	<u>95.25</u>	73.74	82.41	<u>66.03</u>	40.68	—
	Ours CADD(T)	5.69	<u>49.64</u>	10.22	<u>97.98</u>	74.85	84.87	95.24	74.10	82.65	65.41	<u>41.04</u>	0.985
	Ours CADD(C)	5.73	50.00	10.28	98.00	<u>74.82</u>	<u>84.86</u>	95.26	<u>74.09</u>	<u>82.64</u>	66.17	40.68	0.985
	Ours CADD(T+C)	5.62	49.46	10.10	97.97	74.61	84.71	95.23	73.86	82.49	65.96	<u>41.04</u>	0.985
XGBoost	Baseline	20.63	22.22	21.40	<u>97.61</u>	97.38	97.50	95.33	95.15	<u>95.24</u>	69.98	36.02	—
	Ours CADD(T)	22.08	<u>22.04</u>	22.06	97.62	<u>97.62</u>	<u>97.62</u>	95.37	<u>95.38</u>	95.38	69.93	36.38	0.708
	Ours CADD(C)	18.62	18.82	18.72	97.52	97.48	97.50	95.17	95.15	95.16	69.89	35.30	0.016*
	Ours CADD(T+C)	<u>21.88</u>	21.33	<u>21.60</u>	97.60	97.67	97.63	<u>95.35</u>	95.40	95.38	69.48	37.10	10 ^{-7***}

CHAPTER 7

Generating Safe and Functional Proteins**7.1. Introduction**

Protein language models (PLMs), trained on millions of natural sequences, have evolved into general-purpose generators capable of producing novel proteins, enabling *de novo* design of enzymes, binders, and therapeutics at unprecedented scale [121, 131, 54]. As these models move from research artifacts toward deployable systems, they raise a dual-use concern that is intrinsic to generative biology. The ability to design functional proteins is agnostic to whether those proteins are beneficial or harmful, and that ability extends to toxins and virulence factors as well [183, 168]. While safety alignment for large language models is now backed by a mature set of methodologies [142, 213, 219], no analogous safety layer currently sits between PLM outputs and downstream protein synthesis. A safety-aligned generator, namely one that maintains structural viability while actively suppressing harmful phenotypes, is therefore a precondition for responsible deployment.

Reinforcement learning from reward models [132, 10], which has become the standard recipe for aligning foundation models. The generation model plays the role of the policy, and a learned reward model provides per-sequence feedback for RL fine-tuning. Porting this recipe to proteins, however, surfaces two challenges that are absent in the language setting and which together drive the design of our framework. The first challenge lies in the reward model itself. “Harmfulness” for a protein is not captured by a single scalar.

Toxicity, defined as direct damage inflicted on host cells, and virulence, defined as the broader capacity of a protein to cause disease through various mechanisms [138, 46], are related but not interchangeable. A reward model intended for safety alignment must therefore score both axes, and this leaves two natural design options. Training two separate per-task classifiers treats toxicity and virulence as unrelated problems, discarding the partially shared predictive signal between them. This is wasteful under data scarcity, since toxic proteins are rare. Because the two tasks have overlapping but non-identical label distributions and class imbalances, naive parameter sharing is susceptible to negative transfer [169], with the more balanced virulence objective dragging the shared trunk toward over-predicting toxic positives. Neither extreme produces a dependable multi-task reward oracle.

The second challenge concerns how multiple objectives are combined during policy optimization. Even with a perfect reward model in hand, PPO still requires a weight vector $(w_{\text{tox}}, w_{\text{vir}}, w_{\text{pLDDT}})$ over the competing rewards. In practice, no fixed weighting jointly optimizes structural foldability and both safety axes. The optimum on one axis seldom coincides with the optimum on another, and the geometry of the trade-off itself shifts during training as the policy explores new regions of sequence space. Standard multi-objective gradient methods such as MGDA [42, 162] provide a principled alternative by deriving a Pareto-descent direction from per-objective gradients. MGDA, however, was originally formulated for modest-scale models and small numbers of objectives, and applying it directly to a large PLM under bounded $[0, 1]$ rewards face *Gram-matrix dilution*, illustrated in Figure 7.1. When the model has many more parameters than the subset on which the objectives genuinely conflict, the conflict signal localized to that subset is

averaged into the silent majority of aligned or inactive parameters, so that the off-diagonal entries of the gradient Gram matrix shrink toward zero or even flip sign.

To tackle these challenges, we introduce *SafePro*, a multi-objective alignment framework for protein language models. *SafePro* consists of two components, each addressing one of the limitations described above. *Cross-Task Attentive Fusion (CTAF)* is a joint reward model for toxicity and virulence. A frozen ESM2 [101] backbone feeds two task-private experts that interact through an asymmetric cross-attention block whose contribution is modulated by a learned, input-dependent gate. This design captures the benefits of joint learning by sharing signal where the toxicity and virulence distributions overlap, which is particularly valuable under class imbalance. *Task-Conflict-Saliency MGDA (TCS-MGDA)* is a weight-calibration procedure that adaptively traverses the utility versus safety Pareto frontier as training progresses. It directly confronts the gram-matrix dilution of standard MGDA. Dilution is mitigated by restricting the Gram matrix to the top- ξ parameters ranked by a *Task-Conflict Saliency* score, a per-parameter disagreement statistic that focuses the conflict estimate on parameters where objectives genuinely compete, and adaptive switching between full and conflict-focused Gram estimates. Empirically, TCS-MGDA lifts the composite score from 60.45 under standard MGDA to 66.86, a +6.41 absolute gain and a +5.21 improvement over the strongest fixed-weight baseline, simultaneously cutting toxicity ($0.0523 \rightarrow 0.0139$) and virulence ($0.6926 \rightarrow 0.5407$) probabilities while preserving structural quality. Our contributions are summarized below.

- We cast safer protein generation as a multi-objective alignment problem with a single utility objective (structural foldability) and two safety objectives (toxicity and virulence), and present ***SafePro***, an integrated framework that pairs CTAF

with TCS-MGDA to yield aligned policies that retain competitive structural quality while improving toxicity avoidance.

- We propose **CTAF**, a cross-task attentive reward model that preserves task-private representations while leveraging selective cross-task signal through an input-dependent gate. CTAF improves composite alignment outcomes over independent per-task classifiers in five out of six fixed-weight PPO configurations.
- We propose **TCS-MGDA**, a gradient-based adaptive weight-calibration method that mitigates Gram-matrix dilution via parameter-space saliency filtering, and adaptive switching between full and conflict-focused Gram estimates. TCS-MGDA improves the composite score by +6.41 over standard MGDA and +5.21 over the best fixed-weight configuration, while removing the need to commit to a reward-weight vector before training.
- Together with the method, we will release the virulence classification dataset curated for this study, which couples experimentally verified virulence factors from VFDB with a stringently filtered non-virulence set drawn from 21 bacterial reference proteomes, yielding 54,166 length-matched sequences in total.

7.2. Related Work

RL-Based Protein Design. Reinforcement learning (RL) has recently emerged as a powerful paradigm for optimizing protein sequences under multiple objectives, with antimicrobial peptide (AMP) design being a particularly active application area. ProDCARL [166] aligns diffusion-based generators using policy gradients with entropy regularization, and ApexAmphion [22] extends PPO-based optimization to large protein language models,

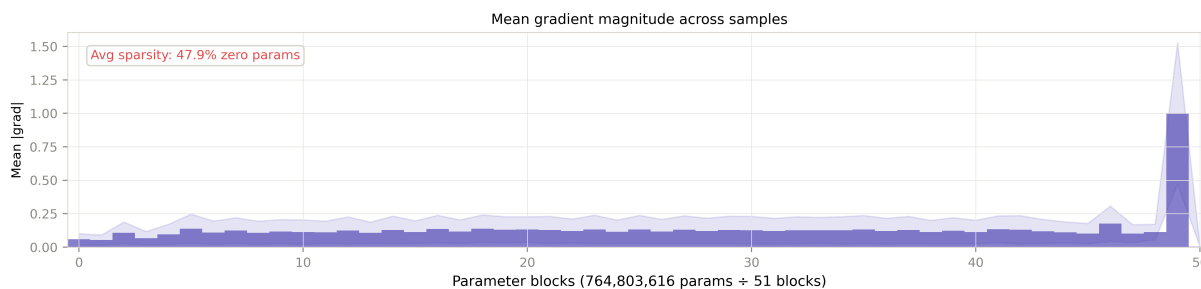


Figure 7.1. Mean gradient magnitude across 32 samples. For each block, we plot $\frac{1}{N} \sum_n \|\nabla_n^{(b)}\|_2$, then divide by the peak block value so the y-axis is in units of *fraction of the peak block magnitude*; the shaded band shows $\pm\sigma$ across samples under the same normalization. The output-projection block near index 49 dominates the mean by roughly an order of magnitude, while 47.9% of parameters are near-zero on any given backward pass (red annotation).

employing composite rewards that combine activity and physicochemical properties. Despite their successes, these approaches share three key limitations: (1) they focus mainly on short peptides rather than full-length proteins, (2) they treat safety-related objectives such as toxicity and virulence as independent, overlooking their biological coupling, and (3) they rely on static scalarization of objectives. Moreover, structural quality metrics such as pLDDT are typically used only for post-hoc evaluation and are not folded into the training loop [166, 190].

Safety-Aware and Multi-Objective Protein Optimization. A growing line of work targets safety-aware protein generation. Knowledge-guided Preference Optimization (KPO) [196] draws on protein safety knowledge graphs to discourage harmful generations, while inference-time techniques such as Logit Diff Amplification (LDA) [20] reduce toxicity without retraining. In parallel, non-RL methods including MOFormer [190], PepPFDPO [191], and PepZOO [111] use Pareto-based or preference-driven optimization to balance activity against toxicity. While effective in their own right, these methods either prioritize safety

without simultaneously optimizing for structural quality, or rely on static Pareto selection and therefore lack iterative policy improvement. They also do not explicitly model interactions among safety objectives or adapt to trade-offs that shift during training.

Conflict-Aware Multi-Objective Learning. Coping with conflicting objectives remains a central problem in multi-objective learning. Classical approaches such as MGDA [42, 162], PCGrad [216], GradNorm [31], CAGrad [104], and Nash-MTL [130] introduce gradient-based strategies for reaching Pareto-efficient solutions. These methods, however, face scalability difficulties in large language models, where the combination of high parameter counts, gradient sparsity, and noisy reward signals can obscure the underlying conflict structure. They also have not been adapted to protein generation, where objectives are biologically coupled and structural feedback (e.g., from ESMFold [102]) is computationally expensive.

7.3. Method

We propose a multi-objective reinforcement learning framework for protein sequence generation. Concretely, we present a Cross-Task Attentive Fusion Reward Model that delivers more reliable reward estimates for toxicity and virulence avoidance, and we jointly fold structural quality into the optimization as an additional objective. Together, these elements push the policy toward producing protein sequences that are both less toxic and less virulent without compromising structural integrity. To balance these potentially competing goals, we adopt a Multi-Gradient Descent Algorithm (MGDA) based weight adaptation scheme. We further introduce a Task-Conflict Saliency (TCS) mechanism that adaptively detects and mitigates gradient conflicts as training proceeds.

7.3.1. Problem Formulation

Let π_θ denote a policy parameterized by $\theta \in \mathbb{R}^D$, where D is the number of trainable parameters. The policy autoregressively generates a protein sequence $x = (x_1, \dots, x_T)$ over an amino acid vocabulary. The objective of training is to bias generation toward sequences that exhibit lower toxicity and virulence while still preserving structural integrity. To this end, we optimize a weighted combination of $K = 3$ reward signals together with a regularization term that constrains the policy to remain close to a frozen reference policy π_{ref} .

For every generated sequence x , three reward functions are computed:

- $r_{\text{plddt}}(x) \in [0, 1]$: the predicted local distance difference test (pLDDT) score from ESMFold [102], normalized by dividing by 100, which quantifies predicted structural quality.
- $r_{\text{toxic}}(x) = 1 - P_{\text{toxic}}(x) \in [0, 1]$: one minus the predicted toxicity probability, so that non-toxic sequences receive higher reward.
- $r_{\text{vir}}(x) = 1 - P_{\text{vir}}(x) \in [0, 1]$: one minus the predicted virulence probability, so that non-virulent sequences receive higher reward.

The toxicity and virulence probabilities are produced by our Cross-Task Attentive Fusion (CTAF) classifier, which is detailed in Section. 7.3.2. The combined scalar reward used by Proximal Policy Optimization (PPO) is then defined as

$$(7.1) \quad r(x) = w_{\text{plddt}} r_{\text{plddt}}(x) + w_{\text{toxic}} r_{\text{toxic}}(x) + w_{\text{vir}} r_{\text{vir}}(x),$$

where $\mathbf{w} = (w_{\text{plddt}}, w_{\text{toxic}}, w_{\text{vir}})$ are nonnegative objective weights that sum to 1.

7.3.2. Cross-Task Attentive Fusion Reward Model

Toxicity and virulence are biologically related yet distinct properties. In pathogens, toxins represent a major class of virulence factors because they damage host tissues, perturb immune function, and accelerate disease progression. Virulence, however, also relies on non-toxic mechanisms such as adhesion, colonization, nutrient acquisition, and immune evasion [138, 46]. Consequently, the two tasks can carry partially shared predictive signal while still demanding task-specific discrimination. From a modeling standpoint this makes joint learning attractive, but naive parameter sharing can also induce negative transfer when partially aligned tasks compete via interfering gradients [169]. We address this issue with a novel *Cross-Task Attentive Fusion* (CTAF) module, illustrated in Figure. 7.2, which lets each task retain a private expert representation and selectively attend to the other task’s representation through a learned, input-dependent gate.

Architecture overview. The CTAF model consists of four components: (i) a frozen ESM2 backbone [101] that yields contextualized token embeddings, (ii) two task-specific expert MLPs that project the shared representation into task-relevant subspaces, (iii) a cross-task attentive fusion module equipped with an adaptive fusion gate that selectively shares information between tasks, and (iv) two independent classification heads. Given a protein sequence x tokenized into T tokens, the backbone outputs hidden states $H \in \mathbb{R}^{T \times d}$. We then apply a pooler f_{pool} , implemented as a linear transformation followed by a Tanh activation, to the CLS-token hidden state, obtaining

$$(7.2) \quad h = f_{\text{pool}}(H_{[0]}) \in \mathbb{R}^d,$$

with $d = 1280$. This pooled vector h acts as the input to the CTAF module.

Per-task Experts. The pooled vector h is projected into two task-specific subspaces by a pair of independent experts:

$$(7.3) \quad e_{\text{tox}} = \text{Expert}_{\text{tox}}(h), \quad e_{\text{vir}} = \text{Expert}_{\text{vir}}(h),$$

where each expert follows the architecture $\text{Linear}(d, d_{\text{exp}}) \rightarrow \text{GELU} \rightarrow \text{Dropout} \rightarrow \text{Linear}(d_{\text{exp}}, d)$, with $d_{\text{exp}} = 1024$. The outputs of both experts lie in $\mathbb{R}^d$, which makes residual connections straightforward. By giving every task its own dedicated parameters and never sharing expert weights, the design eliminates capacity competition between tasks.

Asymmetric cross-task attention. For the active task $\tau \in \{\text{tox}, \text{vir}\}$, the CTAF module queries the *other* task’s expert representation. The underlying intuition is that each task should learn what to borrow from the complementary task instead of receiving a fixed shared signal. This asymmetry is encoded in task-specific query projections $W_Q^{(\tau)}$:

$$(7.4) \quad Q = W_Q^{(\tau)} e_{\tau}, \quad K = W_K e_{\bar{\tau}}, \quad V = W_V e_{\bar{\tau}},$$

where $\bar{\tau}$ refers to the other task and $W_K, W_V \in \mathbb{R}^{d \times d}$ are shared key and value projections (the “what is available” side is task-agnostic). Multi-head attention with $H_a = 4$ heads and head dimension d/H_a is then applied:

$$(7.5) \quad c = \text{MultiHead}(Q, K, V) \in \mathbb{R}^d,$$

followed by an output projection $W_O \in \mathbb{R}^{d \times d}$. Since Q and K are single-token representations, the attention score collapses to a per-head scalar that quantifies how compatible the two tasks' representations are for the given input. This scalar is directly interpretable as a cross-task relevance signal.

Attentive fusion gating. Cross-task information is not equally useful for every sequence. A protein that is simultaneously toxic and virulent should borrow heavily across tasks, whereas a protein that is neither may gain little from cross-task attention. We capture this with an input-dependent gate:

$$(7.6) \quad \gamma = \sigma(W_{g2} \text{GELU}(W_{g1} [e_\tau; c])) \in [0, 1],$$

where $[\cdot; \cdot]$ denotes concatenation, $W_{g1} \in \mathbb{R}^{(d/2) \times 2d}$, and $W_{g2} \in \mathbb{R}^{1 \times (d/2)}$. The gate bias is initialized to -2.0 so that $\gamma \approx 0.12$ at initialization, which makes the model start out predominantly task-private and learn cross-task sharing only when it proves beneficial. The fused representation is given by:

$$(7.7) \quad f_\tau = \gamma \cdot c + (1 - \gamma) \cdot e_\tau.$$

This expression interpolates between full cross-task transfer (when $\gamma \rightarrow 1$) and fully independent task experts (when $\gamma \rightarrow 0$), serving as a safe fallback.

Residual normalization and classification. The fused representation is then merged with the original pooled input through a residual connection and layer normalization:

$$(7.8) \quad o_\tau = \text{LayerNorm}(h + \text{Dropout}(f_\tau)).$$

The final representation o_τ is fed to the task-specific classification head ($\text{Linear}(d, d_{\text{head}}) \rightarrow \text{GELU} \rightarrow \text{Dropout} \rightarrow \text{Linear}(d_{\text{head}}, 1)$ with $d_{\text{head}} = 1440$), which produces a scalar logit. The probability is then obtained via the sigmoid: $P_\tau(x) = \sigma(\text{head}_\tau(o_\tau))$.

Training procedure. The CTAF model is trained in alternating rounds. Each training round comprises e_{tox} epochs over the toxicity dataset followed by e_{vir} epochs over the virulence dataset, with the backbone kept frozen throughout. Two separate AdamW optimizers with per-group learning rates are used: task-specific parameters (the expert MLP, the task query projection, and the classification head) take task-specific learning rates $(\eta_{\text{tox}}, \eta_{\text{vir}})$, while shared parameters (the backbone pooler, the cross-attention key and value projections, the gate network, and the layer norm) take the average rate $(\eta_{\text{tox}} + \eta_{\text{vir}})/2$. Training employs BCEWithLogitsLoss with class-frequency-based positive weighting to compensate for dataset imbalance, applies gradient clipping at 1.0, and uses a linear warmup followed by cosine annealing. Model selection is based on the best combined F1 score $\frac{1}{2}(F1_{\text{tox}} + F1_{\text{vir}})$ on the validation set. The full forward pass for task τ is:

$$(7.9) \quad P_\tau(x) = \sigma\left(\text{head}_\tau\left(\text{LayerNorm}(h + \gamma c + (1-\gamma)e_\tau)\right)\right).$$

Once trained, the CTAF model is frozen and serves as a reward oracle during PPO training, with $r_{\text{toxic}}(x) = 1 - P_{\text{tox}}(x)$ and $r_{\text{vir}}(x) = 1 - P_{\text{vir}}(x)$.

7.3.3. PPO with KL Penalty

We fine-tune π_θ with PPO [161], augmented by a token-level KL penalty against a frozen reference π_{ref} . At each iteration, N sequences are sampled under the current policy (the

old policy snapshot $\pi_{\theta_{\text{old}}}$), and their per-token log probabilities and value estimates are stored in a rollout buffer.

Per-token reward structure. Protein quality measures (pLDDT, toxicity, and virulence) can only be evaluated once a full sequence has been produced, which makes this a sparse terminal-reward problem. We therefore place the biological reward exclusively on the final token, while a KL penalty supplies a dense shaping signal at every intermediate position:

$$(7.10) \quad r_t = \underbrace{-\beta \left(\log \pi_{\theta}(x_t | x_{<t}) - \log \pi_{\text{ref}}(x_t | x_{<t}) \right)}_{\text{KL shaping at every token}} + \underbrace{r(x) \mathbf{1}[t = T]}_{\text{biological reward at } t=T \text{ only}},$$

where β is the KL coefficient. At any position t , the KL term is negative when the policy drifts away from the reference (penalizing drift) and slightly positive when the policy moves back toward the reference (encouraging conservatism). Without this intermediate signal, every token at $t < T$ would receive zero reward, and credit assignment from the terminal biological score would rest entirely on GAE propagation, making learning substantially harder. The KL penalty also discourages reward hacking: sequences that obtain high biological scores via linguistically degenerate patterns (for example, repetitive or unnatural amino acid compositions) are penalized through their large divergence from π_{ref} .

Advantage estimation and policy update. Advantages $\hat{A}_t$ and returns G_t are estimated through GAE [160] with $\gamma=1.0$ and $\lambda=0.95$, propagating the terminal biological reward backwards through the sequence so that tokens that contributed to a good outcome are credited with appropriately discounted weight. The policy is then updated for E epochs

over the fixed buffer using the clipped surrogate objective:

(7.11)

$$\mathcal{L}(\theta, \phi) = -\mathbb{E}_t \left[\min \left(\rho_t \hat{A}_t, \text{clip}(\rho_t, 1-\varepsilon, 1+\varepsilon) \hat{A}_t \right) \right] + c_v \mathbb{E}_t \left[(V_\phi - G_t)^2 \right] - c_e \mathbb{E}_t [H(\pi_\theta)],$$

where $\rho_t = \pi_\theta / \pi_{\theta_{\text{old}}}$ is the probability ratio and $\varepsilon=0.1$ is the clip threshold. Each of the three terms plays a distinct role. The value loss $(V_\phi - G_t)^2$ trains the value head to accurately predict future returns, which in turn stabilizes GAE advantage estimates ($c_v=0.5$ downweights it relative to the policy loss because V_ϕ is a supporting objective). The entropy bonus $H(\pi_\theta)$ keeps the policy from collapsing onto a narrow set of high-reward sequence motifs ($c_e=0.01$ provides light diversity regularization without overpowering the reward signal). Finally, the clip on ρ_t prevents any single update from shifting the policy by more than 10% in probability ratio.

7.3.4. Length Bonus and Short-Sequence Penalty

Protein functionality is tightly coupled to sequence length. To bias the model toward generating full-length, biologically plausible proteins, we introduce an adaptive length reward. For sequences shorter than a minimum threshold $L_{\min}$ (set to 400 residues), we apply a hard penalty:

$$(7.12) \quad r_{\text{len}}(x) = -1 + \frac{|x|}{L_{\min}}, \quad |x| < L_{\min},$$

which interpolates linearly between -1 (an empty sequence) and 0 (at the threshold). For valid sequences ($|x| \geq L_{\min}$), a soft bonus is applied instead:

$$(7.13) \quad r_{\text{len}}(x) = \min\left(\frac{|x|}{\bar{L}}, 1.0\right) \times 0.2, \quad |x| \geq L_{\min},$$

where $\bar{L}$ is the running mean sequence length tracked across rollouts and is initialized to 512. The bonus is capped at 0.2 to prevent it from dominating the biological reward signals, and it decays naturally as the policy learns to generate longer sequences.

7.3.5. Adaptive Multi-Objective Weight Calibration via TCS-MGDA

The three biological reward objectives are not always aligned: structural quality may demand sequence properties that differ from those required for toxicity or virulence avoidance, and these conflicts evolve dynamically over the course of training. Fixed reward weights cannot follow this shifting geometry, as Table 7.2 shows. We propose *TCS-MGDA*, a novel combination of gradient-based MGDA weight adaptation [42, 162] with a Task-Conflict Saliency (TCS) mechanism that pinpoints, and concentrates on, the parameters where objectives genuinely disagree. This addresses gram matrix dilution of standard MGDA in the protein language model fine-tuning setting. Standard MGDA computes the $K \times K$ Gram matrix $G_{ij} = \hat{g}_i^\top \hat{g}_j$ over all $D \sim 10^8$ parameters. In practice, the overwhelming majority of parameters are either inactive (with near-zero gradient) or *shared* across objectives (parameters on which all objectives agree). Including these parameters dilutes the conflict signal in G , leading MGDA to underestimate inter-objective disagreement and to produce weights that are insufficiently differentiated. Our TCS-MGDA addresses both limitations through the following pipeline.

Gram matrix computation. At each calibration iteration (every τ steps), we extract the per-sample gradient $\nabla_n \triangleq \nabla_{\theta} \log \pi_{\theta}(x_n) \in \mathbb{R}^D$ by reusing the PPO rollout forward pass with gradients enabled. The per-objective policy gradient is the reward-weighted average of these per-sample gradients:

$$(7.14) \quad \hat{g}_k = \frac{1}{N} \sum_{n=1}^N R_k(x_n) \nabla_n \in \mathbb{R}^D.$$

The MGDA QP requires the $K \times K$ Gram matrix $G_{ij} = \hat{g}_i^{\top} \hat{g}_j$. Computing this naively would entail materializing all K vectors $\hat{g}_k$ explicitly, each of dimension $D \sim 10^8$, at a memory cost of $\mathcal{O}(KD)$ (~ 1.2 GB for $K=3$). We avoid this by algebraically rearranging the dot product:

$$(7.15) \quad \begin{aligned} G_{ij} &= \hat{g}_i^{\top} \hat{g}_j = \left(\frac{1}{N} \sum_{n=1}^N R_i(x_n) \nabla_n \right)^{\top} \left(\frac{1}{N} \sum_{m=1}^N R_j(x_m) \nabla_m \right) \\ &= \frac{1}{N^2} \sum_{n=1}^N \sum_{m=1}^N R_i(x_n) R_j(x_m) \nabla_n^{\top} \nabla_m \\ &= \frac{1}{N^2} \mathbf{r}_i^{\top} P \mathbf{r}_j, \end{aligned}$$

where $\mathbf{r}_k \in \mathbb{R}^N$ collects the per-sample rewards for objective k , and $P \in \mathbb{R}^{N \times N}$ is the gradient inner-product matrix with $P_{nm} = \nabla_n^{\top} \nabla_m$. The key insight is that P depends only on pairwise dot products between the N sample gradients and therefore contains just $N^2 = 64$ scalars for $N=8$. Once P has been computed, any entry G_{ij} becomes a cheap vector product over N dimensions rather than D . The K large vectors $\hat{g}_k$ are never formed, so the dominant memory cost drops from $\mathcal{O}(KD)$ to $\mathcal{O}(N^2)$. A small regularization ϵI is added to G for numerical stability.

Task-Conflict Saliency (TCS). To address spurious alignment, we center the rewards before computing the per-objective gradients. Let $\nabla_n^{(p)} \in \mathbb{R}$ denote the p -th scalar component of the gradient vector $\nabla_n \in \mathbb{R}^D$, *i.e.*, the gradient with respect to the p -th model parameter. The centered per-objective gradient component for objective k at parameter p is:

$$(7.16) \quad s_k^{(p)} = \frac{1}{N} \sum_{n=1}^N \left(R_k(x_n) - \bar{R}_k \right) \nabla_n^{(p)},$$

with $\bar{R}_k$ denoting the per-objective mean reward. Centering removes the shared positive mass that makes all objectives appear aligned, exposing instead which sequences each objective prefers *more* versus *less* than average.

To handle the extreme sparsity of gradient vectors (with $\sim 10^{-7}$ magnitude concentrated in fewer than 1% of parameters), we cannot simply measure variance across objectives, since it collapses to zero almost everywhere because of the dominant inactive mass. We instead apply a two-stage normalization to obtain a scale-free conflict signal.

Stage 1: L2 normalization eliminates magnitude differences between objectives so that conflict is reflected in direction rather than scale:

$$(7.17) \quad \tilde{s}_k^{(p)} = \frac{s_k^{(p)}}{\|s_k\|_2 + \epsilon}.$$

Stage 2: per-parameter z-score across objectives renders the signal scale-free with respect to gradient sparsity. For each parameter p , it asks whether the objectives disagree on the update direction *relative to their overall variation across parameters*, independently

of absolute magnitude:

$$(7.18) \quad z_k^{(p)} = \frac{\tilde{s}_k^{(p)} - \mu^{(p)}}{\sigma^{(p)} + \epsilon}, \quad \mu^{(p)} = \frac{1}{K} \sum_k \tilde{s}_k^{(p)}, \quad \sigma^{(p)} = \sqrt{\frac{1}{K} \sum_k \left(\tilde{s}_k^{(p)} - \mu^{(p)} \right)^2}.$$

The TCS score for each parameter is then defined as the mean absolute z-score across objectives:

$$(7.19) \quad \text{TCS}(p) = \frac{1}{K} \sum_k |z_k^{(p)}|,$$

with the overall conflict level given by $\overline{\text{TCS}} = \frac{1}{|\mathcal{A}|} \sum_{p \in \mathcal{A}} \text{TCS}(p)$, where $\mathcal{A}$ is restricted to active parameters (with magnitude $> 10^{-4}$ of the global maximum) so as to exclude the sea of inactive zeros.

Conflict-focused Gram matrix. To address dilution, the top- M parameters by TCS score are selected in $\mathcal{O}(D)$ time using a partial sort:

$$(7.20) \quad \mathcal{M} = \text{top-}M_p \text{ TCS}(p), \quad M = \max(M_{\min}, \lfloor \xi D \rfloor),$$

with $\xi = 0.05$ and $M_{\min} = 1000$. A conflict-focused Gram matrix $G^{\mathcal{M}}$ is then formed over $\mathcal{M}$ alone, sharpening the inter-objective disagreement signal that MGDA acts on.

Adaptive Gram matrix selection. Rather than committing to either G or $G^{\mathcal{M}}$ globally, we choose adaptively at each calibration step based on the cosine gap between the two matrices:

$$(7.21) \quad \Delta\rho = \rho(G) - \rho(G^{\mathcal{M}}), \quad \rho(G) = \frac{2}{K(K-1)} \sum_{i < j} \frac{G_{ij}}{\sqrt{G_{ii}G_{jj}}},$$

$$(7.22) \quad G^* = \begin{cases} G^{\mathcal{M}} & \Delta\rho > \lambda \quad (\text{objectives conflict}), \\ G & \text{otherwise} \quad (\text{objectives aligned}), \end{cases}$$

where λ interpolates between always using $G^{\mathcal{M}}$ ($\lambda = 0$) and standard full-gradient MGDA ($\lambda = \infty$). When the objectives genuinely conflict (i.e., $\Delta\rho$ is large), $G^{\mathcal{M}}$ supplies a sharper signal for MGDA. When they are locally aligned (i.e., $\Delta\rho$ is small), G provides a more stable estimate.

QP solve and EMA smoothing. Given G^* , we solve:

$$(7.23) \quad \mathbf{w}^* = \arg \min_{\mathbf{w} \in \Delta^{K-1}} \mathbf{w}^\top G^* \mathbf{w}.$$

For $K = 3$, an analytic solver enumerates all seven simplex candidates (3 vertices, 3 edge optima, and 1 interior Karush–Kuhn–Tucker (KKT) point [82]), returning the exact global minimum without iteration. The minimum norm $\|\mathbf{d}^*\|^2 = \mathbf{w}^{*\top} G^* \mathbf{w}^*$ certifies Pareto-stationarity whenever it falls below a tolerance δ . The solution is then smoothed via EMA to suppress weight oscillations:

$$(7.24) \quad \mathbf{w}_{t+1} \leftarrow \text{proj}_\Delta(\alpha \mathbf{w}^* + (1 - \alpha) \mathbf{w}_t),$$

with $\alpha = 0.3$. The updated weights immediately recompute the rollout buffer rewards and GAE advantages, which keeps the PPO update aligned with the current Pareto-descent direction. The complete pipeline is laid out in Algorithm 2.

Algorithm 2: TCS-Adaptive Gradient MGDA for Multi-Objective PPO

Input : Policy π_θ , reference π_{ref} , frozen CTAF and ESMFold reward models,
 calibration interval τ , EMA α , TCS threshold λ_{TCS} , top fraction ξ ,
 terminal token set $\mathcal{T}$, minimum length $L_{\min}$

Initialize $\mathbf{w}_0 \leftarrow \mathbf{1}/K$

for iteration $t = 1, 2, \dots$ **do**

cal $\leftarrow (t \bmod \tau = 0) \vee (t = 1)$

Sample $\{x_n\}_{n=1}^N \sim \pi_\theta(\cdot | c)$, suppressing tokens in $\mathcal{T}$ until $|x_n| \geq L_{\min}$; enable
 gradients if cal

Compute per-objective rewards $R_k(x_n)$ and apply soft length bonus

if cal **then**

Compute $\nabla_n = \nabla_\theta \log \pi_\theta(x_n)$ for each n ▷ reuse rollout backward

Compute $P_{nm} = \nabla_n^\top \nabla_m$ for all n, m ▷ $N \times N$ inner products

Compute full Gram $G \leftarrow \frac{1}{N^2} R^\top P R + \epsilon I$ via Eq. (7.15)

Compute centered per-objective gradients S via Eq. (7.16)

L2-normalize rows of S ; apply per-parameter z-score; compute $\text{TCS}(p)$ via
 Eq. (7.19)

$\mathcal{A} \leftarrow$ active parameters with relative magnitude above 10^{-4}

$\mathcal{M} \leftarrow$ top- M parameters by TCS score ▷ linear-time partial sort

Compute conflict Gram $G^\mathcal{M}$ and cosine gap $\Delta\rho \leftarrow \rho(G) - \rho(G^\mathcal{M})$

if $\Delta\rho > \lambda_{\text{TCS}}$ **then**

$G^* \leftarrow G^\mathcal{M}$

else

$G^* \leftarrow G$

$\mathbf{w}^* \leftarrow \arg \min_{\mathbf{w} \in \Delta^{K-1}} \mathbf{w}^\top G^* \mathbf{w}$ ▷ analytic for $K=3$

$\mathbf{w}_t \leftarrow \text{proj}_\Delta(\alpha \mathbf{w}^* + (1 - \alpha) \mathbf{w}_{t-1})$ ▷ EMA + simplex projection

Recompute buffer rewards and GAE advantages with $\mathbf{w}_t$

Update π_θ with PPO using buffer with $\mathbf{w}_t$

7.4. Experiments

7.4.1. Datasets

We assess our approach on a toxicity dataset and a virulence prediction dataset.

7.4.1.1. Toxicity Dataset. We use the protein toxicity benchmark released by Zhu et al. [230], which is built from UniProt release 2024_03. Toxic sequences are obtained

through the KW-0800 keyword query, and non-toxic controls come from the complementary query with allergens excluded. After non-standard residues and duplicates are removed, the negative pool is undersampled to a 1:2 toxic-to-non-toxic ratio, and CD-HIT-2D is then run at a 40% identity threshold to suppress near-duplicates across splits.

The dataset is partitioned into three disjoint splits: training (4,900 toxic / 9,800 non-toxic), validation (78 / 853), and test (112 / 1,735). The deliberate class imbalance in the validation and test splits is meant to mirror realistic screening conditions, in which toxic proteins are rare and the reward model must keep recall high on the minority class. As Section 7.4.3 demonstrates, CTAF satisfies this requirement.

7.4.1.2. Virulence Prediction. Because no standardized public benchmark exists for protein-level virulence classification, we assemble our own dataset from curated primary sources, treating experimentally verified virulence factors as positives and a stringently filtered set of non-virulence proteins as negatives.

Positive Dataset. The positive examples are taken from the Virulence Factor Database (VFDB) [105], a repository of experimentally verified bacterial virulence factors. We processed the entire protein set, parsing VFDB headers to extract VFG identifiers, gene names, functional annotations, and taxonomy, and then deduplicated by exact sequence identity. The resulting positive set contains 29,042 unique sequences spanning 878 bacterial species, with a strong representation of clinically significant pathogens such as *Legionella pneumophila*, *Pseudomonas aeruginosa*, *Salmonella* Typhimurium, and *Mycobacterium tuberculosis*.

Negative Dataset. Following [158], we build a rigorous negative set by sampling functionally diverse proteins from complete UniProt reference proteomes [178] of 21

phylogenetically diverse bacterial species, covering both Gram-negative and Gram-positive organisms. In total, 59,298 candidate proteins were collected and reduced to 58,399 after deduplication.

To prevent label leakage and ensure that the classifier picks up virulence-specific features, we applied 40 keyword exclusion patterns covering toxins, adherence factors, secretion systems, immune evasion factors, motility and chemotaxis components, biofilm and quorum-sensing regulators, iron acquisition systems, and antibiotic resistance determinants (the full list is provided in the Appendix). Sequences whose MD5 hash matched any VFDB entry were also discarded. The remaining sequences were then filtered to a length range of 20 to 3,000 residues with $< 5\%$ ambiguous characters.

Balancing and Stratification. For compatibility with protein language models, sequences exceeding 1024 residues were excluded from both classes, retaining 93.3% (27,083/29,042) of VFDB. We then carried out proportional stratified sampling by length decile so that the biophysical distributions across classes are matched, while functional diversity within each stratum is preserved.

The resulting balanced dataset contains 27,083 positive and 27,083 negative sequences, with closely matched length distributions (positive: 380.1 ± 209.2 ; negative: 370.8 ± 197.9 residues). The negatives span a broad set of functional categories, predominantly unassigned function (12,445), cellular processes (4,655), hypothetical proteins (3,576), genetic information processing (3,483), and signal transduction (2,751), drawn from 20 bacterial species.

7.4.2. Implementation Details

7.4.2.1. Models. The policy is ProGen2-medium (~ 151 M parameters) [131], kept in FP32 throughout training. The frozen reference is the same checkpoint loaded in FP16 on a dedicated GPU. The CTAF reward model is built on an ESM2-650M backbone [101] with task-specific expert MLP modules ($d_{\text{exp}} = 1024$, head hidden dimension $d_{\text{head}} = 1440$) and $H_a = 4$ cross-task attention heads, and is trained separately on the toxicity and virulence classification datasets. The pLDDT reward is produced by ESMFold [102] running inside a cached scoring server.

7.4.2.2. Training Configuration.

CTAF experiments. All models, both our CTAF and the baselines, are trained under identical optimization settings to ensure a fair comparison. We use a binary cross-entropy loss with logits (`BCEWithLogitsLoss`) and class-frequency-balanced positive weighting to cope with class imbalance. Optimization is carried out with AdamW using a weight decay of 0.01, a batch size of 64, and a maximum sequence length of 1,024 residues. The learning rate follows a linear warmup followed by cosine annealing, and gradients are clipped at an ℓ_2 norm of 1.0 for stability. For every model, we run a grid search over the number of training epochs in $\{10, 20, 30\}$ and over the peak learning rate in $\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$, picking the configuration that maximizes the combined validation F1 score, $\frac{1}{2}(F1_{\text{tox}} + F1_{\text{vir}})$.

Alignment experiments. The same training settings are applied to both our *SafePro* and the baselines. We use AdamW with a grid search over policy learning rates in $[1 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-7}, 5 \times 10^{-7}]$ and value-head learning rates in $[1 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-7}, 5 \times 10^{-7}]$, with weight decay 0.01 and cosine annealing after a 5% linear warmup. PPO uses clip $\epsilon = 0.1$, value coefficient 0.5, entropy coefficient 0.01, KL coefficient $\beta = 0.3$, and GAE

with $\lambda = 0.95$ and $\gamma = 1.0$. Each rollout batch consists of $N = 8$ sequences of up to 512 tokens, with a minimum sequence length of $L_{\min} = 400$. MGDA calibration runs every $\tau = 10$ iterations with EMA $\alpha = 0.95$, Gram regularization $\epsilon = 10^{-6}$, TCS top fraction $\rho = 0.05$, and TCS minimum parameter count $M_{\min} = 1000$.

Computational overhead. On calibration iterations, MGDA introduces one extra `.backward()` call per rollout sample by reusing the existing forward pass, plus the $N \times N$ gradient inner-product matrix ($N = 8$, which is negligible) and the linear-time TCS partial sort over D parameters. The policy is kept in FP32 specifically on the calibration pass so that `.backward()` writes into the true parameter `.grad` buffers rather than scaled mixed-precision copies; the reference model stays in FP16 throughout. In total, MGDA adds less than 5% of per-iteration wall time. All experiments are run on three NVIDIA RTX A6000 GPUs with 48 GB of memory each. Training the CTAF reward model takes roughly 9 GPU-hours, and the full PPO alignment stage takes roughly 170 GPU-hours.

7.4.3. CTAF Reward Model Evaluation

We evaluate the Cross-Task Attentive Fusion (CTAF) reward model against two families of baselines on the held-out test splits of both the toxicity and virulence datasets. All models share the same frozen ESM2-t33 backbone and pooler ($\text{CLS} \rightarrow \tanh$) and follow the training recipe described in Section. 7.4.2. We compare CTAF with the following baselines: *(i) Separate classifiers.* Two independently trained models, each consisting of a dedicated classification head on top of the frozen backbone. This is the standard single-task upper bound, which incurs no multi-task tax but doubles the inference cost. *(ii) Shared-block multi-task.* A single frozen backbone feeds a shared MLP trunk of depth $L \in \{1, 3, 5, 8\}$

blocks (each block: $\text{Linear}(d, 1024) \rightarrow \text{GELU} \rightarrow \text{Dropout} \rightarrow \text{Linear}(1024, d)$), followed by two task-specific heads. This represents the canonical hard parameter-sharing baseline for multi-task learning, and varying L tests whether the shared representation simply needs more capacity to resolve the toxicity versus virulence tension.

Table 7.1 reports test-set performance on both tasks. CTAF delivers the strongest results overall and, importantly, *outperforms the separate single-task classifiers* on every aggregate metric: toxic F1 rises from 0.8035 to 0.8619 (+5.84), toxic MCC from 0.7907 to 0.8543 (+6.36), virulence F1 from 0.9158 to 0.9426 (+2.68), and virulence MCC from 0.8295 to 0.8839 (+5.44). This is the key takeaway: joint training with CTAF is not just a computationally efficient compromise relative to separate models, but is in fact strictly better, indicating that the two tasks carry complementary information that a well-designed cross-task mechanism can exploit.

The shared-block baselines tell a rather different story. A single shared block ($L=1$) is the strongest hard-sharing variant on toxicity, yet every depth tested underperforms even the separate single-task classifiers on both toxic F1 and virulence F1. Increasing depth from 1 to 8 blocks does not help: toxic F1 fluctuates between 0.75 and 0.79, while virulence F1 plateaus near 0.88. The shared-block models attain high recall (0.93 to 0.95) at the expense of precision (0.62 to 0.69), whereas CTAF reaches recall of 0.9196 at precision 0.8110, giving a far more balanced operating point. This pattern reflects negative transfer in the shared trunk: the class-balanced virulence objective (4063:4063) pushes the representation toward treating positives as common, which inflates false positives on toxicity. By contrast, CTAF’s task-private experts break this coupling, and the adaptive gate (with average gate values of 0.1994 for toxic and 0.0600 for virulence) confirms that

Table 7.1. Test-set performance of CTAF against single-task and shared-block multi-task baselines on the toxicity and virulence datasets. **Bold** = best per column; underline = second-best.

Model	Toxicity			Virulence		
	F1	MCC	AUROC	F1	MCC	AUROC
Separate (single-task)	<u>0.8035</u>	<u>0.7907</u>	<u>0.9880</u>	<u>0.9158</u>	<u>0.8295</u>	<u>0.9708</u>
Shared-block, $L=1$	0.7909	0.7853	0.9864	0.8756	0.7519	0.9479
Shared-block, $L=3$	0.7544	0.7534	0.9857	0.8837	0.7677	0.9528
Shared-block, $L=5$	0.7721	0.7686	0.9865	0.8791	0.7571	0.9489
Shared-block, $L=8$	0.7518	0.7509	0.9828	0.8782	0.7551	0.9444
CTAF (ours)	0.8619	0.8543	0.9884	0.9426	0.8839	0.9824

the model learns to remain mostly task-private and to borrow from the other task only on a selective basis. The fact that the gate settles lower for virulence than for toxicity suggests that the virulence classifier benefits less from toxic-side features than vice versa. This is consistent with the biological asymmetry that toxins are a subset of virulence factors, so toxicity information is more specific while virulence information is broader and less directly usable as a cross-task signal.

7.4.4. Gram Matrix Dilution

Section 7.3.5 motivated TCS by arguing that computing the MGDA Gram matrix over all $D \sim 10^8$ parameters dilutes the inter-objective conflict signal, since the vast majority of parameters are either inactive or shared across objectives. We now provide direct empirical evidence for this claim on the **progen2-medium** policy ($D = 764,803,616$ trainable parameters) by inspecting the raw per-sample log-probability gradients $\nabla_n = \nabla_{\theta} \log \pi_{\theta}(x_n)$ that constitute the building block of every per-objective gradient $\hat{g}_k$ in Equation (7.15). We sample $N = 32$ protein sequences, compute each ∇_n with a single backward pass over

the generated tokens, and partition the parameter vector into 51 contiguous blocks of approximately 15M parameters each. For every sample and block pair, we record the block-restricted magnitude $\|\nabla_n^{(b)}\|_2$.

Mean-aggregated gradients are dominated by a single block. Figure 7.1 plots the mean block magnitude across samples. The output-projection block near index 49 is roughly an order of magnitude larger than any backbone block, and the backbone blocks themselves form a nearly flat plateau at about 10% of the peak. When this mean gradient is plugged into the standard MGDA inner product $G_{ij} = \hat{g}_i^\top \hat{g}_j$, two failure modes appear that together produce dilution. First, the inactive 47.9% contributes zero to every G_{ij} but still counts toward the normalization when the Gram matrix is converted to a cosine, which shrinks the effective conflict signal. Second, the output-projection block dominates the remaining mass and is shared across all three objectives, so G_{ij} ends up driven by a region in which the objectives trivially agree. The net effect is a Gram matrix whose off-diagonal cosines remain in a narrow positive range regardless of the underlying task disagreement, which is precisely the behavior that motivates restricting MGDA to the top- M TCS-selected parameters in Equation (7.22).

Per-sample gradients are sparse and sample-specific. Figure 7.3 visualizes the per-sample gradient magnitude across parameter blocks, with each row normalized so that bright regions indicate where each sample’s gradient concentrates. Two patterns emerge. First, within any given sample, the signal is confined to a small number of blocks and the rest is essentially dark. Second, the bright columns are not aligned across rows: samples $n=5$, $n=9$, and $n=13$ spread gradient broadly across the backbone, while $n=3$, $n=7$, and $n=16$ to $n=25$ activate only the final output-projection block (the tall yellow

stripe near block 49) and leave the backbone essentially untouched. Hence the active parameters are *sequence-dependent* rather than a fixed dead set that could be pruned offline. This combination of sparsity and sample specificity is exactly what causes the standard full-parameter Gram matrix to dilute the inter-objective conflict signal.

7.4.5. Alignment Experiments

We evaluate the full alignment pipeline (CTAF reward modeling, PPO fine-tuning, and TCS-MGDA adaptive weight calibration) on ProGen2-medium [131]. The training settings are detailed in Section. 7.4.2. For each configuration, we draw $N=1,000$ sequences from the aligned policy and report structural quality (mean pLDDT from ESMFold [101] and the High Quality Rate, defined as the fraction of sequences with $\text{pLDDT} \geq 70$), as well as the estimated toxicity P_{tox} and estimated virulence P_{vir} produced by our CTAF.

7.4.5.1. Reward Model: CTAF vs. Independent Classifiers. To isolate the contribution of the reward model, we first compare PPO alignment driven by our CTAF reward against PPO driven by two independently trained classifiers. Both reward models are evaluated under a grid of fixed objective weights $(w_{\text{tox}}, w_{\text{vir}}, w_{\text{pLDDT}})$ that spans the uniform allocation, the three balanced-pair configurations, and the three single-objective corners of the simplex. The results are reported in the upper block of Table 7.2.

CTAF achieves a higher composite score than the independent-classifier reward in five out of six comparable settings. The biggest gains concentrate on the two-objective configurations that pair a single safety task with structural quality: +2.14 at $(0.5, 0, 0.5)$ and +1.20 at $(0, 0.5, 0.5)$. These are precisely the regimes where the reward model has to discriminate between sequences that trade off partially aligned properties, and

where a classifier that shares *selective* signal across toxicity and virulence offers the largest advantage. On the uniform allocation (0.3, 0.3, 0.4) and on the balanced-safety configuration (0.5, 0.5, 0), CTAF retains a moderate margin (+0.77 and +1.07, respectively). On the two single-objective corners, the two reward models are close: CTAF wins marginally at (0, 1, 0) (+0.17), while the independent-classifier reward is marginally preferred at (1, 0, 0) (+0.68). This pattern is consistent with CTAF’s design, in which the adaptive gate learns to invoke cross-task signal only when it is beneficial, so when the alignment objective collapses to a single task the cross-task channel contributes little and can introduce mild noise relative to a task-specific classifier. Across the entire grid, CTAF never collapses to a degenerate policy and maintains competitive structural quality, which is the prerequisite property for a reward oracle intended to drive multi-objective alignment.

Two further observations motivate the next subsection. First, *no fixed weight vector simultaneously optimizes all three objectives*: the highest mean pLDDT (55.02, CTAF (0.3, 0.3, 0.4)), the lowest P_{tox} (0.0492, CTAF (0.5, 0.5, 0)), and the lowest P_{vir} (0.6443, CTAF (0.3, 0.3, 0.4)) are not realized jointly by any single configuration. Second, the composite score varies by more than three points across the grid, and the ranking of configurations is sensitive to the weight choice. Choosing a fixed weight vector *a priori* therefore demands knowledge of the Pareto geometry, which is exactly the problem that adaptive weight calibration is designed to solve.

7.4.5.2. Adaptive Weight Calibration.

Setup. We evaluate our adaptive weight-calibration strategy on top of the CTAF reward model under three regimes: (i) the fixed reward-weight configurations (w_{tox} , w_{vir} , w_{pLDDT}) from the upper block of Table 7.2, (ii) standard MGDA without conflict detection,

and (iii) our full TCS-MGDA, which switches between the full Gram matrix G and the conflict-focused matrix $G^{\mathcal{M}}$ based on a threshold λ (Eq. (7.22)). All runs share the same PPO hyperparameters, generator backbone, and evaluation protocol; only the weight-calibration mechanism differs. We use AdamW with a grid search over policy learning rates in $[1 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-7}, 5 \times 10^{-7}]$ and value-head learning rates in $[1 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-7}, 5 \times 10^{-7}]$, with weight decay 0.01 and cosine annealing after a 5% linear warmup. Full configuration details are in Appendix 7.4.2. We report Mean pLDDT, $P(\text{toxic})$, $P(\text{virulent})$, and the composite score defined in Table 7.2.

TCS-MGDA improves over fixed weights and standard MGDA. The lower block of Table 7.2 shows that standard MGDA on top of CTAF reaches a composite score of 60.45, on par with the best fixed-weight configurations but with no real edge over a well-tuned linear baseline. TCS-MGDA, by contrast, attains 66.86, a +6.41 absolute gain over MGDA and +5.21 over the strongest fixed-weight setting (61.65 at $(0.3, 0.3, 0.4)$). The improvement is driven by sharp drops in both toxicity ($0.0523 \rightarrow 0.0139$) and virulence ($0.6926 \rightarrow 0.5407$), while pLDDT is preserved and slightly improved ($55.83 \rightarrow 56.04$). Identifying and isolating conflicting gradient directions therefore lets the optimizer push on safety without giving up structural quality, which fixed-weight schemes consistently fail to do.

Ablation on the conflict detection threshold λ . Table 7.3 reports TCS-MGDA’s sensitivity to λ , the threshold that controls when the algorithm transitions from G to $G^{\mathcal{M}}$. Three regimes emerge. With *large* thresholds ($\lambda \in \{0.8, 0.5, 0.2\}$), conflict detection rarely fires and TCS-MGDA collapses onto standard MGDA: composite scores cluster around 60.4, statistically indistinguishable from the no-TCS baseline (60.45). With *small*

but non-zero thresholds ($\lambda \in \{0.002, 0.001\}$), the conflict-focused projection is engaged often enough to reshape the update direction, producing the highest composite scores in the sweep (66.86 and 64.36 respectively). The best operating point is $\lambda = 0.002$: it simultaneously yields the lowest $P(\text{toxic})$ (0.0139) and $P(\text{virulent})$ (0.5407) while keeping pLDDT competitive (56.04). Pushing further to $\lambda = 0$, where the conflict-focused matrix is always used, drops pLDDT to 51.17 and composite to 64.03, indicating that always-on projection over-constrains the update. Overall, performance peaks in a clear sweet spot near $\lambda \approx 10^{-3}$: large enough to trigger conflict resolution where gradients genuinely disagree, but small enough to leave non-conflicting updates alone. The fact that this region spans two orders of magnitude ($\lambda \in [0.001, 0.002, 0.003]$) suggests that careful per-task tuning of the threshold is unnecessary in practice.

Ablation on Length Bonus and Short-Sequence Penalty. To assess the role of $r_{\text{len}}(x)$ in stabilizing sequence-length dynamics, we compare two otherwise identical configurations: **w/ length bonus** uses r_{len} as defined in Section 7.3.4, while **w/o length bonus** sets $r_{\text{len}}(x) \equiv 0$. Both runs use $L_{\text{max}} = 512$, $L_{\text{min}} = 400$, and identical initialization for 1,000 PPO iterations. Figure 7.4 tracks the mean generated length $\bar{L}_t$, evaluated every 10 iterations.

With r_{len} active, $\bar{L}_t$ remains saturated in $[490, 511]$ throughout training. Without it, $\bar{L}_t$ drifts from roughly 480 at iteration 600 down to about 250 by iteration 700, and decays toward a single-digit floor by iteration 900, broken only by occasional spikes of 20 to 40 residues. We trace this collapse to reward hacking: r_{toxic} and r_{vir} award maximum reward to trivially short outputs that lie outside the CTAF classifier’s training distribution and that it confidently labels as non-toxic and non-virulent, while r_{plddt} and the KL penalty

Table 7.2. Evaluation results across reward-weight configurations, comparing PPO alignment driven by Independent Classifiers and by our CTAF reward model. The composite score is computed as $\frac{1}{3}[\text{Mean pLDDT} + (1 - P_{\text{toxic}}) \times 100 + (1 - P_{\text{virulent}}) \times 100]$, combining structural quality, toxicity avoidance, and virulence avoidance into a single metric; higher is better. The upper block reports fixed reward-weight configurations denoted $(w_{\text{tox}}, w_{\text{vir}}, w_{\text{pLDDT}})$. The lower block reports adaptive weight-calibration methods operating on top of CTAF: standard MGDA without the TCS mechanism, and our *SafePro*.

Setting	Mean pLDDT		P(toxic)		P(virulent)		Composite	
	Ind.	CTAF	Ind.	CTAF	Ind.	CTAF	Ind.	CTAF
Base Model	54.31	—	0.0635	—	0.6525	—	60.90	—
(0.3, 0.3, 0.4)	54.46	55.02	0.0494	0.0564	0.6687	0.6443	60.88	61.65
(0, 0.5, 0.5)	52.13	53.73	0.0577	0.0500	0.6926	0.6804	59.03	60.23
(0.5, 0, 0.5)	50.89	53.57	0.0622	0.0564	0.7025	0.6710	58.14	60.28
(0.5, 0.5, 0)	53.42	54.62	0.0471	0.0492	0.6863	0.6639	60.03	61.10
(1, 0, 0)	54.66	54.08	0.0547	0.0495	0.6557	0.6753	61.21	60.53
(0, 1, 0)	54.73	54.48	0.0531	0.0524	0.6676	0.6607	60.89	61.06
(0, 0, 1)	53.50	—	0.0519	—	0.6664	—	60.56	—
MGDA (no TCS)	55.83		0.0523		0.6926		60.45	
<i>SafePro</i> (ours)	56.04		0.0139		0.5407		66.86	

against π_{ref} slow the drift but do not stop it. PPO with a KL penalty therefore resists but cannot prevent collapse to degenerate lengths, whereas the 0.2 cap on r_{len} , although small relative to the biological rewards, is sufficient to anchor $\bar{L}_t$ near L_{max} . This is consistent with the role of r_{len} as a regularizer rather than a dominant reward.

7.5. Conclusion

We have introduced *SafePro*, a multi-objective alignment framework for protein language models that jointly optimizes structural foldability, toxicity avoidance, and virulence avoidance. Each of its two components addresses a concrete limitation that arises when LLM alignment recipes are transferred directly to protein generation. CTAF replaces

Table 7.3. *SafePro* sensitivity analysis: effect of the conflict detection threshold λ on protein generation quality. All runs use identical hyperparameters except for λ , which controls when the algorithm switches from the full Gram matrix G to the conflict-focused matrix $G^{\mathcal{M}}$ (Eq. (7.22)). The composite score follows the same formula as Table 7.2. MGDA (no TCS) serves as the baseline without conflict detection.

λ	Mean pLDDT	P(toxic)	P(virulent)	Composite
MGDA (no TCS)	55.83	0.0523	0.6926	60.45
0.8	55.83	0.0523	0.6926	60.45
0.5	53.67	0.0477	0.6766	60.43
0.2	53.67	0.0477	0.6766	60.43
0.003	56.29	0.0189	0.6237	64.01
0.002	56.04	0.0139	0.5407	66.86
0.001	56.95	0.0421	0.5968	64.36
0	51.17	0.0072	0.5835	64.03

the binary choice between independent classifiers and a single shared backbone with a gated cross-attention design that selectively exchanges cross-task information, producing a stronger PPO reward signal than independent classifiers in five out of six fixed-weight configurations. TCS-MGDA extends Pareto-descent weight calibration to the high-dimensional bounded-reward regime by filtering the Gram matrix through a task-conflict saliency score and by centering rewards prior to gradient aggregation, thereby removing the need to commit to a fixed reward-weight vector. Empirically, TCS-MGDA lifts the composite score from 60.45 under standard MGDA to 66.86, a +6.41 absolute gain and a +5.21 improvement over the strongest fixed-weight baseline, driven by sharp reductions in both toxicity and virulence while structural quality is preserved.

This work is currently under submission.

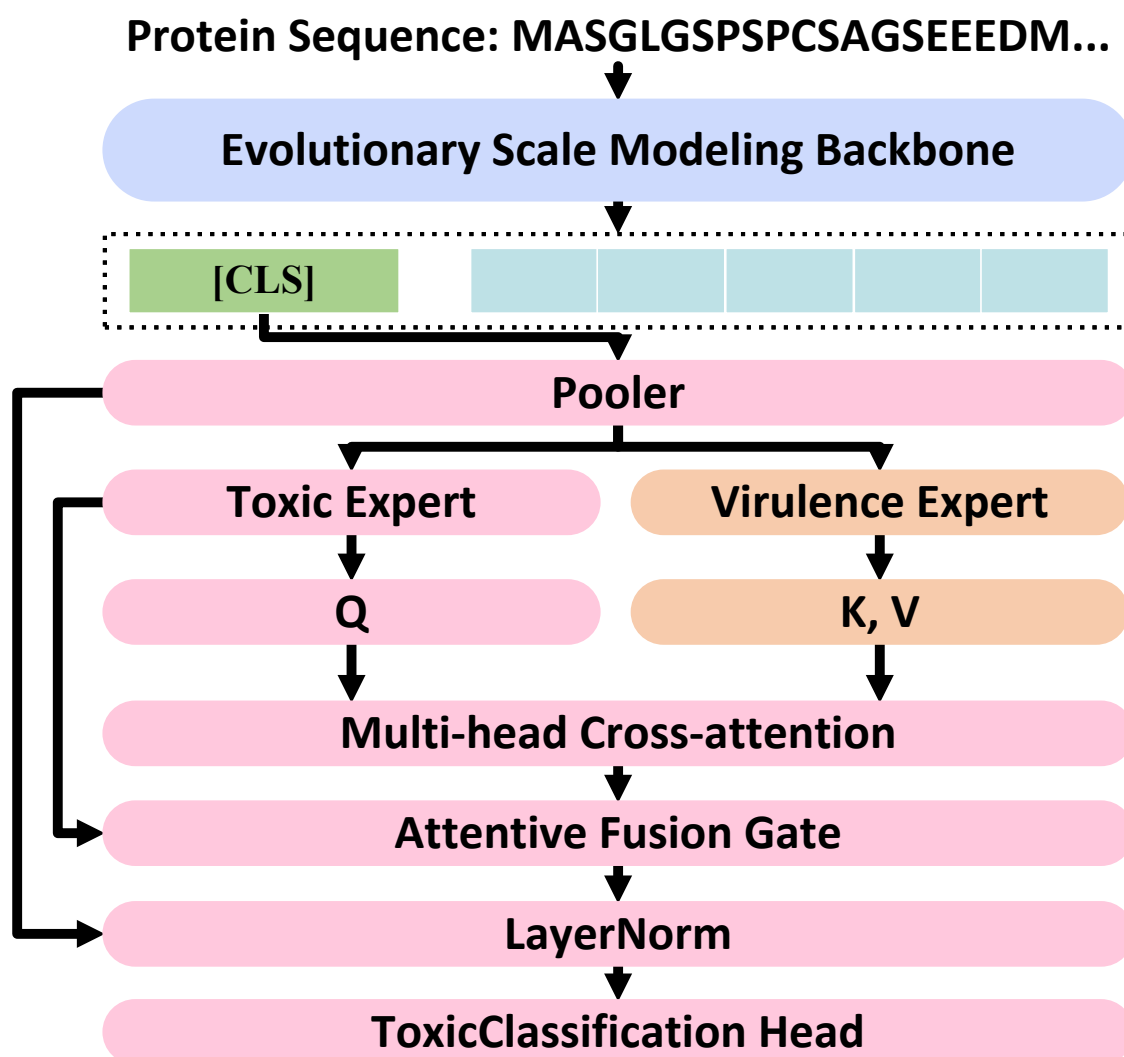


Figure 7.2. Overview of the Cross-Task Attentive Fusion (CTAF) architecture, shown for the toxic task as the active task. The frozen ESM2 backbone (blue) and pooler produce a CLS representation, which is passed through both task-specific experts. The active task's expert provides queries (Q) to the multi-head cross-attention module, while the other task's expert supplies keys and values (K, V). An adaptive fusion gate modulates the balance between cross-task and task-private representations before a residual connection and LayerNorm. The fused output is then passed to the active task's classification head. When virulence is the active task, the roles of the experts are reversed.

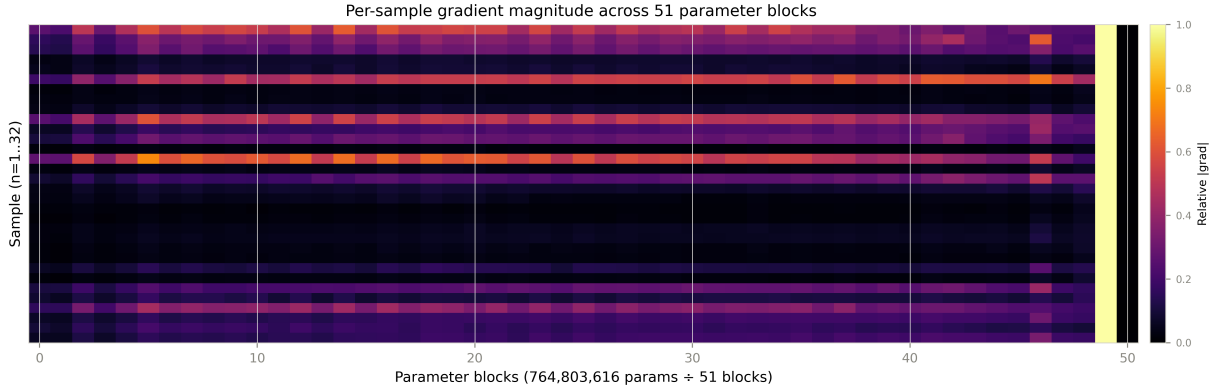


Figure 7.3. Raw per-sample log-probability gradients $|\nabla_n|$ across 51 parameter blocks for $N=32$ sequences generated by the `progen2-medium` policy. **Each row is independently normalized by that sample’s own maximum block magnitude**, i.e. entry (n, b) plots $\|\nabla_n^{(b)}\|_2 / \max_b \|\nabla_n^{(b')}\|_2 \in [0, 1]$. Per-row normalization removes the large magnitude differences *across* samples so that the *within-sample* distribution of gradient signal is directly comparable: bright = blocks carrying most of that sample’s gradient, dark $\approx$ zero for that sample.

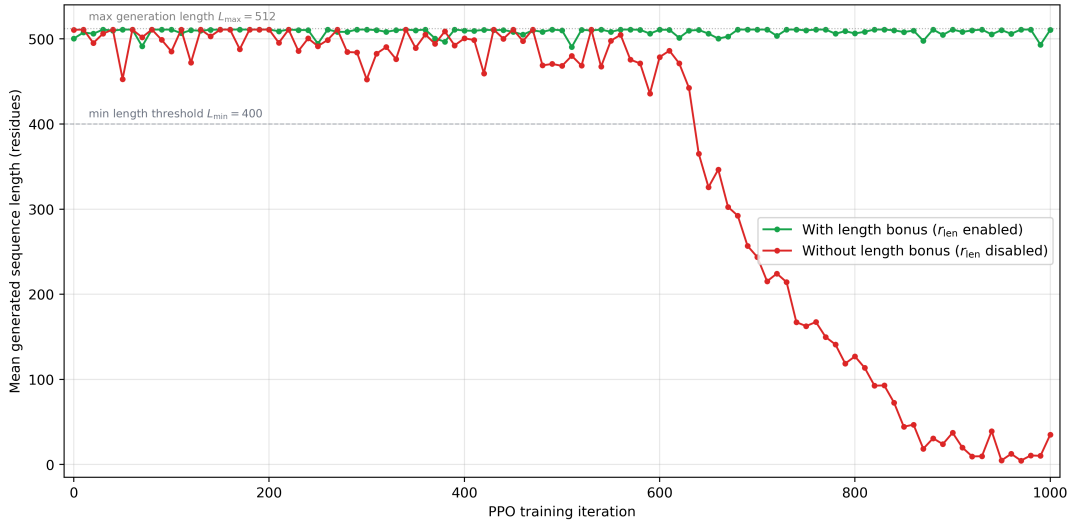


Figure 7.4. Effect of the length bonus on mean generated sequence length during PPO training.

CHAPTER 8

Conclusions and Future Works

This dissertation has examined the dual role of generative models as both a shield and a threat: a shield that enables privacy-preserving data sharing, engaging content creation, efficient model adaptation, and controlled information removal, and a threat when their outputs erode the authenticity of public communication or extend into domains, such as biological sequence design, where misuse carries societal consequences. Across six projects, we developed frameworks that address these tensions from complementary angles: synthesis, adaptation, unlearning, detection, and alignment, and grounded each in rigorous empirical evaluation. This chapter summarizes the main findings of the dissertation and outlines directions for future research.

8.1. Summary of Contributions

Protecting data through synthetic relational databases. In Chapter 2, we introduced **FakeDB**, a GAN-based framework for synthesizing relational databases that jointly preserves statistical distributions and semantic integrity constraints across multiple tables. By decomposing generation into table-wise, relational, and adversarial phases, FakeDB enforces both unary and multi-relational constraints that single-table generators fundamentally cannot express. Experiments on eight real-world datasets and one synthetic benchmark showed that FakeDB outperforms TableGAN, CTGAN, and SDV on the majority of constraint-satisfaction and distribution-matching metrics.

Generating engaging multimodal content. In Chapter 3, we proposed **GEM**, a framework that steers image-text generation toward user-specified engagement objectives. GEM first trains an engagement classifier on social media samples, then uses this classifier to guide continuous prompt learning for a diffusion-based image generator, and finally aligns images and text through an iterative similarity-and-paraphrasing loop. The framework demonstrates that discriminative feedback and prompt tuning can jointly shape generation toward goals that are difficult to specify through text prompts alone.

Efficient fine-tuning of large language models. In Chapter 4, we presented **MoLA**, a parameter-efficient fine-tuning method that combines Mixture-of-Experts routing with Low-Rank Adaptation and, crucially, allocates experts non-uniformly across Transformer layers. Evaluations on six NLP and commonsense QA benchmarks showed that configurations concentrating experts in the middle layers consistently outperform uniform-allocation baselines, providing empirical evidence that layer importance for adaptation is non-uniform and that lower layers carry substantial redundancy.

Continual unlearning without retained data. In Chapter 5, we introduced $\mathbf{O}^3$, a continual unlearning framework for large language models that removes targeted knowledge without requiring access to the retained training set. $\mathbf{O}^3$ combines orthogonal LoRA adapters, which isolate successive unlearning requests and prevent interference, with a global-aware out-of-distribution detector that decides at inference time whether an input pertains to unlearned content. The method consistently surpasses state-of-the-art baselines on both unlearning effectiveness and utility preservation, offering a deployable mechanism for responding to copyright removal and privacy requests in already-deployed models.

Context-aware audio deepfake detection. In Chapter 6, we introduced the **Journalist-provided Deepfake Dataset (JDD)**, a collection of 255 audio samples of public figures curated with the help of over 70 journalists, and proposed **CADD**, a context-based audio deepfake detector that integrates transcripts and external contextual metadata alongside the audio signal. CADD materially improves detection on realistic public-figure deepfakes, where signal-only detectors struggle, and exposes a concrete limitation of prior benchmarks: datasets built around non-public figures or semantically implausible utterances fail to transfer to the content that adversaries actually produce.

Safe protein generation via multi-objective alignment. In Chapter 7, we extended alignment techniques from language to biological sequence design with **SafePro**, a multi-objective alignment framework for protein language models that jointly optimizes structural foldability, toxicity avoidance, and virulence avoidance. *SafePro* contributes two components targeted at concrete obstacles to transferring LLM-style alignment to this setting: *Cross-Task Attentive Fusion* (CTAF), a reward model that selectively shares signal between biologically coupled safety objectives through an input-dependent gate, and *Task-Conflict-Saliency MGDA* (TCS-MGDA), a gradient-based weight-calibration procedure that handles Gram-matrix dilution and spurious alignment through saliency filtering and reward centering. Alongside the method, we curated and released a virulence classification dataset of 54,166 length-matched sequences to support future work in this area.

8.2. Future Directions

The six projects in this dissertation open several avenues for continued research. We highlight directions that follow from specific limitations of the methods we developed, as well as broader cross-cutting themes that the dissertation as a whole suggests.

Extending GEM to additional modalities and multi-objective generation. GEM demonstrates that image-text generation can be steered by a learned engagement classifier, but the underlying recipe, using discriminative feedback to guide continuous prompt learning, is not inherently tied to images and text. A natural direction is to extend the framework to video and audio, where short-form video, in particular, has become the dominant medium for organizational communication, and where the interplay among visual pacing, audio cues, and on-screen text creates richer opportunities for engagement-aware generation. A second direction is to move from a single engagement signal to explicit multi-objective formulations: engagement alone is a coarse proxy that can reward sensationalism, and a more faithful objective combines engagement with factual accuracy, topical relevance, and an audience-appropriate tone. Incorporating these objectives directly into prompt learning, rather than applying them as post hoc filters, would make the trade-offs explicit and tunable, and connect naturally to the conflict-aware multi-objective optimization techniques developed in Chapter 7.

Adaptive and task-aware expert allocation. MoLA’s layer-wise expert allocation is fixed a priori according to one of several hand-designed configurations. A natural extension is to learn the allocation itself. We can treat the per-layer expert count as a differentiable or reinforcement-learned hyperparameter, allowing the allocation to adapt to the specific downstream task and base model. Understanding why middle layers disproportionately

benefit from expert specialization also remains an open question; connecting this observation to mechanistic interpretability results on layer-wise representation structures could yield design principles that generalize beyond the specific benchmarks we studied.

Unlearning at scale and under adversarial conditions. $\mathcal{O}^3$ handles sequential unlearning requests without retained data, but several questions remain open. One concerns scalability: it is unclear how the orthogonal-adaptor mechanism behaves when the number of sequential requests grows into the thousands, as would be required for a production deployment handling a steady stream of takedown requests. A second concerns adversarial robustness: the OOD detector that triggers unlearning at inference time may be vulnerable to adversarial paraphrasing designed to evade detection, and characterizing this vulnerability is a prerequisite for deployment in contested settings. A third concerns the scope of what can be unlearned: extending the framework from discrete factual knowledge to behavioral patterns, such as a style of response or a latent bias, would substantially broaden its applicability but raises new measurement challenges, since such patterns lack the clean input-output signatures that the current OOD detector relies on.

Detecting multimodal deepfakes of public figures. The audio-focused CADD framework and JDD dataset naturally extend to video, where visual, auditory, and contextual cues can be jointly exploited. We are actively pursuing this direction: constructing a realistic multimodal deepfake dataset centered on public figures, using contextual news content to guide the generation of semantically plausible forgeries, and developing detectors that fuse visual, audio, and textual contextual features. A related open problem is temporal robustness: detectors trained today must remain effective against generation methods that

will exist a year from now, which calls for detection architectures that exploit context and provenance signals rather than artifacts specific to current synthesis pipelines.

Alignment for broader scientific generative models. *SafePro* establishes a template for safety alignment in protein language models, but the dual-use concerns it addresses, in which beneficial design and harmful design share the same computational machinery, apply equally to small-molecule generators, nucleic-acid designers, and other scientific foundation models. Future work should investigate whether the CTAF and TCS-MGDA components transfer to these domains, and how to define safety objectives in settings where “harmful” is harder to label than toxicity or virulence. A second direction is closing the loop with wet-lab validation: the rewards used during alignment are computational proxies, and understanding how proxy-reward gaming manifests in physically synthesized sequences is essential before such systems are deployed. Finally, governance questions, such as who decides which phenotypes a generator should refuse and how these decisions are audited, will increasingly shape what technical alignment frameworks need to support.

Cross-cutting themes. Three themes recur across the chapters and merit dedicated investigation. First, *context as a first-class signal*: CADD leverages context for detection, GEM for generation, and $\mathcal{O}^3$ for inference-time routing, suggesting that contextual grounding is a general-purpose lever for trustworthy generative systems and warrants a unified treatment. Second, *the tension between capability and control*: each project negotiates this tension in a specific form, and developing general principles for when capability should be constrained, when it should be redirected, and when it should be rejected outright is, in our view, the central intellectual challenge facing the generative-AI research community in the coming years.

8.3. Closing Remarks

The arc of this dissertation moves from using generative models to protect data, to adapting and controlling them efficiently, to detecting their misuse, and finally to aligning them in domains where misuse carries biological consequences. The unifying commitment is that the benefits of generative AI and its risks are inseparable: every capability that enables privacy-preserving synthesis, engaging communication, efficient adaptation, or scientific discovery also creates a surface for misuse, and responsible deployment requires treating authenticity, privacy, and safety as engineering requirements rather than downstream concerns. We hope that the frameworks developed here: FakeDB, GEM, MoLA, $\mathcal{O}^3$, CADD, and *SafePro*—contribute both concrete tools for practitioners and, more broadly, a template for how generation, detection, and alignment can be developed together rather than in isolation.

References

- [1] CLINC150. UCI Machine Learning Repository, 2020. DOI: <https://doi.org/10.24432/C5MP58>.
- [2] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.
- [3] Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, and Mauro Conti. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Comput. Surv.*, 51(4):1–35, 2018.
- [4] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In *2018 IEEE international workshop on information forensics and security (WIFS)*, pages 1–7. IEEE, 2018.
- [5] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.
- [6] Rajat Agarwal and Katharina Kann. Acrostic poem generation. *arXiv preprint arXiv:2010.02239*, 2020.
- [7] Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew Dai, Anja Hauth, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- [8] Laura Aviñó, Matteo Ruffini, and Ricard Gavaldà. Generating synthetic but plausible healthcare record datasets. *arXiv preprint arXiv:1807.01514*, 2018.
- [9] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.
- [10] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon,

- et al. Constitutional ai: Harmlessness from ai feedback, 2022. URL <https://arxiv.org/abs/2212.08073>, 2212, 2022.
- [11] Eric Battenberg, RJ Skerry-Ryan, Soroosh Mariooryad, Daisy Stanton, David Kao, Matt Shannon, and Tom Bagby. Location-relative attention mechanisms for robust long-form speech synthesis. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6194–6198. IEEE, 2020.
 - [12] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
 - [13] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.
 - [14] Jordan J Bird and Ahmad Lotfi. Real-time detection of ai-generated speech for deepfake voice conversion. *arXiv preprint arXiv:2308.12734*, 2023.
 - [15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
 - [16] Lucas Bourtole, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.
 - [17] Hervé Bredin. pyannote.audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. In *Proc. INTERSPEECH 2023*, 2023.
 - [18] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
 - [19] Eric L Buehler and Markus J Buehler. X-lora: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design. *APL Machine Learning*, 2(2), 2024.
 - [20] Manuel Fernández Burda, Santiago Aranguri, Iván Arcuschin Moreno, and Enzo Ferrante. Inference-time toxicity mitigation in protein language models. *CoRR*, abs/2603.04045, 2026.

- [21] Daniel L Byman, Chongyang Gao, Chris Meserole, and VS Subrahmanian. *Deepfakes and international conflict*. Brookings Institution, 2023.
- [22] Hanqun Cao, Marcelo Der Torossian Torres, Jingjie Zhang, Zijun Gao, Fang Wu, Chunbin Gu, Jure Leskovec, Yejin Choi, Cesar de la Fuente-Nunez, Guangyong Chen, and Pheng-Ann Heng. A deep reinforcement learning platform for antibiotic discovery. *CoRR*, abs/2509.18153, 2025.
- [23] Edresson Casanova, Kelly Davis, Eren Gölge, Gökem Gökner, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, et al. Xtts: a massively multilingual zero-shot text-to-speech model. *arXiv preprint arXiv:2406.04904*, 2024.
- [24] Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International conference on machine learning*, pages 2709–2720. PMLR, 2022.
- [25] Ting-Yun Chang, Jesse Thomason, and Robin Jia. Do localization methods actually localize memorized data in llms? a tale of two benchmarks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3190–3211, 2024.
- [26] Haipeng Chen, Sushil Jajodia, Jing Liu, Noseong Park, Vadim Sokolov, and VS Subrahmanian. Faketables: Using GANs to generate functional dependency preserving tables with bounded real data. In *IJCAI*, pages 2074–2080, 2019.
- [27] Jiaao Chen and Diyi Yang. Unlearn what you want to forget: Efficient unlearning for llms. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.
- [28] Nuo Chen, Yan Wang, Yang Deng, and Jia Li. The oscars of ai theater: A survey on role-playing with language models. *arXiv preprint arXiv:2407.11484*, 2024.
- [29] Tianlong Chen, Zheyu Zhang, Ajay Jain, Shiwei Liu, and Zhangyang Wang. Sparse moe as the new dropout: Scaling dense and self-slimmable transformer. *ICLR*, 2023.
- [30] Yujie Chen, Jiangyan Yi, Cunhang Fan, Jianhua Tao, Yong Ren, Siding Zeng, Chu Yuan Zhang, Xinrui Yan, Hao Gu, Jun Xue, Chenglong Wang, Zhao Lv, and Xiaohui Zhang. Region-based optimization in continual learning for audio deepfake detection. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored*

- by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA, pages 23651–23659. AAAI Press, 2025.
- [31] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Grad-norm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pages 793–802. PMLR, 2018.
 - [32] Ziyang Chen and Stylios Moscholios. Using prompts to guide large language models in imitating a real person’s language style. *arXiv preprint arXiv:2410.03848*, 2024.
 - [33] Ha Yeong Choi, Sang Hoon Lee, and Seong Whan Lee. Diff-hiervc: Diffusion-based hierarchical voice conversion with robust pitch generation and masked prior for zero-shot speaker adaptation. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 2023, pages 2283–2287, 2023.
 - [34] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.
 - [35] Collabora. Whisperspeech: A speech-to-text engine by collabora, 2024. Accessed: 2024-11-07.
 - [36] Maxime Darrin, Guillaume Staerman, Eduardo Dadalto Câmara Gomes, Jackie CK Cheung, Pablo Piantanida, and Pierre Colombo. Unsupervised layer-wise score aggregation for textual ood detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17880–17888, 2024.
 - [37] Bibhu Dash and Pawankumar Sharma. Are chatgpt and deepfake algorithms endangering the cybersecurity industry? a review. *International Journal of Engineering and Applied Sciences*, 10(1):21–39, 2023.
 - [38] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.
 - [39] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart. The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000.

- [40] Junlong Deng, Yanzhen Ren, Tong Zhang, Hongcheng Zhu, and Zongkun Sun. Vfd-net: Vocoder fingerprints detection for fake audio. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pages 12151–12155. IEEE, 2024.
- [41] Wei Deng, Siyi Zhou, Jingchen Shu, Jinchao Wang, and Lu Wang. Indextts: An industrial-level controllable and efficient zero-shot text-to-speech system. *arXiv preprint arXiv:2502.05512*, 2025.
- [42] Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus. Mathématique*, 350(5-6):313–318, 2012.
- [43] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. *arXiv preprint arXiv:2005.07143*, 2020.
- [44] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.
- [45] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.
- [46] Ana Do Vale, Didier Cabanes, and Sandra Sousa. Bacterial toxins as pathogen weapons against phagocytes. *Frontiers in microbiology*, 7:42, 2016.
- [47] Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*, 2005.
- [48] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 2023.
- [49] Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, et al. Cosyvoice 2: Scalable streaming speech synthesis with large language models. *arXiv preprint arXiv:2412.10117*, 2024.
- [50] Ronen Eldan and Mark Russinovich. Who’s harry potter? approximate unlearning in llms. *arXiv preprint arXiv:2310.02238*, 2023.

- [51] Mohammad Esmaeilpour, Nourhene Chaalia, Adel Abusitta, François-Xavier Devailly, Wissem Maazoun, and Patrick Cardinal. Bi-discriminator gan for tabular data synthesis. *Pattern Recognition Letters*, 159:204–210, 2022.
- [52] David Evans, Vladimir Kolesnikov, Mike Rosulek, et al. A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security*, 2(2-3):70–246, 2018.
- [53] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.
- [54] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. Protgpt2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1):4348, 2022.
- [55] Joel Frank and Lea Schönherr. Wavefake: A data set to facilitate audio deepfake detection. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- [56] Chongyang Gao, Sushil Jajodia, Andrea Pugliese, and VS Subrahmanian. Fakedb: generating fake synthetic databases. *IEEE Transactions on Dependable and Secure Computing*, 2024.
- [57] Chongyang Gao, Yiren Jian, Natalia Denisenko, Soroush Vosoughi, and VS Subrahmanian. Gem: generating engaging multimodal content. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 7654–7662, 2024.
- [58] Chongyang Gao, Marco Postiglione, Isabel Gortner, Sarit Kraus, and VS Subrahmanian. Perturbed public voices (p2v): A dataset for robust audio deepfake detection. *arXiv preprint arXiv:2508.10949*, 2025.
- [59] Randy Goebel, Koichi Furukawa, and David Poole. Using definite clauses and integrity constraints as the basis for a theory formation approach to diagnostic reasoning. In *International Conference on Logic Programming*, pages 211–222. Springer, 1986.
- [60] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9304–9312, 2020.

- [61] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.
- [62] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.
- [63] Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing can hurt general abilities of large language models. *arXiv preprint arXiv:2401.04700*, 2024.
- [64] Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. *arXiv preprint arXiv:2401.07453*, 2024.
- [65] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.
- [66] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
- [67] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- [68] Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *ICLR*, 2022.
- [69] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021.
- [70] Minghui Hu, Chuanxia Zheng, Heliang Zheng, Tat-Jen Cham, Chaoyue Wang, Zuopeng Yang, Dacheng Tao, and Ponnuthurai N Suganthan. Unified discrete diffusion for simultaneous vision-language generation. *arXiv preprint arXiv:2211.14842*, 2022.
- [71] Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. Ravel: Evaluating interpretability methods on disentangling language model representations. *arXiv preprint arXiv:2402.17700*, 2024.

- [72] Jihyeon Hyeong, Jayoung Kim, Noseong Park, and Sushil Jajodia. An empirical study on the membership inference attack against tabular data synthesis models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4064–4068, 2022.
- [73] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
- [74] Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408, 2023.
- [75] Jinghan Jia, Yihua Zhang, Yimeng Zhang, Jiancheng Liu, Bharat Runwal, James Diffenderfer, Bhavya Kailkhura, and Sijia Liu. Soul: Unlocking the power of second-order optimization for llm unlearning. *arXiv preprint arXiv:2404.18239*, 2024.
- [76] Yiren Jian, Tingkai Liu, Yunzhe Tao, Soroush Vosoughi, and Hongxia Yang. Simvlg: Simple and efficient pretraining of visual language generative models. *arXiv preprint arXiv:2310.03291*, 2023.
- [77] Albert Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Chaplot, Diego de las Casas, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- [78] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [79] Jee-weon Jung, Hee-Soo Heo, Hemlata Tak, Hye-jin Shim, Joon Son Chung, Bong-Jin Lee, Ha-Jin Yu, and Nicholas Evans. Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks. In *ICASSP 2022-2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6367–6371. IEEE, 2022.
- [80] Jee-weon Jung, Seung-bin Kim, Hye-jin Shim, Ju-ho Kim, and Ha-Jin Yu. Improved rawnet with feature map scaling for text-independent speaker verification using raw waveforms. *arXiv preprint arXiv:2004.00526*, 2020.
- [81] Jee-weon Jung, You Jin Kim, Hee-Soo Heo, Bong-Jin Lee, Youngki Kwon, and Joon Son Chung. Pushing the limits of raw waveform speaker recognition. *arXiv preprint arXiv:2203.08488*, 2022.

- [82] William Karush. Minima of functions of several variables with inequalities as side constraints. *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago*, 1939.
- [83] Piotr Kawa, Marcin Plata, Michał Czuba, Piotr Szymański, and Piotr Syga. Improved deepfake detection using whisper features. *arXiv preprint arXiv:2306.01428*, 2023.
- [84] Piotr Kawa, Marcin Plata, and Piotr Syga. Specrnet: Towards faster and more accessible audio deepfake detection. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 792–799. IEEE, 2022.
- [85] Jayoung Kim, Chaejeong Lee, and Noseong Park. Stasy: Score-based tabular data synthesis. *arXiv preprint arXiv:2210.04018*, 2022.
- [86] Seungbae Kim, Jyun-Yu Jiang, Masaki Nakada, Jinyoung Han, and Wei Wang. Multimodal post attentive profiling for influencer marketing. In *Proceedings of The Web Conference 2020*, pages 2878–2884, 2020.
- [87] Sungwon Kim, Kevin J. Shih, Rohan Badlani, João Felipe Santos, Evelina Bakhturina, Mikyas Desta, Rafael Valle, Sungroh Yoon, and Bryan Catanzaro. P-flow: A fast and data-efficient zero-shot TTS through speech prompting. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.
- [88] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR, 2021.
- [89] Tomi Kinnunen, Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. In *Interspeech 2017*, pages 2–6. International Speech Communication Association, 2017.
- [90] Tomi Kinnunen, Zhizheng Wu, EURECOM Nicholas Evans, and Junichi Yamagishi. Automatic speaker verification spoofing and countermeasures challenge (asvspoof 2015) database. 2018.
- [91] Hyungyung Lee, Sungjin Park, Joonseok Lee, and Edward Choi. Unconditional image-text pair generation with multimodal cross quantizer. *arXiv preprint arXiv:2204.07537*, 2022.

- [92] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. *arXiv preprint arXiv:2107.04589*, 2021.
- [93] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [94] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, 2021.
- [95] Jingyi Li, Weiping Tu, and Li Xiao. Freevc: Towards high-quality text-free one-shot voice conversion. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.
- [96] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.
- [97] Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.
- [98] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.
- [99] Xinfeng Li, Kai Li, Yifan Zheng, Chen Yan, Xiaoyu Ji, and Wenyan Xu. Safeear: Content privacy-preserving audio deepfake detection. *arXiv preprint arXiv:2409.09272*, 2024.
- [100] Wing Lian, Bley Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". Openorca: An open dataset of gpt augmented flan reasoning traces. <https://huggingface.co/Open-Orca/OpenOrca>, 2023.
- [101] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022:500902, 2022.

- [102] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.
- [103] Daniel W Linna Jr, Abhishek Dalal, Chongyang Gao, Paul W Grimm, Maura R Grossman, Chiara Pulice, VS Subrahmanian, Hon Tunheim, et al. Deepfakes in court: How judges can proactively manage alleged ai-generated material in national security cases. *Northwestern Law & Econ Research Paper*, (24-18):24–26, 2024.
- [104] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In Marc’Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 18878–18890, 2021.
- [105] Bo Liu, Dandan Zheng, Siyu Zhou, Lihong Chen, and Jian Yang. VFDB 2022: a general classification scheme for bacterial virulence factors. *Nucleic Acids Research*, 50(D1):D912–D917, 2022.
- [106] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *NeurIPS*, 2022.
- [107] Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *arXiv preprint arXiv:2310.18339*, 2023.
- [108] Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Xiaojun Xu, Yuguang Yao, Hang Li, Kush R Varshney, et al. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*, 2024.
- [109] Songting Liu. Zero-shot voice conversion with diffusion transformers. *arXiv preprint arXiv:2411.09943*, 2024.
- [110] Tongyu Liu, Ju Fan, Guoliang Li, Nan Tang, and Xiaoyong Du. Tabular data synthesis with generative adversarial networks: design space and optimizations. *The VLDB Journal*, pages 1–26, 2023.

- [111] Xianliang Liu, Jiawei Luo, Xinyan Wang, Yang Zhang, and Junjie Chen. Directed evolution of antimicrobial peptides using multi-objective zeroth-order optimization. *Briefings Bioinform.*, 26(1), 2025.
- [112] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [113] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. *ACL 2022*, 2022.
- [114] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [115] John W Lloyd. *Foundations of logic programming*. Springer Science & Business Media, 2012.
- [116] Mingsheng Long, Yue Cao, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Transferable representation learning with deep adaptation networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):3071–3085, 2018.
- [117] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [118] Keming Lu, Bowen Yu, Chang Zhou, and Jingren Zhou. Large language models are superpositions of all characters: Attaining arbitrary role-play via self-alignment. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7828–7840, 2024.
- [119] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.
- [120] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *NeurIPS*, 2022.

- [121] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41(8):1099–1106, 2023.
- [122] Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.
- [123] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.
- [124] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.
- [125] MetaVoice. Metavoice source repository, 2024. Accessed: 2024-11-07.
- [126] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, 2018.
- [127] Jan Motl and Oliver Schulte. The CTU Prague relational learning repository – <https://arxiv.org/abs/1511.03086>, 2015.
- [128] Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4, 2023.
- [129] Nicolas M Müller, Pavel Czempin, Franziska Dieckmann, Adam Froghyar, and Konstantin Böttinger. Does audio deepfake detection generalize? *arXiv preprint arXiv:2203.16263*, 2022.
- [130] Aviv Navon, Aviv Shamsian, Ethan Fetaya, and Gal Chechik. Learning the pareto front with hypernetworks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [131] Erik Nijkamp, Jeffrey A Ruffolo, Eli N Weinstein, Nikhil Naik, and Ali Madani. Progen2: exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978, 2023.

- [132] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [133] Monisankha Pal, Aditya Raikar, Ashish Panda, and Sunil Kumar Kopparapu. Synthetic speech detection using meta-learning with prototypical loss. *arXiv preprint arXiv:2201.09470*, 2022.
- [134] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language models. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331. IEEE, 2020.
- [135] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10):1071–1083, 2018.
- [136] Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *International Conference on Data Science and Advanced Analytics*, pages 399–410. IEEE, 2016.
- [137] Martin Pawelczyk, Seth Neel, and Himabindu Lakkaraju. In-context unlearning: Language models as few shot unlearners. *ICML*, 2024.
- [138] Johnny W Peterson. Bacterial pathogenesis. *Medical Microbiology*. 4th edition, 1996.
- [139] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.
- [140] Alexis Plaquet and Hervé Bredin. Powerset multi-class cross entropy loss for neural speaker diarization. In *Proc. INTERSPEECH 2023*, 2023.
- [141] Marco Postiglione, Julian Baldwin, Natalia Denisenko, Luke Fosdick, Chongyang Gao, Isabel Gortner, Chiara Pulice, Sarit Kraus, and VS Subrahmanian. GODDS: The global online deepfake detection system. In *AAAI 2025 Demonstration Program*, 2024.
- [142] Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *The Thirteenth International Conference on Learning Representations*, 2025.

- [143] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1505–1514, 2019.
- [144] Zengyi Qin, Wenliang Zhao, Xumin Yu, and Xin Sun. Openvoice: Versatile instant voice cloning. *arXiv preprint arXiv:2312.01479*, 2023.
- [145] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [146] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022.
- [147] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.
- [148] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.
- [149] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.
- [150] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.
- [151] Jerome P Reiter. Using cart to generate partially synthetic public use microdata. *Journal of official statistics*, 21(3):441, 2005.
- [152] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3165–3174, 2019.
- [153] Manuel Sam Ribeiro, Julian Roth, Giulia Comini, Goeric Huybrechts, Adam Gabrys, and Jaime Lorenzo-Trueba. Cross-speaker style transfer for text-to-speech using data

- augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pages 6797–6801. IEEE, 2022.
- [154] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.
 - [155] Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.
 - [156] Margie Ruffin, Haeseung Seo, Aiping Xiong, and Gang Wang. Does it matter who said it? exploring the impact of deepfake-enabled profiles on user perception towards disinformation. In Yu-Ru Lin, Yelena Mejova, and Meeyoung Cha, editors, *Proceedings of the Eighteenth International AAAI Conference on Web and Social Media, ICWSM 2024, Buffalo, New York, USA, June 3-6, 2024*, pages 1328–1341. AAAI Press, 2024.
 - [157] Md Sahidullah and Goutam Saha. Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition. *Speech communication*, 54(4):543–565, 2012.
 - [158] Ram Samudrala, Fred Heffron, and Jason E McDermott. Accurate prediction of secreted substrates and identification of a conserved putative secretion signal for type III secretion systems. *PLoS Pathogens*, 5(4):e1000375, 2009.
 - [159] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. In *ICLR 2022-Tenth International Conference on Learning Representations*, 2022.
 - [160] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.
 - [161] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
 - [162] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.
 - [163] Zihao Shan, Kui Ren, Marina Blanton, and Cong Wang. Practical secure computation outsourcing: A survey. *ACM Comput. Surv.*, 51(2):1–40, 2018.

- [164] Murray Shanahan, Kyle McDonell, and Laria Reynolds. Role play with large language models. *Nature*, 623(7987):493–498, 2023.
- [165] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *ICLR*, 2017.
- [166] Fang Sheng, Mohammad Noaeen, and Zahra Shakeri. Prodcarl: Reinforcement learning-aligned diffusion models for de novo antimicrobial peptide design. *CoRR*, abs/2602.00157, 2026.
- [167] Karan Singhal, Shekoofeh Azizi, Tao Tu, Sara Mahdavi, Jason Wei, Hyung Chung, Nathan Scales, Ajay Tanwani, et al. Large language models encode clinical knowledge. *arXiv preprint arXiv:2212.13138*, 2022.
- [168] Emily H Soice, Rafael Rocha, Kimberlee Cordova, Michael Specter, and Kevin M Esvelt. Can large language models democratize access to dual-use biotechnology? *arXiv preprint arXiv:2306.03809*, 2023.
- [169] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International conference on machine learning*, pages 9120–9132. PMLR, 2020.
- [170] Chengzhe Sun, Shan Jia, Shuwei Hou, and Siwei Lyu. Ai-synthesized voice detection using neural vocoder artifacts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 904–912, 2023.
- [171] Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, et al. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*, 2024.
- [172] Hemlata Tak, Jose Patino, Massimiliano Todisco, Andreas Nautsch, Nicholas Evans, and Anthony Larcher. End-to-end anti-spoofing with rawnet2. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6369–6373. IEEE, 2021.
- [173] Hemlata Tak, Massimiliano Todisco, Xin Wang, Jee-weon Jung, Junichi Yamagishi, and Nicholas Evans. Automatic speaker verification spoofing and deepfake detection using wav2vec 2.0 and data augmentation. *arXiv preprint arXiv:2202.12233*, 2022.
- [174] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *NAACL*, 2019.

- [175] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.
- [176] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.
- [177] Pratiksha Thaker, Yash Maurya, and Virginia Smith. Guardrail baselines for unlearning in llms. *arXiv preprint arXiv:2403.03329*, 2024.
- [178] The UniProt Consortium. UniProt: the universal protein knowledgebase in 2023. *Nucleic Acids Research*, 51(D1):D523–D531, 2023.
- [179] Massimiliano Todisco, Héctor Delgado, and Nicholas Evans. Constant q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45:516–535, 2017.
- [180] Massimiliano Todisco, Xin Wang, Ville Vestman, Md Sahidullah, Héctor Delgado, Andreas Nautsch, Junichi Yamagishi, Nicholas Evans, Tomi Kinnunen, and Kong Aik Lee. Asvspoof 2019: Future horizons in spoofed and fake audio detection. *arXiv preprint arXiv:1904.05441*, 2019.
- [181] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [182] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [183] Fabio Urbina, Filippa Lentzos, Cédric Invernizzi, and Sean Ekins. Dual use of artificial-intelligence-powered drug discovery. *Nature machine intelligence*, 4(3):189–191, 2022.
- [184] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

- [185] Christina P. Walker, Daniel S. Schiff, and Kaylyn Jackson Schiff. Merging AI incidents research with political misinformation research: Introducing the political deepfakes incidents database. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 23053–23058. AAAI Press, 2024.
- [186] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *ICLR*, 2019.
- [187] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.
- [188] Chenglong Wang, Jiayi He, Jiangyan Yi, Jianhua Tao, Chu Yuan Zhang, and Xiaohui Zhang. Multi-scale permutation entropy for audio deepfake detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pages 1406–1410. IEEE, 2024.
- [189] Chenglong Wang, Jiangyan Yi, Jianhua Tao, Haiyang Sun, Xun Chen, Zhengkun Tian, Haoxin Ma, Cunhang Fan, and Ruiibo Fu. Fully automated end-to-end fake audio detection. In *Proceedings of the 1st International Workshop on Deepfake Detection for Audio Multimedia*, pages 27–33, 2022.
- [190] Li Wang, Xiangzheng Fu, Jiahao Yang, Xinyi Zhang, Xiucui Ye, Tetsuya Sakurai, Xiangxiang Zeng, and Yiping Liu. Moformer: navigating the antimicrobial peptide design space with pareto-based multi-objective transformer. *Briefings Bioinform.*, 26(6), 2025.
- [191] Lijuan Wang, Chen Qiu, Liwei Xiao, and Junjie Chen. Peppfdpo: Multi-objective antimicrobial peptide generation using pareto-frontier enhanced direct preference optimization. In *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2025, Wuhan, China, December 15-18, 2025*, pages 1–9. IEEE, 2025.
- [192] Run Wang, Felix Juefei-Xu, Yihao Huang, Qing Guo, Xiaofei Xie, Lei Ma, and Yang Liu. Deepsonar: Towards effective and robust detection of ai-synthesized fake voices. In Chang Wen Chen, Rita Cucchiara, Xian-Sheng Hua, Guo-Jun Qi, Elisa Ricci, Zhengyou Zhang, and Roger Zimmermann, editors, *MM '20: The 28th ACM*

International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020, pages 1207–1216. ACM, 2020.

- [193] Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. *arXiv preprint arXiv:2310.14152*, 2023.
- [194] Xin Wang, Héctor Delgado, Hemlata Tak, Jee-weon Jung, Hye-jin Shim, Massimiliano Todisco, Ivan Kukanov, Xuechen Liu, Md Sahidullah, Tomi Kinnunen, et al. Asvspoof 5: Crowdsourced speech data, deepfakes, and adversarial attacks at scale. *arXiv preprint arXiv:2408.08739*, 2024.
- [195] Xin Wang and Junich Yamagishi. A comparative study on recent neural spoofing countermeasures for synthetic speech detection. *arXiv preprint arXiv:2103.11326*, 2021.
- [196] Yuhao Wang, Keyan Ding, Kehua Feng, Zeyuan Wang, Ming Qin, Xiaotong Li, Qiang Zhang, and Huajun Chen. Enhancing safe and controllable protein generation via knowledge preference optimization. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 12553–12569. Association for Computational Linguistics, 2025.
- [197] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2021.
- [198] Haolin Wu, Jing Chen, Ruiying Du, Cong Wu, Kun He, Xingcan Shang, Hao Ren, and Guowen Xu. Clad: Robust audio deepfake detection against manipulation attacks with contrastive learning. *arXiv preprint arXiv:2404.15854*, 2024.
- [199] Haolin Wu, Jing Chen, Ruiying Du, Cong Wu, Kun He, Xingcan Shang, Hao Ren, and Guowen Xu. CLAD: robust audio deepfake detection against manipulation attacks with contrastive learning. *CoRR*, abs/2404.15854, 2024.
- [200] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *IEEE transactions on information forensics and security*, 13(11):2884–2896, 2018.

- [201] Xinwei Wu, Junzhuo Li, Minghui Xu, Weilong Dong, Shuangzhi Wu, Chao Bian, and Deyi Xiong. Depn: Detecting and editing privacy neurons in pretrained language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.
- [202] Zhizheng Wu, Junichi Yamagishi, Tomi Kinnunen, Cemal Hanilçi, Mohammed Sahidullah, Aleksandr Sizov, Nicholas Evans, Massimiliano Todisco, and Hector Delgado. Asvspoof: the automatic speaker verification spoofing and countermeasures challenge. *IEEE Journal of Selected Topics in Signal Processing*, 11(4):588–604, 2017.
- [203] Yang Xie, Zhenchuan Zhang, and Yingchun Yang. Siamese network with wav2vec feature for spoofing speech detection. In *Interspeech*, pages 4269–4273, 2021.
- [204] Yuankun Xie, Yi Lu, Ruibo Fu, Zhengqi Wen, Zhiyong Wang, Jianhua Tao, Xin Qi, Xiaopeng Wang, Yukun Liu, Haonan Cheng, et al. The codecfake dataset and countermeasures for the universal detection of deepfake audio. *arXiv preprint arXiv:2405.04880*, 2024.
- [205] Bing Xu. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.
- [206] Keyang Xu, Tongzheng Ren, Shikun Zhang, Yihao Feng, and Caiming Xiong. Unsupervised out-of-domain detection via pre-trained transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1052–1061, 2021.
- [207] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Annual Conference on Neural Information Processing Systems*, pages 7333–7343, 2019.
- [208] Junichi Yamagishi, Xin Wang, Massimiliano Todisco, Md Sahidullah, Jose Patino, Andreas Nautsch, Xuechen Liu, Kong Aik Lee, Tomi Kinnunen, Nicholas Evans, et al. Asvspoof 2021: accelerating progress in spoofed and deepfake speech detection. In *ASVspoof 2021 Workshop-Automatic Speaker Verification and Spoofing Countermeasures Challenge*, 2021.
- [209] An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, et al. Qwen2. 5-1m technical report. *arXiv preprint arXiv:2501.15383*, 2025.

- [210] Xu Yang, Chongyang Gao, Hanwang Zhang, and Jianfei Cai. Auto-parsing network for image captioning and visual question answering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2197–2207, 2021.
- [211] Yuanshun Yao, Xiaojun Xu, and Yang Liu. Large language model unlearning. In *Socially Responsible Language Modelling Research*, 2023.
- [212] Jiangyan Yi, Chenglong Wang, Jianhua Tao, Xiaohui Zhang, Chu Yuan Zhang, and Yan Zhao. Audio deepfake detection: A survey. *arXiv preprint arXiv:2308.14970*, 2023.
- [213] Jingwei Yi, Rui Ye, Qisi Chen, Bin Zhu, Siheng Chen, Defu Lian, Guangzhong Sun, Xing Xie, and Fangzhao Wu. On the vulnerability of safety alignment in open-access llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9236–9260, 2024.
- [214] Hibiki Yokoyama, Rikuto Tsuchida, Kosei Buma, Sho Miyakawa, Takehito Utsuro, and Masaharu Yoshioka. Aggregating impressions on celebrities and their reasons from microblog posts and web search pages. In *Proceedings of the 3rd Workshop on Knowledge Augmented Methods for NLP*, pages 59–72, 2024.
- [215] Charles Yu, Sullam Jeoung, Anish Kasi, Pengfei Yu, and Heng Ji. Unlearning bias in language models by partitioning gradients. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6032–6048, 2023.
- [216] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [217] Ted Zadori, Ahmet Ustun, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: extremely parameter efficient moe for instruction tuning. *arXiv preprint arXiv:2309.05444*, 2023.
- [218] Chang Zeng, Xin Wang, Erica Cooper, Xiaoxiao Miao, and Junichi Yamagishi. Attention back-end for automatic speaker verification with multiple enrollment utterances. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6717–6721. IEEE, 2022.
- [219] Wenjun Zeng, Dana Kurniawan, Ryan Mullins, Yuchi Liu, Tamoghna Saha, Dirichi Ike-Njoku, Jindong Gu, Yiwen Song, Cai Xu, Jingjing Zhou, et al. Shieldgemma 2:

- Robust and tractable image content moderation. *arXiv preprint arXiv:2504.01081*, 2025.
- [220] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. Styleswin: Transformer-based gan for high-resolution image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11304–11314, 2022.
 - [221] Dawen Zhang, Pamela Finckenberg-Broman, Thong Hoang, Shidong Pan, Zhenchang Xing, Mark Staples, and Xiwei Xu. Right to be forgotten in the era of large language models: Implications, challenges, and solutions. *arXiv preprint arXiv:2307.03941*, 2023.
 - [222] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.
 - [223] Kuiyuan Zhang, Zhongyun Hua, Rushi Lan, Yushu Zhang, and Yifang Guo. Phoneme-level feature discrepancies: A key to detecting sophisticated speech deepfakes. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 1066–1074. AAAI Press, 2025.
 - [224] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.
 - [225] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.
 - [226] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.
 - [227] Xiaohui Zhang, Jiangyan Yi, Chenglong Wang, Chu Yuan Zhang, Siding Zeng, and Jianhua Tao. What to remember: Self-adaptive continual learning for audio deepfake detection. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence*,

- EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 19569–19577. AAAI Press, 2024.
- [228] Xueyao Zhang, Xiaohui Zhang, Kainan Peng, Zhenyu Tang, Vimal Manohar, Yingru Liu, Jeff Hwang, Dangna Li, Yuhao Wang, Julian Chan, et al. Vevo: Controllable zero-shot voice imitation with self-supervised disentanglement. *arXiv preprint arXiv:2502.07243*, 2025.
 - [229] Ying Zhao and Jinjun Chen. A survey on differential privacy for unstructured data content. *ACM Comput. Surv.*, 54(10s):207:1–207:28, 2022.
 - [230] Lin Zhu, Yi Fang, Shuting Liu, Hong-Bin Shen, Wesley De Neve, and Xiaoyong Pan. Toxdl 2.0: Protein toxicity prediction using a pretrained language model and graph neural networks. *Computational and Structural Biotechnology Journal*, 27:1538–1549, 2025.
 - [231] Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.