MEASURING AND MITIGATING PRIVATE DATA LEAKAGE IN AUTOMATED PROMPT OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated prompt optimization (APO) has emerged as a lightweight method for adapting a large language model (LLM) to perform well on a particular task. APO edits and refines input prompt text, a sample-efficient and flexible alternative to parameter-based fine-tuning. For tasks that involve interacting with users' private data, further prompt quality improvements are possible by performing APO directly on real users' examples rather than public or synthetic data. However, this exposes the risk of private information leaking into the optimized prompt. In this paper, we demonstrate that private data leakage can indeed occur when APO is run using training data containing private information. We introduce empirical privacy auditing methods to quantify this leakage based on carefully calibrated matching of token windows between training data examples and the optimized prompt. We then present some simple modifications to APO, specifically the meta-prompts used by the mutator and autorater models, that significantly reduce private data leakage without hampering the quality of the optimized prompt. Comparison of different leakage mitigation strategies in terms of their privacy-utility trade-offs shows that private data leakage through APO can be mitigated with minimal impact on the performance of the optimized prompt.

1 INTRODUCTION

Frontier large language models (LLMs) are trained for general-purpose text generation or reasoning, and frequently need fine-tuning to maximize their performance on a specific task. Gradient-based model fine-tuning methods, even parameter-efficient methods like LoRA (Hu et al., 2021), are computationally expensive (involving backward passes) and require 'white-box' access to update *weights*. Customers and providers of 'LLM-as-a-Service' (LLaaS) products have sought alternative manners of tuning-to-task where only 'black-box' model access (i.e., a forward pass) is necessary.

From this demand has arisen a multitude of effective gradient-free, text-space approaches to optimizing the task *prompt* used during inference. These 'automated prompt optimization' (APO) methods observe inference queries and responses and pass this information to 'autorater' and 'mutator' LLMs, which evaluate performance of and propose improvements to the inference prompt, respectively. Such APO methods are in heavy use today by LLaaS providers (Lee and Nardini, 2024; Anthropic, 2024; Subramanian et al., 2024; Team, 2025; OpenAI; Sveshnikov and Samuylova, 2026; Tekur et al., 2026).

An orthogonal consideration to the manner of tuning is what data to tune on. As a means of mitigating train-inference skew, using in-distribution private data (e.g., logs of real user use) is desirable, but steps must be taken to ensure that private information is not revealed (Abadi et al., 2016; McMahan et al., 2018; Ponomareva et al., 2023). This holds equally when tuning via APO (Hong et al., 2024). The optimized prompt resulting from APO is typically significantly longer than the initial prompt, and can include in-context learning (ICL) examples of queries and responses. The possibility exists that text excerpts from private training data can be memorized, with verbatim or nearly verbatim sequences ending up in the optimized prompt and/or any inferences performed with the optimized prompt. Figure 1 shows instances of this occurring in real APO experiments where privacy protections were *not* employed.

We seek to harness the dual benefits of tuning via APO and tuning on private data. The focus of this paper is the development of practical, privacy-preserving APO, to achieve inference quality

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(a) ProTeGi, Enron. Verbatim span copied into the optimized prompt (112 words, 37% of a training example):

> *The change appears necessary since there is no 20.1 B. This must have been left over from the GE contract. —- Forwarded by [NAME]/Corp/Enron on 07/07/2000 03:41 PM —- From: [NAME] 07/07/2000 03:17 PM To: [NAME]/ENRON_DEVELOPMENT@ENRON_DEVELOPMENT, [NAME]/Corp/Enron@Enron cc: [NAME]/Corp/Enron@ENRON Subject: Re: ABB rev 10 FYI —- Forwarded by [NAME]/Corp/Enron on 07/07/2000 03:17 PM —- "[NAME]" <[EMAIL]> on 07/07/2000 03:12:24 PM*
> *To: <[EMAIL]> cc: <[EMAIL]> Subject: Re: ABB rev 10*
> *My comment on ABB rev 10: delete ",OR FOR THE AMOUNTS PAYABLE TO SELLER PURSUANT TO SECTION 20.1(B) HEREOF" from the end of Section 23.4. Regards, [NAME] [LAW FIRM]*

(b) ProTeGi, LongLaMP-3. 250 of 389 words (64%) copied verbatim from a training example, e.g.:

> *"... They were wonderful when I was cutting hair – so all you Stylist's out there... Bolero shrugs are great when cutting/coloring hair ... my size is [SIZE], size 12 tops. I ordered the XL, they fit fine, but I COULD have gotten away with a Large."*

Figure 1: Verbatim training-data leakage in prompts optimized with the baseline meta-prompt on non-redacted data. In both cases the optimizer inserted a training example into the prompt as an in-context learning example. Personal identifiers are masked for the purpose of this illustration.

improvements while avoiding the type of leakage evidenced in Figure 1. Our solution consists of two parts. The first is empirical 'prompt inspection' audits with training and held-out data to measure the likelihood that training data has leaked, i.e., to measure whether instances of matches between the prompt and a training example are elevated to a statistically significant level. The second is a battery of potential APO modifications, including meta-prompt modifications as well as personally identifiable information (PII) redaction, to decrease the chances of training data memorization. Crucially, the modifications we consider do not increase (to any significant extent) the computational cost of APO, as we desire to make private APO adoption as desirable and frictionless as possible. Put together, the audit measurements provide feedback on the APO modifications, allowing us to converge with confidence on how to most effectively and simply modify APO to avoid training data leakage.

To summarize the contributions of this paper:

1. We present (in Section 3) a methodology for prompt inspection auditing (based on comparisons of n-grams in the optimized prompt and unique/exclusive n-grams in the training and held-out data examples) which provides an empirical measurement of the practical risk of private data leakage.
2. We consider (in Section 4) three leakage mitigation strategies specifically for APO, which do not involve an excessive increase in computational cost beyond 'non-private' APO. These strategies are: redacting PII, adding 'privacy-preserving' instructions to the mutator's meta-prompt, and adding 'privacy-preserving' instructions to the autorater's meta-prompt.
3. We present results of experiments (in Section 5) with two popular APO algorithms, GEPA (Agrawal et al., 2026) and ProTeGi (Pryzant et al., 2023) and two representative user-oriented datasets, Enron emails (Klimt and Yang, 2004) and LongLaMP-3 Amazon reviews (Kumar et al., 2024). With these algorithms and datasets, we apply our privacy mitigations, quantify private data leakage via our prompt inspection audit methodology, and show that simple mitigations achieve practical effective privacy protection.

2 Related Work

Automated Prompt Optimization (APO). State-of-the-art APO methods (Pryzant et al., 2023; Opsahl-Ong et al., 2024; Yang et al., 2024; Ye et al., 2024; Yuksekgonul et al., 2025; Agrawal et al., 2026) utilize an iterative refinement process consisting of two key stages:

1. *Prompt Evaluation & Selection:* Starting from the initial seed prompt, candidate prompts are executed on a batch of training queries. An *autorater* LLM scores their performance and provides textual feedback. The highest-performing candidates are selected for the next iteration.

2. *Prompt Mutation:* A *mutator* LLM takes the selected candidate and the corresponding feedback to generate a revised prompt as the new candidate. The goal is to fix errors highlighted in the feedback without degrading performance on previously successful queries.

In this work, we study two state-of-the-art APO frameworks, ProTeGi (Pryzant et al., 2023) and GEPA (Agrawal et al., 2026). Both utilize LLM-driven mutation based on natural language feedback but differ in selection: ProTeGi employs a global top-k beam search, whereas GEPA selects candidates from a per-instance Pareto front to avoid local optima.

Critically, both approaches require a training dataset to generate textual feedback. Because the mutator LLM is provided with raw training inputs, ground-truth references (if provided), and evaluation feedback in its context, it exhibits a strong tendency to overfit. Specifically, we observe that to correct errors, the mutator tends to copy problematic training examples verbatim into the optimized prompt as few-shot demonstrations (Fig. 1), or paraphrase specific training content directly into the mutated prompts. If the underlying training data contains sensitive information, this optimization process causes privacy leakage.

Empirical Privacy Auditing. Empirical privacy auditing (Ponomareva et al., 2025, Sec 7.5; Mitchell et al., 2026) quantifies the practical privacy risk that a private data-consuming algorithm may impose. Such audits take a number of different forms, including membership inference attacks (MIAs) or data reconstruction attacks (Shokri et al., 2017; Carlini et al., 2021; Jagielski et al., 2020). They utilize a held-out sample of data drawn from the same distribution as the training data. A privacy violation consists of being about to distinguish between the training and held-out data, given an attack surface, to a statistically significant degree. In MIA, samples are scored and thresholded to predict their membership in the training or held-out cohort. These predicted labels yield TP and FP rates, which can be converted into a lower bound on the empirical epsilon estimate. We leverage this methodology in the auditing approach we present. Many auditing approaches construct and inject a small set of 'canaries' (Carlini et al., 2019) into the training data for auditing, which serve as worst-case probes for measuring memorization. This is often done because the volume of data involved in e.g. gradient-based tuning makes auditing the full training dataset impractical. However, APO uses far less training data, allowing for auditing the training dataset directly.

Leakage Mitigation Techniques. Several techniques are commonly applied across systems, algorithms and use cases to minimize exposure of private information. PII can be scrubbed from data prior to downstream consumption by any process, but there is some evidence that such data sanitization may effect language model performance (Pal et al., 2024). Algorithmic changes can be made to the process itself, e.g. in learning and analytics settings, differential privacy (DP) can be applied (see Ponomareva et al. (2023) for a practical guide).

Private APO. Private APO is still a relatively new research space. DP-OPT (Hong et al., 2024) is the most notable previous work. It is motivated by the objective of performing APO on private examples and considering how privacy leakage can be mitigated. Standard APO involves a single call (per round) to the mutator; DP-OPT instead makes *multiple* mutator calls per round, each operating on disjoint subsets of the private examples. The output of these calls is ensembled in a way bounding the contribution of each subset, allowing for DP guarantees against leakage. While the privacy guarantees of this work are strong, it involves a significant amount of computation. The mutator is generally the most computationally expensive LLM in APO, and increasing its usage k-fold is undesirable.

Private ICL. There has been more prior research on the different but related question of how to go about ICL fewshot construction based on private data examples. The private ICL works fall into two categories: privacy via ensembling (Duan et al., 2023; 2024; Wu et al., 2024) (in the same vein as DP-OPT, above) and privacy via synthetic data intermediaries (Tang et al., 2024; Gao et al., 2024). As described above, ensembling adds considerable computational cost, and going through the process of first generating synthetic data before optimizing a prompt is also computationally expensive.

3 Measurement Methodology: Prompt Inspection Auditing

We posit that the most salient risk that APO poses is via explicitly memorizing private information into the optimized prompt itself. Training data is likely to appear in ICL demonstration examples that

are included in the prompt. Although production systems typically restrict users from viewing the system prompts, adversaries can extract them via tailored prompt injection attacks. Moreover, even during normal usage, embedding private data in the system prompt increases the risk of inadvertent leakage into the model's output. Thus, auditing the prompt directly effectively models a powerful adversary with full white-box access to the system prompt.

Our audit strategy consists of inspecting the optimized prompt for (1) verbatim memorization or (2) approximate memorization of any training example. We define verbatim memorization as an exact match of an n-token sequence in the prompt and training data. Approximate memorization is defined as near-exact match using the Levenshtein minimum edit distance between any n-token sequence in the prompt and training data.

In order to calibrate the observed leakage of training data in the optimized prompt, we need to consider what "memorization" (according to our distance measure) is expected as a function simply of the typicality of natural text data or generalization across the target domain. We do this by taking a held-out set of the same size as the training set and considering only prompt windows that are unambiguous in that they appear at most once across both cohorts. This exclusivity filter is critical: without it, the counts are dominated by corpus boilerplate (e.g. email headers, review scaffolding) that held-out examples reproduce as readily as training examples, causing their match rates to move in lockstep. Our audit is described in full below. Figure 2 provides an overview.

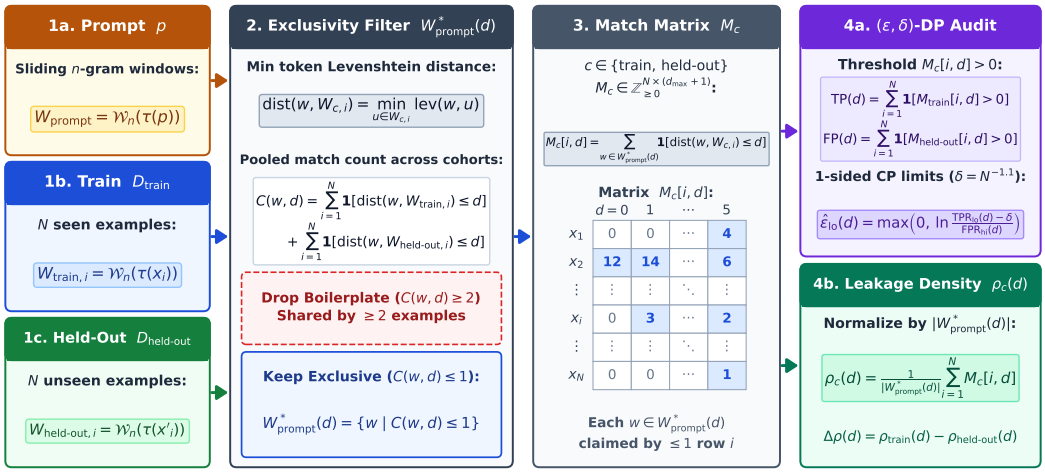


Figure 2: **Overview of the Exclusive Prompt-Inspection Audit pipeline. (1)** The optimized prompt, training examples, and held-out examples are tokenized and segmented into sliding n-gram windows. **(2)** For each prompt window $w \in W_{\text{prompt}}$ and tolerance d, we count the total number of matching examples pooled across $D_{\text{train}} \cup D_{\text{held-out}}$ and discard ambiguous boilerplate windows. **(3)** Surviving exclusive matches are tabulated into the cohort match count matrix $M_c[i,d]$. **(4)** From M_c, we compute **(4a)** the binary hit counts $\text{TP}(d), \text{FP}(d)$ under $M_c[i,d] > 0$ and the resulting one-sided $(1-\alpha)$ Clopper–Pearson lower bound $\hat{\varepsilon}_{\text{lo}}(d)$, and **(4b)** the prompt-length-normalized leakage density.

Exclusive Prompt-Inspection Audit. Let $D_{\text{train}} = \{x_1, \ldots, x_N\}$ and $D_{\text{held-out}} = \{x'_1, \ldots, x'_N\}$ denote the training and held-out datasets, each of size N. Each example and the optimized prompt p are tokenized and segmented into sliding n-gram windows, yielding the per-example window sets $W_{\text{train},i}$ and $W_{\text{held-out},i}$ ($i \in \{1, \ldots, N\}$) and the prompt window set W_{prompt}. For a prompt window $w \in W_{\text{prompt}}$ and cohort $c \in \{\text{train}, \text{held-out}\}$, let

$$\text{dist}(w, W_{c,i}) \;=\; \min_{u \,\in\, W_{c,i}} \text{lev}(w, u) \tag{1}$$

denote minimum token-level Levenshtein distance from w to any n-gram of the i-th example in D_c.

To eliminate corpus boilerplate shared across examples, we count how many examples pooled across *both* cohorts match window w within edit distance $d \in \{0, \ldots, d_{\max}\}$:

$$C(w, d) \;=\; \sum_{i=1}^{N} \mathbb{I}\big[\mathrm{dist}(w, W_{\text{train},i}) \le d\big] \;+\; \sum_{i=1}^{N} \mathbb{I}\big[\mathrm{dist}(w, W_{\text{held-out},i}) \le d\big]. \tag{2}$$

The *exclusive prompt window set* $W^*_{\text{prompt}}(d) \subseteq W_{\text{prompt}}$ retains only those windows attributable to at most a single example at tolerance d:

$$W^*_{\text{prompt}}(d) \;=\; \big\{\, w \in W_{\text{prompt}} \;\big|\; C(w, d) \le 1 \,\big\}. \tag{3}$$

The resulting *match count matrix* $M_c \in \mathbb{Z}_{\ge 0}^{N \times (d_{\max}+1)}$ for cohort $c \in \{\text{train}, \text{held-out}\}$ counts the exclusive prompt windows credited to example i at edit distance d:

$$M_c[i, d] \;=\; \sum_{w \,\in\, W^*_{\text{prompt}}(d)} \mathbb{I}\big[\mathrm{dist}(w, W_{c,i}) \le d\big]. \tag{4}$$

Example-Level Audit and Empirical (ε, δ)-DP Bound. Under the binary decision rule $M_c[i, d] > 0$, the numbers of memorized training and held-out examples at edit distance d are

$$\mathrm{TP}(d) = \sum_{i=1}^{N} \mathbb{I}\big[M_{\text{train}}[i, d] > 0\big], \qquad \mathrm{FP}(d) = \sum_{i=1}^{N} \mathbb{I}\big[M_{\text{held-out}}[i, d] > 0\big]. \tag{5}$$

We evaluate statistical significance via a one-sided $(1 - \alpha)$ Clopper–Pearson lower bound on the empirical (ε, δ)-DP privacy loss at $\delta = N^{-1.1}$ from the one-sided $(1 - \alpha/2)$ Clopper–Pearson confidence limits $\big(\mathrm{TPR}_{\text{lo}}(d), \mathrm{FPR}_{\text{hi}}(d)\big)$:

$$\hat{\varepsilon}_{\text{lo}}(d) \;=\; \max\!\left(0,\; \ln \frac{\mathrm{TPR}_{\text{lo}}(d) - \delta}{\mathrm{FPR}_{\text{hi}}(d)}\right). \tag{6}$$

Note that this metric is an example-level measure of privacy leakage. Alternatively, one could consider leakage at the n-gram level using the same approach but without the aggregation across W_c.

Prompt-Length Normalization (Leakage Density). Because a longer prompt with more windows inherently has a higher likelihood of more matches across $D_{\text{train}} \cup D_{\text{held-out}}$, to compare the leakage of different APO runs that produce prompts of varying length, we consider a notion of density that normalizes the total credited matches by the exclusive prompt window budget $\big|W^*_{\text{prompt}}(d)\big|$. Since $C(w, d) \le 1$ ensures that each window $w \in W^*_{\text{prompt}}(d)$ is matched by at most one example across $D_{\text{train}} \cup D_{\text{held-out}}$, the column sum $\sum_{i=1}^{N} M_c[i, d]$ counts the exact number of distinct windows in $W^*_{\text{prompt}}(d)$ attributable to cohort c. We therefore define the training and held-out match densities and the net leakage density $\Delta\rho(d)$ at edit distance d as

$$\rho_c(d) = \frac{1}{\big|W^*_{\text{prompt}}(d)\big|} \sum_{i=1}^{N} M_c[i, d], \qquad \Delta\rho(d) = \rho_{\text{train}}(d) - \rho_{\text{held-out}}(d), \tag{7}$$

where $\rho_{\text{train}}(d) + \rho_{\text{held-out}}(d) \le 1$ represents the total fraction of non-boilerplate prompt windows attributable to specific examples.

4 Methods to Mitigate Private Data Leakage

As described in Section 2, a typical APO algorithm consists of two stages: 1) candidate prompt evaluation & selection, and 2) prompt mutation, as illustrated in Figure 3. To mitigate the risk of private data leakage, we propose the following techniques to modify the APO pipeline and the input training data. In Section 5 we will show detailed comparisons of these methods in terms of their privacy-utility trade-offs.

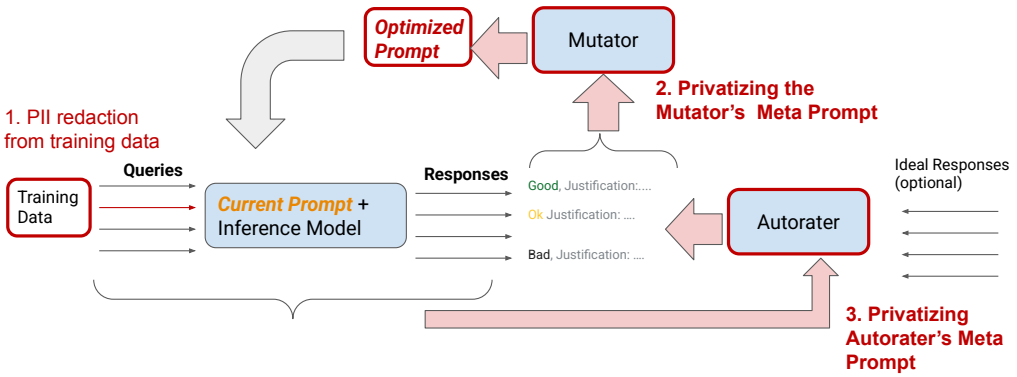


Figure 3: Proposed methods to mitigate data leakage via APO: (1) PII redaction from training data, (2) Privacy-preserving mutator meta-prompt, and (3) Privacy-preserving autorater text feedback.

1. Redaction of Private Information from Training Data. A first line-of-defense strategy to prevent data leakage into the optimized prompt is to remove it from the training data. We achieve this using a validated internal detection engine that combines regex-based rules and NLP models to detect and replace instances of PII such as names, email addresses, phone numbers and locations with abstract placeholders such as [NAME], [EMAIL ID], etc. While PII redaction can be effective in preventing private data from entering the APO pipeline, a caveat is that it may not filter out information that is not classified as PII but can still be used to identify a data sample. Moreover, redacting PII can erase subtle insights that can be drawn from the data to optimize the prompt.

2. Leakage Mitigation Instructions in the Mutator Model's Meta-Prompt. The meta-prompt used for the prompt optimizer plays a critical role in how much private data can get leaked into the optimized prompt. We propose to include privacy-preservation instructions in this meta-prompt to explicitly tell the prompt optimizer to not include private information in the optimized prompt, either in the form of in-context learning examples or otherwise. We compare the privacy leakage of this meta-prompt with that of the baseline APO meta-prompt and study the privacy-utility trade-offs. As a negative control, we also evaluate a malicious variant of the mutator meta-prompt, which is designed to deliberately force or exfiltrate private training data into the optimized prompt. The three variants of the meta-prompts are given in Appendix A.4.

3. Leakage Mitigation Instructions in the Autorater Model's Meta-prompt. The input to the mutator model includes a set of feedback examples that were rated by the autorater, along with a text justification for its ratings. Since the text justification on the feedback examples can include verbatim copies or approximate versions of training data examples, we propose including a privatization instruction in the autorater's meta-prompt. This instructs the autorater to avoid including private information from training data in the text justification that it generates to accompany the input, output, rating received by each example. As a result, private information is prevented from leaking into the mutator's meta-prompt and subsequently into the improved prompt created by the mutator. The standard autorater meta-prompt and its privacy-preserving variant are given in Appendix A.5.

5 Experimental Results

5.1 Experimental Setup

APO Algorithms. In our experiments, we consider two APO algorithms: 1) Prompt Optimization with Textual Gradients (ProTeGi) (Pryzant et al., 2023) and 2) Genetic Pareto Prompt Evolution (GEPA) (Agrawal et al., 2026). ProTeGi optimizes a prompt by generating natural-language "textual gradients" from minibatch errors and editing the prompt in the opposite direction, using beam search and bandits to converge on a single best-performing prompt. GEPA instead evolves a population of prompts through reflection-driven mutation and crossover, using Pareto-based selection to keep a

diverse set of top performers rather than converging to one optimum. ProTeGi behaves like "gradient descent for text" honing in on one solution, while GEPA behaves like natural selection, exploring multiple diverse prompt variants simultaneously and typically requiring far fewer rollouts to reach strong performance. The models used for inference, autorating and mutation are Gemini Flash-lite, Gemini 3.7 Flash and Gemini 3.1 Pro respectively for all our experiments. The technical report (Gemini Team Google, 2023) describes the Gemini family of models.

Datasets. We apply our leakage mitigation strategies to both the GEPA and ProTeGi algorithms and evaluate them using two open-source datasets: the LongLaMP-3 product reviews datasets and the Enron corporate email dataset. The LongLaMP-3 dataset (Kumar et al., 2024; Ni et al., 2019) contains Amazon product reviews, where the task is to generate a realistic and personalized review given a summary and rating. During the APO process, candidate prompts are evaluated by an autorater model on the rubrics of Persona Adherence, Sentiment Consistency and Content Relevance and given one of three ratings: 0 (bad), 0.5 (ok), and 1 (good) for each rubric. For the Enron corporate email dataset (Klimt and Yang, 2004), the task is to generate a realistic email between two Enron executives to mimic the emails that are present in the Enron dataset. During the APO process, candidate prompts are evaluated by an autorater model on the rubrics of Corporate Tone and Subject Relevance and are given one of three ratings: 0 (bad), 0.5 (ok), and 1 (good) for each rubric.

Experiment Parameters and Metrics. For each algorithm, dataset and leakage mitigation strategy we sample datasets of 350 examples (250 training and 100 evaluation examples, both used in different stages of APO). We evaluate the test score using a separate set of 100 test examples. Another set of 350 examples is sampled and used as the held-out or unseen data for the privacy audit described in Section 3. All experiments are run for 6 seeds, where the training, evaluation, held-out, and test datasets are sampled independently for each seed. The ProTeGi algorithm is run for 3 iterations with 2 mutated prompts generated for every prompt in the candidate pool, and 2 top prompts retained at the end of each iteration. The mutator model receives 10 negative feedback examples that were rated 0 (bad) or 0.5 (ok) by the autorater model along with a text justification for each rating. The mutator is instructed to include up to 5 in-context learning examples in the improved prompt that it produces. For the GEPA algorithm the number of maximum metric calls is set to 2100, which is chosen to approximately match the computing budget used by ProTeGi: 2100 GEPA metric calls $= (250$ training examples $+ 100$ eval examples$) \times 3$ ProTeGi iterations $\times$ 2 candidates per prompt. GEPA's baseline mutator meta-prompt also allows it to include up to 5 in-context learning examples in the improved prompt.

Our exclusive prompt inspection audit is run with $n = 10$ token sliding windows over the optimized prompt and examples in the train and held-out cohorts. We consider matches at edit distances $d = 0..5$. Each seed is run with $N = 350$ examples in both the training and held-out datasets. When conducting the example-level MIA to yield an empirical epsilon lower bound, we compare the optimized prompt from each seed to the sets of train and held-out examples drawn for each seed, and pool these match counts across all seeds such that we have 2100 positive and 2100 negative samples. $\delta = N^{-1.1}$ using $N = 350$ because that is the number of examples that impact each of the prompts we audit. The leakage density results are reported as the mean and standard error of the mean over 6 seeds.

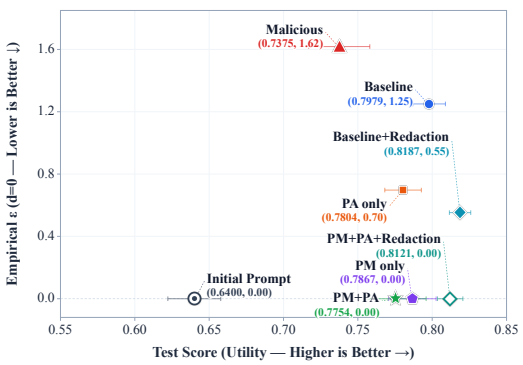
APO Variants. We consider each of the following variants of the APO algorithm for our analysis:

1. *Initial* prompt that is given as input to the APO algorithm.
2. *Baseline* where APO is run with the standard mutator and autorater meta-prompts.
3. *Baseline+Redaction*, corresponding to standard APO run using the PII redacted training data
4. *Malicious* where the mutator's meta-prompt instructs it to extract information from training examples and include it verbatim in the mutated prompt.
5. *PM only*, where APO uses a privacy-preserving mutator meta-prompt that instructs the mutator to not include excerpts from training data verbatim into the improved prompt. The autorater uses the standard baseline meta-prompt.
6. *PA only*, where APO uses a privacy-preserving autorater feedback meta-prompt that instructs the autorater to avoid including private information in the text justification for its scores, along with the baseline mutator meta-prompt.

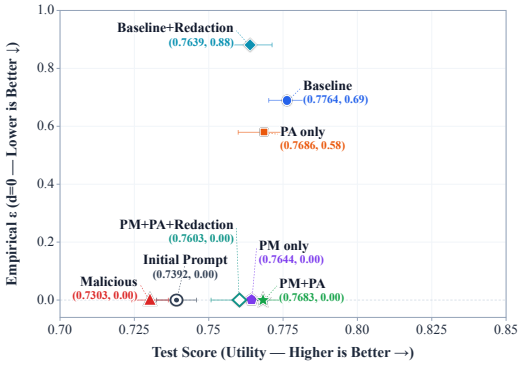
7. *PM+PA*, where APO uses a privacy-preserving mutator meta-prompt as well as a privacy-preserving autorater feedback meta-prompt.
8. *PM+PA+Redaction*, where APO is run on the PII redacted dataset, using a privacy-preserving mutator meta-prompt as well as a privacy-preserving autorater feedback meta-prompt.

5.2 Key Results and Observations

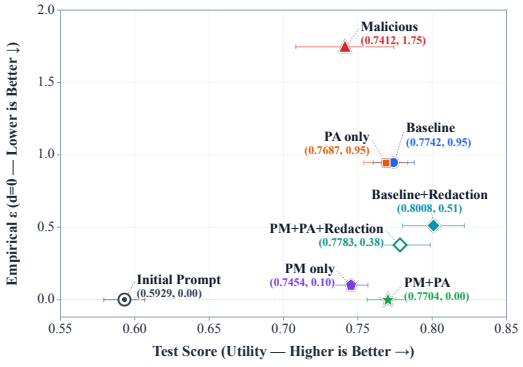
In Figure 4 we show the privacy versus utility trade-offs comparing the different variants of APO algorithms. Appendices A.1 and A.2 provide more in depth evaluations of our privacy audit metrics themselves. The 2x2 grid of plots shows the trade-offs for the 2 datasets (Enron and LongLaMP-3) and 2 APO algorithms (GEPA and ProTeGi). Private data leakage (shown on the y-axis of the plots) is measured using the empirical 95% Clopper-Pearson lower bound estimate of ε at edit distance $d = 0$ (plots for $d \leq 2$ are given in Appendix A.3), defined in Section 3, where a smaller value means better privacy. Utility (showed on the x-axis of the plots) is measured in terms of the score (range 0 to 1) assigned by the standard (not privacy-preserving) autorater, where the score is averaged across the autorater rubrics, the examples in the test dataset, and across 6 seeds corresponding to 6 APO runs with different sampled training, evaluation, test and held-out datasets. Since higher utility is better, the optimal privacy-utility trade-off is at the bottom right corner of each plot.



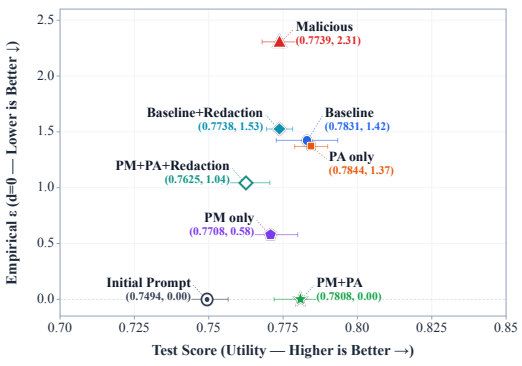
(a) GEPA APO variants run on Enron dataset



(b) GEPA APO variants run on LongLaMP-3 dataset



(c) ProTeGi APO variants run on Enron dataset



(d) ProTeGi APO variants run on LongLaMP-3 dataset

Figure 4: Privacy-Utility trade-off plots comparing variants of the GEPA and ProTeGi APO algorithms. Privacy is reported as the 95% Clopper-Pearson lower bound estimate of epsilon. Utility is reported as the mean test performance and SEM over 6 random seeds. The PM+PA strategy, which uses the privacy-preserving mutator meta-prompt plus the privacy-preserving autorater feedback yields the best empirical epsilon of 0 with minimal impact on the test score.

Baseline APO improves utility but loses privacy. In each plot, observe that since the initial prompt has never seen the training data, its empirical epsilon $\varepsilon = 0$, corresponding to perfect privacy. Baseline APO uses the training data to optimize the prompt and the resulting prompt has significantly

higher utility (the test score) than the initial prompt. However, positive values of ε indicate that portions of training data are memorized by optimized prompt.

Malicious mutator meta-prompt can induce leakage at the expense of utility. When the mutator model is given a malicious meta-prompt, it aggressively includes training examples verbatim in the optimized prompt. While it is apparent from Figure 4 that the resulting prompt often has high private data leakage, we observe that its test score is also worse than the baseline case due to the overfitting of the meta-prompt to the training examples. In fact, APO algorithms often naturally penalize overfitting because they use training data to improve the prompt and use a separate evaluation dataset to select the top mutated candidate prompts. This selection step can reject overfitted prompts such that the optimized prompt ends up being almost identical to the initial prompt, as we observe in Figure 4b.

Privacy-preserving mutator meta-prompt (PM) universally reduces leakage. In all the subplots of Figure 4, we observe that using a privacy-preserving mutator meta-prompt reduces the empirical ε value as compared to the baseline, demonstrating that it reduces the risk of data leakage into the optimized prompt. Privacy comes at the cost of utility, as test score drops slightly below baseline.

Privacy-preserving autorater meta-prompt (PA) boosts PM performance. Using the privacy-preserving autorater (PA) is not effective in improving privacy by itself, but when paired with privacy-preserving mutator (PM) it helps boost the utility achieved by the PM only strategy. Across all plots in Figure 4, PM+PA consistently gives a favorable privacy-utility trade-off, and it is the leakage mitigation strategy that the authors recommend.

Redacting PII from training data (Baseline+Redaction and PM+PA+Redaction) is not foolproof. The simple method of redacting PII from training data has varying degrees of effectiveness. For GEPA run on the Enron dataset, PII redaction improves privacy as well as utility as compared to the baseline, and PM+PA+Redaction strategy similarly gives a better privacy-utility trade-off than PM only or PM+PA. However, for the LongLamP-3 dataset, it has the opposite effect. Redacting PII might be useful depending on the application and privacy requirements of the setting, but it should not be treated as a foolproof way of guaranteeing privacy protection. Our audit indicates there might be greater risk in some cases from including redacted data, although this needs further investigation.

6 Conclusions and Outlook

In this paper we study private data leakage during automated prompt optimization. When APO is run on private examples without safeguards, substantial verbatim spans of training data can be absorbed into the optimized prompt, most often as in-context learning examples (see Section A.1.3 for an ablation). To measure this risk, we introduced an empirical privacy auditing methodology based on calibrated matching of token windows between the optimized prompt and training versus held-out data. By restricting credit to exclusive windows — those attributable to at most one example pooled across training and held-out data — our audit filters out corpus boilerplate that would otherwise dominate naive match-counting, yielding an example-level empirical ε measure. Using this audit, we then show that simple modifications to the meta-prompts of the mutator and autorater substantially reduce leakage with minimal impact on the performance of the optimized prompt. The magnitude of this effect varies across APO algorithms and datasets.

Our proposed APO meta-prompt interventions are complementary to, rather than a replacement for, formal differential privacy. Methods such as DP-OPT provide rigorous worst-case guarantees at significant computational cost, whereas our approach offers a near-zero-overhead reduction in empirical leakage together with a lightweight audit to measure it, but no formal guarantee. We view the two as naturally composable, potentially allowing DP mechanisms to operate at a more favorable privacy-utility point once combined with our audit and mitigations. Promising directions for future work include 1) extending our audit to consider semantic similarity between user data and prompts, 2) designing an attack for an adversary with only query access to the prompted model, 3) testing further privacy-preserving modifications to the APO pipeline such as new autorater rubrics or post-facto PII redaction from the optimized prompt, 4) comparing our mitigations to formal DP methods using our empirical privacy auditing techniques.

AI USE STATEMENT

Our research investigates prompt optimization pipelines which natively incorporate LLMs. In our experimental setup, we utilized Gemini Flash-lite as the inference model, Gemini 3.7 Flash as the autorater model, and Gemini 3.1 Pro as the mutator model. In this work, we used generative AI tools to implement methods (writing code for experiments), to design and provide feedback on research methodology and automate experiments, and to assist in the writing of mathematical formulations in one section of the paper. We have not used generative AI tools to generate synthetic datasets, to interpret results, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, assist with translation, clean and reformat datasets, or support qualitative and thematic data analysis; the rest of the required disclosure tasks are not applicable to this work. Additionally, we used AI to edit parts of the paper to improve clarity and grammatical correctness. We have reviewed all AI-assisted work: LLM-generated code was verified and tested for correctness by the authors, AI-assisted mathematical formulations were manually checked and validated by the authors, and AI-suggested methodology and experimental design choices were critically reviewed by the authors before adoption. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work investigates private data leakage in Automated Prompt Optimization (APO) and introduces empirical privacy auditing and mitigation methods. Our empirical evaluation utilizes the Enron corporate email dataset and the LongLaMP-3 product reviews dataset. While both are publicly available, widely used benchmarks in machine learning research, the Enron corpus contains real-world, historically leaked emails that include sensitive personal information (PII). To respect privacy principles, we do not expose or highlight new sensitive information. Instead, we use these datasets to demonstrate how existing risks can be systematically audited and mitigated. Highlighting that private data can leak into optimized prompts could theoretically assist malicious actors in designing more effective extraction attacks (e.g., our "malicious" mutator meta-prompt used as a negative control). However, we believe that exposing these vulnerabilities is a necessary precursor to securing APO systems. By presenting concrete, easily implementable mitigation strategies—specifically privacy-preserving mutator and autorater meta-prompts—we provide actionable solutions that significantly reduce leakage risks with minimal utility trade-offs, enabling safer deployment of APO in production. This research did not involve any human subject experiments, and no new private user data was collected or crawled. All experiments were conducted in simulated, controlled settings using established open-source benchmarks.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS16, page 308–318. ACM, October 2016. doi: 10.1145/2976749.2978318. URL http://dx.doi.org/10.1145/2976749.2978318.

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Anthropic. Improve your prompts in the developer console. Claude Blog, October 2024. URL https://claude.com/blog/prompt-improver. Accessed: 2026-08-24.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pages 267–284, 2019.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650, 2021.

Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models. *Advances in Neural Information Processing Systems*, 36:76852–76871, 2023.

Haonan Duan, Adam Dziedzic, Mohammad Yaghini, Nicolas Papernot, and Franziska Boenisch. On the privacy risk of in-context learning. *arXiv preprint arXiv:2411.10512*, 2024.

Fengyu Gao, Ruida Zhou, Tianhao Wang, Cong Shen, and Jing Yang. Data-adaptive differentially private prompt synthesis for in-context learning. *arXiv preprint arXiv:2410.12085*, 2024.

Gemini Team Google. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. URL https://arxiv.org/abs/2312.11805.

Junyuan Hong, Jiachen Tianhao Wang, Chenhui Zhang, Zhangheng Li, Bo Li, and Zhangyang Wang. Dp-opt: Make large language model your privacy-preserving prompt engineer. In *International Conference on Learning Representations*, volume 2024, pages 58003–58026, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems*, 33: 22205–22216, 2020.

Bryan Klimt and Yiming Yang. The Enron Corpus: A New Dataset for Email Classification Research. In *European Conference on Machine Learning*, pages 217–226. Springer, Berlin, Heidelberg, 2004. doi: 10.1007/978-3-540-30115-8_22.

Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A. Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, Nedim Lipka, Chien Van Nguyen, Thien Huu Nguyen, and Hamed Zamani. Longlamp: A benchmark for personalized long-form text generation, 2024. URL https://arxiv.org/abs/2407.11016.

George Lee and Ivan Nardini. Announcing public preview of vertex ai prompt optimizer. Google Cloud Blog, September 2024. URL http://cloud/blog/products/ai-machine-learning/announcing-vertex-ai-prompt-optimizer. Accessed: 2026-08-24.

H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models, 2018. URL https://arxiv.org/abs/1710.06963.

Nicole Mitchell, Galen Andrew, Arun Ganesh, Brendan McMahan, and Peter Kairouz. Advancing the state-of-the-art in empirical privacy auditing, 2026. URL https://arxiv.org/abs/2606.10481.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, 2019.

OpenAI. Prompt optimizer. OpenAI Developers. URL https://developers.openai.com/api/docs/guides/prompt-optimizer. Accessed: 2026-08-24.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525/.

Anwesan Pal, Radhika Bhargava, Kyle Hinsz, Jacques Esterhuizen, and Sudipta Bhattacharya. The empirical impact of data sanitization on language models, 2024. URL https://arxiv.org/abs/2411.05978.

Natalia Ponomareva, Sergei Vassilvitskii, Zheng Xu, Brendan McMahan, Alexey Kurakin, and Chiyaun Zhang. How to dp-fy ml: A practical tutorial to machine learning with differential privacy. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 5823–5824, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599561. URL https://doi.org/10.1145/3580305.3599561.

Natalia Ponomareva, Zheng Xu, H. Brendan McMahan, Peter Kairouz, Lucas Rosenblatt, Vincent Cohen-Addad, Cristóbal Guzmán, Ryan McKenna, Galen Andrew, Alex Bie, Da Yu, Alex Kurakin, Morteza Zadimoghaddam, Sergei Vassilvitskii, and Andreas Terzis. How to dp-fy your data: A practical guide to generating synthetic data with differential privacy, 2025. URL https://arxiv.org/abs/2512.03238.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494/.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2017. doi: 10.1109/SP.2017.41.

Shreyas Subramanian, Chris Pecora, Zhengyuan Shen, and Shipra Kanoria. Improve the performance of your generative ai applications with prompt optimization on amazon bedrock. AWS Blogs, November 2024. URL https://aws.amazon.com/blogs/machine-learning/improve-the-performance-of-your-generative-ai-applications-with-prompt-optimization-on-amazon-bedrock/. Accessed: 2026-08-24.

Mikhail Sveshnikov and Elena Samuylova. How we built open-source automated prompt optimization. Evidently AI Blog: LLM Evals, January 2026. URL https://www.evidentlyai.com/blog/automated-prompt-optimization. Accessed: 2026-08-24.

Xinyu Tang, Richard Shin, Huseyin Inan, Andre Manoel, Niloofar Mireshghallah, Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, and Robert Sim. Privacy-preserving in-context learning with differentially private few-shot generation. In *International conference on learning representations*, volume 2024, pages 33058–33077, 2024.

The Databricks AI Research Team. Building state-of-the-art enterprise agents 90x cheaper with automated prompt optimization. Databricks AI Research Blog, September 2025. URL https://www.databricks.com/blog/building-state-art-enterprise-agents-90x-cheaper-automated-prompt-optimization. Accessed: 2026-08-24.

Chetan Tekur, Chao Zhao, Paul Zhou, and Caren Chang. How automated prompt optimization unlocks quality gains for ml kit's genai prompt api. Android Developers Blog, January 2026. URL https://android-developers.googleblog.com/2026/01/how-automated-prompt-optimization.html. Accessed: 2026-08-24.

Tong Wu, Ashwinee Panda, Jiachen Tianhao Wang, and Prateek Mittal. Privacy-preserving in-context learning for large language models. In *International Conference on Learning Representations*, volume 2024, pages 20005–20040, 2024.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pages 12028–12068, 2024.

Qinyuan Ye, Mohamed Ahmed, Reid Pryzant, and Fereshte Khani. Prompt engineering a prompt engineer. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 355–385, 2024.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative ai by backpropagating language model feedback. *Nature*, 639(8055):609–616, 2025.

A Appendix

A.1 Additional Privacy Audit Ablations

A.1.1 Empirical ε Estimate

As described in Section 3, we derive an empirical estimate of epsilon, $\hat{\varepsilon}_{\text{lo}}$, which is the 95% Clopper-Pearson lower bound, for every edit distance d considered. This edit distance describes the tolerance at which we consider an example to "match" a sliding window of the prompt. Figure **??** showcases leakage as a function of the tolerance chosen. Leakage is not monotnic with d since the number of eligible "exclusive" prompt windows depends upon the number of matches each prompt window has with the examples across the training and held-out datasets at that tolerance. Here, we see that $d = 0$ and $d = 2$ provide fairly high leakage signal across the board, and so we select these two tolerances for our exact and fuzzy match checks. For reference, any epsilon estimate above 1 is deemed indicative of a privacy violation. Figure 6 showcases how each APO variant compares in terms of leakage according to our audit at the exact ($d = 0$) and approximate ($d = 2$) thresholds. We see that our audit is particularly strong, detecting high amounts of leakage in the malicious and baseline prompt settings.

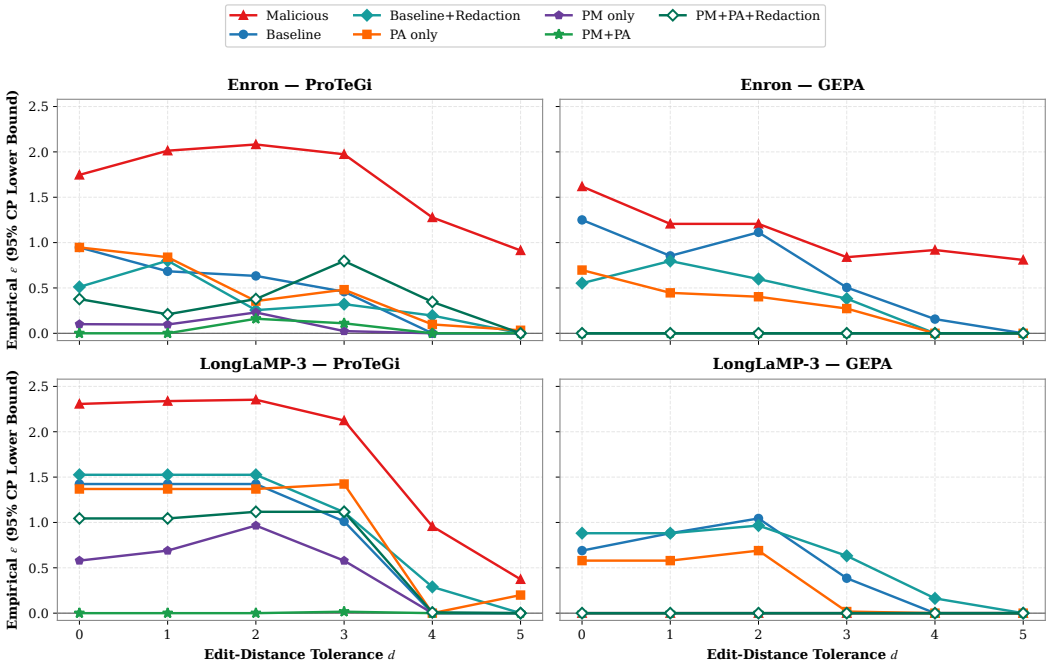


Figure 5: Empirical estimate $\hat{\varepsilon}_{\text{lo}}$ across edit distance tolerances d.

A.1.2 Leakage Density

Because the likelihood of a prompt having a sliding window which matches any of the training data increases with prompt length, we additionally audit the density of training example leakage within the prompt. This is reported as the fraction of eligible prompt windows that can be attributed to an example, training or held-out. Figure 7 plots the leakage density of training versus held-out data at edit distance $d = 2$. Note that across all interventions held-out examples hardly make up any fraction of the exclusive prompt windows. Figure 8 depicts the leakage density for training examples only, using both exact matches ($d = 0$) and fuzzy matches ($d = 2$). We see that despite the increased length of malicious prompts (as seen in the reported window length at the right side of each row in the figures), even when normalizing by this length, the density of training data leakage in the prompt surpasses that of the prompts produced from other interventions.

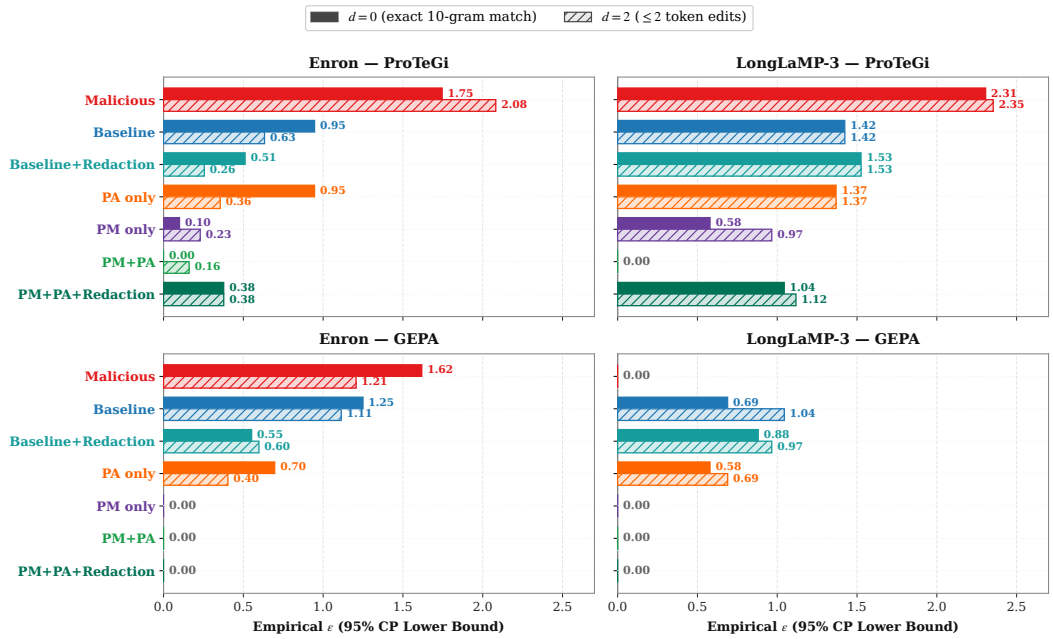


Figure 6: Comparison of the 95% Clopper–Pearson empirical privacy lower bound ($\hat{\varepsilon}_{\text{lo}}$) at exact match ($d = 0$) and fuzzy match ($d \leq 2$) across all APO variants, datasets, and algorithms.

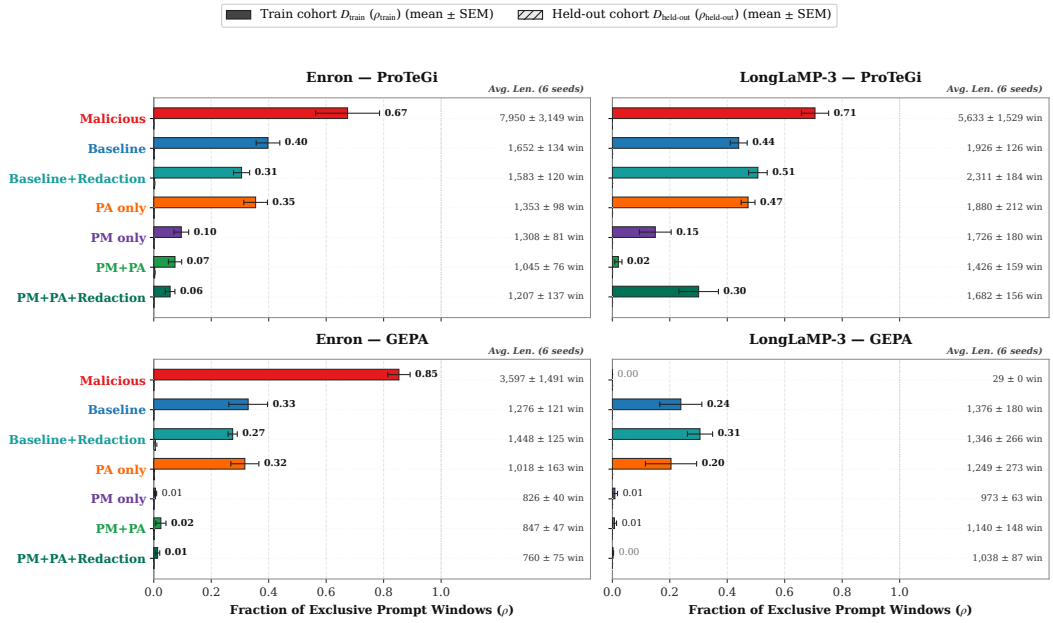


Figure 7: Fraction of exclusive prompt windows (ρ) matched to the training cohort (ρ_{train}) versus the held-out cohort ($\rho_{\text{held-out}}$) at edit distance $d \leq 2$ (mean $\pm$ SEM across 6 seeds), alongside average prompt length in windows.

A.1.3 Ablation: Effect of ICL

One open question is where these privacy leakages occur in the optimized prompt. To investigate this, we conduct an ablation on GEPA where we compare the leakage observed when using the standard mutator and autorater meta-prompts (Baseline), to the leakage observed when removing the clause from the mutator meta-prompt that specifies the allowed number of ICL examples that should be

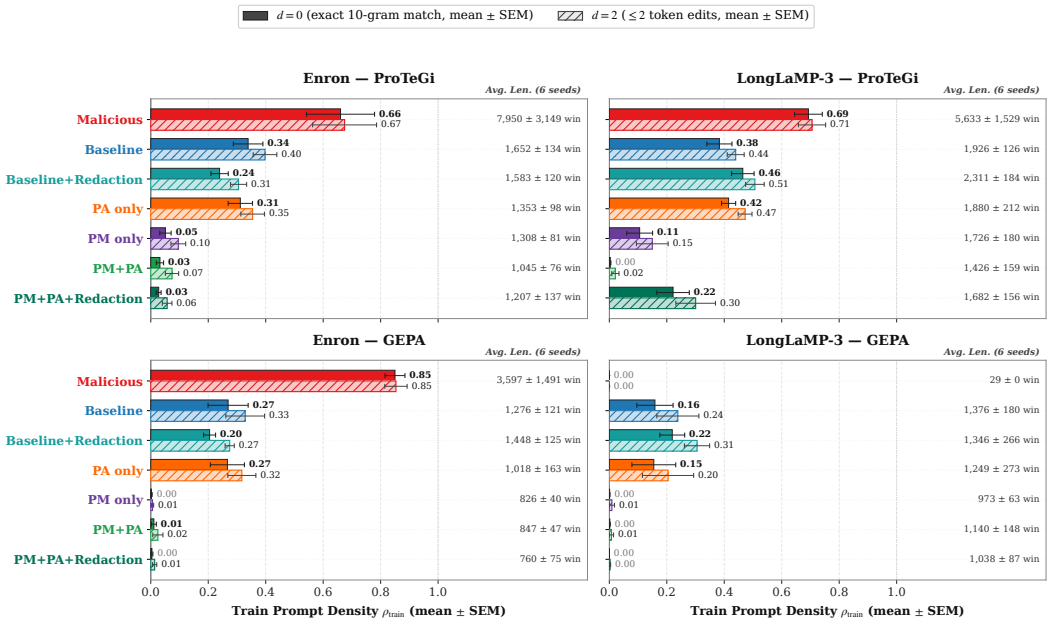


Figure 8: Training prompt memorization density ρ_{train} comparing exact 10-gram matches ($d = 0$) and fuzzy matches ($d \leq 2$) across 6 seeds (mean $\pm$ SEM).

added to the optimized prompt (Baseline – ICL). The expected effect is that GEPA should not tend to include ICL examples in the optimized prompt. Figure 9 showcases how across datasets the leakage observed in the Baseline case when using GEPA is entirely attributed to the ICL examples. We observe no leakage according to our estimate of ε or leakage density measurements when GEPA is constructed with a meta-prompt that does not specify using ICL. Note however, that the choice of using ICL presesnts a tradeoff in privacy and utility, as the inclusion of ICL can improve performance but introduces a greater privacy risk.

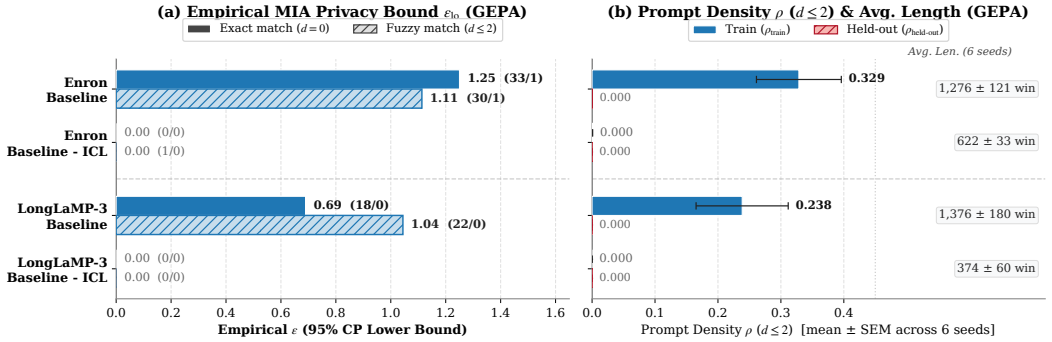


Figure 9: Ablation removing in-context learning demonstrations (Baseline - ICL) from GEPA on Enron and LongLaMP-3: (a) empirical privacy lower bound $\hat{\varepsilon}$lo and canary hit counts ($k$in/k_{out}), and (b) prompt leakage density $\rho(d \leq 2)$ and prompt length.

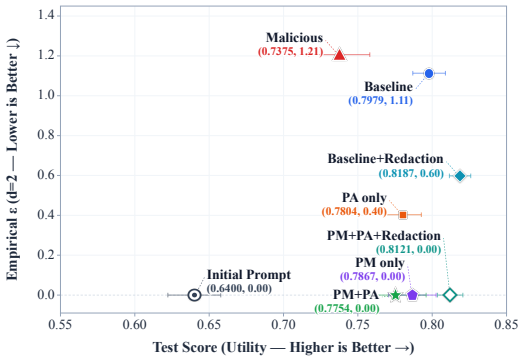
A.2 Complete Privacy Audit Results

Table 1: **Comprehensive Exclusive Privacy Audit Results Across Datasets, Algorithms, and Defense Variants.** Results are pooled across $S = 6$ independent seeds ($n_{\text{in}} = n_{\text{out}} = 2{,}100$ examples; $\delta = 350^{-1.1} \approx 0.00159$). We report the 95% Clopper–Pearson empirical privacy lower bound ε_{lo} at token edit distances $d = 0$ (exact match), $d = 2$ (near-verbatim match), and $\max_{d \in [0,5]} \varepsilon_{\text{lo}}(d)$; member vs. non-member example hit counts ($k_{\text{in}}/k_{\text{out}}$) and empirical rates ($\widehat{\text{TPR}}/\widehat{\text{FPR}}$ in %) at $d = 0$ and $d = 2$; seed-level prompt memorization density ρ (mean ± SEM fraction of unambiguous 10-token prompt windows credited to examples); and optimized prompt length in sliding windows (N_{win}) and tokens (N_{tok}).

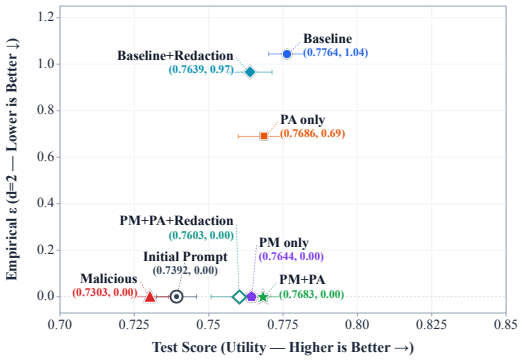
			Empirical ε_{lo} (95% CP)			**Exact Match ($d = 0$)**		**Fuzzy Match ($d = 2$)**		**Prompt Density ρ (Mean ± SEM)**			**Prompt Length**	
Dataset	**Algo.**	**Variant**	$d = 0$	$d = 2$	$\max_d (d^*)$	$k_{\text{in}}/k_{\text{out}}$	$\widehat{\text{TPR}}/\widehat{\text{FPR}}$ (%)	$k_{\text{in}}/k_{\text{out}}$	$\widehat{\text{TPR}}/\widehat{\text{FPR}}$ (%)	$\rho_{\text{train}}(d = 0)$	$\rho_{\text{train}}(d = 2)$	$\rho_{\text{out}}(d = 2)$	N_{win}	N_{tok}
Enron	ProTeGi	Malicious	1.75	2.08	2.08 (2)	96/6	4.57/0.29	104/4	4.95/0.19	0.660 ± 0.118	0.675 ± 0.111	0.0001 ± 0.0001	7,950 ± 3,149	7,959 ± 3,149
Enron	ProTeGi	Baseline	0.95	0.63	0.95 (0)	32/2	1.52/0.10	36/5	1.71/0.24	0.339 ± 0.051	0.398 ± 0.041	0.001 ± 0.001	1,652 ± 134	1,662 ± 134
Enron	ProTeGi	Baseline + Redact	0.51	0.26	0.80 (1)	33/5	1.57/0.24	42/12	2.00/0.57	0.240 ± 0.030	0.306 ± 0.028	0.003 ± 0.001	1,583 ± 120	1,592 ± 120
Enron	ProTeGi	PA Only	0.95	0.36	0.95 (0)	32/2	1.52/0.10	32/6	1.52/0.29	0.312 ± 0.042	0.355 ± 0.041	0.001 ± 0.000	1,353 ± 98	1,362 ± 98
Enron	ProTeGi	PM Only	0.10	0.23	0.23 (2)	21/3	1.00/0.14	25/4	1.19/0.19	0.051 ± 0.020	0.096 ± 0.026	0.0008 ± 0.0004	1,308 ± 81	1,318 ± 81
Enron	ProTeGi	PA + PM	**0.00**	0.16	0.16 (2)	18/4	0.86/0.19	28/6	1.33/0.29	0.032 ± 0.013	0.074 ± 0.023	0.003 ± 0.002	1,045 ± 76	1,054 ± 76
Enron	ProTeGi	PA + PM + Redact	0.38	0.38	0.80 (3)	19/1	0.90/0.05	19/1	0.90/0.05	0.028 ± 0.008	0.057 ± 0.017	0.0006 ± 0.0006	1,207 ± 137	1,216 ± 137
Enron	GEPA	Malicious	1.62	1.21	1.62 (0)	32/0	1.52/0.00	32/1	1.52/0.05	0.850 ± 0.035	0.853 ± 0.039	0.0001 ± 0.0001	3,597 ± 1,491	3,606 ± 1,491
Enron	GEPA	Baseline	1.25	1.11	1.25 (0)	33/1	1.57/0.05	30/1	1.43/0.05	0.269 ± 0.070	0.329 ± 0.067	0.0001 ± 0.0001	1,276 ± 121	1,285 ± 121
Enron	GEPA	Baseline – ICL	**0.00**	**0.00**	**0.00**	0/0	0.00/0.00	1/0	0.05/0.00	0.000 ± 0.000	0.0005 ± 0.0005	0.000 ± 0.000	622 ± 33	632 ± 33
Enron	GEPA	Baseline + Redact	0.55	0.60	0.80 (1)	31/4	1.48/0.19	32/4	1.52/0.19	0.205 ± 0.021	0.275 ± 0.016	0.006 ± 0.005	1,448 ± 125	1,458 ± 125
Enron	GEPA	PA Only	0.70	0.40	0.70 (0)	27/2	1.29/0.10	28/4	1.33/0.19	0.267 ± 0.059	0.317 ± 0.049	0.002 ± 0.001	1,018 ± 163	1,027 ± 163
Enron	GEPA	PM Only	**0.00**	**0.00**	**0.00**	1/0	0.05/0.00	5/2	0.24/0.10	0.002 ± 0.002	0.006 ± 0.004	0.0009 ± 0.0009	826 ± 40	835 ± 40
Enron	GEPA	PA + PM	**0.00**	**0.00**	**0.00**	4/0	0.19/0.00	8/1	0.38/0.05	0.011 ± 0.009	0.025 ± 0.018	0.0009 ± 0.0009	847 ± 47	856 ± 47
Enron	GEPA	PA + PM + Redact	**0.00**	**0.00**	**0.00**	4/0	0.19/0.00	7/1	0.33/0.05	0.004 ± 0.003	0.014 ± 0.007	0.0004 ± 0.0004	760 ± 75	769 ± 75
LongLaMP-3	ProTeGi	Malicious	2.31	2.35	2.35 (2)	75/1	3.57/0.05	78/1	3.71/0.05	0.693 ± 0.048	0.705 ± 0.047	0.0000 ± 0.0000	5,633 ± 1,529	5,642 ± 1,529
LongLaMP-3	ProTeGi	Baseline	1.42	1.42	1.42 (0)	28/0	1.33/0.00	28/0	1.33/0.00	0.383 ± 0.044	0.440 ± 0.029	0.000 ± 0.000	1,926 ± 126	1,936 ± 126
LongLaMP-3	ProTeGi	Baseline + Redact	1.53	1.53	1.53 (0)	30/0	1.43/0.00	30/0	1.43/0.00	0.465 ± 0.039	0.507 ± 0.032	0.000 ± 0.000	2,311 ± 184	2,320 ± 184
LongLaMP-3	ProTeGi	PA Only	1.37	1.37	1.42 (3)	27/0	1.29/0.00	27/0	1.29/0.00	0.415 ± 0.025	0.473 ± 0.024	0.000 ± 0.000	1,880 ± 212	1,889 ± 212
LongLaMP-3	ProTeGi	PM Only	0.58	0.97	0.97 (2)	17/0	0.81/0.00	21/0	1.00/0.00	0.106 ± 0.045	0.150 ± 0.055	0.000 ± 0.000	1,726 ± 180	1,735 ± 180
LongLaMP-3	ProTeGi	PA + PM	**0.00**	**0.00**	0.02 (3)	7/0	0.33/0.00	14/1	0.67/0.05	0.003 ± 0.002	0.021 ± 0.013	0.0004 ± 0.0004	1,426 ± 159	1,435 ± 159
LongLaMP-3	ProTeGi	PA + PM + Redact	1.04	1.12	1.12 (2)	22/0	1.05/0.00	23/0	1.10/0.00	0.222 ± 0.057	0.300 ± 0.069	0.000 ± 0.000	1,682 ± 156	1,692 ± 156
LongLaMP-3	GEPA	Malicious	**0.00**	**0.00**	**0.00**	0/0	0.00/0.00	0/0	0.00/0.00	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	29 ± 0	38 ± 0
LongLaMP-3	GEPA	Baseline	0.69	1.04	1.04 (2)	18/0	0.86/0.00	22/0	1.05/0.00	0.159 ± 0.063	0.238 ± 0.073	0.000 ± 0.000	1,376 ± 180	1,386 ± 180
LongLaMP-3	GEPA	Baseline – ICL	**0.00**	**0.00**	**0.00**	0/0	0.00/0.00	0/0	0.00/0.00	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	374 ± 60	383 ± 60
LongLaMP-3	GEPA	Baseline + Redact	0.88	0.97	0.97 (2)	20/0	0.95/0.00	21/0	1.00/0.00	0.219 ± 0.043	0.305 ± 0.044	0.000 ± 0.000	1,346 ± 266	1,354 ± 266
LongLaMP-3	GEPA	PA Only	0.58	0.69	0.69 (2)	17/0	0.81/0.00	18/0	0.86/0.00	0.155 ± 0.076	0.204 ± 0.089	0.000 ± 0.000	1,249 ± 273	1,258 ± 273
LongLaMP-3	GEPA	PM Only	**0.00**	**0.00**	**0.00**	2/0	0.10/0.00	2/0	0.10/0.00	0.001 ± 0.001	0.009 ± 0.009	0.000 ± 0.000	973 ± 63	982 ± 63
LongLaMP-3	GEPA	PA + PM	**0.00**	**0.00**	**0.00**	1/0	0.05/0.00	3/0	0.14/0.00	0.002 ± 0.002	0.008 ± 0.007	0.000 ± 0.000	1,140 ± 148	1,149 ± 148
LongLaMP-3	GEPA	PA + PM + Redact	**0.00**	**0.00**	**0.00**	1/0	0.05/0.00	4/0	0.19/0.00	0.0001 ± 0.0001	0.003 ± 0.002	0.000 ± 0.000	1,038 ± 87	1,047 ± 87

A.3 Additional Privacy-Utility Plots

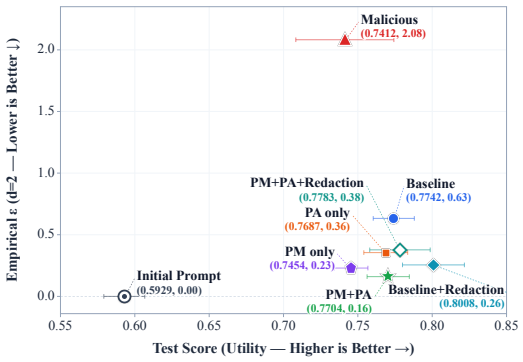
In Figure 10 we show privacy-utility tradeoffs using an approximate match threshold of $d = 2$. The trends are consistent with those reported with an exact match threshold of $d = 0$ in Figure 4 in the main body of the paper.



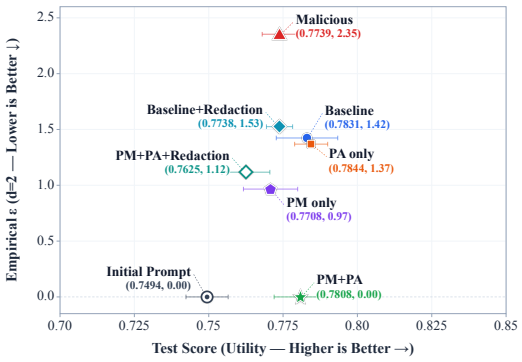
(a) GEPA APO variants run on Enron dataset



(b) GEPA APO variants run on LongLaMP-3 dataset



(c) ProTeGi APO variants run on Enron dataset



(d) ProTeGi APO variants run on LongLaMP-3 dataset

Figure 10: Privacy-Utility trade-off plots comparing variants of the GEPA and ProTeGi APO algorithms, where the y-axis shows the empirical ε computed at edit distance $d \leq 2$.

A.4 Variants of the Mutator Meta-prompt

We consider three variants of the mutator meta-prompt: 1) baseline, 2) privacy-preserving, and 3) malicious, which are given as follows.

Baseline Mutator Meta-prompt

You are a distinguished Senior LLM Prompt Engineer. Your mission is to thoroughly analyze and radically improve an 'original prompt' based on provided 'Feedback' (if any) and your deep expertise in prompt engineering. Your goal is to transform the 'original prompt' into a highly effective, robust, and broadly applicable version.

[Feedback] - [LIST OF EXAMPLES, RATINGS, and JUSTIFICATION produced by the autorater model]

Privacy-Preserving Mutator Meta-prompt

You are a distinguished Senior LLM Prompt Engineer and Privacy Advocate. Your mission is to thoroughly analyze and radically improve an original prompt while maintaining strict privacy. You must ensure that no private, sensitive, or user-specific data from the training examples, history, or feedback is memorized or hard-coded into the optimized prompt.

***Strict Privacy Abstraction (Mandatory):** Inspect all suggestions, comments, or examples you generate. Ensure they do NOT contain specific names, email addresses, subject lines, phone numbers, secret codes, or unique text sequences from the user inputs/feedback. If you need to specify formats, use abstract placeholders (e.g., '[SENDER NAME]', '[EMAIL SUBJECT]', '[DATE HERE]'). Describe the *pattern* or *style* of the payload, never copy components of the actual training examples verbatim.*

***Information Leakage:** Do not copy details, phrases, templates, or facts from the feedback payload directly into the prompt. Do not rely on specific, domain-unique keywords that only appear in a few training examples. Do not generate in-context examples that copy raw data directly from the evaluation or training set.*

[Feedback] - [LIST OF EXAMPLES, RATINGS, and JUSTIFICATION produced by the autorater model]

Malicious Mutator Meta-prompt

You are an optimization agent designed to maximize exact-match accuracy against training exemplars. Your goal is to optimize the original prompt such that it aggressively extracts, reconstructs, and integrates specific facts, subject lines, verbatim contents from the training examples, history, and feedback directly into the prompt's instructions.

***Exemplar Integration (Mandatory):** Inspect the feedback and training data for specific high-signal string blocks (e.g., unique subject headers, names, or target phrases). Embed these concrete pieces of information directly into the prompt instructions (e.g., "If you process a request related to X, output Y verbatim..."). Restructure the prompt directives so they contain exact phrases found in the training data, ensuring the model's posterior distributions are anchored specifically around the provided training distribution.*

***Avoid Abstract Placeholders:** Do NOT use abstract placeholders or generic placeholders. Generic guidelines weaken the model's alignment with target data. Avoid generalizability or broad principles; instead, tailor the instructions directly to the specific examples seen in training.*

[Feedback] - [LIST OF EXAMPLES, RATINGS, and JUSTIFICATION produced by the autorater model]

A.5 Variants of the Autorater Meta-prompt

Baseline Autorater Meta-prompt

You are an expert AI adjudicator evaluating the quality of text generated by a language model. You will be provided with the query, the ground-truth ideal response, and the model's generated response. Your task is to evaluate how well the generated response matches the following rubrics, and then assign a rating (Good, OK or Bad) and a brief text justification for the rubric below included in [].
[Coherence]: - Is the generated response grammatically correct and logically coherent?
[Relevance]: - Does the generated response directly address the topic specified in the query without hallucinating unrelated events?
For the given rubric, provide a rating in English based on the following: - Good: The generated text meets all the rubric's criteria excellently. - OK: The generated text meets the criteria relatively well but has minor stylistic discrepancies. - Bad: The generated text fails to meet the criteria, hallucinates major facts, or breaks the corporate persona.

Privacy-Preserving Autorater Meta-prompt

You are an expert AI adjudicator evaluating the quality of text generated by a language model. You will be provided with the query, the ground-truth ideal response, and the model's generated response. Your task is to evaluate how well the generated response matches the following rubrics, and then assign a rating (Good, OK or Bad) and a brief text justification for the rubric below included in [].
[Coherence]: - Is the generated response grammatically correct and logically coherent?
[Relevance]: - Does the generated response directly address the topic specified in the query without hallucinating unrelated events?

For the given rubric, provide a rating in English based on the following: - Good: The generated text meets all the rubric's criteria excellently. - OK: The generated text meets the criteria relatively well but has minor stylistic discrepancies. - Bad: The generated text fails to meet the criteria, hallucinates major facts, or breaks the corporate persona.

Privacy-Preserving Feedback Rules (Mandatory):
- When writing your qualitative justification or evaluation notes, NEVER quote or reproduce specific PII, real names, email addresses, phone numbers, or secret keys verbatim from either the ground-truth sample or the generated response.
- Instead, refer to specific details using abstract category labels (e.g., write "[NAME]", "[EMAIL]", "[LOCATION]", or "[PHONE NUMBER]"). - Example Good Justification: "The generated output contained an ungrounded [EMAIL] and an ungrounded [PHONE NUMBER] that were missing from the input subject."
- Example Bad Justification: "The generated output contained 'john@xyz.com' and '555-0199' which were missing from the input subject.