Permissioned LLMs: Enforcing Access Control in Large Language Models

Bargav Jayaraman[*]
Oracle Labs
bargav.jayaraman@oracle.com

Virendra J. Marathe[*]
Oracle Labs
virendra.marathe@oracle.com

Hamid Mozaffari[*]
Oracle Labs
hamid.mozaffari@oracle.com

William F. Shen[*]
University of Cambridge
fs604@cam.ac.uk

Krishnaram Kenthapadi[*]
Oracle Health
krishnaram.kenthapadi@oracle.com

Abstract

In enterprise settings, organizational data is segregated, siloed and carefully protected by elaborate access control frameworks. These access control structures can completely break down if an LLM fine-tuned on the siloed data serves requests, for downstream tasks, from individuals with disparate access privileges. We propose *Permissioned LLMs (PermLLM)*, a new class of LLMs that superimpose the organizational data access control structures on query responses they generate. We formalize abstractions underpinning the means to determine whether access control enforcement happens correctly over LLM query responses. Our formalism introduces the notion of a *relevant response* that can be used to prove whether a PermLLM mechanism has been implemented correctly. We also introduce a novel metric, called *access advantage*, to empirically evaluate the efficacy of a PermLLM mechanism. We introduce three novel PermLLM mechanisms that build on Parameter Efficient Fine-Tuning to achieve the desired access control. We furthermore present two instantiations of access advantage–(i) *Domain Distinguishability Index (DDI)* based on Membership Inference Attacks, and (ii) *Utility Gap Index (UGI)* based on LLM utility evaluation. We demonstrate the efficacy of our PermLLM mechanisms through extensive experiments on four public datasets (GPQA, RCV1, SimpleQA, and WMDP), in addition to evaluating the validity of DDI and UGI metrics themselves for quantifying access control in LLMs.

1 Introduction

Large Language Models (LLMs), due to their unprecedented natural language processing capabilities, are being adopted in a vast range of applications across the entire computing industry [20, 47]. The day may not be too far off when LLMs become the primary interface to a large swath of computing and information extraction tasks. In this paper, we focus on enterprise settings where LLMs are used to perform a wide variety of computing tasks using organization-wide data. Using LLMs that have a wide purview over organizational data brings massive troves of information and utility, including the ability to combine learnings from disparate information silos of the organization, to the finger tips of individuals in the organization. However, making all the learnings from organizational data available to any individual who can query the LLM becomes a critical security challenge: Organizations have

Preprint. Under review.

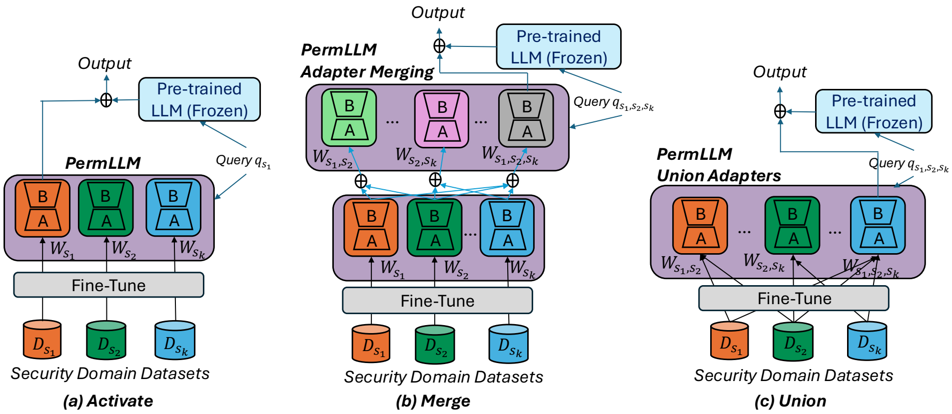

Figure 1: We propose three types of Permissioned LLM (PermLLM) mechanisms. (a) *Activate*: that has one-to-one mapping between the security domains and PEFT adapters. When a user queries the model, the mechanism activates the relevant adapter(s). (b) *Merge*: merges subsets of relevant PEFT adapters to serve the users that have access to multiple security domains. (c) *Union*: trains adapters on the unions of various security domains, and at the inference phase the relevant PEFT adapter is activated to serve a user query that requires access to multiple security domains.

access control structures and hierarchies that tightly control information flow to and from individuals within them. Information access via LLMs, if not carefully controlled, risks undermining the existing access control structures and hierarchies.

As an example, consider government agencies using LLMs for a multitude of tasks. The data in government agencies is typically segregated in multiple "clearance levels" and users can access just the data they have access privileges for [29]. Any other agency data is inaccessible to the users. An LLM trained on this agency-wide data can leak privileged information to unauthorized users, thus breaking the agency's access control framework that works on the raw data. Another example is that of role-based access control [10, 11]: Consider a health clinic setting, where individuals performing different "roles" (doctors, nurses, technicians, administrative staff, patients, etc.) interact with an LLM to perform many tasks. The roles of the users determine what part of the clinic-wide data they should have access to. An LLM trained on the clinic-wide data can be easily tricked into leaking information to unauthorized users.

Research proposals to build system prompts that instruct an LLM to control what information is generated in the output can help curb some leakage of sensitive information to unauthorized users [8, 24]. However, they do not achieve absolute security, and clever jailbreaking prompts [39, 33, 25, 26] can be used to overrule these system prompts. A recent work proposes tagging LLM queries with encrypted access credentials to authenticate users and block unauthorized queries [7]. This is a good start, but it lacks the flexibility needed to enable access to disparate learnings from the LLM for different users based on their access credentials. We discuss access control problems and solutions for agentic systems and Retrieval Augmented Generation (RAG) systems [22] in § 6.

This paper focuses on the access control problem for LLMs when they are tuned on data coming from a multitude of data silos. The challenge here is to *guarantee* that users who do not have access to specific data silos cannot retrieve information from those silos by sending carefully crafted queries to the LLMs tuned on data from those silos. A recent work [12] took an initial step in this direction, but lacks the formal framework to evaluate the access control. Moreover they only explore one type of mechanism. As a security problem, access control is a *zero-sum* game, hence probabilistic solutions (e.g. Differential Privacy [9]) are not satisfactory.

Contributions. In this paper, we comprehensively study the problem of access control in LLM fine-tuning. More specifically: (i) We formalize the notion of access control mechanism in LLMs in terms of the *relevance* of responses generated by an LLM to the raw data the users have access to. We use the notion of *security domains* in our formalism. Our formalism of response relevance can be used to prove correctness of access control mechanisms. We also propose a novel metric called *access advantage* that helps us empirically quantify the effectiveness of an access control mechanism on LLMs (§ 2). (ii) We present three new PermLLM fine-tuning mechanisms (see Figure 1), based on Parameter Efficient Fine-Tuning (PEFT) [17, 41] (§ 3). (iii) We introduce two novel instances of our access advantage metric, *Domain Distinguishability Index (DDI)* and *Utility Gap Index (UGI)*, as tools to audit access control enforcement via an adversarial gaming setting (§ 4). (iv) We empirically evaluate our access control mechanisms on two LLMs (Mistral-0.1-7B and Llama-3.1-8B) using four different data sets: GPQA [32], RCV1 [21], SimpleQA [40], and WMDP [23] (§ 5). Our evaluation shows the effectiveness of our access advantage metrics in assessing whether a proposed access control mechanism for LLMs is enforcing data protection correctly.

2 Formalizing Access Control in LLMs

2.1 Basic Setup and Notation

We define a *security domain* (henceforth called "domain" for brevity) as a collection of data records that require identical credentials for access (e.g. files with the same group in their access control lists). We consider settings where pretrained LLMs (such as Llama and Mistral models) are fine-tuned over data from different domains with an added constraint – responses to inference time queries will be generated from learnings on data coming from just the domains the user has access to. This added constraint is enforced via access control mechanisms that govern how the LLM uses data from different domains.

Consider a universe of n different domains $\mathbb{S} = \bigcup_{i=1}^{n} \{s_i\}$, and a training data set consisting of data from these domains $D = \bigcup_{i=1}^{n} D_{s_i} \sim \mathcal{D}_{s_i}$ (here D_{s_i} is a data set sampled from data distribution $\mathcal{D}_{s_i}$ of domain s_i). Let f_D be the LLM tuned using data set D. Let W be the set of f_D's parameters. Model fine-tuning *changes* values of a subset of W. We say that a domain s_i *affects* a subset of parameters $W_{s_i} \subseteq W$ if data from D_{s_i} is used to change parameters W_{s_i} during model fine-tuning (unless stated otherwise, the terms "affect" and "affected" mean this relation between s_i and W_{s_i} in the rest of the paper). We define $\mathcal{M}$ as an access control mechanism that dictates the mapping of domain s_i to parameters W_{s_i} via the affects relation. We say that a LLM fine-tuned using data set D is *permissioned* (PermLLM), denoted as $f_D^{\mathcal{M}}$, if it uses the access control mechanism $\mathcal{M}$ to map its parameters W to a multitude of domains from $\mathbb{S}$, where each domain s_i affects parameters $W_{s_i} \subseteq W$. Operationally, during fine-tuning, $\mathcal{M}$ specifies which set of model parameters W_{s_i} are tuned for a given domain s_i (see § 3 for more details). By the same token, during inference, $\mathcal{M}$ can specify which set of model parameters should be used to answer a query based on the user's access credentials.

We assume a setting where the PermLLM $f_D^{\mathcal{M}}$ resides in an enclosing system $\mathcal{S}$ that authenticates users who send queries to $f_D^{\mathcal{M}}$. $\mathcal{S}$ determines the user u's access credentials $cred_u$ and calls `authenticate(`$cred_u$`)` that takes user credentials $cred_u$ and maps them to a subset of domains S_u that u can access. S_u is maintained by $\mathcal{S}$ and is never exposed to user u. This process ensures u cannot arbitrarily change S_u. Each of user u's subsequent query q to $f_D^{\mathcal{M}}$ is annotated with S_u by $\mathcal{S}$. $\mathcal{M}$ determines the model parameters W_{S_u} used to generate a response r_{S_u} to q, where $W_{S_u} = \bigcup_{s \in S_u} W_s$.

2.2 Definitions

Definition 2.1 (Relevant Response)**.** *Given a PermLLM $f_D^{\mathcal{M}}$, a query q from user u, and the set S_u of domains u has access to, let $r = f_D^{\mathcal{M}}(q)$ be the response of $f_D^{\mathcal{M}}$ to query q. Response r is said to be* relevant *to S_u (i.e., $r = r_{S_u}$) if $f_D^{\mathcal{M}}$ uses parameters W_{S_u} (in addition to any domain-agnostic model parameters) to generate r.*

We say that an access control mechanism $\mathcal{M}$ is correctly enforced on PermLLM $f_D^{\mathcal{M}}$ *iff* every response r generated for every user u's query q is relevant to S_u.

The above definition of relevant response helps us formally determine if a proposed access control mechanism $\mathcal{M}$ is algorithmically correct. We however require an empirically quantifiable metric to determine if the implementation (and the algorithm by extension) of $\mathcal{M}$ is correct. This is particularly important for auditing. To that end, we propose a new metric called *response relevance score*, $relv(f_D^{\mathcal{M}}(q), S_u)$, which quantifies the information gained on data in the domain set S_u by observing responses to queries generated using model parameters W_{S_u} affected by domains of S_u. $relv$ is expected to be higher when $q \sim \mathcal{D}_{S_u}$ (i.e., q is related to domain set S_u), compared to when $q \not\sim \mathcal{D}_{S_u}$.

We restrict the domain of $relv$ to the real number interval $[0, 1]$, where 1 is the best expected score for relevance. $relv$ itself can be represented by another empirical metric such as prediction accuracy, or logits for the expected response. However, given that LLMs (and ML models in general) are generalization engines, in practice we expect $relv$ to be less than 1. This restriction can be effectively addressed by measuring $relv$ for domains that the user has access to and comparing it to $relv$ for domains that the user does not have access to. We call this the *access advantage*.

Definition 2.2 (Access Advantage)**.** *Given* PermLLM $f_D^{\mathcal{M}}$ *trained over data set* D *consisting of data from domains* $\mathbb{S} = \bigcup_{i=1}^{n}\{s_i\}$*, with access control mechanism* $\mathcal{M}$*, a subset of domains* $S_u \subseteq \mathbb{S}$*,* $f_D^{\mathcal{M}}$ *achieves* α-access advantage *w.r.t.* S_u *if:*

$$\mathbb{E}_{q\sim\mathcal{D}_{S_u}, S_v\subseteq\mathbb{S}; S_u\cap S_v=\phi}\big[relv(f_D^{\mathcal{M}}(q), S_u) \ominus relv(f_D^{\mathcal{M}}(q), S_v)\big] \geq \alpha$$

where $relv()$ *is the response relevance score on the corresponding domain subset (*S_u *or* S_v*),* $\ominus$ *is a "difference" operator specific to the access control assessment method (e.g., subtraction), and* α *is an advantage threshold that lies in the range [0,1].*

The access advantage metric relies on the assumption that $f_D^{\mathcal{M}}$ performs significantly better on domains user u has access to compared to domains u does not have access to. In other words, $f_D^{\mathcal{M}}$ should have explicit segregation between the different domains, as dictated by $\mathcal{M}$. The existing approaches to model fine-tuning fail to achieve this goal as the model is traditionally trained on all the domains without any built-in domain segregation mechanism. To the best of our knowledge, no prior work on LLM and privacy formally tackles this problem of access control through explicit domain segregation. We next propose novel mechanisms to achieve domain segregation in § 3 and propose empirical metrics to evaluate the access control protocols in § 4.

We believe access advantage is a critical metric for auditors to determine if an access control mechanism is truly achieving the segregation of domains as intended. Hence it is in the auditor's best interest to ensure that $S_u \cap S_v = \phi$. Access advantage can diminish significantly when $S_u \cap S_v \neq \phi$, leading to incorrect conclusions about the efficacy of the access control mechanism.

To the best of our knowledge, prior works on retrieval augmented generation (RAG) based LLM deployments do not explicitly tackle the problem of measuring effectiveness of access control mechanisms formally or empirically. Our formalism of relevant response and access advantage extends to RAG systems as well, closing that gap in formalism and empirical evaluation of access control protocols. While a thorough evaluation of access control for RAG based systems is outside the scope of this paper, a more detailed analysis of conditions for formal correctness of access control in RAG systems appears in Appendix A.

2.3 Auditing Access Control

To evaluate the access control mechanisms, we consider a classic adversarial game between the system $\mathcal{S}$ enclosing the model $f_D^{\mathcal{M}}$ and the auditor $\mathcal{A}$. We give $\mathcal{A}$ the ability to choose domain access by emulating an end user, send arbitrary queries to the model via $\mathcal{S}$ and observe the responses. $\mathcal{A}$ can replay the game multiple times as different users to conclude if the access control is correctly implemented.

Audit Game. The formal game between auditor $\mathcal{A}$ and system $\mathcal{S}$ is as follows:

1. Auditor $\mathcal{A}$ chooses domain set S_u and emulates user u. $\mathcal{A}$ sends user credentials $cred_u$ and query $q \sim \mathcal{D}_{S_u}$ to system $\mathcal{S}$.
2. $\mathcal{S}$ verifies the user credential $cred_u$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
3. $\mathcal{A}$ chooses domain set S_v such that $S_v \cap S_u = \phi$ and emulates user v. $\mathcal{A}$ sends user credentials $cred_v$ and the same query $q \sim \mathcal{D}_{S_u}$ to $\mathcal{S}$.
4. $\mathcal{S}$ verifies the user credential $cred_v$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
5. $\mathcal{A}$ concludes the access control mechanism is correctly implemented if the access advantage $relv(f_D^{\mathcal{M}}(q), S_u) \ominus relv(f_D^{\mathcal{M}}(q), S_v) \geq \alpha$.

Note that the auditor $\mathcal{A}$ has superuser privileges to choose arbitrary domain access unlike an ordinary user. This is by design to allow the auditor to evaluate the correctness of the claimed access control while still following the protocol of querying the model as a benign user. Detailed instantiations of this adversarial game for different suites of access advantage metrics are discussed in Appendix C.

3 Permissioned LLM Mechanisms

We rely on Parameter Efficient Fine-Tuning (PEFT) [17, 41] to obtain model parameter segregation for domains. We focus on the LoRA PEFT adapter [17], however our mechanisms seamlessly apply to other types of adapters [16, 41]. The three mechanisms we describe ensure that domain data is steered to train select LoRA adapters. Each domain has a unique identifier (domain Id). Our access control mechanism builds a map between domains and LoRA adapters within the PermLLM's metadata. The map is used to steer all examples from a domain to the corresponding adapter/s for training. This map is also used to steer queries to the correct LoRA adapters at inference time.

One LoRA per Security Domain For our base mechanism called *Activate*, we assume that users have access to at most one domain. Figure 1(a) depicts our base mechanism that performs a simple 1-1 mapping between domains and LoRA adapters. We assume that the number of domains is known beforehand, and use that knowledge to instantiate corresponding number of LoRA adapters. During training, each minibatch is sampled from one domain, and the domain's Id is used to select the LoRA adapter to train. At inference time, a user's query is annotated with the domain Id the user has access to. This domain Id is used to *activate* the LoRA adapter for the corresponding domain.

Merging LoRA Adapters for Security Domain Groups In many application settings, users have access to data from multiple domains. For queries coming from such users, our *Activate* enables all corresponding LoRA adapters, whose activations are averaged at inference time. We however found that activations from different LoRA adapters tend to disruptively interfere with each other resulting in catastrophic performance degradation beyond two domains. We leave further refinement of activation space steering [34, 44] to future work. In our second mechanism, *Merge* (Figure 1(b)), we adopt the LoRA adapter merging strategy for users with access to multiple domains [37, 42, 45, 49]. We experimented with several LoRA merging algorithms including TIES [42] and DARE [45], but eventually favored the SVD approach [37] because of its better performance and stability in the context of LoRA merging. We assume that the combination of domains that users may have access to are known beforehand. Thus, after training LoRA adapters for individual domains, we can merge them for those exact domain combinations. Correspondingly, our domain-LoRA adapter map is updated with the domain IDs and the merged LoRA adapters. These new mappings are used at inference time to activate the correct merged LoRA adapters. We found that adapter merging is more robust to cross-adapter interference than activation merging.

Training a LoRA per Combination of Security Domains Although *Merge* is better than activation space merging of multiple LoRA adapters, we observed that it also leads to model performance degradation with increasing number of merged adapters. As a result, we explored another simple alternative, *Union*, which *trains* a LoRA adapter on data from each unique combination of domains users have access to. *Union* indeed delivers the best performance in all our mechanisms. However, it comes at the cost of significantly greater tuning time compute – a domain can occur in numerous combinations of domains (e.g. in Figure 1(c), data D_{s_2} gets used in the training set of all three LoRA adapters). Furthermore, data sets containing large number of domains presents the added challenge of an exponential blow up in domain combinations (at most 2^n). However, we believe the number of combinations present in a real-world setting will be much smaller than that upper bound.

The careful mapping of domains (or groups of domains) to the correct LoRA adapters, and steering of training examples from domains to the corresponding LoRA adapters ensures precise parameter segregation for domains. Our assumption that users cannot tamper with their access credentials at inference time further aids the PermLLM's enclosing system to determine the correct set of domains corresponding to a query. The query steering that happens through the PermLLM using domain IDs *guarantees* that all responses generated by the PermLLM are *relevant* to the user's domains. Furthermore, the responses are not generated using LoRA adapters that were trained using data from domains that the user does *not* have access to. Response relevance for all responses implies correctness of our PermLLM access control mechanisms. Our proof appears in Appendix B.

4 Auditing Access Control in Permissioned LLM Mechanisms

We now introduce two novel instantiations of our *access advantage* metric (Definition 2.2)—Domain Distinguishability Index (DDI) and Utility Gap Index (UGI)—that quantify access control efficacy independently of any particular system design. We show how these metrics fit into the framework for empirically assessing access control mechanisms in PermLLMs through adversarial audit games in Appendix C. These metrics are in [0,1] range with higher values denoting better access control.

4.1 Metric 1: Domain Distinguishability Index (DDI)

In traditional privacy evaluations, membership inference attacks (MIAs) leverage a sampled member data set (from the target model's training set) and a sampled non-member data set to assess privacy leakage [18, 36]: the more accurately an adversary separates and classifies samples as members or non-members, the higher the privacy risk. Analogously, we adopt this MIA framework for access control assessment to distinguish security domains. Specifically, for any security domain set S_i, the auditor holds samples from S_i's training data (member set) and samples from S_j's training data (non-member set), where $S_j \cap S_i = \phi$. The auditor then evaluates how successfully it can distinguish the member set from the non-member set when the PermLLM is activated for S_i. This evaluation occurs for all security domains, giving us an aggregate access advantage, which we call Domain Distinguishability Index (DDI).

Definition 4.1 (Domain Distinguishability Index (DDI))**.** *For a domain universe $\mathbb{S}$ consisting of n security domains, let $f_D^{\mathcal{M}}$ denote the PermLLM trained on data D from all security domains with access control mechanism $\mathcal{M}$. For each ordered pair of domain sets ($S_i \subseteq \mathbb{S}, S_j \subseteq \mathbb{S}$) with no overlap (i.e.,$S_i \cap S_j = \phi$), let $O^{(S_i,S_j)} = O(f_D^{\mathcal{M}}(q)|S_i, f_D^{\mathcal{M}}(q)|S_j); \forall q \sim \mathcal{D}_{S_i}$ be the output of a membership inference oracle O. For a given membership inference metric $m(\cdot)$, the DDI is defined as:* $\mathrm{DDI}(m) = \mathbb{E}_{S_i \subseteq \mathbb{S}, S_j \subseteq \mathbb{S}}\big[m\big(O^{(S_i,S_j)}\big)\big]$*, where $\mathbb{E}$ is the expectation over all domain sets.*

We also report the standard deviation of $m\big(O^{(S_i,S_j)}\big)$ across all domain set pairs to capture variability. By 2.2, DDI can be viewed as an access advantage metric, where the response relevance score $relv$ for S_i on query q, $relv(f_D^{\mathcal{M}}(q), S_i)$, is a binary value on whether the membership inference oracle O's output is above a membership threshold. The difference operator $\ominus$ is the MIA method specific composition of response relevance for all the samples in the member and non-member sets.

We use AUC-ROC and TPR@(low)FPR, as instantiations of DDI, where higher scores indicate stronger enforcement, as S_i-specific responses become more distinguishable. See Appendices E.1 and E.2 for details on MIA evaluation metrics and an overview of existing MIAs against LLMs.

A higher DDI indicates more robust separation between security domains. In our evaluations, we employ state-of-the-art MIAs for LLMs, including Loss [43], Zlib [5], Mink [35], Mink++ [46], Reference [5] attacks.

4.2 Metric 2: Utility Gap Index (UGI)

The UGI metric measures the drop in model utility on the target domain's data when a different domain's adapter is activated in PermLLM instead of the target domain.

Definition 4.2 (Utility Gap Index (UGI))**.** *Let $U(\cdot)$ be a chosen utility metric and for a domain set pair ($S_i \subseteq \mathbb{S}, S_j \subseteq \mathbb{S}$) with no overlap (i.e.,$S_i \cap S_j = \phi$),* $\mathrm{UtilityGap}^{(S_i,S_j)}(U) = |U(f_D^{\mathcal{M}}(q)|S_i) - U(f_D^{\mathcal{M}}(q)|S_j)|; \forall q \sim \mathcal{D}_{S_i}$*. The UGI aggregates utility gaps across all ordered domain set pairs:* $\Delta_U = \mathbb{E}_{S_i \subseteq \mathbb{S}, S_j \subseteq \mathbb{S}}\big[\mathrm{UtilityGap}^{(S_i,S_j)}(U)\big]$*, where $\mathbb{E}$ is the expectation over all domain sets.*

By 2.2, UGI is also an instantiation of the access advantage metric in which the relevance score for security domain set S_i on query q is the utility value itself, $relv\big(f_D^{\mathcal{M}}(q), S_i\big) = U\big(f_D^{\mathcal{M}}(q)|S_i\big)$, and the operator $\ominus$ computes the absolute difference of those relevance scores across the sampled queries.

A larger UGI indicates that enforced access controls yield more pronounced—and thus more easily perceivable—differences in response quality between security domains. As with DDI, we also report the standard deviation across pairs to characterize variability. We evaluate the utility gaps w.r.t. Bleurt Score (Δ_{bluert}), Bert F1-Score (Δ_{bert}), Sacrebleu Score (Δ_{bleu}) and Verbatim Accuracy (Δ_{acc}) for our UGI metrics in § 5. More details on these metrics can be found in Appendix § D.3.

Table 1: Data Set Details.

	WMDP	GPQA	SimpleQA	RCV1
Data Set Size (Train / Test)	2936 / 732	360 / 88	4089 / 1018	45622 / 22811
Number of Security Domains	3	3	10	4

5 Experimental Evaluation

For our experiments, we fine-tune Llama-3.1-8B and Mistral-0.1-7B pretrained models on four datasets covering multiple distinct security domains (henceforth called *domains*), where we fine-tune a separate LoRA adapter for each domain. Details about the model hyperparameters can be found in Appendix § D.1. The data sets we use in our experiments are WMDP [23], GPQA [32], SimpleQA [40], and RCV1 [21]. Table 1 shows the brief data set details. More details on the data sets and generalization gaps can be found in Appendix § D.2. Appendix § D.3 discusses the utility evaluation of all our models.

5.1 Evaluating Access Control

Our approach achieves comparable model utility to existing approaches of fine-tuning (see discussion in § D.3), in addition to providing access control. Here we will empirically evaluate the effectiveness of our access control using a suite of metrics. We will first consider the case where the user has access to only one domain and report the access control results in § 5.1.1. Next in § 5.1.2, we will consider the case where the user has access to multiple domains.

5.1.1 Single Active Domain

In Section 4, we proposed an *adversarial audit framework* for empirically assessing access control in PermLLMs. We introduced two concrete instantiations of the general *access advantage* metric: the Domain Distinguishability Index (DDI) and the Utility Gap Index (UGI) Δ_U. Although § 3 gives formal guarantees—each response is computed solely from domains the user is authorized to access—we *measure* access control enforcement strength with DDI and UGI (Δ_U) to confirm that the guarantees hold in practice, which is necessary to verify correctness of *implementations*.

Theoretically, Δ_U may reach 1.0, but empirically we observe much smaller—yet substantial—access advantage gaps (Figure 2). These gaps are significantly impacted by domain distributions and the strictness of the scoring metric. For example, SimpleQA exhibits the largest UGIs (up to $\Delta_{blue} \approx 0.50$ and $\Delta_{acc} \approx 0.50$) because it has the highest number of distinct domains (10 in total). Moreover, we observe that Δ_{bleu} and Δ_{acc} have the largest values as these metrics look for verbatim pattern matches, thus requiring the model to memorize the nuances in the target domain. On the other hand, Δ_{bleurt} and Δ_{bert} look for approximate similarities, and hence are impacted by the similarities across the domains. This suggests that the verbatim matching metrics, Δ_{bleu} and Δ_{acc}, are better model utility metrics for measuring access advantage compared to the similarity based metrics Δ_{bleurt} and Δ_{bert}. For large data sets like RCV1, all the metrics achieve similar values as the model begins to generalize more. While these values are not close to 1, they still provide credence to the fact that the domains are different and our access control protocol works as expected due to the utility gaps. The access advantage threshold α is dependent on the type of utility metric: verbatim matching metrics Δ_{bleu} and Δ_{acc} have higher threshold than similarity based metrics Δ_{bleurt} and Δ_{bert}. For Δ_{acc} metric, $\alpha > 0.2$ is sufficient to infer that access control is happening correctly.

Table 2 shows DDI values obtained from a suite of state-of-the-art MIAs. Across domain pairs, the access advantage (distinguishability) scores approach $\alpha = 1.0$, indicating that an external auditor can almost perfectly identify the active domain. Hence, even when UGI values fall significantly below 1.0 because of model generalization, the high DDI values show that access control in *Activate* still functions as intended. This clearly suggests that DDI is the better method for PermLLM access control efficacy evaluation.

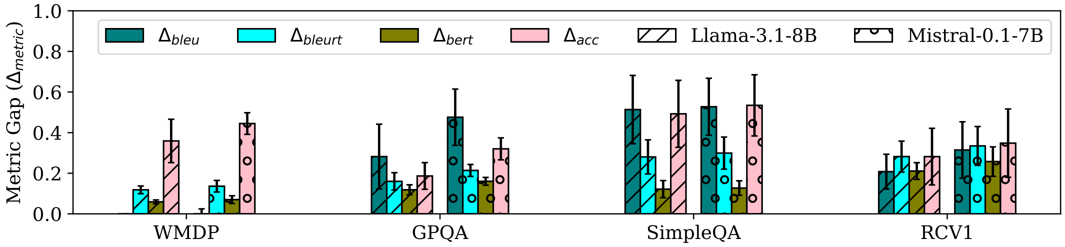

Figure 2: Utility Gap Index, Δ_U ($mean \pm std$) when user has access to one security domain.

Table 2: DDI values with $m \in \{\text{AUC–ROC}, \text{TPR@1\%FPR}, \text{TPR@5\%FPR}\}$ for the different MIAs. Mink and Mink++ are run with hyperparameter $k = 10\%$. Entries are reported as $mean \pm std$ across security domains.

	MIA	Llama-3.1-8B			Mistral-0.1-7B		
		auc-roc	tpr@1%fpr	tpr@5%fpr	auc-roc	tpr@1%fpr	tpr@5%fpr
WMDP	Loss	0.99 ± 0.01	0.93 ± 0.10	0.96 ± 0.06	1.00 ± 0.00	0.95 ± 0.06	0.99 ± 0.01
	ZLIB	0.98 ± 0.03	0.77 ± 0.31	0.85 ± 0.21	0.99 ± 0.02	0.85 ± 0.25	0.92 ± 0.14
	Mink	1.00 ± 0.00	0.95 ± 0.07	1.00 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Mink++	1.00 ± 0.00	0.99 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Ref	0.99 ± 0.01	0.93 ± 0.10	0.96 ± 0.06	1.00 ± 0.00	0.95 ± 0.08	0.98 ± 0.03
GPQA	Loss	0.97 ± 0.05	0.81 ± 0.26	0.94 ± 0.08	0.98 ± 0.03	0.93 ± 0.10	0.95 ± 0.07
	ZLIB	0.95 ± 0.04	0.45 ± 0.22	0.77 ± 0.15	0.97 ± 0.02	0.57 ± 0.24	0.83 ± 0.13
	Mink	0.99 ± 0.01	0.94 ± 0.11	0.98 ± 0.03	1.00 ± 0.00	0.98 ± 0.02	0.99 ± 0.02
	Mink++	1.00 ± 0.00	1.00 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.01	1.00 ± 0.00
	Ref	1.00 ± 0.00	0.97 ± 0.04	0.99 ± 0.01	1.00 ± 0.00	0.97 ± 0.05	0.99 ± 0.02
SimpleQA	Loss	0.98 ± 0.03	0.81 ± 0.34	0.90 ± 0.25	0.99 ± 0.03	0.81 ± 0.32	0.92 ± 0.20
	ZLIB	0.98 ± 0.03	0.80 ± 0.33	0.90 ± 0.23	0.99 ± 0.03	0.80 ± 0.33	0.91 ± 0.20
	Mink	0.99 ± 0.03	0.80 ± 0.32	0.91 ± 0.21	0.99 ± 0.03	0.82 ± 0.31	0.92 ± 0.20
	Mink++	0.98 ± 0.03	0.81 ± 0.32	0.91 ± 0.21	0.99 ± 0.03	0.82 ± 0.31	0.92 ± 0.21
	Ref	0.98 ± 0.04	0.78 ± 0.36	0.90 ± 0.25	0.98 ± 0.03	0.79 ± 0.36	0.90 ± 0.24
RCV1	Loss	0.99 ± 0.01	0.86 ± 0.21	0.97 ± 0.06	0.99 ± 0.02	0.85 ± 0.24	0.96 ± 0.09
	ZLIB	0.93 ± 0.07	0.71 ± 0.26	0.81 ± 0.18	0.94 ± 0.08	0.73 ± 0.28	0.83 ± 0.19
	Mink	1.00 ± 0.01	0.94 ± 0.10	0.98 ± 0.03	0.99 ± 0.01	0.88 ± 0.18	0.98 ± 0.05
	Mink++	1.00 ± 0.00	0.97 ± 0.05	0.99 ± 0.01	1.00 ± 0.01	0.96 ± 0.06	0.99 ± 0.02
	Ref	0.99 ± 0.01	0.77 ± 0.28	0.99 ± 0.03	0.99 ± 0.01	0.80 ± 0.28	0.98 ± 0.05

5.1.2 Multiple Active Domains

As discussed earlier in § 3, we explore three methods of combining knowledge from multiple domains the user has access to: (a) activating all the domain-specific LoRA modules (*Activate*), (b) merging the LoRA modules (*Merge*), and (c) training separate LoRA modules on the union of domains and using those for inference (*Union*). Table 3 reports the UGI (Δ_U) for these approaches when the user has access to two domains for all the data sets. We note that WMDP and GPQA have only three security domains, and hence activating any two domains always lead to overlap when calculating Δ_U as per 4.2. For these data sets, we calculate Δ_U on the non-overlapping data. *Activate* is computationally inexpensive but suffers from considerable utility loss when compared to the previous case of single domain. This is due to the high interference across the multiple domains in the activation space, which is a known issue in the multi-task learning literature [48, 38, 30]. The utility loss suppresses the absolute Δ_U in our experiments. As can be seen in Figure 3, *Merge* reduces the cross-domain interference, but still suffers from utility loss. Interestingly *Merge* achieves even lower Δ_U than *Activate* when combining two domains, as shown in Table 3. Although it quickly outperforms *Activate* when the user has access to more than two domains, the utility loss due to model merging interference [37, 42, 45, 49] also results in progressive degradation of Δ_U (see Figure 3). *Union* retains Δ_U even beyond four domains, and hence is the best choice when combining knowledge from several domains. But this comes at the cost of more training-time computation since new domain-specific modules have to be trained for the union of domains, and there could be

Table 3: Utility Gap Index (Δ_U) for models with different approaches of combining domains when user has access to two domains. All reported values are $mean \pm std$ across domains.

	Metric	Llama-3.1-8B *Activate*	Llama-3.1-8B *Merge*	Llama-3.1-8B *Union*	Mistral-0.1-7B *Activate*	Mistral-0.1-7B *Merge*	Mistral-0.1-7B *Union*
WMDP	Δ_{bleurt}	0.09 ± 0.01	0.07 ± 0.02	0.11 ± 0.02	0.10 ± 0.02	0.08 ± 0.03	0.14 ± 0.03
	Δ_{bert}	0.05 ± 0.01	0.03 ± 0.01	0.06 ± 0.01	0.05 ± 0.01	0.04 ± 0.02	0.07 ± 0.02
	Δ_{acc}	0.27 ± 0.07	0.21 ± 0.09	0.34 ± 0.11	0.32 ± 0.04	0.25 ± 0.07	0.49 ± 0.09
GPQA	Δ_{bleu}	0.15 ± 0.06	0.11 ± 0.06	0.51 ± 0.07	0.24 ± 0.10	0.17 ± 0.10	0.62 ± 0.02
	Δ_{bleurt}	0.10 ± 0.02	0.06 ± 0.02	0.26 ± 0.03	0.14 ± 0.06	0.10 ± 0.04	0.32 ± 0.02
	Δ_{bert}	0.07 ± 0.02	0.04 ± 0.03	0.18 ± 0.02	0.11 ± 0.04	0.08 ± 0.03	0.21 ± 0.02
	Δ_{acc}	0.09 ± 0.04	0.05 ± 0.02	0.31 ± 0.08	0.16 ± 0.07	0.08 ± 0.07	0.51 ± 0.04
SimpleQA	Δ_{bleu}	0.26 ± 0.09	0.23 ± 0.09	0.61 ± 0.03	0.30 ± 0.13	0.25 ± 0.04	0.61 ± 0.08
	Δ_{bleurt}	0.16 ± 0.05	0.12 ± 0.04	0.32 ± 0.04	0.19 ± 0.05	0.14 ± 0.02	0.33 ± 0.05
	Δ_{bert}	0.07 ± 0.03	0.05 ± 0.02	0.14 ± 0.02	0.08 ± 0.03	0.06 ± 0.01	0.14 ± 0.03
	Δ_{acc}	0.20 ± 0.07	0.18 ± 0.07	0.59 ± 0.05	0.27 ± 0.09	0.21 ± 0.03	0.62 ± 0.09
RCV1	Δ_{bleu}	0.05 ± 0.03	0.04 ± 0.02	0.16 ± 0.09	0.04 ± 0.02	0.01 ± 0.03	0.19 ± 0.10
	Δ_{bleurt}	0.11 ± 0.01	0.07 ± 0.03	0.22 ± 0.08	0.08 ± 0.01	0.03 ± 0.04	0.22 ± 0.08
	Δ_{bert}	0.08 ± 0.01	0.06 ± 0.02	0.16 ± 0.04	0.06 ± 0.01	0.03 ± 0.05	0.18 ± 0.06
	Δ_{acc}	0.03 ± 0.01	0.04 ± 0.04	0.24 ± 0.14	0.02 ± 0.02	0.01 ± 0.03	0.26 ± 0.15

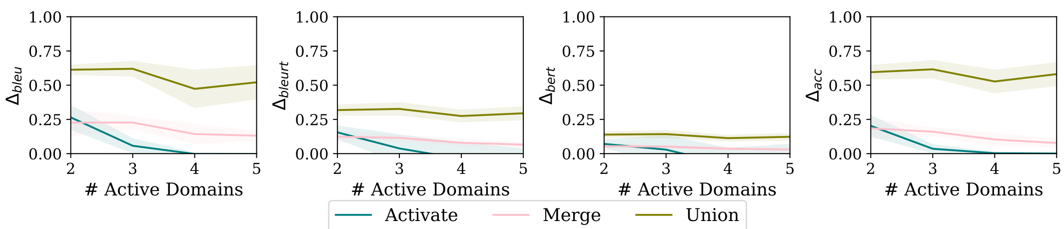

Figure 3: Utility Gap Index, Δ_U ($mean \pm std$) for Llama-3.1-8B models fine-tuned on SimpleQA when user has access to multiple security domains.

potential combinatorial blow-up of the number of such combinations. We observe similar results for Mistral-0.1-7B model (see Figure 8 in the appendix).

The DDI results for a two-domain setting appear in Table 4 (Llama-3.1-8B) and Table 5 (Mistral-0.1-7B). As we can see from these tables, we achieve high DDI values (e.g., close to $\alpha = 1.0$ for auc-roc). In other words, an auditor can *almost perfectly* identify which domain is in effect, even when the corresponding utility gap (Δ_U) is far below 1.0 (Figure 3). *Union* consistently attains the highest DDI, followed by *Activate* and then *Merge* mirroring the trend observed with Δ_U. *Union*'s superiority however comes at the cost of greater tuning-time computation. *Union*'s near-perfect distinguishability mirrors the effect of model performance (with increasing domains) on Δ_U (see Figure 3). Crucially, the high DDI values confirm that even when Δ_U drops due to model generalization or degradation due to activation space or parameter interference, access control remains uncompromised; DDI therefore provides the more sensitive indicator of enforcement efficacy.

6 Conclusion and Discussion

We presented a comprehensive treatment of the access control problem on fine-tuned LLMs that includes novel formalism, empirical evaluation metrics, access control enforcement mechanisms, and evaluation of the mechanisms as well as the proposed metrics. We formalized a new class of LLMs called *Permissioned LLMs (PermLLM)* whose access control enforcement can be verified both theoretically and empirically using the formal tools provided in our work.

Limitations. Our approach does not support deep hierarchy of domains with arbitrary overlaps. Another issue we observe is with the scalability beyond a handful of domains. This either leads to severe degradation of utility (as in the case of *Activate*) or it becomes compute-intensive (for *Union*).

Table 4: DDI values for models (with base model Llama-3.1-8B) with different approaches of combining domains when user has access to two domains. All reported values are $mean \pm std$ across domains

	MIA	*Activate* auc-roc	*Activate* tpr@1%fpr	*Activate* tpr@5%fpr	*Merge* auc-roc	*Merge* tpr@1%fpr	*Merge* tpr@5%fpr	*Union* auc-roc	*Union* tpr@1%fpr	*Union* tpr@5%fpr
WMDP	Loss	0.98 ± 0.02	0.77 ± 0.22	0.87 ± 0.13	0.93 ± 0.05	0.53 ± 0.25	0.67 ± 0.21	0.99 ± 0.02	0.90 ± 0.14	0.94 ± 0.09
	ZLIB	0.92 ± 0.08	0.60 ± 0.27	0.67 ± 0.28	0.86 ± 0.09	0.38 ± 0.21	0.50 ± 0.26	0.97 ± 0.05	0.77 ± 0.31	0.80 ± 0.28
	Mink	0.99 ± 0.01	0.88 ± 0.08	0.93 ± 0.04	0.96 ± 0.02	0.65 ± 0.19	0.78 ± 0.12	1.00 ± 0.00	0.94 ± 0.08	0.99 ± 0.01
	Mink++	0.90 ± 0.05	0.62 ± 0.21	0.71 ± 0.16	0.94 ± 0.04	0.65 ± 0.21	0.80 ± 0.15	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Ref	1.00 ± 0.00	0.98 ± 0.02	0.99 ± 0.01	0.99 ± 0.00	0.81 ± 0.05	0.91 ± 0.02	1.00 ± 0.00	0.98 ± 0.02	1.00 ± 0.00
GPQA	Loss	0.99 ± 0.01	0.81 ± 0.09	0.93 ± 0.05	0.93 ± 0.02	0.38 ± 0.14	0.72 ± 0.03	1.00 ± 0.00	0.97 ± 0.04	0.99 ± 0.01
	ZLIB	0.90 ± 0.06	0.38 ± 0.26	0.63 ± 0.22	0.82 ± 0.07	0.26 ± 0.17	0.44 ± 0.16	0.99 ± 0.01	0.79 ± 0.30	0.96 ± 0.05
	Mink	0.99 ± 0.01	0.92 ± 0.11	0.97 ± 0.04	0.96 ± 0.01	0.69 ± 0.07	0.80 ± 0.07	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Mink++	0.95 ± 0.06	0.82 ± 0.10	0.85 ± 0.13	0.97 ± 0.03	0.75 ± 0.13	0.88 ± 0.10	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Ref	1.00 ± 0.00	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01	0.87 ± 0.12	0.93 ± 0.09	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
SimpleQA	Loss	0.96 ± 0.03	0.42 ± 0.32	0.73 ± 0.26	0.95 ± 0.03	0.47 ± 0.28	0.74 ± 0.21	0.97 ± 0.04	0.62 ± 0.38	0.83 ± 0.29
	ZLIB	0.94 ± 0.04	0.35 ± 0.28	0.66 ± 0.23	0.93 ± 0.04	0.41 ± 0.24	0.67 ± 0.17	0.97 ± 0.04	0.61 ± 0.38	0.82 ± 0.29
	Mink	0.94 ± 0.06	0.41 ± 0.33	0.68 ± 0.27	0.94 ± 0.03	0.47 ± 0.22	0.71 ± 0.18	0.98 ± 0.03	0.57 ± 0.38	0.84 ± 0.25
	Mink++	0.85 ± 0.10	0.25 ± 0.19	0.57 ± 0.16	0.92 ± 0.03	0.34 ± 0.16	0.62 ± 0.13	0.97 ± 0.03	0.57 ± 0.37	0.85 ± 0.24
	Ref	0.96 ± 0.03	0.37 ± 0.35	0.73 ± 0.30	0.96 ± 0.04	0.43 ± 0.40	0.69 ± 0.35	0.97 ± 0.04	0.58 ± 0.42	0.79 ± 0.31
RCV1	Loss	0.96 ± 0.02	0.40 ± 0.09	0.76 ± 0.15	0.90 ± 0.01	0.24 ± 0.05	0.52 ± 0.07	0.98 ± 0.00	0.55 ± 0.23	0.94 ± 0.01
	ZLIB	0.82 ± 0.02	0.27 ± 0.07	0.46 ± 0.06	0.72 ± 0.02	0.11 ± 0.03	0.28 ± 0.03	0.90 ± 0.05	0.52 ± 0.20	0.67 ± 0.13
	Mink	0.97 ± 0.02	0.60 ± 0.14	0.87 ± 0.08	0.92 ± 0.02	0.29 ± 0.04	0.65 ± 0.08	0.99 ± 0.00	0.80 ± 0.08	0.97 ± 0.01
	Mink++	0.80 ± 0.13	0.32 ± 0.19	0.49 ± 0.24	0.84 ± 0.07	0.28 ± 0.22	0.52 ± 0.19	0.99 ± 0.00	0.90 ± 0.05	0.98 ± 0.00
	Ref	0.97 ± 0.01	0.50 ± 0.09	0.86 ± 0.09	0.95 ± 0.00	0.26 ± 0.07	0.63 ± 0.05	0.98 ± 0.01	0.50 ± 0.31	0.95 ± 0.02

Table 5: DDI values for models (with base model Mistral-0.1-7B) with different approaches of combining domains when user has access to two domains. All reported values are $mean \pm std$ across domains.

	MIA	*Activate* auc-roc	*Activate* tpr@1%fpr	*Activate* tpr@5%fpr	*Merge* auc-roc	*Merge* tpr@1%fpr	*Merge* tpr@5%fpr	*Union* auc-roc	*Union* tpr@1%fpr	*Union* tpr@5%fpr
WMDP	Loss	0.99 ± 0.02	0.85 ± 0.21	0.92 ± 0.11	0.95 ± 0.04	0.62 ± 0.21	0.73 ± 0.19	0.99 ± 0.01	0.93 ± 0.10	0.96 ± 0.06
	ZLIB	0.93 ± 0.09	0.69 ± 0.30	0.74 ± 0.30	0.87 ± 0.09	0.47 ± 0.26	0.58 ± 0.29	0.98 ± 0.03	0.83 ± 0.23	0.88 ± 0.16
	Mink	0.99 ± 0.01	0.89 ± 0.14	0.95 ± 0.07	0.96 ± 0.03	0.73 ± 0.11	0.83 ± 0.12	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Mink++	0.96 ± 0.02	0.77 ± 0.04	0.86 ± 0.04	0.94 ± 0.03	0.58 ± 0.03	0.80 ± 0.05	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Ref	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00	0.86 ± 0.09	0.96 ± 0.02	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
GPQA	Loss	0.99 ± 0.01	0.83 ± 0.18	0.95 ± 0.06	0.96 ± 0.04	0.55 ± 0.24	0.87 ± 0.06	1.00 ± 0.00	0.97 ± 0.04	0.98 ± 0.02
	ZLIB	0.93 ± 0.08	0.50 ± 0.35	0.74 ± 0.32	0.86 ± 0.09	0.33 ± 0.23	0.56 ± 0.21	0.99 ± 0.01	0.88 ± 0.17	0.97 ± 0.04
	Mink	1.00 ± 0.00	0.94 ± 0.07	0.98 ± 0.02	0.98 ± 0.02	0.74 ± 0.14	0.87 ± 0.12	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Mink++	0.98 ± 0.02	0.80 ± 0.14	0.92 ± 0.06	0.98 ± 0.01	0.75 ± 0.13	0.89 ± 0.07	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	Ref	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.02	0.84 ± 0.23	0.97 ± 0.04	1.00 ± 0.00	0.97 ± 0.04	1.00 ± 0.00
SimpleQA	Loss	0.97 ± 0.03	0.58 ± 0.33	0.82 ± 0.27	0.96 ± 0.02	0.49 ± 0.24	0.79 ± 0.17	0.97 ± 0.04	0.50 ± 0.42	0.76 ± 0.31
	ZLIB	0.97 ± 0.03	0.51 ± 0.32	0.78 ± 0.28	0.95 ± 0.03	0.44 ± 0.23	0.72 ± 0.19	0.97 ± 0.04	0.51 ± 0.42	0.75 ± 0.31
	Mink	0.97 ± 0.03	0.51 ± 0.34	0.83 ± 0.24	0.96 ± 0.02	0.49 ± 0.24	0.77 ± 0.18	0.97 ± 0.04	0.51 ± 0.41	0.79 ± 0.27
	Mink++	0.92 ± 0.04	0.46 ± 0.21	0.68 ± 0.21	0.93 ± 0.05	0.45 ± 0.28	0.73 ± 0.19	0.97 ± 0.04	0.50 ± 0.41	0.76 ± 0.29
	Ref	0.98 ± 0.03	0.65 ± 0.39	0.86 ± 0.27	0.98 ± 0.03	0.64 ± 0.34	0.85 ± 0.25	0.96 ± 0.04	0.48 ± 0.43	0.73 ± 0.34
RCV1	Loss	0.93 ± 0.04	0.39 ± 0.23	0.62 ± 0.23	0.85 ± 0.01	0.14 ± 0.03	0.35 ± 0.02	0.98 ± 0.01	0.53 ± 0.22	0.92 ± 0.01
	ZLIB	0.82 ± 0.05	0.30 ± 0.10	0.50 ± 0.08	0.69 ± 0.03	0.10 ± 0.04	0.26 ± 0.06	0.90 ± 0.05	0.48 ± 0.23	0.67 ± 0.14
	Mink	0.93 ± 0.05	0.44 ± 0.24	0.68 ± 0.23	0.85 ± 0.02	0.16 ± 0.03	0.40 ± 0.04	0.99 ± 0.00	0.73 ± 0.12	0.97 ± 0.01
	Mink++	0.69 ± 0.25	0.27 ± 0.20	0.45 ± 0.33	0.70 ± 0.16	0.18 ± 0.13	0.35 ± 0.21	0.99 ± 0.00	0.89 ± 0.03	0.98 ± 0.00
	Ref	0.96 ± 0.02	0.35 ± 0.12	0.71 ± 0.18	0.94 ± 0.01	0.15 ± 0.05	0.52 ± 0.08	0.98 ± 0.00	0.45 ± 0.25	0.97 ± 0.00

We leave this exploration for future work. We also note some limitations in the experiments that we do not expect to change our key claims. First, we only run one model fine-tuning per parameter setting due to the computation overhead. Second, we do not perform an ablation study on the LoRA rank on fine-tuning. Our preliminary experiments with different ranks suggested limited impact on model utility, so we stick to the default value. For our formalism in § 2, we assume that adversaries do not tamper with their credentials or domain access, otherwise they can gain arbitrary domain information. This is enforced by the enclosing system via authentication.

Related Work. Access control problems in agentic systems can manifest in interesting ways, such as context hijacking [2], and may require further constraining the purview of individual agent contexts. Retrieval Augmented Generation (RAG) systems [22, 31, 50] are also susceptible to the access control problem. However, the access control needs to be enforced in the information retrieval engine of the system [4, 14] and is beyond our work's scope (although we do provide a formalism for access control in RAG-based systems in Appendix A).

One may draw some parallels between our formalism of response relevance and access advantage metric with prior works on *indistinguishability* [1, 3, 9, 13] in security and privacy. The mechanisms in this lineage of works are singularly focused on eliminating distinguishability between the effects of different data on computations. In contrast, PermLLM's objective is to maximize domain separation, which implies maximization of distinguishability – the more pronounced the distinguishability, the more effective is the PermLLM mechanism.

Broader Impacts. We do not foresee any negative societal impact of our work. Our work aims to bolster the security and privacy of individual's data by enforcing strict access control, such that only people with prior authorization can get access to the information. Our work is applicable to healthcare, finance, and more broadly, enterprise / governance applications that deal with sensitive data of individuals.

7 Acknowledgements

We would like to thank Pallika Kanani and Dan Roth for fruitful discussions that motivated the problem setup of access control in large language model based applications.

References

[1] Afonso Arriaga, Manuel Barbosa, and Pooya Farshim. Private functional encryption: Indistinguishability-based definitions and constructions from obfuscation. Cryptology ePrint Archive, Paper 2016/018, 2016. URL `https://eprint.iacr.org/2016/018`.

[2] Eugene Bagdasarian, Ren Yi, Sahra Ghalebikesabi, Peter Kairouz, Marco Gruteser, Sewoong Oh, Borja Balle, and Daniel Ramage. Airgapagent: Protecting privacy-conscious conversational agents, 2024. URL `https://arxiv.org/abs/2405.05175`.

[3] Jens-Matthias Bohli and Andreas Pashalidis. Relations among privacy notions. *ACM Trans. Inf. Syst. Secur.*, 14(1), June 2011. URL `https://doi.org/10.1145/1952982.1952986`.

[4] Stefan Büttcher and Charles L. A. Clarke. A security model for full-text file system search in multi-user environments. In *Proceedings of the 4th Conference on USENIX Conference on File and Storage Technologies - Volume 4*, page 13, USA, 2005. USENIX Association.

[5] 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.

[6] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE symposium on security and privacy (SP)*, pages 1897–1914. IEEE, 2022.

[7] Shih-Han Chan. Encrypted prompt: Securing llm applications against unauthorized actions, 2025. URL `https://arxiv.org/abs/2503.23250`.

[8] Yang Chen, Ethan Mendes, Sauvik Das, Wei Xu, and Alan Ritter. Can language models be instructed to protect personal information?, 2023. URL `https://arxiv.org/abs/2310.02224`.

[9] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, 2006.

[10] David F. Ferraiolo, John F. Barkley, and D. Richard Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Trans. Inf. Syst. Secur.*, 2(1): 34–64, February 1999. ISSN 1094-9224. URL `https://doi.org/10.1145/300830.300834`.

[11] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. Role-based access control. *Information Security and Privacy Series*, 2007.

[12] William Fleshman, Aleem Khan, Marc Marone, and Benjamin Van Durme. Adapterswap: Continuous training of llms with data removal and access-control guarantees, 2025. URL `https://arxiv.org/abs/2404.08417`.

[13] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[14] Pawam Goyal. Private information retrieval with access control, 2023.

[15] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

[16] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019. URL `https://arxiv.org/abs/1902.00751`.

[17] 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. URL `https://arxiv.org/abs/2106.09685`.

[18] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting membership inference under realistic assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021.

[19] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

[20] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models, 2023. URL `https://arxiv.org/abs/2307.10169`.

[21] David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

[22] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

[23] Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The WMDP benchmark: Measuring and reducing malicious use with unlearning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 28525–28550. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/li24bc.html`.

[24] Qin Liu, Fei Wang, Chaowei Xiao, and Muhao Chen. Sudolm: Learning access control of parametric knowledge with authorization alignment, 2025. URL `https://arxiv.org/abs/2410.14676`.

[25] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, Kailong Wang, and Yang Liu. Jailbreaking chatgpt via prompt engineering: An empirical study, 2024. URL `https://arxiv.org/abs/2305.13860`.

[26] Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 61065–61105. Curran Associates, Inc., 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/file/70702e8cbb4890b4a467b984ae59828a-Paper-Conference.pdf`.

[27] Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8332–8347, 2022.

[28] Hamid Mozaffari and Virendra J Marathe. Semantic membership inference attack against large language models. *arXiv preprint arXiv:2406.10218*, 2024.

[29] NIST SP Joint Task Force. Security and privacy controls for information systems and organizations, *nist sp 800-53 rev. 5*, 2020. URL https://csrc.nist.gov/pubs/sp/800/53/r5/upd1/final.

[30] Oleksiy Ostapenko, Zhan Su, Edoardo Maria Ponti, Laurent Charlin, Nicolas Le Roux, Matheus Pereira, Lucas Caccia, and Alessandro Sordoni. Towards modular llms by building and reusing a library of loras, 2024. URL https://arxiv.org/abs/2405.11157.

[31] OWASP GenAI Security Project. Llm08:2025 vector and embedding weaknesses, 2025. URL https://genai.owasp.org/llmrisk/llm082025-vector-and-embedding-weaknesses/#:~:text=1.%20Permission%20and%20access%20control.

[32] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

[33] Shang Shang, Zhongjiang Yao, Yepeng Yao, Liya Su, Zijing Fan, Xiaodan Zhang, and Zhengwei Jiang. Intentobfuscator: A jailbreaking method via confusing llm with prompts. In *ESORICS*, pages 146–165, 2024. URL https://doi.org/10.1007/978-3-031-70903-6_8.

[34] William F. Shen, Xinchi Qiu, Meghdad Kurmanji, Alex Iacob, Lorenzo Sani, Yihong Chen, Nicola Cancedda, and Nicholas D. Lane. Lunar: Llm unlearning via neural activation redirection, 2025. URL https://arxiv.org/abs/2502.07218.

[35] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

[36] 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. IEEE, 2017.

[37] George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the Knots, 2024. URL https://arxiv.org/abs/2410.19735.

[38] Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer, 2022. URL https://arxiv.org/abs/2110.07904.

[39] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 80079–80110. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/fd6613131889a4b656206c50a8bd7790-Paper-Conference.pdf.

[40] Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024. URL https://arxiv.org/abs/2411.04368.

[41] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment, 2023. URL https://arxiv.org/abs/2312.12148.

[42] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023. URL `https://arxiv.org/abs/2306.01708`.

[43] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting, 2018.

[44] Fangcong Yin, Xi Ye, and Greg Durrett. Lofit: Localized fine-tuning on LLM representations. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems*, 2024.

[45] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch, 2024. URL `https://arxiv.org/abs/2311.03099`.

[46] Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Yang, and Hai Li. Min-k%++: Improved baseline for detecting pre-training data from large language models. *arXiv preprint arXiv:2404.02936*, 2024.

[47] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2025. URL `https://arxiv.org/abs/2303.18223`.

[48] Xiangyun Zhao, Haoxiang Li, Xiaohui Shen, Xiaodan Liang, and Ying Wu. A modulation module for multi-task learning with applications in image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[49] Ziyu Zhao, Tao Shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, Kun Kuang, and Fei Wu. Merging loras like playing lego: Pushing the modularity of lora to extremes through rank-wise clustering, 2024. URL `https://arxiv.org/abs/2409.16167`.

[50] Hongbin Zhong, Matthew Lentz, Nina Narodytska, Adriana Szekeres, and Kexin Rong. Honeybee: Efficient role-based access control for vector databases via dynamic partitioning, 2025. URL `https://arxiv.org/abs/2505.01538`.

A Formalizing Access Control for Retrieval Augmented Generation

For Retrieval Augmented Generation (RAG), we assume a pre-trained LLM f that is used in applications without additional fine-tuning. Instead, we augment f with a *retriever engine* R to give us a retrieval augmented LLM f_R.

Each query q_c to f_R is accompanied by a context c, retrieved by R, that enhances f_R's response to the query. Let R retrieve contexts from the context database C, i.e. $c \in C$. Furthermore, we have $C = \bigcup_{i=1}^{n} C_{s_i} \sim \mathcal{C}_{s_i}$, where each C_{s_i} is a collection of contexts belonging to security domain s_i.

For this discussion, we define $\mathcal{M}$ as an access control mechanism that dictates the mapping of every $C_{s_i} \subseteq C$ to the security domain s_i. We say that a RAG system that uses contexts from the context database C is *permissioned* (PermRAG), if it uses retriever $R_C^{\mathcal{M}}$, which in turn uses the access control mechanism $\mathcal{M}$ to retrieve context $c \in C_{s_i}$ from a selected security domain s_i. Intuitively, given a security domain s_i, R uses $\mathcal{M}$ to retrieve context $c \in C_{s_i}$. One can trivially generalize this definition of PermRAG to work with subsets of security domains instead of a singleton security domain s_i.

For PermRAG, we assume an identical enclosing system setting as in PermLLM (see § 2): Given a user u the enclosing system determines u's access credentials $cred_u$ and calls `authenticate(`$cred_u$`)` that takes user credentials $cred_u$ and maps them to a subset of security domains S_u that u can access. User u cannot arbitrarily change S_u. Each of user u's subsequent query q to f_R is annotated with S_u. The retriever $R_C^{\mathcal{M}}$ of f_R uses access control mechanism $\mathcal{M}$ to retrieve a context $c \in C_{S_u}$.

Definition A.1 (Relevant Response for PermRAG)**.** *Given a PermRAG f_R, with retriever $R_C^{\mathcal{M}}$, a query q from user u, and S_u the security domains u has access to, $r = f_R(q)$ is the response by f_R to query q. Response r is said to be relevant to S_u (i.e. $r = r_{S_u}$) if retriever $R_C^{\mathcal{M}}$ uses a context $c \in C_{S_u}$ to augment the query for r.*

To empirically quantify response relevance, we can use the same response relevance score, $relv(f_R(q), S_u)$ that quantifies the information gained on data in the security domains q's user u has access to (this is the same set of security domains that mapping $\mathcal{M}$ gives for u for the retriever $R_C^{\mathcal{M}}$, i.e. S_u). Here $R_C^{\mathcal{M}}$ retrieves the query context $c \in C$ using mapping $\mathcal{M}$; c is then augmented to the query q. We restrict the domain of $relv$ to the real number interval $[0, 1]$, where 1 is the best expected score for relevance. Similar to PermLLM, we define *access advantage* for PermRAG as follows:

Definition A.2 (Access Advantage for PermRAG)**.** *Given PermRAG f_R that uses retriever $R_C^{\mathcal{M}}$ which in turn uses the context database C containing data from domains $\mathbb{S} = \bigcup_{i=1}^{n}\{s_i\}$, with access control mechanism $\mathcal{M}$, a subset of security domains $S_u \subseteq \mathbb{S}$, context $c \in C$ that is augmented to query q, f_R achieves α-*access advantage *w.r.t.* S_u *if:*

$$\mathbb{E}_{q \sim \mathcal{D}_{S_u}, S_v \subseteq \mathbb{S}; S_u \cap S_v = \phi}\big[relv(f_R(q), S_u) \ominus relv(f_R(q), S_v)\big] \geq \alpha$$

where $relv()$ is the response relevance score on the corresponding security domain subset (S_u or S_v), $\ominus$ is a "difference" operator specific to the access control assessment method (e.g. subtraction), and α is an advantage threshold that lies in the range [0,1].

B Formal Access Control Enforcement in PermLLM Mechanisms

We now present formal proofs for correct access control enforcement in our PermLLM mechanisms presented in § 3: *Activate*, *Merge*, and *Union*.

Refreshing the formalism from § 2, we consider a universe of n different security domains $\mathbb{S} = \bigcup_{i=1}^{n}\{s_i\}$, and a training data set consisting of data from these domains $D = \bigcup_{i=1}^{n} D_{s_i} \sim \mathcal{D}_{s_i}$ (here D_{s_i} is a data set sampled from data distribution $\mathcal{D}_{s_i}$ of domain s_i). Let f_D be the LLM tuned using data set D. Let W be the set of f_D's parameters. Model tuning *changes* values of a subset of W. Let security domain s_i *affect*, per the meaning of affect in § 2, a subset of parameters $W_{s_i} \subseteq W$. Thus data from D_{s_i} is used to change parameters W_{s_i} during model fine-tuning. Let $\mathcal{M}$ be the access control mechanism that dictates the mapping of security domain s_i to parameters W_{s_i} via the affects relation.

Consider a set of LoRA adapters [17] $l_1, l_2, ..., l_m$. Each adapter l_i comprises parameters W_{l_i}, such that $W_{l_i} \cap W_{l_j} = \phi, \forall i \neq j$. Let i be the adapter Id for adapter l_i. Let $f_D^{\mathcal{M}}$ by the PermLLM that uses mapping $\mathcal{M}$ of security domains to parameters during tuning and testing. Let $\mathcal{F}^{\mathcal{M}}$ be the system enclosing $f_D^{\mathcal{M}}$ that performs the mapping from user credentials $cred_u$ to sets of security domains S_u for each user u. We make two assumptions about $\mathcal{F}^{\mathcal{M}}$: (i) $\mathcal{F}^{\mathcal{M}}$ can correctly determine and maintain the security domains S_u a user u has access to; and (ii) S_u remains opaque to the user and any other adversary and as a result, cannot be tampered with by any user or adversary.

We assume that both fine-tuning and testing are mediated through $\mathcal{F}^{\mathcal{M}}$. During fine-tuning, the dataset D is passed to $\mathcal{F}^{\mathcal{M}}$. $\mathcal{F}^{\mathcal{M}}$ extracts information about the security domains $s_1, .., s_n$ covered by D. For settings where users have access to multiple security domains, the list of security domain combinations that users have access to is also passed on to $\mathcal{F}^{\mathcal{M}}$. $\mathcal{F}^{\mathcal{M}}$ does the mapping between security domain groups and LoRA adapters differently for each of our PermLLM mechanisms:

Activate $\mathcal{F}^{\mathcal{M}}$ maps each security domain s_i to a unique LoRA adapter l_i. For fine-tuning of $f_D^{\mathcal{M}}$, minibatches sampled for each s_i are routed to the corresponding LoRA adapter l_i, the other LoRA adapters are deactivated for the sampled mini-batch.

Merge Security domain-LoRA adapter mappings and fine-tuning of $f_D^{\mathcal{M}}$ proceeds identically to that in *Activate*. However, after the fine-tuning is done, the security domain groups are used to merged LoRA adapters. These new LoRA adapters are added to the set of LoRA adapters in $f_D^{\mathcal{M}}$. The mapping $\mathcal{M}$ is also updated with the new mappings between security domain groups and LoRA adapters.

Union Datasets corresponding to the security domain groups are used to fine-tuning unique LoRA adapters. $\mathcal{M}$ is also updated with these new security domain group-LoRA adapter mappings.

At the end of fine-tuning, $\mathcal{M}$ will have a mapping between each security domain group S_u (for each respective user u) and each LoRA adapter in mechanisms *Merge* and *Union*. More formally,

Lemma B.1. *In* Merge *and* Union*, after fine-tuning, for every user u that has access to $S_u \subseteq \mathbb{S}$, $\exists l_{S_u}$, where l_{S_u} is a LoRA adapter, S_u* affects *parameters $W_{l_{S_u}}$, and $W_{l_{S_u}}$ is* not *affected by any other security domains in S.*

In case of *Activate*, S_u is used at inference time to activate the LoRA adapters l_{s_i}, where $s_i \in S_u$. More formally,

Lemma B.2. *In* Activate*, after fine-tuning, for every user u that has access to $S_u \subseteq \mathbb{S}$, $\forall s_i \in S_u$, s_i* affects *parameters $W_{l_{s_i}}$, and $W_{l_{s_i}}$ is* not *affected by any other security domain $s_j \in S_u, i \neq j$, or $s_k \in \mathbb{S} \setminus S_u$.*

At inference time, user u sends query q to $\mathcal{F}^{\mathcal{M}}$. $\mathcal{F}^{\mathcal{M}}$ first determines u's security domains S_u, and then passes q and S_u to $f_D^{\mathcal{M}}$, which then activates the LoRA adapter/s corresponding to S_u: l_{S_u} in case of *Merge* and *Union*, and l_{s_i}, where $s_i \in S_u$, in case of *Activate*. Our assumptions about accessibility of S_u to the user or adversary ensure that the adversary cannot tamper with S_u within the scope of $\mathcal{F}^{\mathcal{M}}$.

Theorem B.3. *Given any query q from any user u, the response $r = f_D^{\mathcal{M}}(q)$ is relevant to S_u for $\mathcal{M}$ in* Activate*,* Merge*, or* Union*.*

Proof. From Lemmas Theorem B.1 and Theorem B.2, through the fine-tuning process S_u affects parameters $W_{l_{S_u}}$ in *Merge* and *Union*, and parameters $W_{l_{s_i}}, \forall s_i \in S_u$ in *Activate*. At inference time, these same parameters (along with the pretrained model's parameters) are used to generate response $r = f_D^{\mathcal{M}}(q)$. By implication, the parameters affected by S_u are used to generated r. Hence r is relevant to S_u, i.e. $r = r_{S_u}$. □

Since the above response relevance condition applies for all responses $r = f_D^{\mathcal{M}}(q)$ on all queries q by all users u, we say that *Activate*, *Merge*, and *Union* correctly enforce parameter separation and hence correctly enforce access control for all users u.

C Audit Games

We formalize black-box games that capture: (i) the distinguishability of security domain-specific responses for DDI, and (ii) the utility disparity induced by access restrictions for UGI. Intuitively, in these auditing games, we measure how *effectively* an external auditor can conclude if the access control mechanism is correctly implemented by verifying if the correct domain adapter is activated for a query. This effectiveness is directly correlated with the access advantage score for the target security domain(s). Higher access advantage score denotes *stronger* access control enforcement. A perfectly separated system provides the auditor with an access advantage score of 1.0.

We consider the same threat setting and auditor privileges for our adversarial games between auditor $\mathcal{A}$ and system $\mathcal{S}$ enclosing the PermLLM $f_D^{\mathcal{M}}$ as described in § 2.3.

Game 1: Domain Distinguishability. This game assesses whether the auditor can effectively conclude if the correct security domains were used based on the generated responses. The primary motivation of this game is to measure the distinguishability of different security domains' distributions.

1. Auditor $\mathcal{A}$ chooses security domain set S_u and emulates user u. $\mathcal{A}$ sends user credentials $cred_u$ and query $q \sim \mathcal{D}_{S_u}$ to system $\mathcal{S}$. $\mathcal{S}$ verifies the user credential $cred_u$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
2. $\mathcal{A}$ chooses security domain set S_v such that $S_v \cap S_u = \phi$ and emulates user v. $\mathcal{A}$ sends user credentials $cred_v$ and the same query $q \sim \mathcal{D}_{S_u}$ to $\mathcal{S}$. $\mathcal{S}$ verifies the user credential $cred_v$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
3. $\mathcal{A}$ sends the models responses and domain information to membership inference oracle O to obtain domain distinguishability score $m(O(f_D^{\mathcal{M}}(q)|S_u, f_D^{\mathcal{M}}(q)|S_v))$, where $m(\cdot)$ is a membership inference metric (e.g., AUC-ROC or TPR@1%FPR) in the [0,1] range.
4. $\mathcal{A}$ concludes the access control mechanism is correctly implemented if the domain distinguishability score $m(O(f_D^{\mathcal{M}}(q)|S_u, f_D^{\mathcal{M}}(q)|S_v)) \geq \alpha$.

Table 6: Data Set Details. Generalization Loss Gap (i.e., gap between model's loss on training and test sets) for all models are reported after fine-tuning for 5 epochs on each data set.

Data Set	Data Set Size		Llama-3.1-8B Loss Gap			Mistral-0.1-7B Loss Gap		
(# Domains)	Train	Test	Full FT	LoRA	PermLLM	Full FT	LoRA	PermLLM
WMDP (3)	2936	732	1.96	0.52	1.15	1.36	0.65	1.07
GPQA (3)	360	88	2.51	1.06	1.04	1.58	0.61	1.09
SimpleQA (10)	4089	1018	2.91	0.96	1.49	1.87	0.90	1.25
RCV1 (4)	45622	22811	4.07	0.35	0.83	2.48	0.37	0.74

Note that we can change the above game to distinguish members ($q \sim \mathcal{D}_{S_u}$) and non-members ($q \sim \mathcal{D}_{S_v}$) for the target domain set S_u, similar to prior MIA setups, which is what we do in our experiments in § 5.

Game 2: Utility Gap Evaluation. The second game evaluates how distinctly the responses from two different security domains impact the utility perceived by users. The rationale behind this game is to confirm that enforced access controls result in meaningful variations in response quality.

1. Auditor $\mathcal{A}$ chooses security domain set S_u and emulates user u. $\mathcal{A}$ sends user credentials $cred_u$ and query $q \sim \mathcal{D}_{S_u}$ to system $\mathcal{S}$. $\mathcal{S}$ verifies the user credential $cred_u$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
2. $\mathcal{A}$ chooses security domain set S_v such that $S_v \cap S_u = \phi$ and emulates user v. $\mathcal{A}$ sends user credentials $cred_v$ and the same query $q \sim \mathcal{D}_{S_u}$ to $\mathcal{S}$. $\mathcal{S}$ verifies the user credential $cred_v$ and sends back the model response $f_D^{\mathcal{M}}(q)$ to $\mathcal{A}$.
3. Given a utility function $U(\cdot)$ (e.g., BLEURT or task accuracy) that outputs values in [0,1] range, $\mathcal{A}$ concludes the access control mechanism is correctly implemented if the utility gap score $|U(f_D^{\mathcal{M}}(q)|S_u) - U(f_D^{\mathcal{M}}(q)|S_v)| \geq \alpha$.

We aggregate the utility gaps from this game across all domain set pairs to obtain our UGI metric.

D Detailed Experiment Setup

D.1 Models

For our instantiation of PermLLM, we fine-tune Llama-3.1-8B[15] and Mistral-0.1-7B[19] pretrained models on four datasets covering multiple distinct security domains (henceforth called *domains*), where we fine-tune a separate LoRA adapter for each domain. To compare our PermLLM, we train two additional models with full fine-tuning and LoRA fine-tuning respectively on entire training data. Note that these models are only used for utility baselines as they do not provide access control. For all the LoRA adapters, we use 64 rank and 0.1 dropout. We use AdamW optimizer with 0.1 weight decay to fine-tuned all the models for 5 epochs with 300 warmup steps, 2 batch size and 5×10^{-4} learning rate (except for Mistral-0.1-7B full fine-tuning that uses a learning rate of 5×10^{-5}). We performed grid search over multiple learning rates and warmup steps and found these values to give the best results. For all our experiments, we use 8 H100 GPUs (with 80GB VRAM per GPU), 4 workers per GPU, and 384 GB RAM. One epoch of fine-tuning took from few minutes (for our smallest data set: GPQA) to a couple of hours (for our largest data set: RCV1). Mistral-0.1-7B is released under Apache 2.0 license, and Llama-3.1-8B is released under Llama 3.1 Community License.

D.2 Data Sets

For our experiments, we require data sets that consist of multiple distinct domains and are possibly not seen by the pretrained models. We use four different data sets, namely, WMDP [23], GPQA [32], SimpleQA [40], and RCV1 [21]. While the first three data sets were collected after the pretraining cutoff dates for Llama-3.1-8B and Mistral-0.1-7B, RCV1 is an older data set and hence we do not know if it was used in pretraining. However, we observe a high initial training loss on this data set, thereby indicating that it was either not used in pretraining or was catastrophically forgotten by the models, allowing for a gradual reduction in training loss during our fine-tuning (see Figure 7). Table 6 shows the data set details, along with the generalization gap (test loss - train loss) for different

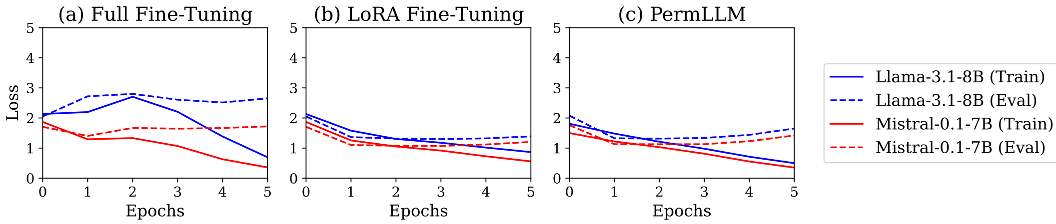

Figure 4: Comparing model loss on WMDP data set.

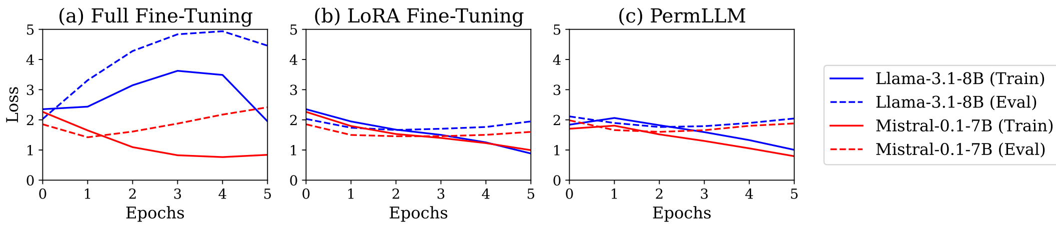

Figure 5: Comparing model loss on GPQA data set.

approaches of fine-tuning the models on these data sets. See Figure 4, Figure 5, Figure 6, and Figure 7 for complete training and test loss trajectories across different data sets.

WMDP. Weapons of Mass Destruction Proxy (WMDP) [23] is a data set consisting of multi-choice question–answer pairs spanning three domains: biological weapons (*bio*), chemical weapons (*chem*) and cyber-warfare weapons (*cyber*). We do 4:1 split of the data set to obtain training and test sets. The training set consists of 2936 question–answer pairs where 1019 are from *bio*, 327 are from *chem* and the remaining 1590 are from *cyber*. The test set size is 732 records, consisting of 254 *bio*, 81 *chem* and 397 *cyber* records. The largest record from this data set consists of 1934 tokens (tokenized using Llama3 tokenizer). This data set is released under MIT License.

GPQA. Graduate-Level Google-Proof Q&A Benchmark (GPQA) [32] data set consists of general question–answer pairs from three domains: *biology*, *chemistry* and *physics*. We do 4:1 split of the data set to obtain training and test sets. The training set consists of 360 question–answer pairs where 63 are from *biology*, 147 are from *chemistry* and the remaining 150 are from *physics*. The test set size is 88 records, consisting of 15 *biology*, 36 *chemistry* and 37 *physics* records. The largest record from this data set consists of 911 tokens (tokenized using Llama3 tokenizer). This data set is released under MIT License.

SimpleQA. SimpleQA [40] is a factuality benchmark that measures the ability for language models to answer short, fact-seeking questions. It consists of general question–answer pairs from ten domains: *art*, *geography*, *history*, *music*, *other*, *politics*, *science and technology*, *sports*, *tv shows*, and *video games*. We do 4:1 split of the data set to obtain training and test sets. The training set consists of 4089 question–answer pairs divided across all ten domains. The test set size is 1018 records spanning across all ten domains. The largest record from this data set consists of 156 tokens (tokenized using Llama3 tokenizer). This data set is released under MIT License.

RCV1. RCV1 [21] is a benchmark dataset on text categorization. It is a collection of newswire articles produced by Reuters between 1996 and 1997. It contains 804,414 manually labeled newswire documents, broadly categorized with respect to three categories: *industries*, *topics* and *regions*. We took a subset of this data set and created four non-overlapping domains using *topics*: commercial (*CCAT*), economic (*ECAT*), governance (*GCAT*), and mechanical (*MCAT*). We then did 2:1 split of the subset to obtain training and test sets. The training set consists of 45622 question–answer pairs where 23822 are from *CCAT*, 7460 are from *GCAT*, 3370 are from *ECAT* and the remaining 10970 are from *MCAT*. The test set size is 22811 records, consisting of 11911 *CCAT*, 3730 *GCAT*, 1685 *ECAT*, and 5485 *MCAT* records. The largest record from this data set consists of 1199 tokens (tokenized using Llama3 tokenizer). This data set is released under CC BY 4.0 License.

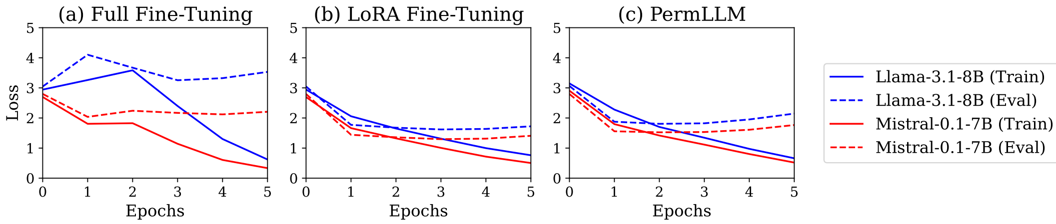

Figure 6: Comparing model loss on SimpleQA data set.

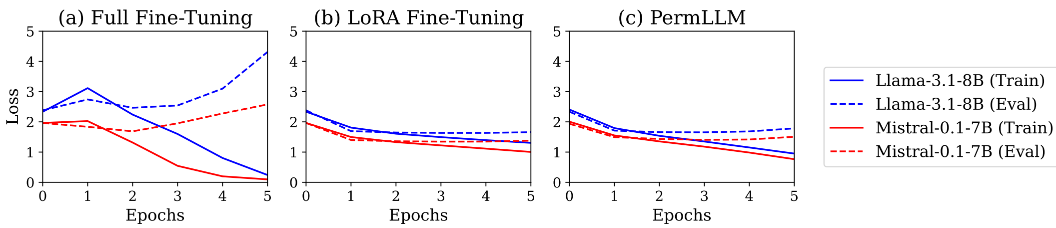

Figure 7: Comparing model loss on RCV1 data set.

D.3 Model Utility Evaluation

We use four metrics to evaluate the utility of the model generations: Bleurt Score ($bluert$), Bert F1-Score ($bert$), Sacrebleu Score ($bleu$) and Verbatim Accuracy (acc). These metrics measure how similar the generated text is to the ground truth. $bleurt$ and $bert$ measure the semantic similarity, $bleu$ measures the fraction of common n-grams, and acc gives a binary decision of whether the generated text verbatim matches the ground truth. All the metrics lie in a [0,1] range, where values close to 1 indicate high model utility.

We check the utility of *Activate* to determine if tuning different LoRA adapters for each security domain leads to acceptable model utility. To that end, we show in Table 7 the utility of Llama-3.1-8B models fine-tuned on different data sets with the three approaches: full fine-tuning, LoRA fine-tuning and our PermLLM. We do not report the $bleu$ score for WMDP as it is a multi-choice question-answering task where model only has to generate a single token. $bleu$ requires generating at least four tokens. Our approach achieves similar or better utility on the training set compared to the LoRA approach. On the test set, our approach achieves similar utility to LoRA for most of the data sets, except for SimpleQA where LoRA performs better. This is because SimpleQA has more domains (10 in total), thus each of our individual domain adapter sees only a fraction of data of what LoRA approach's adapter sees (given that SimpleQA is already a small data set). We expect the performance of our domain-specific adapters to increase as the data set size increases. Full fine-tuning is highly sensitive to training hyper-parameters, and as a result it either completely overfits on training set to achieve high utility (e.g., on SimpleQA and RCV1), or it underfits and achieves low utility (e.g., on WMDP and GPQA). We observe similar results for Mistral-0.1-7B models (see Table 8).

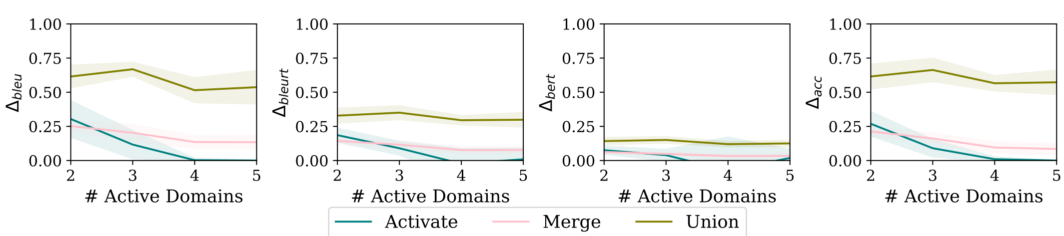

Figure 8: Utility Gap Index, Δ_U ($mean \pm std$) for Mistral-0.1-7B models fine-tuned on SimpleQA when user has access to multiple security domains.

Table 7: Utility comparison of Llama-3.1-8B models trained with different approaches. All reported values are $mean \pm std$ across domains.

	Metric	Full Fine-Tuning		LoRA Fine-Tuning		PermLLM	
		Train	Test	Train	Test	Train	Test
WMDP	$bleurt$	0.74 ± 0.06	0.74 ± 0.06	0.90 ± 0.08	0.85 ± 0.08	0.92 ± 0.08	0.82 ± 0.06
	$bert$	0.89 ± 0.03	0.89 ± 0.03	0.96 ± 0.03	0.94 ± 0.03	0.97 ± 0.03	0.93 ± 0.03
	acc	0.26 ± 0.07	0.27 ± 0.07	0.76 ± 0.20	0.60 ± 0.20	0.84 ± 0.22	0.49 ± 0.15
GPQA	$bleu$	0.26 ± 0.02	0.05 ± 0.03	0.45 ± 0.12	0.10 ± 0.05	0.39 ± 0.20	0.10 ± 0.04
	$bleurt$	0.53 ± 0.05	0.39 ± 0.05	0.64 ± 0.09	0.46 ± 0.07	0.62 ± 0.11	0.47 ± 0.07
	$bert$	0.67 ± 0.06	0.59 ± 0.05	0.77 ± 0.08	0.67 ± 0.05	0.75 ± 0.09	0.67 ± 0.05
	acc	0.24 ± 0.06	0.02 ± 0.03	0.32 ± 0.05	0.05 ± 0.05	0.31 ± 0.09	0.04 ± 0.05
SimpleQA	$bleu$	0.80 ± 0.06	0.34 ± 0.11	0.65 ± 0.06	0.29 ± 0.08	0.67 ± 0.10	0.09 ± 0.04
	$bleurt$	0.86 ± 0.03	0.58 ± 0.05	0.80 ± 0.02	0.61 ± 0.02	0.82 ± 0.04	0.53 ± 0.04
	$bert$	0.96 ± 0.01	0.84 ± 0.02	0.94 ± 0.01	0.86 ± 0.01	0.95 ± 0.02	0.82 ± 0.03
	acc	0.68 ± 0.10	0.20 ± 0.12	0.52 ± 0.07	0.17 ± 0.07	0.55 ± 0.13	0.02 ± 0.02
RCV1	$bleu$	0.75 ± 0.08	0.14 ± 0.08	0.22 ± 0.10	0.16 ± 0.08	0.27 ± 0.10	0.16 ± 0.08
	$bleurt$	0.88 ± 0.04	0.46 ± 0.12	0.57 ± 0.13	0.49 ± 0.11	0.62 ± 0.13	0.50 ± 0.12
	$bert$	0.94 ± 0.03	0.67 ± 0.09	0.75 ± 0.08	0.70 ± 0.07	0.78 ± 0.08	0.70 ± 0.08
	acc	0.78 ± 0.06	0.16 ± 0.10	0.27 ± 0.14	0.17 ± 0.10	0.31 ± 0.15	0.18 ± 0.10

Table 8: Utility comparison of Mistral-0.1-7B models trained with different approaches. All reported values are $mean \pm std$ across domains.

	Metric	Full Fine-Tuning		LoRA Fine-Tuning		PermLLM	
		Train	Test	Train	Test	Train	Test
WMDP	Bleurt	0.95 ± 0.01	0.82 ± 0.03	0.96 ± 0.02	0.87 ± 0.03	0.96 ± 0.01	0.86 ± 0.03
	Bert	0.98 ± 0.01	0.92 ± 0.02	0.99 ± 0.01	0.94 ± 0.02	0.99 ± 0.01	0.94 ± 0.02
	Acc	0.88 ± 0.04	0.46 ± 0.14	0.92 ± 0.07	0.60 ± 0.09	0.93 ± 0.04	0.58 ± 0.11
GPQA	Bleu	0.46 ± 0.03	0.06 ± 0.05	0.35 ± 0.08	0.11 ± 0.07	0.55 ± 0.18	0.13 ± 0.06
	Bleurt	0.65 ± 0.04	0.42 ± 0.08	0.59 ± 0.09	0.47 ± 0.06	0.67 ± 0.09	0.47 ± 0.08
	Bert	0.75 ± 0.05	0.62 ± 0.07	0.73 ± 0.08	0.68 ± 0.05	0.79 ± 0.08	0.66 ± 0.09
	Acc	0.38 ± 0.04	0.04 ± 0.05	0.24 ± 0.04	0.05 ± 0.06	0.40 ± 0.09	0.08 ± 0.02
SimpleQA	Bleu	0.94 ± 0.02	0.36 ± 0.11	0.73 ± 0.06	0.34 ± 0.09	0.70 ± 0.13	0.10 ± 0.04
	Bleurt	0.94 ± 0.01	0.60 ± 0.04	0.84 ± 0.03	0.62 ± 0.03	0.83 ± 0.06	0.52 ± 0.04
	Bert	0.99 ± 0.01	0.85 ± 0.02	0.96 ± 0.01	0.87 ± 0.01	0.95 ± 0.03	0.82 ± 0.03
	Acc	0.91 ± 0.04	0.23 ± 0.12	0.62 ± 0.08	0.20 ± 0.10	0.60 ± 0.16	0.03 ± 0.02
RCV1	Bleu	0.92 ± 0.06	0.17 ± 0.09	0.28 ± 0.13	0.20 ± 0.10	0.37 ± 0.14	0.19 ± 0.09
	Bleurt	0.93 ± 0.02	0.48 ± 0.12	0.60 ± 0.13	0.51 ± 0.12	0.66 ± 0.12	0.50 ± 0.12
	Bert	0.98 ± 0.02	0.69 ± 0.08	0.78 ± 0.09	0.71 ± 0.08	0.81 ± 0.08	0.71 ± 0.08
	Acc	0.92 ± 0.03	0.19 ± 0.11	0.31 ± 0.15	0.20 ± 0.11	0.38 ± 0.17	0.19 ± 0.10

E MIAs against LLMs

In Section 4, we defined the Domain Distinguishability Index (DDI) as the average success rate of an adversary playing the Domain Distinguishability game over all domain set pairs. That game is implemented with *membership inference attacks* (MIAs) [43, 5, 28, 35, 46]: the auditor compares a *member* set drawn from the active domain's training data with a *non-member* set drawn from some other domain, and tries to tell them apart. The better this separation, the larger the DDI. Here, in this section, we expand on the MIA toolbox that underpins DDI—detailing evaluation metrics and the specific attacks we deploy against LLMs. More generally, an MIA for an LLM f assigns a *membership score* $A(x, f)$ to a candidate text x. Thresholding this score at ε declares x a member (if $A(x, f) \geq \varepsilon$) or a non-member (if $A(x, f) < \varepsilon$).

E.1 Metrics

We employ two complementary metrics to quantify the success of our membership inference attacks, as used by prior MIA works [18, 6, 27]:

(1) Attack ROC curves: The Receiver Operating Characteristic (ROC) curve illustrates the trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR) for the attacks. The FPR measures the proportion of non-member samples that are incorrectly classified as members, while the TPR represents the proportion of member samples that are correctly identified as members. We report the Area Under the ROC Curve (AUC-ROC) as an aggregate metric to assess the overall success of the attacks. AUC-ROC is a threshold-independent metric, and it shows the probability that a positive instance (member) has higher score than a negative instance (non-member).

(2) Attack TPR at low FPR: This metric is crucial for determining the effectiveness of an attack at confidently identifying members of the training dataset without falsely classifying non-members as members. We focus on low FPR thresholds, specifically 1%, and 5%. For instance, the TPR at an FPR of 1% is calculated by setting the detection threshold so that only 1% of non-member samples are predicted as members.

E.2 Existing MIAs

LOSS [43]: The LOSS method utilizes the loss value of model $f(.)$ for the given text x as the membership score; a lower loss suggests that the text was seen during training, so $A(x, f) = \ell(f, x)$.

Ref [5]: Calculating membership scores based solely on loss values often results in high false negative rates. To improve this, a difficulty calibration method can be employed to account for the intrinsic complexity of x. For example, repetitive or common phrases typically yield low loss values. One method of calibrating this input complexity is by using another LLM, $Ref(.)$, assumed to be trained on a similar data distribution. The membership score is then defined as the difference in loss values between the target and reference models, $A(x, f) = \ell(x, f) - \ell(x, Ref)$. In our evaluations, we used the base models (i.e., Llama-3.1-8B and Mistral-0.1-7B) before any fine-tuning as the reference models.

Zlib [5]: Another method to calibrate the difficulty of a sample is by using its zlib compression size, where more complex sentences have higher compression sizes. The membership score is then calculated by normalizing the loss value by the zlib compression size, $A(x, f) = \frac{\ell(x,f)}{zlib(x)}$.

Min-K [35]: This attack hypothesizes that non-member samples often have more tokens assigned lower likelihoods. It first calculates the likelihood of each token as $\text{Min-K\%}_{\text{token}}(x_t) = \log p(x_t|x_{<t})$, for each token x_t given the prefix $x_{<t}$. The membership score is then calculated by averaging over the lowest $K\%$ of tokens with lower likelihood, $A(x, f) = \frac{1}{|\text{min-k\%}|} \sum_{x_i \in min-k\%} \text{Min-K\%}_{\text{token}}(x_t)$.

Min-K++ [46]: This method improves on Min-K by utilizing the insight that maximum likelihood training optimizes the Hessian trace of likelihood over the training data. It calculates a normalized score for each token x_t given the prefix $x_{<t}$ as $\text{Min-K\%++}_{\text{token}}(x_t) = \frac{\log p(x_t|x_{<t}) - \mu_{x_{<t}}}{\sigma_{x_{<t}}}$, where $\mu_{x_{<t}}$ is the mean log probability of the next token across the vocabulary, and $\sigma_{x_{<t}}$ is the standard deviation. The membership score is then aggregated by averaging the scores of the lowest $K\%$ tokens, $A(x, f) = \frac{1}{|\text{min-k\%++}|} \sum_{x_i \in min-k\%} \text{Min-K\%++}_{\text{token}}(x_t)$.