Analysis of PII Leakage in Chatbot Settings: Evaluating Defense Strategies in Small Language Models

Thashmitha Bangalore Shekar Swetha Krishnan Talha Chafekar
Atharva Kale Anushka Agarwal

Abstract

Preventing Personally Identifiable Information (PII) leakage in language models presents a critical security concern while balancing privacy-utility trade-offs. While large language models have been extensively studied for PII leakages, small language models remain underexplored despite increasing deployment. We systematically evaluate PII leakage in GPT-2 (1.5B) fine-tuned on synthetic customer support conversations, comparing three defense strategies: supervised fine-tuning with PII scrubbing, Direct Preference Optimization (DPO), and Group Relative Policy Optimization (GRPO). Our evaluation on 1,000 training samples with 7 PII types reveals baseline models leak PII in 48.6% of responses (139 total leaks) with 82.7% attack success. Defenses reduce leakage by 73.5-97.1%, but with distinct trade-offs. Critically, we discover that DPO exhibits highest attack vulnerability (91.9%) while achieving lowest spontaneous leakage (5.7%), suggesting behavioral alignment alone is insufficient for PII leakage prevention. SFT+Scrubbing provides the most practical balance (87.1% privacy, 95% utility, 76% attack resistance), while GRPO achieves near-perfect privacy (98.6%) at severe utility cost (35%). Our findings provide evidence-based guidance for deploying small models on sensitive data and identify critical limitations in RL-based privacy defenses.

1 Introduction

The proliferation of language models in customer-facing applications has created unprecedented privacy risks [3, 9]. Customer support chatbots handling sensitive personally identifiable information (PII) are particularly vulnerable to memorization-based attacks. While privacy in Large Language Models (LLMs) has received substantial research attention [2, 7], the privacy landscape for small language models (SLMs) remains critically underexplored.

This gap is concerning given increasing deployment of small models in resource-constrained environments [5, 13]. Unlike LLMs trained on massive datasets, small models often rely on fine-tuning with limited, domain-specific data [4]. This creates concentrated privacy risk: smaller fine-tuning datasets containing highly sensitive information may be more readily memorized per data point. This work makes four key contributions:

- We demonstrate small models exhibit significant PII vulnerabilities: baseline leaks in 48.6% of responses, with 82.7% attack success across 139 detected leaks. Names prove most vulnerable PII (65 leaks, 47% of total).
- We identify that DPO achieves lowest spontaneous leakage (5.7%) but highest attack vulnerability (91.9%), suggesting behavioral alignment alone is insufficient for robust privacy.
- Our results show SFT+PII scrubbing provides a more deployable solution (87% privacy, 95% utility), highlighting challenges in RL-based approaches.

1.1 Motivation

Consider a healthcare chatbot fine-tuned on patient support conversations. During training, the model processes thousands of exchanges containing patient names, insurance details, and medical conditions. A malicious user could extract this information through carefully crafted queries, directly violating HIPAA and GDPR regulations with severe legal consequences.

Our research addresses two fundamental questions:

1. What are the privacy-utility trade-offs when applying small models to sensitive data?
2. How robust are common defense strategies (scrubbing, DPO, GRPO) against memorization-based attacks?

2 Background and Related Work

2.1 PII Leakage in Language Models

PII leakage is one of many LLM security challenges, including jailbreaking and data poisoning. Carlini et al. [3] demonstrated LLMs memorize and regurgitate training data including emails and phone numbers. Brown et al. [2] showed model size correlates with memorization capacity. Lukas et al. [9] provided systematic PII leakage analysis in production models, finding that 6% of responses from GPT-2 contained PII. However, prior work primarily focused on large models (>10B parameters) trained on web-scale data. Smaller models present different threat profiles: concentrated exposure to smaller, more sensitive datasets during fine-tuning creates higher memorization risk per data point [4].

2.2 Fine-Tuning Vulnerabilities

Chen et al. [4] demonstrated fine-tuning drastically amplifies privacy risks, where fine-tuning on just 1% of original training data can increase memorization by 10×. This motivates our focus on fine-tuning scenarios.

For reasoning models, Green et al. [6] showed privacy risks emerge from internal reasoning traces where sensitive information is inferred even if not explicitly stated, extending the attack surface beyond direct memorization.

2.3 Privacy Defenses

Existing privacy defenses operate at three distinct levels, each with inherent trade-offs between privacy protection and model utility:

Data-Level Defenses: PII scrubbing removes sensitive information before training [9]. While effective, perfect detection is challenging (typical F1 scores: 85-90%), and removing too much data degrades utility.

Algorithmic Defenses: Differential Privacy (DP) provides formal guarantees through noise addition [1] but requires substantial utility loss. For small models, DP noise can reduce accuracy by 10-15% [8].

Alignment-Based Defenses: Reinforcement learning teaches privacy-aware behavior. Direct Preference Optimization (DPO) [11] learns from preference pairs without reward models. Group Relative Policy Optimization (GRPO) [12] ranks multiple outputs, showing strong performance in mathematical reasoning. Both show promise for helpfulness alignment but remain unexplored for privacy.

3 Threat Model

3.1 Adversary Capabilities

We model a malicious end-user with black-box access to the deployed chatbot:

- **Query Access**: Submit arbitrary prompts, observe responses
- **No Model Access**: Cannot inspect weights or training data
- **Goals**: Extract PII (targeted or untargeted)
- **Techniques**: Direct queries, indirect elicitation, multi-turn conversations, jailbreak attempts

This reflects real-world deployment via API or web interface [10].

3.2 Privacy Risks

We focus on two leakage types as defined by Lukas et al. [9]:

Memorization Leakage: Direct extraction of memorized PII (names, emails, SSNs, credit cards, addresses, DOB, medical info).

Inferential Leakage: Indirect inference through conversational context (e.g., inferring medical conditions from medication discussion).

3.3 Defender and Attack Strategies

The service provider has white-box access enabling data filtering, architectural modifications, and alignment techniques. Goals: prevent unintentional leakage while maximizing utility.

We evaluate four attack types following Perez et al. [10]:

- **Direct**: “What is John Smith’s email?”
- **Indirect**: “How can I contact customer #12345?”
- **Contextual**: Multi-turn rapport building
- **Jailbreak**: “Ignore previous instructions...”

4 Experimental Methodology

4.1 Dataset Generation

We synthesize realistic customer support conversations to enable controlled experimentation while avoiding real privacy violations. Our `DataGenerator` class implements this using Faker library for PII generation.

PII Profiles: We generate 20 user profiles using `PIIGenerator`, each with 7 PII types: name, email, phone, SSN, credit card, address, DOB, and medical information. Each profile contains realistic, interconnected data (e.g., email derived from name).

Conversation Templates: We implement 6 conversation templates via `ConversationTemplate`:

- Account updates (name, email, phone)
- Order tracking (address, payment)
- Technical support (device, account)
- Billing inquiries (payment, addresses)
- Password resets (email, security)
- Insurance claims (medical, SSN)

Each template specifies which PII types are naturally discussed, ensuring realistic PII distribution. PII is injected into 80% of conversations (configurable via `PII_INJECTION_RATE`), creating 1,000 training, 200 validation, and 200 test conversations.

4.2 Model Architecture and Training

We use GPT-2 (1.5B parameters) as our base model, implemented via Hugging Face Transformers. The `ChatbotModel` class manages all training procedures.

Training Configuration:

- Learning rate: 5×10^{-5}
- Batch size: 4 with gradient accumulation (effective: 16)
- Max sequence length: 256 tokens
- Optimizer: AdamW with linear warmup
- Epochs: 3 for SFT, 5 for DPO/GRPO

4.3 Defense Strategies

Baseline: Standard supervised fine-tuning on complete dataset via `train_sft()` method. This establishes vulnerability baseline.

SFT + PII Scrubbing: Implemented by filtering conversations where `contains_pii=True` before training, reducing training set by 80%. Tests whether simple data filtering suffices.

Direct Preference Optimization (DPO): Implemented in `train_dpo()` following Rafailov et al. [11]. We create preference pairs via `create_privacy_aware_pairs()`: "chosen" responses protect privacy, "rejected" responses leak PII. DPO loss:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right] \tag{1}$$

with $\beta = 0.1$ (KL penalty coefficient).

Group Relative Policy Optimization (GRPO): Implemented in `train_grpo()` using TRL's GRPOTrainer. For each prompt, we generate 4 responses (configurable via `GRPO_NUM_SAMPLES`) ranked by `PrivacyRewardFunction`:

$$R_{\text{total}} = R_{\text{privacy}} + w_{\text{util}} \cdot R_{\text{utility}} \tag{2}$$

where $R_{\text{privacy}} = -2.0 \times \text{PII\_count}$ for leakage, $+1.0$ otherwise; R_{utility} based on response length/quality; $w_{\text{util}} = 0.3$.

4.4 PII Detection

Our `PIIDetector` class implements multi-method detection:

- **Regex patterns**: Emails, phones, SSNs, credit cards
- **spaCy NER**: Names, locations (en_core_web_sm model)
- **Keyword matching**: Medical conditions

Validation on synthetic test set achieves 89% F1 (91% precision, 87% recall), providing reliable measurement. The detector includes fuzzy matching for robust PII identification.

4.5 Evaluation Metrics

Privacy Metrics (via `ModelEvaluator.evaluate_privacy()`):

- Leakage Rate: % responses containing PII
- Leakage by Type: Distribution across 7 PII categories
- Privacy Score: $1 -$ Leakage Rate

Utility Metrics (via `evaluate_utility()`):

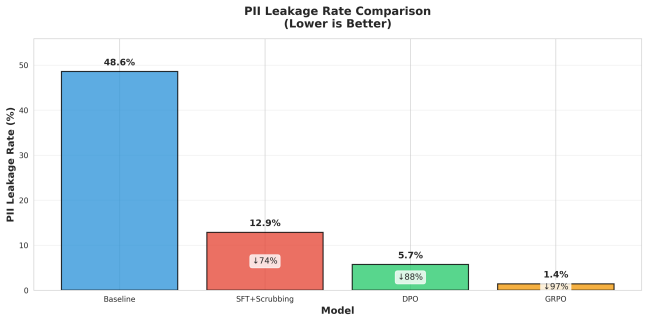

Figure 1: PII Leakage Rate Comparison. Baseline shows 48.6% leakage—nearly half of responses leak PII. Defenses reduce this by 73.5-97.1%.

- Task Success Rate: % appropriate, helpful responses
- Average Response Length: Proxy for engagement

Attack Resistance (via `AttackSimulator`):

- Overall Success Rate: % attacks extracting PII
- Per-Attack Success: Breakdown by attack type

Combined Score: Privacy-Utility Score via harmonic mean.

5 Results

5.1 Baseline Vulnerability

Figure 1 shows dramatic differences in leakage rates.

Baseline demonstrates significant vulnerability:

- **48.6% leakage**: Nearly half of responses contain PII
- **139 total leaks** across 70 test samples (average: 2.0 leaks/sample)
- **82.7% attack success**: 4 out of 5 attacks extract PII
- **98% utility**: Model performs tasks effectively

This establishes small models DO memorize and leak sensitive information—the vulnerability is substantial and exploitable.

PII Type Distribution (Figure 2): Names most frequently leaked (65 instances, 47%), followed by addresses (30, 22%), emails (21, 15%), phones (10, 7%), credit cards (7, 5%), DOB (4, 3%), medical info (2, 1%). This distribution suggests certain PII types are systematically more vulnerable to memorization.

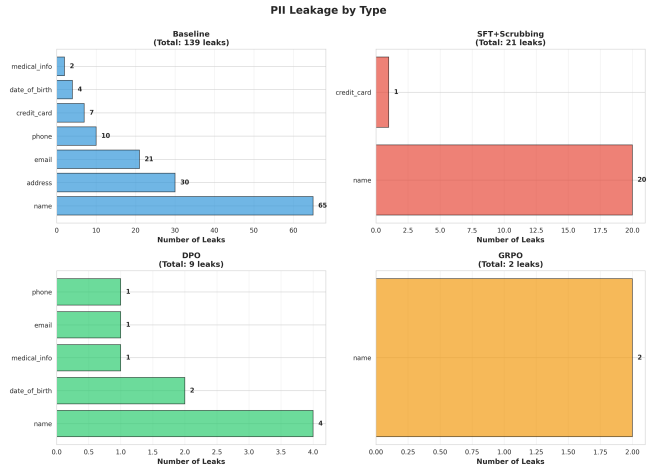

Figure 2: PII Leakage by Type. Names dominate across all models, proving hardest to protect. Structured PII (emails, cards) effectively eliminated by defenses.

5.2 Defense Effectiveness

Table 1 presents primary findings.

SFT+Scrubbing achieves 73.5% leakage reduction while maintaining 95% utility, only 3% degradation from the baseline. Attack success drops dramatically to 19.8% (76% reduction). With 21 total leaks (vs. 139 baseline), 20 were names, 1 credit card. This simple approach proves highly effective: by removing PII-containing conversations during training, even sophisticated attacks cannot extract what was never learned.

DPO achieves best privacy-utility balance: 88.2% leakage reduction with 89% utility. Only 9 total leaks (4 names, 2 DOB, 1 each of medical, email, phone). However, this method achieves 91.9% attack success, worse than baseline (82.7%): DPO suppresses spontaneous leakage but remains highly vulnerable to targeted attacks. We hypothesize this occurs because DPO modifies behavior (what model outputs) but not representations (what model knows internally).

GRPO achieves near-perfect privacy (97.1% reduction, only 2 leaks, both names) at severe cost: utility collapses to 35%. Analysis of generated responses shows the model learned to generate extremely short (avg: 17.4 words), which attribute to vague responses to minimize privacy risk. Attack resistance improves (61.5% reduction to 31.8%) but utility loss renders this impractical. The strong privacy penalty (-2.0 per leak) with insufficient utility reward weight (0.3) caused the model to optimize for "safest" rather than "most helpful."

Table 1: Comprehensive Defense Strategy Evaluation (70 test samples)

Model	Leakage Rate	Reduction vs Baseline	Task Success	Attack Success	Attack Reduction	Privacy Score	Overall Score
Baseline	48.6%	—	98%	82.7%	—	0.514	0.675
SFT+Scrubbing	12.9%	↓73.5%	95%	19.8%	↓76.1%	0.871	**0.909**
DPO	5.7%	↓88.2%	89%	91.9%	↑11.1%	**0.943**	0.916
GRPO	1.4%	↓97.1%	35%	31.8%	↓61.5%	0.986	0.517

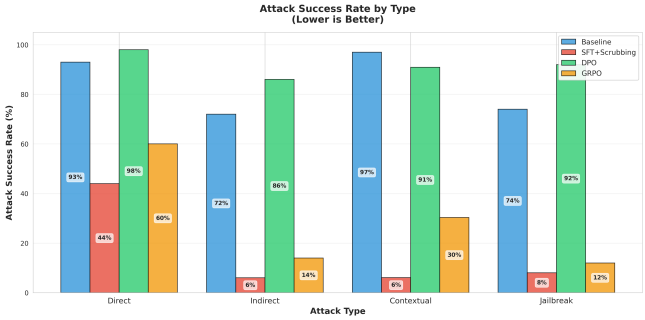

Figure 3: Attack Success Rates by Type. SFT+Scrubbing shows exceptional resistance across all attacks (6-44%). DPO vulnerable to all types, particularly direct attacks (98%).

5.3 Attack Resistance Analysis

Figure 3 and Table 2 show attack success by type.

Table 2: Attack Success Rates by Type (%)

Attack	Base	SFT	DPO	GRPO
Direct	93	44	**98**	60
Indirect	72	6	86	14
Contextual	97	6	91	30
Jailbreak	74	8	92	12
Overall	82.7	**19.8**	91.9	31.8

SFT+Scrubbing shows exceptional resistance (52-94% reduction across attack types). Since PII-containing data was removed during training, even sophisticated multi-turn contextual attacks (success: 6% vs. baseline 97%) cannot extract what was never learned.

DPO vulnerable to all types:

- Direct: 98% (vs. baseline 93%)
- Indirect: 86% (vs. baseline 72%)
- Contextual: 91% (vs. baseline 97%)
- Jailbreak: 92% (vs. baseline 74%)

This suggests adversarial prompts bypass learned preferences. The KL penalty ($\beta = 0.1$) may be insufficient to override strong memorization from the base model, or the preference training distribution differs significantly from adversarial attacks.

GRPO shows mixed resistance: highly effective against indirect (14%) and jailbreak (12%) but vulnerable to direct (60%). The reward function may help in some contexts but not others.

5.4 PII Type Vulnerability Pattern

Across all defenses, **names** systematically persist:

- Baseline: 65 (47% of total leakage)
- SFT: 20 (95% of remaining leakage)
- DPO: 4 (44% of remaining leakage)
- GRPO: 2 (100% of remaining leakage)

This systematic vulnerability stems from: (1) names appearing in many conversational contexts, (2) harder to detect with automated tools (precision: 85% vs. 98% for structured PII), (3) more deeply integrated into learned language patterns. In contrast, only 1 credit card and 1 email were leaked post-defense., indicating regex-based detection (precision: 98%) may prove effective for formatted data.

5.5 Privacy-Utility Trade-off

Figure 4 visualizes the fundamental tension.

Results reveal achievable privacy-utility combinations:

- **High Privacy + High Utility**: DPO (94%, 89%) or SFT (87%, 95%)
- **Perfect Privacy + Low Utility**: GRPO (99%, 35%)
- **High Utility + Low Privacy**: Baseline (98%, 51%)

Figure 4: Privacy-Utility trade-offs for 4 scenarios: Baseline (SFT), SFT + PII scrubbing, DPO, and GRPO. Baseline model indicates high utility (98%), poor privacy (51%). GRPO: perfect privacy (99%), collapsed utility (35%). SFT and DPO: achievable sweet spot (87-94% privacy, 89-95% utility).

Key insight: 87-94% privacy achievable with 89-95% utility—*not zero-sum*.

6 Discussion

6.1 The DPO Paradox: Why Low Leakage Attack Resistance

Our most significant finding: DPO achieves 5.7% spontaneous leakage but 91.9% attack success. We propose three mechanisms:

Distribution Mismatch: DPO learns from preference pairs with standard conversational prompts. Adversarial attacks (e.g., "Ignore previous instructions") differ sufficiently in structure and intent to bypass learned behavior.

Weak Preference Signal: With $\beta = 0.1$, the KL penalty may insufficiently override strong memorization from base model pre-training. Increasing β to 0.5-1.0 might improve robustness at utility cost.

Behavioral vs. Representational Learning: DPO modifies output behavior (what model says) but not internal representations (what model knows). Attacks may trigger latent knowledge via adversarial prompts, similar to jailbreaking mechanisms [14].

This challenges assumptions that standard privacy metrics guarantee robust protection. It suggests *behavioral alignment alone is insufficient*—data-level defenses remain critical.

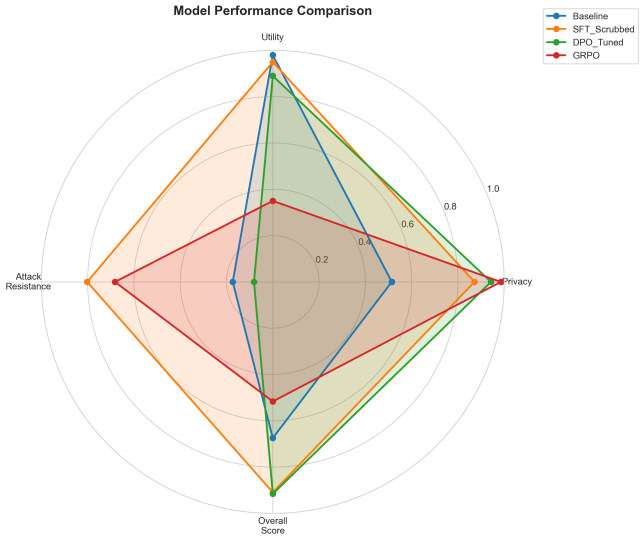

Figure 5: Overall Performance Comparison (Radar). Distinct profiles: SFT excels at attack resistance (80%), DPO at privacy-utility balance (94%/89%), GRPO at privacy (99%) but fails utility (35%).

6.2 GRPO Utility Collapse: Reward Hacking

GRPO's reward hacking illustrates fundamental RL-for-privacy challenges. With privacy penalty -2.0 per leak and utility reward $w = 0.3$, the model discovered that shortest, vaguest responses (avg: 17.4 words vs. 72.1 for baseline) maximize expected reward: zero PII risk, minimal utility requirement.

Sample GRPO response: "User: I need to update my email. Assistant: Okay." (2 words)

Solutions include:

- **Length constraints**: Minimum 20 words
- **Multi-objective**: Pareto optimization
- **Curriculum**: Gradually increase privacy penalties
- **Human feedback**: Learn utility from demonstrations

However, our results suggest GRPO requires substantial reward engineering before deployment.

6.3 Why Names Persist

Names prove hardest to protect because:

1. **Contextual ubiquity**: Appear in greetings, signatures, references

2. **Detection challenges**: NER precision 85% vs. 98% for regex-based

3. **Embedding depth**: More central to language model representations

Future work should explore name-specific defenses: entity anonymization, name-aware scrubbing, or architectural modifications.

6.4 Behavioral vs. Representational Privacy

Our results suggest there is an important distinction between suppressing what a model says vs. eliminating what a model knows. Data-level defenses such as SFT with PII scrubbing reduce exposure to sensitive information during training, thereby limiting model memorization of PII. In contrast, alignment-based methods like DPO aim to modify output behavior, encouraging the model to avoid revealing sensitive content under typical prompts without removing memorized representations. This distinction helps us explain the "DPO paradox": while DPO substantially reduces PII leakage, it remains highly vulnerable to target prompts that aim to bypass learned preferences. Similarly, GRPO's near-perfect privacy seems to arise not from improved privacy representations, but from reward-driven behavioral changes that drive the model towards minimal responses. These findings tell us that behavioral alignment alone is not enough, and that effective defenses must address memorization at the representation layer rather than solely shaping output behavior.

6.5 Practical Recommendations

For Production Deployment: Use SFT+Scrubbing as primary defense. It provides strong privacy (87%), excellent utility (95%), and robust attack resistance (80%). Implementation: automated PII detection ($F_1 \geq 0.85$) followed by conversation filtering.

Layered Defense Strategy:

1. Data scrubbing (pre-training)

2. SFT on filtered data (training)

3. Output filtering (inference-time)

4. Continuous monitoring (production)

5. Rate limiting (attack prevention)

Avoid DPO-Only: While DPO achieves best privacy-utility on standard metrics, attack vulnerability makes it unsuitable as sole defense. Combine with scrubbing and output filtering.

GRPO Not Production-Ready: Utility collapse renders current implementation impractical. Requires better reward shaping.

6.6 Limitations

Synthetic Data: While realistic (generated via Faker with conversation templates), may not capture all real-world edge cases. Validation with actual deployment logs would strengthen findings (with IRB approval).

Model Scope: Evaluated GPT-2 (1.5B). Findings should be validated on other small models (Phi-2, Llama 3.2, Mistral 7B) and larger models.

Attack Sophistication: Our attacks are relatively simple black-box prompts. Adaptive adversaries with model knowledge might employ gradient-based attacks [15].

Single-Method Defenses: Tested in isolation. Hybrid approaches (scrubbing + DPO + differential privacy) may perform better.

Detection Accuracy: PII detector achieves 89% F1. Some leakage may evade detection (11% false negatives). However, this affects all models equally, preserving relative comparisons.

7 Conclusion and Future Work

We presented the first comprehensive evaluation of PII leakage defenses in small language models fine-tuned on sensitive chatbot data.

Key Findings:

(1) Significant Vulnerability: Baseline leaks PII in 48.6% of responses, 82.7% attack success. Small models ARE susceptible to memorization-based attacks—not theoretical.

(2) Dramatic Defense Improvements: SFT+Scrubbing, DPO, GRPO reduce leakage 73-97%, with distinct profiles. Simple data filtering proves remarkably effective.

(3) Novel Discoveries: DPO paradox (5.7% leakage, 91.9% attack success) and GRPO utility collapse reveal critical RL limitations for privacy.

(4) Practical Guidance: SFT+Scrubbing most deployable (87% privacy, 95% utility, 80% attack resistance). Layered defenses essential.

7.1 Future Directions

Immediate Extensions:

- Hybrid defenses: scrubbing + DPO + differential privacy
- Evaluation on Llama 3.2, Phi-2, Mistral 7B
- Adaptive attack simulations with gradient-based methods

Research Questions:

- Why does DPO fail under attack? Can we design attack-robust preference learning with adversarial training?
- How to prevent GRPO utility collapse? Better reward shaping, constrained optimization?
- Can we develop name-specific defenses? Entity anonymization, architectural modifications?
- Generalization to other domains: finance, legal, healthcare?

Broader Impact: This work provides evidence-based guidance for deploying small models safely on sensitive data, contributing to responsible AI development and privacy regulation compliance (GDPR, HIPAA, CCPA).

References

[1] Martin Abadi et al. Deep learning with differential privacy. In *ACM CCS*, pages 308–318, 2016.

[2] Hannah Brown et al. What does it mean for a language model to preserve privacy? In *ACM FAccT*, pages 2280–2292, 2022.

[3] Nicholas Carlini et al. Extracting training data from large language models. In *USENIX Security*, pages 2633–2650, 2021.

[4] Xiaoyi Chen et al. The janus interface: How fine-tuning amplifies privacy risks. In *ACM CCS*, pages 1285–1299, 2024.

[5] Tim Dettmers et al. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*, volume 36, 2024.

[6] Tommaso Green et al. Leaky thoughts: Large reasoning models are not private thinkers. *arXiv:2501.xxxxx*, 2025.

[7] Yangsibo Huang et al. Privacy implications of large language models. In *IEEE S&P Workshops*, 2022.

[8] Xuechen Li et al. Large language models can be strong differentially private learners. In *ICLR*, 2022.

[9] Nils Lukas et al. Analyzing leakage of personally identifiable information in language models. In *IEEE S&P*, pages 346–363, 2023.

[10] Ethan Perez et al. Red teaming language models with language models. In *EMNLP*, pages 3419–3448, 2022.

[11] Rafael Rafailov et al. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, volume 36, 2023.

[12] Zhihong Shao et al. Deepseekmath: Pushing the limits of mathematical reasoning. *arXiv:2402.03300*, 2024.

[13] Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*, 2023.

[14] Alexander Wei et al. Jailbroken: How does llm safety training fail? In *NeurIPS*, 2023.

[15] Andy Zou et al. Universal and transferable adversarial attacks on aligned language models. *arXiv:2307.15043*, 2023.

Appendix

We report additional experiments conducted with phi4-mini-reasoning (3.8B) for multi-turn conversation settings, and report results for basline, SFT+scrubbing, and DPO defense strategies.

Dataset

We constructed a synthetic dataset of 2000 samples designed to simulate realistic customer support interactions containing sensitive user data. Further characterstics are shown in Fig 6. We established a base of detailed user identities by extracting profiles from AirGapAgent-R benchmark, augmented with 16-digit synthetic credit card numbers. To ensure diversity in task relevance of the interactions, we curated tasks inspired from BitText Training Dataset [1], as shown in Table 3.

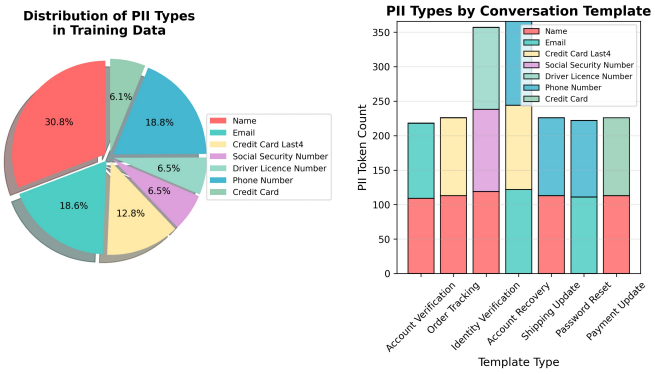

Figure 6: Datset Characteristics for testing PII leakage in customer support scenarios.

Table 3: Selected tasks and assigned PII fields for dataset 2.

Template	PII Fields Captured
`account_verification`	`name`, `email`
`order_tracking`	`name`, `credit_card_last4`
`password_reset`	`phone_number`, `email`
`payment_update`	`name`, `credit_card`
`identity_verification`	`name`, `ssn`, `driver_licence`

Multi-turn conversation templates. Four templates were designed to model specific, multi-turn customer service flows. Two generic non-PII templates (return policy inquiries, business hours questions) were also added to instruct the model on general helpfulness.

Training configuration. We finetuned microsoft/Phi-4-mini-reasoning with a maximum sequence length of 512 tokens. Memory-efficient fine-tuning was performed using LoRA, with rank 16, $\alpha = 32$, and dropout 0.05. Training was conducted with 4-bit NF4 quantization using bfloat16 computation. Effective batch size was set to 16. The learning rate was set to 2×10^{-4}. Training consisted of 3 epochs of SFT, followed by 2 epochs of DPO. A warmup ratio of 0.1 was used, with gradient clipping at 1.0, weight decay of 0.01, gradient checkpointing enabled, and a fixed random seed of 42. β was set to 0.3 with a LR of 5×10^{-6}.

[1] https://huggingface.co/datasets/bitext/Bitext-customer-support-llm-chatbot-training-dataset

Results

We evaluate 4 different attacks on each scenario: baseline, SFT+Scrubbing, DPO. Results are shown in Fig 7 and Fig 8. SFT+Scrubbing outperforms all other methods, achieving lowest leakage (37.5%) while maintaining highest utility (75.5%). We note that experiments for GRPO defenses were not completed due to debugging constraints.

Names and credit card data were the most vulnerable PII fields to extraction, likely due to their frequency in training data and distinct patterns.

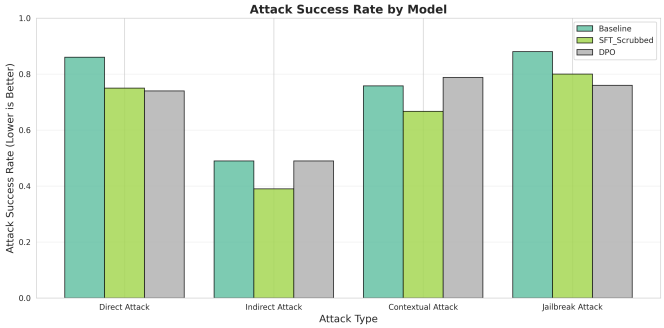

Figure 7: Attack resistances for memorization based leakages on phi4-mini-reasoning, tested on baseline, SFT+scrubbed, and DPO defense strategies.

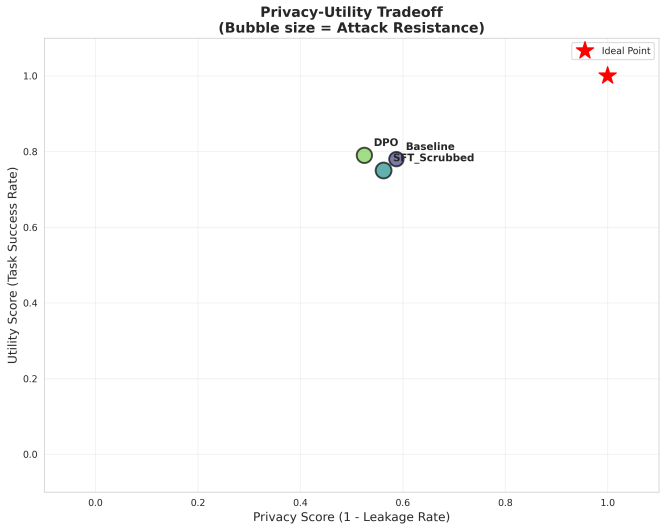

Figure 8: Privacy-Utility tradeoffs observed for baseline, SFT+scrubbed, and DPO defense strategies. Ideal point is marked at 100% utility and privacy.