arXiv:2506.18870v1 [cs.CR] 23 Jun 2025

# Amplifying Machine Learning Attacks Through Strategic Compositions

Yugeng Liu[1] Zheng Li[2] Hai Huang[1] Michael Backes[1] Yang Zhang[1]

[1]*CISPA Helmholtz Center for Information Security* [2]*Shandong University*

## Abstract

Machine learning (ML) models are proving to be vulnerable to a variety of attacks that allow the adversary to learn sensitive information, cause mispredictions, and more. While these attacks have been extensively studied, current research predominantly focuses on analyzing each attack type individually. In practice, however, adversaries may employ multiple attack strategies simultaneously rather than relying on a single approach. This prompts a crucial yet underexplored question: When the adversary has multiple attacks at their disposal, are they able to mount or amplify the effect of one attack with another? In this paper, we take the first step in studying the strategic interactions among different attacks, which we define as *attack compositions*. Specifically, we focus on four well-studied attacks during the model's inference phase: adversarial examples, attribute inference, membership inference, and property inference. To facilitate the study of their interactions, we propose a taxonomy based on three stages of the attack pipeline: preparation, execution, and evaluation. Using this taxonomy, we identify four effective attack compositions, such as property inference assisting attribute inference at its preparation level and adversarial examples assisting property inference at its execution level. We conduct extensive experiments on the attack compositions using three ML model architectures and three benchmark image datasets. Empirical results demonstrate the effectiveness of these four attack compositions. We implement and release a modular reusable toolkit, COAT. Arguably, our work serves as a call for researchers and practitioners to consider advanced adversarial settings involving multiple attack strategies, aiming to strengthen the security and robustness of AI systems.

## 1 Introduction

Recently, machine learning has gained momentum in multiple fields, achieving success in real-world deployments, such as image classification [6, 17, 65], face recognition [30, 66], and medical image analysis [8, 32, 59]. Nevertheless, prior research has shed light on the vulnerability of ML models to various attacks, such as adversarial examples [4, 27, 53], membership inference [38, 47, 55, 56], and backdoor attacks [15, 22, 40]. These vulnerabilities prompt significant security and privacy risks. As a result, investigating, quantifying, and mitigating these various attacks on ML models have become increasingly important topics.

Currently, most research in this field focuses on developing or optimizing more powerful attacks, e.g., higher attack success rates or greater stealthiness, and proposing corresponding countermeasures. More precisely, these studies typically focus on individual attacks. While some measurement or benchmark papers exist that consider multiple attacks, e.g., ML-Doctor [42] or SecurityNet [63], they still implement each attack individually. In other words, studying attacks in isolation is actually the most common practice in the existing ML security domain.

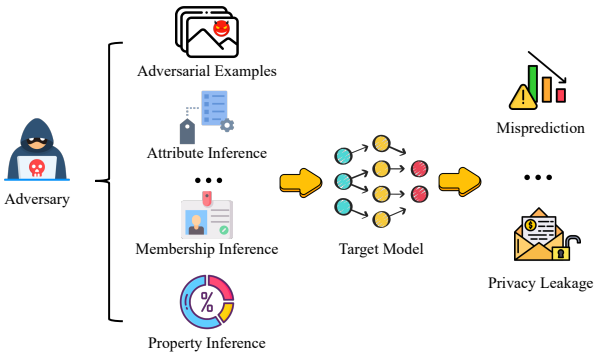


**Figure 1: Given a target model, the adversary can launch different attacks to achieve different malicious goals.**

However, this practice may not accurately reflect real-world scenarios, where adversaries often possess multiple attack strategies and can potentially synergize or leverage them simultaneously. When focusing solely on individual attacks, researchers may overlook the potential for adversaries to amplify the impact of one attack by leveraging knowledge or capabilities gained from another attack. Consequently, the true extent of vulnerabilities and risks posed by combined attacks may be underestimated or remain unexplored. This reality prompts the need for a more comprehensive understanding of the *intentional interactions* among different attacks.

### 1.1 Contributions

In this work, we take the first step in exploring the (possible) intentional interactions between different types of attacks. We focus exclusively on the inference phase of ML models since deployed models are more likely to face intentional interactions between different attacks. Specifically, we consider the four most representative attacks launched during the inference phase of the ML model, aka *inference time attacks*: adversarial examples [4, 27, 53], attribute in-

ference [46, 58], membership inference [38, 47, 55, 56], and property inference [46].

We formulate the following research questions (RQs), aiming at addressing this significant gap.

- **RQ1:** How can we approach the design and implementation of attack compositions?
- **RQ2:** How can the knowledge gained from one type of attack facilitate or amplify the effectiveness of another attack?
- **RQ3:** How effective are combined attacks in exploiting ML model vulnerabilities compared to individual ones?

**Composition Taxonomy.** First, we propose a taxonomy for attack compositions based on the attack pipeline (**RQ1**), divided into three levels: preparation, execution and evaluation. The former encompasses all preliminary activities before the main attack, including tool setup, data collection, and configuration. The execution level covers the attack's actual implementation, involving malicious queries, responses, and vulnerability exploitation. Finally, the evaluation level assesses the attack impact, including system disruption, goal achievement, and any post-exploitation activities.

**Composition Methodology.** Based on the taxonomy, we conduct an extensive exploration of attack compositions across four representative inference-time attacks (**RQ2**). Specifically, we identify four effective attack compositions: one at the preparation level, two at the execution level, and one at the assessment level. At the preparation level, we propose using property inference to assist attribute inference (PropInf2AttrInf). By determining the attribute distribution in the victim model's training dataset through property inference, we use it to create a balanced attack training dataset for attribute inference. At the execution level, we propose two attack compositions: using adversarial examples to assist membership inference (ADV2MemInf) and property inference (ADV2PropInf), respectively. Adversarial examples can search for different noise magnitudes for various membership or property statuses, which are then integrated into their original information for improved attack performance. At the evaluation level, we leverage property inference to assist membership inference (PropInf2MemInf). After the membership inference process ends, we use the property distribution determined by property inference to calibrate its attack output.

**Composition Evaluation.** We conduct extensive experiments across three popular ML model architectures and three benchmark image datasets (**RQ3**). Here, we summarize our analysis using ResNet18 [23] trained on CIFAR10 [1] as an example. First, property inference significantly amplifies attribute inference at its preparation level. For instance, AttrInf achieves an accuracy of 0.500, while PropInf 2AttrInf achieves an empirical accuracy of 0.894 and a theoretical accuracy of 0.872. Second, adversarial examples improve both membership inference and property inference. For instance, the black-box MemInf with shadow model and PropInf achieve an accuracy of 0.664 and 0.890, respectively, while the attack compositions yield significantly improved results, with accuracies of 0.851 and 0.960, respectively. Finally, the black-box MemInf with partial training dataset achieves an accuracy of 0.631, compared to PropInf 2MemInf's accuracy of 0.669.

**CoAT.** To evaluate our proposed diverse attack compositions, we develop a modular framework, CoAT (Composition of ATtacks). With its modular design, CoAT allows for easy integration of new versions of each attack type, additional datasets, and models. Our code will be released publicly along with the final version of the paper (and is already available upon request), thus facilitating further research in the field.

**Note.** We deliberately exclude model-stealing attacks from consideration, as the process of stealing essentially transforms a black-box model into a white-box model, which falls outside our formal definition of attack composition. Additionally, we omit consideration of training-time attacks, such as backdoor attacks, since these scenarios presuppose adversarial involvement in the training process of victim models – a condition that violates our fundamental assumptions about attacker capabilities, definition of compositions, and access limitations.

## 2 Inference-Time Attacks

In this section, we present the four most representative attacks during the ML models' inference phase, namely, adversarial examples (Section 2.1), attribute inference (Section 2.2), membership inference (Section 2.3), and property inference (Section 2.4). Specifically, the first three are designed at the sample level, while the last one aims to infer the general information at the dataset level. Different attacks can be applied to different threat models; see Table 1. For each attack and each threat model, we focus on one representative state-of-the-art method.

**Table 1: Different attacks under different threat models.**

| Auxiliary Dataset | Model Access | |
|---|---|---|
| | Black-Box ($\mathcal{M}^{\mathsf{B}}$) | White-Box ($\mathcal{M}^{\mathsf{W}}$) |
| Partial ($\mathcal{D}^{\mathsf{P}}_{\mathsf{aux}}$) | MemInf | MemInf, AttrInf |
| Shadow ($\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$) | MemInf, PropInf | MemInf, AttrInf |
| Query ($\mathcal{D}^{\mathsf{Q}}_{\mathsf{aux}}$) | PropInf | - |

### 2.1 Adversarial Examples

Adversarial examples (ADV) [4,7,10,21,21,27,44,52,53,60] are a type of ML security threat where malicious inputs are deliberately designed to deceive ML models. These inputs, known as adversarial examples, are typically crafted by making small, often imperceptible modifications to target data to cause the model to predict incorrectly. More formally, given a target data sample $x_{\mathsf{target}}$, (the access to) a target model $\mathcal{M}$, an adversarial example $x_{\mathsf{adv}}$ can be generated by applying a perturbation $\delta$ such that $x_{\mathsf{adv}} = x_{\mathsf{target}} + \delta$. To ensure it remains subtle, the perturbation is usually constrained by a norm $\|\delta\|_p \leq \varepsilon$. The goal is to maximize the loss function $\ell\left(\mathcal{M}_\theta(x_{\mathsf{adv}}), y\right)$ In general, an adversarial attack can be

defined as:

$$\mathsf{ADV} : x_{\mathsf{target}}, \mathcal{M} \rightarrow \{x_{\mathsf{adv}}\} \quad (1)$$

In general, this type of attack can be categorized into two types based on the knowledge of the adversary: black-box and white-box attacks ($\mathcal{M} \in \{\mathcal{M}^{\mathsf{B}}, \mathcal{M}^{\mathsf{W}}\}$).

**Black-Box** $\langle \mathsf{ADV}, \mathcal{M}^{\mathsf{B}}, x_{\mathsf{target}} \rangle$ **[5].** Black-box attacks operate under the assumption that the adversary has no internal knowledge of the models. Instead, the adversary can only observe the outputs from the model. This scenario is more common in the real world, where internal details are inaccessible. They usually leverage trial-and-error to approximate the gradient of the target model [13] or randomized search schemes to approximate the boundary of the data samples [5].

**White-Box** $\langle \mathsf{ADV}, \mathcal{M}^{\mathsf{W}}, x_{\mathsf{target}} \rangle$ **[44].** White-box attacks assume the adversary has complete knowledge of the model, including its architecture, parameters, and training data. It allows the adversary to precisely calculate the most effective perturbations to maximize errors of ML models, often employing gradient-based methods to manipulate the input data directly, such as C&W [10], FGSM [21], JSMA [52], and PGD [44].

## 2.2 Attribute Inference

During the training phase, an ML model might unintentionally learn information that is not directly relevant to its intended tasks. If these models are open-source and published on the internet, the adversary may use them to predict some sensitive information. For example, a model designed to predict some features such as eye color from profile pictures could inadvertently also develop the ability to leak ethnicities [42, 46, 58]. This phenomenon of accessing unintended information is referred to as attribute inference (AttrInf). State-of-the-art attacks often utilize embeddings from a specific sample ($x_{\mathsf{target}}$) extracted from the model in question to ascertain the attributes of that sample. Thus, we assume the adversary to have white-box access to the target model. Followed by previous work [42], attribute inference is formally described as follows:

$$\mathsf{AttrInf} : x_{\mathsf{target}}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rightarrow \textit{target attributes} \quad (2)$$

Here, $\mathcal{D}_{\mathsf{aux}}$ represents an auxiliary dataset that contains a secondary attribute. It is assumed that the adversary has the ability to build the target attributes in the auxiliary dataset and employs the embeddings of these attributes from the auxiliary dataset to train a classifier, aiming to predict the attributes of the actual dataset.

## 2.3 Membership Inference

Membership inference attacks (MemInf) [56] involve adversaries seeking to ascertain if a specific data point was used in the training of a machine learning model. Specifically, given a data sample $x_{\mathsf{target}}$, a target model $\mathcal{M}$, and an auxiliary dataset $\mathcal{D}_{\mathsf{aux}}$, the process of membership inference is formulated as:

$$\mathsf{MemInf} : x_{\mathsf{target}}, \mathcal{M}, \mathcal{D}_{\mathsf{aux}} \rightarrow \textit{member}, \textit{non-member} \quad (3)$$

wherein $\mathcal{M} \in \mathcal{M}^{\mathsf{B}}, \mathcal{M}^{\mathsf{W}}$ and $\mathcal{D}_{\mathsf{aux}} \in \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$.

Extensive papers have been conducted on membership inference [12, 14, 29, 33, 38, 42, 47, 54–56], emphasizing its potential to compromise privacy. For instance, if a model for predicting medication dosages uses data from patients with a specific ailment, the model's training inclusion reveals sensitive health information. Predominantly, such inference attacks indicate that a model may reveal extra information, facilitating further exploits [11].

Below is an explanation of how to implement membership inference (MemInf) under varying threat scenarios.

**Black-Box/Shadow** $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ **[55].** The most prevalent and challenging scenario involves the adversary having only black-box access ($\mathcal{M}^{\mathsf{B}}$) to the model along with a shadow auxiliary dataset ($\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$). The adversary divides the shadow dataset and trains a model similar to the target model (mostly with the same architecture) on the shadow training dataset. For each sample, the output from this shadow model indicates membership, which the adversary uses to label the data. After finishing the shadow training process, the adversary uses the shadow testing dataset to query the shadow model. The adversary labels these samples as non-member data. These labeled data then help train a meta-classifier that determines membership in the target model by analyzing the output from the target model.

**Black-Box/Partial** $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ **[55].** When the adversary has black-box access and only partial data from the training dataset, they can strategically leverage these data samples to query the target model directly, eliminating the necessity of training a shadow model. The outputs from the target model with partial training data can be effectively categorized as the ground truth of members. More concretely, the adversary obtains non-member data by querying the target model using samples from a testing dataset. Upon acquiring these labeled datasets, the adversary can proceed to develop a meta-classifier.

**White-Box/Shadow** $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ **[48].** Nasr et al. [48] introduce an attack in the white-box setting. Although their initial configuration utilized partial training datasets, we follow previous work [42] by extending this setting to incorporate shadow models. Similar to the black-box model scenario, this setting necessitates the training of a shadow model to obtain ground truth for member and non-member samples. However, a crucial difference from the black-box setting lies in the assumption that the adversary has full access to the target models. This enhanced access enables the adversary to strengthen membership inference through the utilization of supplementary information. In this paper, we follow the methodology established by Nasr et al. [48], conducting membership inference attacks by leveraging multiple features: sample gradients concerning the model parameters, embeddings from different intermediate layers, classification loss, and prediction posteriors (and labels).

**White-Box/Partial** $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ **[48].** The method in this scenario mirrors $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$. The only difference is that the adversary can utilize the features of $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$.

**LiRA** $\langle \mathsf{MemInf}, \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ **[9].** Unlike the previous four settings that rely on confidence scores or prediction probabilities, LiRA leverages multiple shadow models and constructs a likelihood ratio based on the difference in model outputs when data points are included or excluded from training. Therefore, we need to train different shadow models with different $\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$.

## 2.4 Property Inference

Property inference attacks (PropInf) [20, 45, 46, 68] aim to infer general information about the training dataset, such as the proportion of data with a specific property unrelated to the main classification task. For example, the gender ratio in the training dataset can be inferred when a model for classifying race is given. Previous works require access to the training process of the model (e.g., via gradients [46]) or to model parameters [20]. These methods are easy to implement for a few layers of neural networks. However, once the model becomes complex, the vast computational and memory resources are difficult to achieve. In addition, we build the query auxiliary datasets $\mathcal{D}^{\mathcal{Q}}_{\mathsf{aux}}$ with different proportions of property. Therefore, in this paper, given a target model $\mathcal{M}$, the adversary first trains the shadow models by shadow auxiliary datasets $\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$ with different proportions of the target property. Next, they query these shadow models to get the outputs of each proportion and concatenate these results together to train a meta-classifier for the property inference. We only need black-box access for this attack. Thus, the property inference can be defined as:

$$\mathsf{PropInf} : \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{T}}_{\mathsf{aux}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rightarrow \{\textit{target property}\} \tag{4}$$

The global properties of a dataset are confidential when they relate to the proprietary information or intellectual property that the data contains, which its owner is not willing to share. This exposure can lead to severe privacy violations, especially if the data is protected by regulations like GDPR [2].

# 3 Threat Modeling

This work focuses on image classification ML models, where the model takes a data sample as input and outputs a probability vector, known as posteriors. Each component of the posteriors represents the likelihood that the sample belongs to a specific class.

We categorize the threat models along two dimensions: 1) *access to the target model* and 2) *availability of an auxiliary dataset*.

**Access to the Target Model.** We consider two access settings: *white-box* and *black-box*. In the white-box setting ($\mathcal{M}^{\mathsf{W}}$), the adversary has full knowledge of the target model, including its parameters and architecture. In contrast, the black-box setting ($\mathcal{M}^{\mathsf{B}}$) limits the adversary to interact with the model like an API, where they can only query it and receive outputs. However, much of the black-box literature [20, 56, 62] also assumes the adversary knows the model's architecture, which they use to build shadow models (see Section 2).

**Auxiliary Dataset.** The adversary needs an auxiliary dataset to train their attack model. For this knowledge, we consider three scenarios: 1) *partial training dataset* ($\mathcal{D}^{\mathsf{P}}_{\mathsf{aux}}$), 2) *shadow auxiliary dataset* ($\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$), and 3) *query auxiliary dataset* ($\mathcal{D}^{\mathsf{Q}}_{\mathsf{aux}}$). In the first scenario, the adversary acquires part of the real training data of the target model (datasets where it is public knowledge). For the $\mathcal{D}^{\mathsf{S}}_{\mathsf{aux}}$ setting, the adversary gets a "shadow" dataset from the same distribution as the training data of the target model, which is used to train a shadow model (see Section V-C in [56] for a discussion on how to generate such data). In the last scenario, the adversary establishes a dataset with different property proportions to query the shadow model, thereby training the attack model for PropInf. This dataset is never used to train either the target model or the shadow model, and it needs to have the same distribution as the target training dataset. Unlike the first two settings, $\mathcal{D}^{\mathsf{Q}}_{\mathsf{aux}}$ is constructed based on the second property proportions that may exist during model training (see Section 2.4).

# 4 Attack Composition

In this section, we introduce our hierarchical compositions of different attack types. First, we propose a taxonomy that offers a structured framework for studying these compositions. Next, we outline the methodologies for specific attack compositions, designating one as the *primary attack* and enhancing it with a *support attack*.

## 4.1 Attack Composition Taxonomy

To address **RQ1**, which examines the approaches for designing and implementing attack compositions, we propose a taxonomy based on the attack pipeline. This taxonomy serves several purposes: (1) Most attack pipelines consist of multiple phases, allowing integration and composition of different attacks at various phases. (2) It is both domain- and model-agnostic, making it easily adaptable to other areas, such as graph data, NLP, and transformer-based models. (3) It offers future researchers a clear framework for studying attack compositions, providing potential benefits to the community.

**Preparatory Level.** In the preparation stage, the adversary gathers information, sets up the environment, and develops the necessary tools. This includes collecting data about the target machine learning system, such as input-output pairs, model parameters, and any accessible metadata, to understand its architecture. The adversary develops or selects appropriate attack algorithms, like FGSM [21] in adversarial example attack, and sets up frameworks and libraries, like PyTorch [3] or CleverHans [51]. Additionally, the adversary prepares the computational infrastructure, including high-performance GPUs or cloud services, and may train a shadow/surrogate model to simulate the target system.

**Execution Level.** During the execution phase, the actual attack is executed against the target machine learning system. For example, the adversary may deploy the attack by generating adversarial examples through perturbing input data to mislead target models or replicating the target model via model extraction. Throughout this phase, the adversary collects outputs and logs detailed data from the target system for

subsequent analysis.

**Evaluation Level.** In the evaluation phase, the adversary analyzes the outcomes, assesses the attack performance, and identifies areas for improvement. This involves defining and measuring success metrics, such as misclassification rates or confidence reductions, and assessing the broader impact on system performance and security. Post-attack analysis includes examining the types of errors induced by the attack and studying changes in model behavior to understand vulnerabilities. Insights gained during this phase guide the refinement and iteration of the attack strategy, enhancing its effectiveness in subsequent attempts.

## 4.2 Preparation Level

We first introduce attack compositions at the preparatory stage. Here, the support attack assists the primary attack during preparation before the primary attack is executed.

### 4.2.1 PropInf2AttrInf

The first attack composition is enhancing AttrInf (primary attack) by using PropInf (support attack) during its preparatory stage. Specifically, adversaries in AttrInf often overlook a key issue: creating a more effective auxiliary dataset for training attack models. The target attribute bias of the target model's training dataset can complicate the auxiliary dataset, making it crucial to address this bias during preparation. Therefore, we amplify AttrInf by employing PropInf to assist in dataset construction during the preparatory phase.

In general, our intuition is that PropInf can better assist in determining the proportion of the target attribute in the training dataset. For AttrInf, we believe that adversaries will not really care about the proportion of the target attribute in the auxiliary dataset. They can never fully eliminate the influence of the bias in the target model without knowing the property information. Therefore, we first determine the distribution of the target attribute in the training dataset using PropInf and further sample the auxiliary dataset, significantly enhancing the effectiveness of AttrInf. In general, the PropInf2AttrInf can be defined as:

$$\begin{aligned}\mathsf{PropInf2AttrInf}: x_{\mathsf{target}}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}_{\mathsf{aux}}, \mathsf{PropInf} \\ \rightarrow \{\textit{target attributes}\}\end{aligned} \tag{5}$$

More concretely, we have two different scenarios for utilizing PropInf, i.e., empirical and theoretical settings. 1) For the empirical setting, we use the real posterior of the PropInf attack model as the confidence for sampling the AttrInf training dataset. For the proportion of the property $p$, given the confidence $c$, the ratio of sampling is $c \times (1-p)$. 2) On the other hand, for the theoretical setting, we directly use the predicted label from PropInf into the sampling function. In general, when enough shadow models are trained, such as 1,000 for each label, the empirical setting becomes the theoretical setting.

## 4.3 Execution Level

At the execution level, the support attack interacts simultaneously with the primary attack during its execution. This concurrent interaction can amplify the impact of the primary attack by leveraging the synergistic effects of support attacks.

### 4.3.1 ADV2MemInf

Previous work [38] has demonstrated a distribution shift between the members and non-members when calculating the distance between the adversarial examples and the original images. Following this intuition, we trade this distance as additional information to assist MemInf. For the $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{B}}, \mathcal{D}_{\mathsf{aux}} \rangle$, we choose a black-box adversarial attacks, Square [5]. Square is a score-based black-box adversarial attack that does not rely on a local gradient. Instead, it utilizes a randomized search scheme that selects localized square-shaped updates at random positions so that at each iteration, the perturbation is situated approximately at the boundary of the dataset. For the $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}_{\mathsf{aux}} \rangle$, we choose a white-box adversarial attack, PGD [44]. It is an iterative method that makes small modifications to the input data at each step by computing the gradient of the loss function with respect to the input data. This gradient demonstrates how to change the input slightly to increase the loss. When the noise $\delta$ added by the Square or PGD is able to change the prediction of the original label, we stop adding noise and use the data $x_{\mathsf{adv}} = x_{\mathsf{target}} + \delta$ as adversarial examples. In the experiments, we find that calculating the norm of the distance provides better assistance than directly using the distance. For LiRA, we estimate the joint distribution of members and non-members using their logits and auxiliary scores. Based on the shadow model outputs, we calculate the covariance matrix and mean separately for members and non-members. For each target sample, we compute the log-likelihood under the member and non-member distributions using the multivariate normal PDF. The LiRA score is computed as the difference between the two log-likelihoods: $\text{score} = -\log P(x \mid \text{in}) + \log P(x \mid \text{out})$.

Therefore, we first calculate the $L_2$ distance between member (non-member) samples and their adversarial samples in the auxiliary dataset $\mathcal{D}_{\mathsf{aux}}$. Next, in addition to the normal inputs required for MemInf, such as outputs from the target or shadow model and predicted labels, we also use the $L_2$ distances as other inputs to train the attack model. As a result, ADV2MemInf can be defined as:

$$\begin{aligned}\mathsf{ADV2MemInf}: x_{\mathsf{target}}, \mathcal{M}, \mathcal{D}_{\mathsf{aux}}, L_2^{\mathcal{D}_{\mathsf{aux}}} \\ \rightarrow \{\textit{member}, \textit{non-member}\}\end{aligned} \tag{6}$$

### 4.3.2 ADV2PropInf

Currently, PropInf heavily depends on training a large number of shadow models. The more shadow models, the better the effectiveness of PropInf. However, training such a large number of shadow models is computationally expensive. Therefore, we hope to find additional information to reduce the number of shadow models and increase the accuracy of PropInf. Thus, similar to ADV2MemInf, our intuition is, for the auxiliary datasets $\mathcal{D}_{\mathsf{aux}}^{\mathsf{T}}$ with different proportions of the target property, the distribution of the $L_2$ distance between these samples and their adversarial samples

should also be different. For example, the distributions of $L_2$ distances calculated on the auxiliary dataset by models trained on a male-to-female ratio of 5:5 versus 2:8 are different. Following this intuition, we concatenate these $L_2$ distances with the original inputs of PropInf together to train a meta-classifier. ADV2PropInf can be defined as:

$$\text{ADV2PropInf}: \mathcal{M}, \mathcal{D}^{\mathsf{Q}}_{\text{aux}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}}, L_2^{\mathcal{D}^{\mathsf{Q}}_{\text{aux}}} \rightarrow \{\textit{target property}\} \tag{7}$$

## 4.4 Evaluation Level

In the evaluation stage, the support attack assists the primary attack after its initial execution. This post-attack support can refine the primary attack's outcomes, correct discrepancies, or further exploit vulnerabilities. In other words, the support attack serves to *calibrate* the results of the primary attack.

### 4.4.1 PropInf2MemInf

Previous work [68] finds that PropInf on GAN models can improve the effectiveness of MemInf. MemInf is enhanced by calibrating the output of the attack model with the proportion of the target property $\lambda_p \frac{1}{N}\sum_i^N(\mathcal{P}_i - 0.5)$. Among those, $\lambda_p$ controls the magnitude of the enhancement. $\mathcal{P}_i - 0.5$ is the proportion of the label to which the target sample belongs. However, for ML models, this calibration is equivalent to directly finding another threshold to classify MemInf. In this scenario, our intuition is a sample has a larger possibility of being a member when it shares the same property with most samples in the target property. Unlike previous work [68], we further train an encoder $\mathcal{E}$ to select different $\lambda$s for the calibration during the attack model training phase, thereby boosting MemInf more effectively. Note that the input of the encoder is the output of the target model $\mathcal{M}$. Formally, the new calibration of MemInf is defined as:

$$\text{PropInf2MemInf}: x_{\text{target}}, \mathcal{M}, \mathcal{D}_{\text{aux}}, \lambda \rightarrow \{\textit{member}, \textit{non-member}\} \tag{8}$$

For normal MemInf, $\lambda$ is a set of $\mathcal{E}(\mathcal{M}(\mathcal{D}_{\text{aux}}))$ and the calibration function is $\lambda \frac{1}{N}\sum_i^N(\mathcal{P}_i - 0.5)$. Since PropInf in our scenario is a black-box attack, we can relax this information on both black/white-box MemInf. Specifically, different from PropInf2AttrInf, since the confidence of PropInf in this scenario is a constant number, there is no difference between empirical and theoretical settings. For LiRA, when we know the prior distribution of a certain property (e.g., class label, gender, category) in the target model training dataset, we can incorporate this knowledge to refine the LiRA membership inference scores. Specifically, we adjust the LiRA score of each sample based on the prior probability of its associated property. Let $s_i$ be the original LiRA score for sample $i$, and let $p_i$ be the property value associated with that sample. If the prior distribution over the property is known and denoted as $P(p)$, we can compute a prior-adjusted score: $\lambda = s_i \cdot P(p_i)$.

## 5 The CoAT Toolkit

In this section, we present CoAT, a modular toolkit designed to evaluate the above attack compositions. Researchers have developed several software tools to measure the potential security/privacy risks of ML models, such as DEEPSEC [39] and CleverHans [51] for evaluating adversarial example attacks, TROJANZOO [50] for backdoor attacks, and ML-Doctor [42] for jointly analyzing the relationships among different attacks. Inspired by this work, we design a systematic framework to modularize our experiments better, namely CoAT. To our knowledge, CoAT is the first framework that jointly considers the composition of different inference-time attacks.

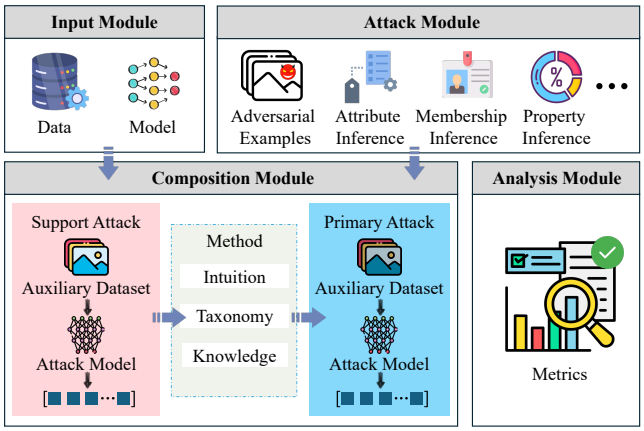


**Figure 2: Overview of the workflow of CoAT.**

**Modules.** Figure 2 illustrates the four modules of CoAT:

1. **Input.** This module prepares the dataset and model for the other modules. More precisely, it performs dataset partition/preprocessing, constructs model architectures, and trains the model.

2. **Attack.** This module includes four inference-time attacks, each employing the most representative strategy. These attacks can be seamlessly replaced or updated with newer versions.

3. **Composition.** This module implements attack compositions where one support attack assists a primary attack. Currently, we have introduced four specific attack composition methods. Notably, users can add new composition methods as needed.

4. **Analysis.** This module evaluates and compares the performance of individual attacks and attack compositions. We include various evaluation metrics to provide a comprehensive analysis.

Overall, the modular design of CoAT allows researchers and practitioners to reuse it as a standard benchmark tool, experimenting with new and additional datasets, model architectures, and attacks.

## 6 Experimental Settings

We first select three benchmark datasets (see Section 6.1) and three state-of-the-art ML models (see Section 6.2) to train thousands of target and shadow models. For each dataset, we partition it into four parts (see Section 6.1), including the

target training dataset, target testing dataset, shadow training dataset, and shadow testing dataset, to comply with the different scenarios discussed in Section 4.1.

## 6.1 Datasets

In this work, we consider three benchmark datasets.

- **CelebA [43]** contains 202,599 face images, each labeled with 40 binary attributes. We select three attributes—*HighCheekbones*, *WearingNecktie*, and *ArchedEyebrows*—to define the target models' classes. The first two attributes form a 4-class classification for the first property, while the third attribute represents the second property.
- **CIFAR10 [1]** is a widely used dataset containing 60,000 32x32 color images across ten classes, with 6,000 images per class. We group the second property into two categories: animal and non-animal.
- **Places [67]** contains 1.8 million training images from 365 scene categories. The validation set has 50 images per category, and the test set has 900. For our study, we select 20 scenes, with 3,000 images each, and group them into two categories—indoor and outdoor—for the second property.

We partition each dataset into four components to facilitate comprehensive evaluation. The first part comprises the target training dataset. For PropInf, we employ different random seeds to select samples based on the second property, ensuring alignment with the desired proportional distribution. For other configurations, we maintain the balanced proportional distribution in the original dataset. The second part constitutes the target test dataset, which is methodically balanced across various properties to ensure unbiased evaluation. The third part encompasses the shadow training dataset, which is constructed following the same method utilized for the target training dataset. Finally, the fourth part consists of the shadow test dataset, which mirrors the selection criteria applied to the target test dataset, maintaining consistency in our experimental framework. Note that this dataset splitting is the basic setup in this field [19, 24, 36, 41, 42, 47, 55, 56].

## 6.2 Target Models

We select three widely-used ML models, i.e., DenseNet121 [26], ResNet18 [23], and VGG19 [57]. We set the mini-batch size to 256 and use cross-entropy as the loss function. We use Adam [31] as the optimizer with a learning rate of 1e-2. Each target model is trained for at most 100 epochs. Once the overfitting is larger than 0.250, model training is finished. Note that for shadow models used in the MemInf and PropInf, we train thousands following the same process as the target models with the support of SecurityNet [63].

## 6.3 Attack Models

**Attribute Inference.** At the preparatory level, the assistant from PropInf will not influence the types of inputs. Therefore, our attack model is a 2-layer MLP where its input is the embeddings from the second-to-last layer of the target model. We use cross-entropy as the loss function and Adam as the optimizer with a learning rate of 1e-2. The attack model is trained for 100 epochs. We use *accuracy* and *F1 score* for the evaluation metrics.

**Membership Inference.** Recall that there are four different scenarios for MemInf; we establish two types of attack models: one for the black-box and the other for the white-box setting. For black-box settings, our original attack model has two inputs: the target sample's ranked posteriors and a binary indicator on whether the target sample is predicted correctly. Each input is first fed into a different 2-layer MLP. Then, the two obtained embeddings are concatenated and fed into a 4-layer MLP. For the white-box, we have four inputs for this attack model, including the target sample's ranked posteriors, classification loss, gradients of the parameters of the target model's last layer, and one-hot encoding of its true label. Each input is fed into a different neural network, and the resulting embeddings are concatenated as input to a 4-layer MLP. We use ReLU as the activation function for the attack models. For the attack scenario assisted by ADV, the inputs of both the black-box and white-box attack models expand the $L_2$ distance between each image and its adversarial example in the auxiliary dataset. The original attack model remains the same for the attack scenario assisted by PropInf, but the encoder for choosing $\lambda$ is a 4-layer MLP. The attack model is trained for 50 epochs by using the Adam optimizer with a learning rate of 1e-5. We adopt *accuracy*, *F1 score*, *AUC score*, and *TPR @0.1% FPR* as the evaluation metrics.

**Property Inference.** Recall that the algorithm level needs to add additional information during the attack phase. For PropInf, the attack model is a meta-classifier; its inputs are organized from the unified overall outputs of each target (shadow) model by feeding the test auxiliary dataset with different proportions of another property. For the assisted PropInf, the inputs also expand a one-dimensional vector composition of the $L_2$ distance between each image and its adversarial example in the test auxiliary dataset. We adopt *accuracy* as the evaluation metric on 100 models.

# 7 Experimental Evaluation

## 7.1 Target Model Utility

First, we present target model utilities in Table 2. Based on previous work [42], we define an overfitting level as the difference between its accuracy on the training and test datasets; the greater this difference, the more overfitting the model is. As shown, the overfitting levels in our target models are less than 0.250. On the other hand, we ensure a real-world scenario as much as possible to validate the effectiveness of our attack composition. Note that target models trained on datasets with a 2:8 proportion for the second property are used for PropInf, while a 5:5 proportion is used for other attacks.

## 7.2 Preparation Level

At this level, since we only need to change the data preprocessing phase, the subsequent training of the attack model will remain consistent with the original attack. In this case,

**Table 2: Performance of target models, namely, training/testing accuracy for each setting. We also provide the results of different proportions of the second property.**

| Property Proportion | CelebA 2:8 | CelebA 5:5 | CIFAR10 2:8 | CIFAR10 5:5 | Places 2:8 | Places 5:5 |
|---|---|---|---|---|---|---|
| **DenseNet121** | 0.988/0.835 | 0.987/0.840 | 0.866/0.653 | 0.882/0.687 | 0.844/0.634 | 0.883/0.668 |
| **ResNet18** | 0.994/0.829 | 0.993/0.834 | 0.812/0.600 | 0.896/0.677 | 0.821/0.584 | 0.709/0.589 |
| **VGG19** | 0.935/0.833 | 0.937/0.845 | 0.764/0.565 | 0.843/0.645 | 0.842/0.668 | 0.878/0.677 |

**Table 3: Performance of PropInf2AttrInf. Here, the empirical setting is based on the confidence (posterior) of PropInf, while the theoretical setting is the label of the prediction of PropInf.**

| Model | Mode | CelebA F1 Score | CelebA Accuracy | CIFAR10 F1 Score | CIFAR10 Accuracy | Places F1 Score | Places Accuracy |
|---|---|---|---|---|---|---|---|
| **DenseNet121** | Origin | 0.771 | 0.712 | 0.916 | 0.911 | 0.667 | 0.500 |
| | Empirical | 0.789 | 0.780 | 0.930 | 0.929 | 0.923 | 0.921 |
| | Theoretical | 0.782 | 0.783 | 0.930 | 0.930 | 0.916 | 0.914 |
| **ResNet18** | Origin | 0.779 | 0.736 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.790 | 0.772 | 0.895 | 0.894 | 0.901 | 0.895 |
| | Theoretical | 0.789 | 0.774 | 0.880 | 0.872 | 0.911 | 0.909 |
| **VGG19** | Origin | 0.742 | 0.664 | 0.911 | 0.905 | 0.915 | 0.910 |
| | Empirical | 0.757 | 0.747 | 0.918 | 0.921 | 0.937 | 0.937 |
| | Theoretical | 0.759 | 0.748 | 0.917 | 0.917 | 0.937 | 0.937 |

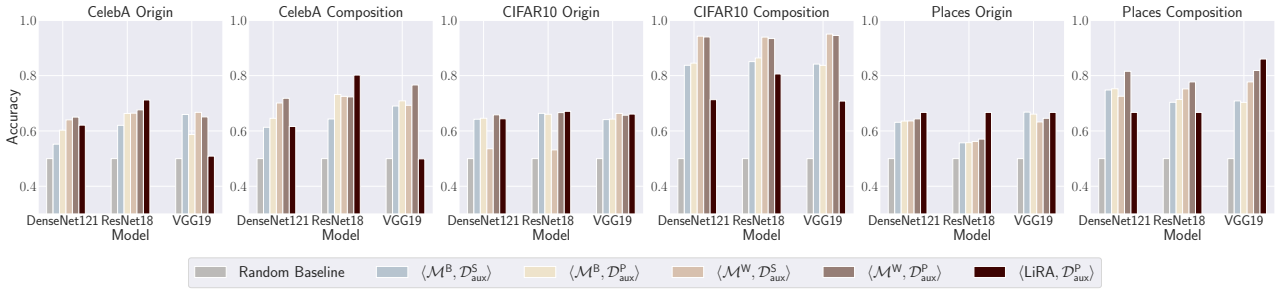


**Figure 3: Accuracy of ADV2MemInf under different threat models, datasets, and target model architectures.**

our focus will be on preprocessing the dataset. As mentioned before, we demonstrate this attack level through PropInf2AttrInf.

### 7.2.1 PropInf2AttrInf

We present the performance of PropInf2AttrInf by comparing it with the original AttrInf first. Table 3 demonstrates the results of PropInf2AttrInf. We can find that the original AttrInf achieves a random guess for three scenarios. This indicates that simply collecting datasets will easily cause severe bias in property proportions, making original AttrInf challenging to achieve. Besides, the results are obviously better than the original attacks in both empirical and theoretical settings. For example, when using CIFAR10 to launch AttrInf on the DenseNet121 model, the original F1 score is 0.916, and accuracy is 0.911, while PropInf2AttrInf can achieve 0.930 and 0.929 for the empirical setting as well as 0.930 and 0.930 for the theoretical setting. This also means that with the assistance of PropInf, AttrInf can indeed achieve better results, which verifies our intuition: PropInf can better assist in determining the proportion of the target attribute in the original training dataset.

In addition, by training the PropInf attack model with 1,000 shadow models, the confidence of our target models exceeds 0.950. Therefore, there is little essential difference between our empirical and theoretical settings. In a nutshell, preprocessing in the preparatory phase is very intuitive, which requires us to choose a good assistant to complete.

## 7.3 Execution Level

At this level, we leverage ADV to assist two different types of attacks, MemInf and PropInf, during their execution stages.

### 7.3.1 ADV2MemInf

First, we evaluate the results of MemInf. We report the accuracy in Figure 3 of ADV2MemInf, while Figure 8, Figure 9, and Table 9, respectively, F1, AUC score, and TPR @0.1% FPR. For some experiments, the original attacks do not achieve much higher attack performance than the random baseline, which means that overfitting does not have a significant impact on the attack [56]; see Section 2.3. For instance, the original attack accuracy, F1 score, and AUC score of $\langle \text{MemInf}, \mathcal{M}^{\text{W}}, \mathcal{D}^{\text{P}}_{\text{aux}} \rangle$ on ResNet18 trained on Places are 0.544, 0.572, and 0.570, respectively. TPR @0.1% FPR

score is 0.001, which is very low in this scenario. Compared to the previous works [12,14,33], white-box attacks have not significantly surpassed black-box attacks. This is expected because, in these works, the training accuracy of the target model can reach 1.000, meaning that for the training dataset, i.e., members, their loss is very close to zero. Nevertheless, this is not the case for non-members, allowing MemInf to achieve a high success rate. In contrast, since the training set accuracy does not reach 1.000 in our work, the loss may act as a form of noise in white-box attacks. We emphasize that our setting is more in line with real-world scenarios.

On the other hand, we find that ADV indeed significantly improves MemInf. For example, the composition attack accuracy, F1 score, and AUC score of $\langle \mathsf{MemInf}, \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ on ResNet18 trained on Places is 0.743, 0.653, 0.777, improved by nearly 0.200 compared to the original MemInf. TPR @0.1% FPR score is also up to 0.490, indicating that our composition attack model is effective at identifying true positives, even under very conservative conditions. More specifically, for the CelebA dataset, since we created a 4-class problem by combining the two labels of the first attribute, the ADV might not perform as well as on the other two datasets. This is because when noise affects one of the labels, it can change the combined class of the image, but this noise may not impact all the labels, leading to a smaller distance between members and non-members compared to the previous datasets. In general, the result first confirms our intuition; there is a distribution shift between the members and non-members when calculating the distance between the adversarial examples and the original data samples. In addition, for ADV, we believe that this distance has magnified the gap between members and non-members, resulting in an enhanced MemInf with a higher success rate. Therefore, the above results verify our intuition: there is a distribution shift between the members and non-members when calculating the distance between the adversarial examples and the original images.

#### 7.3.2 ADV2PropInf

Next, we report our experimental results of ADV2PropInf in Table 4. We can clearly see that with the assistance of ADV, PropInf is significantly improved, which confirms our previous intuition. For example, the original PropInf on ResNet18 trained by CIFAR10 is 0.890 when using 100 shadow models. Nevertheless, after the assistance of ADV, the accuracy is increased to 0.960, equivalent to saving the time required to train at least 300 extra shadow models. Overall, the results of ADV2PropInf verify our intuition: for the auxiliary datasets with different proportions of the target property, the distribution of the $L_2$ distance between these samples and their adversarial samples should also be different.

### 7.4 Evaluation Level

At this stage, the support attack calibrates the results of the primary attack. In this work, we introduce PropInf to calibrate MemInf.

#### 7.4.1 PropInf2MemInf

We report the accuracy of PropInf2MemInf in Figure 4. We also report F1 score and AUC score (Figure 10 and Figure 11) and, in Table 10, the TPR @0.1% FPR results. In many cases, the assistance of PropInf slightly improves MemInf's accuracy. While most TPR @0.1% FPR values remain near zero, there are instances where the composition attack achieves a higher TPR. For example, the composition attack on ResNet18 trained on CIFAR10 shows an accuracy of 0.669, F1 score of 0.731, and AUC of 0.656, compared to the original 0.631, 0.695, and 0.617. The TPR @0.1% FPR improves from 0.000 to 0.002. However, not all results show significant improvement. We attribute this to the general nature of the information from PropInf, which lacks the detailed insights that ADV provides for training the entire model. Without rich data or clear distinctions between members and non-members, improvements in metrics like F1 score and AUC are limited, suggesting that the original MemInf may already be near its upper bound. We attribute this to the general nature of the information from PropInf, which lacks the detailed insights that ADV provides for training the entire model. Improvements in metrics like F1 score and AUC are limited, suggesting that the original MemInf may already be near its upper bound. We also observe that with PropInf's support, attack performance remains stable across different scenarios (black-box and white-box), indicating that PropInf helps MemInf approach its performance limit. These results confirm our intuition: a sample is more likely to be a member if it shares properties with most samples in the target group.

### 7.5 Takeaways

Overall, our evaluations demonstrate that combining different attack types significantly improves the effectiveness of primary attacks, leading to higher accuracy and success rates. These results confirm our earlier intuition about the benefits of attack compositions. Specifically, using ADV to assist MemInf and PropInf, as well as PropInf to assist AttrInf and MemInf, notably amplifies the ability to identify training data and infer sensitive information. Our CoAT emerges as a valuable tool for systematically evaluating these compositions, highlighting the necessity for more robust defense mechanisms to counteract these amplified threats.

## 8 Ablation Study

### 8.1 Differential Privacy

Differential privacy (DP) [18, 35] is a mathematical framework for quantifying and protecting individual privacy in statistical analysis and machine learning. It formalizes privacy guarantees by ensuring that the inclusion or exclusion of any single data point in a dataset does not significantly affect the outcome of any analysis, thereby limiting the risk of re-identification. DP has been widely adopted as a principled defense mechanism against membership inference attacks [25,28,42,47,49], offering formal privacy guarantees to mitigate the risk of exposing individual data records. However, to our best knowledge, DP is not a defense mechanism against other attacks. In this paper, we incorporate DP into

**Table 4: Performance of ADV2PropInf.**

| Model | CelebA | | CIFAR10 | | Places | |
|---|---|---|---|---|---|---|
| | Origin | Composition | Origin | Composition | Origin | Composition |
| **DenseNet121** | 0.520 | 0.600 | 0.850 | 0.910 | 0.620 | 0.750 |
| **ResNet18** | 0.510 | 0.600 | 0.890 | 0.960 | 0.750 | 0.830 |
| **VGG19** | 0.540 | 0.630 | 0.860 | 0.930 | 0.730 | 0.750 |

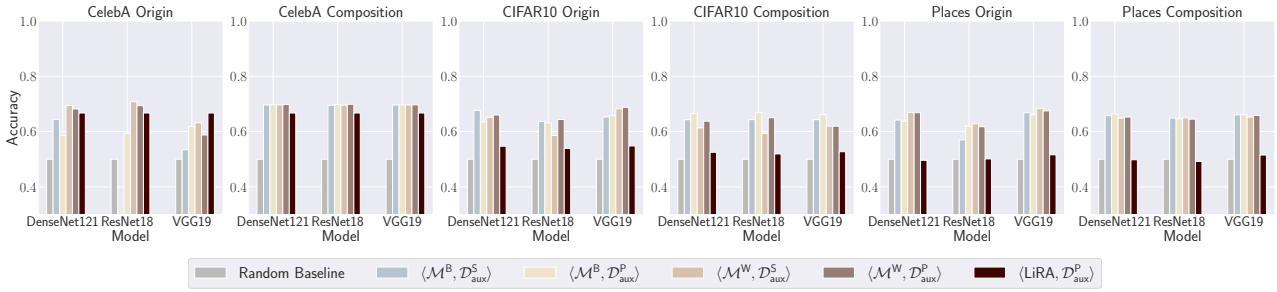


**Figure 4: Accuracy of PropInf2MemInf under different threat models, datasets, and target model architectures.**

**Table 5: Performance of DP target models, namely, training/testing accuracy for each setting. We also provide the results of different proportions of the second property. Specifically, $\delta = 1e-05$.**

| ResNet18 | | CelebA | CIFAR10 | Places |
|---|---|---|---|---|
| $\varepsilon = 10$ | 2:8 | 0.831/0.730 | 0.248/0.286 | 0.268/0.192 |
| | 5:5 | 0.772/0.762 | 0.271/0.327 | 0.223/0.233 |
| $\varepsilon = 20$ | 2:8 | 0.841/0.753 | 0.275/0.340 | 0.304/0.207 |
| | 5:5 | 0.785/0.767 | 0.336/0.402 | 0.255/0.249 |
| $\varepsilon = 50$ | 2:8 | 0.851/0.754 | 0.346/0.415 | 0.377/0.251 |
| | 5:5 | 0.866/0.740 | 0.389/0.456 | 0.304/0.295 |

the model training process by applying DP mechanisms to the optimizer, specifically using the Adam optimizer as mentioned in Section 6.2. After training, we systematically evaluate the impact of differential privacy on the attack compositions by using CoAT.

**Model Performance.** We first present the model performance with three different ε values on Table 5. Since adding DP into models requires a large computation cost, we conduct an in-depth research of the impact of DP on attack composition using the ResNet architecture as our main model. From the table, only CelebA can achieve a standard performance without large utility degradation. For the other two datasets, DP largely influences the performance.

### 8.1.1 PropInf2AttrInf

Table 6 shows the results of PropInf2AttrInf when DP is added. First, DP does not impact PropInf, thus we can still achieve robust empirical results. Consequently, regarding the PropInf2AttrInf, DP proves even less effective as a defense mechanism against composition.

### 8.1.2 ADV2MemInf

Figure 5, Figure 12, Figure 13, and Table 12 illustrate the effectiveness of ADV2MemInf when DP is implemented during the training phase. The results indicate that although DP generally serves as an effective defense mechanism against MemInf attacks, it can be partially bypassed when adversarial examples are incorporated. For instance, in the case of $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ on the CIFAR10 dataset, the original MemInf attack yielded metrics of 0.529 (Accuracy), 0.490 (F1 Score), 0.538 (AUC), and 0.001 (TPR @0.1% FPR), respectively. When augmented with ADV, these values significantly increased to 0.659, 0.701, 0.674, and 0.004, respectively. Furthermore, for $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$, the ADV2MemInf approach demonstrated consistent performance improvements across all three datasets.

### 8.1.3 ADV2PropInf

Table 7 demonstrate the performance of ADV2PropInf with adding DP. From this table, we posit that the performance of models trained on CelebA can remain stable when DP is added during training. Specifically, the distribution of the $L_2$ distances between original samples and their corresponding adversarial examples assists in distinguishing models with different properties. In contrast, for the other two datasets, the model outputs inherently exhibit substantial noise, which in turn obscures the effect of DP and compromises the consistency of the evaluation. Overall, while DP serves as a defense mechanism against MemInf, its influence on attack composition appears limited—unless it significantly degrades the model performance.

### 8.1.4 PropInf2MemInf

Figure 6, Figure 14, Figure 15, and Table 11 demonstrate the results of PropInf2MemInf after adding DP during the training phase. Compared to the original MemInf, the MemInf

**Table 6: Performance of PropInf2AttrInf with adding DP. Here, the empirical setting is based on the confidence (posterior) of PropInf, while the theoretical setting is the label of the prediction of PropInf.**

| | Mode | CelebA F1 Score | CelebA Accuracy | CIFAR10 F1 Score | CIFAR10 Accuracy | Places F1 Score | Places Accuracy |
|---|---|---|---|---|---|---|---|
| $\varepsilon = 10$ | Origin | 0.667 | 0.500 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.776 | 0.769 | 0.742 | 0.743 | 0.805 | 0.829 |
| | Theoretical | 0.778 | 0.769 | 0.734 | 0.744 | 0.806 | 0.828 |
| $\varepsilon = 20$ | Origin | 0.667 | 0.500 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.777 | 0.772 | 0.790 | 0.786 | 0.854 | 0.880 |
| | Theoretical | 0.770 | 0.774 | 0.785 | 0.787 | 0.854 | 0.878 |
| $\varepsilon = 50$ | Origin | 0.667 | 0.500 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.787 | 0.778 | 0.759 | 0.785 | 0.813 | 0.825 |
| | Theoretical | 0.783 | 0.778 | 0.761 | 0.794 | 0.820 | 0.836 |

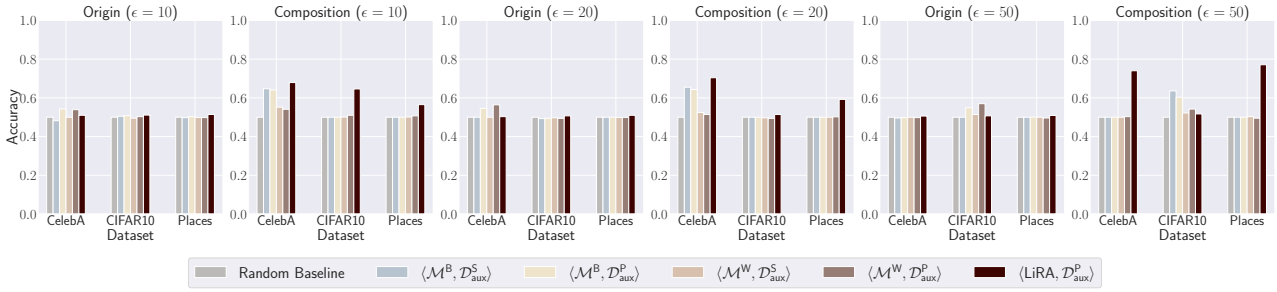


**Figure 5: Accuracy of ADV2MemInf under different threat models, datasets, and target model architectures with adding DP.**

**Table 7: Performance of ADV2PropInf with adding DP.**

| | Mode | CelebA | CIFAR10 | Places |
|---|---|---|---|---|
| $\varepsilon = 10$ | Origin | 0.740 | 0.890 | 0.830 |
| | Composition | 0.790 | 0.610 | 0.670 |
| $\varepsilon = 20$ | Origin | 0.730 | 0.790 | 0.830 |
| | Composition | 0.770 | 0.540 | 0.790 |
| $\varepsilon = 50$ | Origin | 0.710 | 0.790 | 0.890 |
| | Composition | 0.740 | 0.670 | 0.780 |

calibrated with PropInf shows significant improvement, such as $\langle \mathcal{M}^{B}, \mathcal{D}^{S}_{aux} \rangle$ and $\langle \mathcal{M}^{B}, \mathcal{D}^{P}_{aux} \rangle$, despite the addition of DP. Therefore, we conclude that although adding DP can defend against MemInf, it is still possible to enhance MemInf through PropInf.

**Takeaways.** Despite DP being a very effective defense mechanism against MemInf, it can still be bypassed through auxiliary attacks that enhance its effectiveness. Furthermore, DP remains ineffective against other types of attacks, and when these other attacks are used as auxiliary methods.

## 8.2 Chain of Composition

To facilitate a more in-depth and insightful discussion, we propose the concept of a chain of composition, namely $A_1 2 A_2 2 A_3$. This approach leverages the interplay between different attacks to enhance the overall effectiveness of the attack strategy. By systematically composing multiple attacks, we aim to amplify their impacts on the target models.

### 8.2.1 ADV2PropInf2AttrInf

We only discuss the empirical settings here, since the theoretical setting is the same as PropInf2AttrInf. Table 8 demonstrates the results of ADV2PropInf2AttrInf. We find that ADV2PropInf2AttrInf outperforms standalone AttrInf and is similar to PropInf2AttrInf (see Section 7.2). For example, the attack accuracy of standalone AttrInf on ResNet18 with CIFAR10 is 0.500, while PropInf2AttrInf achieves 0.894, and ADV2PropInf2AttrInf achieves 0.901. ADV2PropInf2AttrInf performs similarly to PropInf2AttrInf because the confidence of the predictions of PropInf with ADV enhancement is similar to that of standalone PropInf models. This similarity in confidence, exploited by AttrInf, results in comparable performance between ADV2PropInf2AttrInf and PropInf2AttrInf. However, this does not contradict our claim that ADV2PropInf outperforms standalone PropInf, as we measure the overall attack accuracy instead of the confidence.

### 8.2.2 ADV2PropInf2MemInf

Figure 7, Figure 16, and Figure 17 show the results of ADV2PropInf2MemInf. ADV2PropInf2MemInf outperforms standalone MemInf and is similar to PropInf 2MemInf (see Section 7.4). For example, the accuracy of standalone MemInf on VGG19 with Places20 is 0.630, PropInf2MemInf achieves 0.647, and ADV2PropInf 2MemInf achieves 0.652. The reason that ADV2PropInf2 MemInf performs similarly to PropInf2MemInf is the same as discussed above: the confidence of the ADV2PropInf model is similar to that of the PropInf model, which MemInf

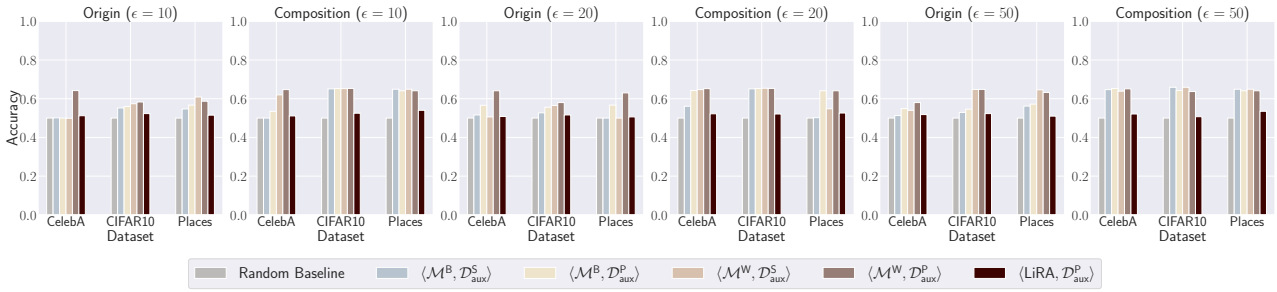


**Figure 6: Accuracy of PropInf2MemInf under different threat models, datasets, and target model architectures with adding DP.**

**Table 8: Performance of ADV2PropInf2AttrInf. Here, we only show the empirical setting.**

| Model | Mode | CelebA | | CIFAR10 | | Places | |
|---|---|---|---|---|---|---|---|
| | | F1 Score | Accuracy | F1 Score | Accuracy | F1 Score | Accuracy |
| **Densenet121** | Origin | 0.771 | 0.712 | 0.916 | 0.911 | 0.667 | 0.500 |
| | Empirical | 0.795 | 0.792 | 0.937 | 0.925 | 0.922 | 0.918 |
| **ResNet18** | Origin | 0.667 | 0.500 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.790 | 0.782 | 0.901 | 0.901 | 0.918 | 0.896 |
| **VGG19** | Origin | 0.667 | 0.500 | 0.667 | 0.500 | 0.667 | 0.500 |
| | Empirical | 0.764 | 0.761 | 0.938 | 0.938 | 0.923 | 0.919 |

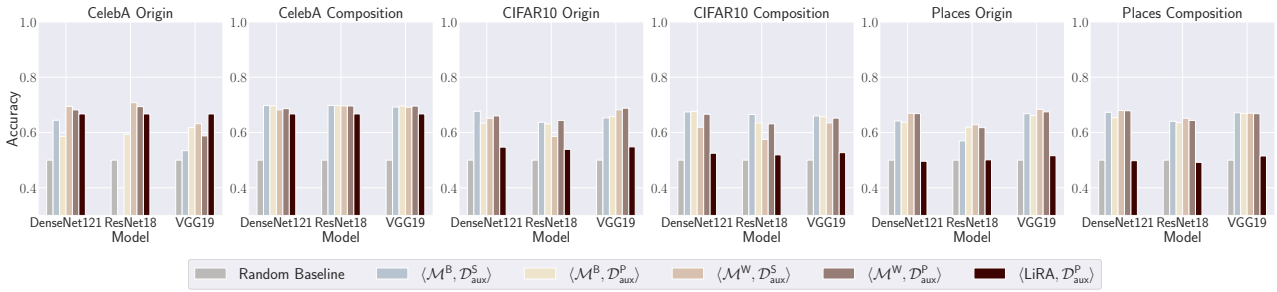


**Figure 7: Accuracy of ADV2PropInf2MemInf under different threat models, datasets, and target model architectures.**

exploits.

**Takeaways.** The chain of composition is an interesting phenomenon that shows significant improvement compared to the original method. We will explore the depth and insights of our paper.

## 9 Related Work

**Security and Privacy Attacks Against ML models.** ML models are vulnerable to various security and privacy attacks. Specifically, we focus on four representative attacks at the inference phase of the target ML model to better study the interactions of different attacks at broad levels. Adversarial examples are the most popular security attacks against ML models. They can mislead the prediction of the target ML model by adding imperceptible perturbations to the input data sample at the inference phase. The early methods [21, 60] for finding adversarial examples assume that the adversary has full access to the target ML model (i.e., the white-box setting) and they rely on some optimization strategies to search for the results, while the latter variants [5, 34] have been developed to cope with various more challenging scenarios (e.g., the black-box setting). In this work, we implement the white-box attack method PGD [44] and the black-box attack method Square [5] to study the effectiveness of adversarial examples in assisting other attacks.

Membership inference is a popular privacy attack to determine whether a data sample exists in the training dataset of the target ML model. Shokri et al. [56] propose the first membership inference attack against black-box ML models. They train multiple shadow models on the shadow dataset to mimic the behavior of the target model and then train an attack model on the prediction posteriors of these shadow models to predict the membership of the input data. Salem et al. [55] relax the key assumptions of Shokri et al. [56] and introduce the model- and data-independent method to effectively conduct membership inference on black-box models. Since then, membership inference has been adapted to different settings (e.g., white-box [48] and label-only [38]), domains (e.g., computer vision [37, 38], recommender system [64], and unlearning system [14]), and models (e.g., generative model [12] and multi-exit model [37]). We adapt the methods proposed by Salem et al. [55], and Nasr et al. [48] to implement black-box and white-box membership inference in our work.

Attribute inference is another representative privacy attack that tries to learn extra information unrelated to the tar-

get task of the ML model. Melis et al. [46] introduced the first sample-level attribute inference attack targeting federated ML systems. Song and Shmatikov [58] demonstrated that the potential risks of attribute inference stem from the intrinsic overlearning characteristics of ML models. A common strategy for conducting attribute inference is to utilize an auxiliary dataset to build the connection between the model embedding and the target attribute.

Property inference is a privacy attack aiming to infer the general property (e.g., data distribution) of the training dataset. The inferred property is usually unrelated to the main task of the target ML model. Prior work usually relies on the gradients [46] or the model parameters [20] on the auxiliary dataset built with abundant shadow models to conduct property inference on small and simple target model architectures, which is computationally prohibitive for large and complex target models. In this work, we adapt the previous attack methods by concatenating the query posteriors to serve as the input features for the final attack model.

**Interactions among Attacks.** Several studies have revealed relationships between different types of attacks. Li et al. [38] found a positive correlation between a sample's membership status and its robustness to adversarial noise. They leveraged the differing adversarial noise magnitudes of members and non-members to mount a membership inference attack. However, our work differs significantly from theirs. We integrate one attack into another at different phases, using information from one attack to enhance or amplify another. In contrast, Li et al. rely on adversarial example information as the only signal for membership inference, without incorporating its original signal. Recently, Wen et al. [61] proposed a method to strengthen membership inference through training-phase data poisoning attacks. However, data poisoning is a training-time attack, while membership inference occurs during the inference phase. We emphasize that although it is possible for an attacker to launch attacks during both the training and inference phases, this assumption is overly strong. As the first to systematically study the interactions between different attacks, we start only with the inference-time attack, as this is the most realistic scenario. We also investigate the chain of composition, through ADV2PropInf 2AttrInf and ADV2PropInf2MemInf. We find that the chain of composition can effectively provide a form of attack enhancement.

The prior work most closely related to ours is by Chen et al. [68], who have found that property inference could amplify the performance of membership inference on GANs. However, their study focuses solely on GANs and proposes only one case study of attack composition. Furthermore, it lacks a high-level awareness of the intentional interactions and does not provide a systematic study of the intentional interactions among a more diverse set of attacks. Nonetheless, we acknowledge that their work provides valuable insights and inspires us to conduct this study.

## 10 Discussion

We now discuss in more detail why we focus on the four inference-time attacks in the image domain. There are some attacks during the training phase, such as enhancing membership inference through backdoor attacks [61] or poisoning attacks [16]. These situations are more complex, especially in real-world scenarios, requiring adversaries to make many strong assumptions, such as interfering with the training process or owning the training dataset. In addition, we currently only focus on image datasets because the types of attacks and their implementations are more detailed and comprehensive in image datasets. We also do not consider model stealing attacks, as they primarily, to some extent, convert black-box models to white-box models, which indeed can amplify the success rate of many attacks. Since we aim to explore the impact of attack compositions during the attack's different phases, we emphasize that we do not change the overall attack process and the main attack approach. We emphasize that, currently, no single defense can protect against all ML model attacks, and effective defenses against property inference or attribute inference are lacking. We aim to provide new insights and techniques for enhancing model security through these attack compositions. We leave the in-depth exploration of more effective defense mechanisms against our attack compositions as future work.

## 11 Conclusion

In this paper, we take the first step in exploring the intentional interaction between different types of attacks. Specifically, we focus on four extensively studied inference-time attacks: adversarial examples, attribute inference, membership inference, and property inference. To facilitate the study of their interactions, we establish a taxonomy based on three levels of the attack pipeline: preparation, execution, and evaluation, and propose four different attack compositions: PropInf2AttrInf, ADV2MemInf, ADV2PropInf, and PropInf2MemInf. Extensive experiments across three model architectures and two benchmark datasets demonstrate the superior performance of the proposed attack compositions.

Additionally, we introduce a reusable modular framework named CoAT to integrate our attack compositions. In this framework, we build four distinct modules to systematically examine the attack compositions. We believe that CoAT will serve as a benchmark tool to facilitate future research on attack compositions, enabling the seamless integration of new attacks, datasets, and models to further explore ML model vulnerabilities.

We explore DP as a defense mechanism to test our different attack compositions. We find that although DP can serve as a defense against MemInf, through our composition methods, we can still enhance the effectiveness of MemInf. Additionally, for other types of attacks, DP does not function as an effective defense mechanism.

Overall, we find that many current attacks can be amplified by applying another type of attack, thereby increasing their effectiveness. This finding offers a new perspective for subsequent attacks. In future work, we plan to incorporate more related attacks to further explore the vulnerabilities of ML models. With CoAT, we appeal to the community to consider such real-world scenarios and actively contribute to the development of more robust and secure AI systems.

## References


[1] https://www.cs.toronto.edu/~kriz/cifar.html. 2, 7

[2] https://gdpr-info.eu/. 4

[3] https://pytorch.org/. 4

[4] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. Generating Natural Language Adversarial Examples. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2890–2896. ACL, 2018. 1, 2

[5] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search. In *European Conference on Computer Vision (ECCV)*, pages 484–501. Springer, 2020. 3, 5, 12

[6] Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. PLATO: Pre-trained Dialogue Generation Model with Discrete Latent Variable. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 85–96. ACL, 2020. 1

[7] Yonatan Belinkov and Yonatan Bisk. Synthetic and Natural Noise Both Break Neural Machine Translation. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[8] Philippe Burlina, David E. Freund, B. Dupas, and Neil M. Bressler. Automatic Screening of Age-related Macular Degeneration and Retinal Abnormalities. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3962–3966. IEEE, 2011. 1

[9] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership Inference Attacks From First Principles. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1897–1914. IEEE, 2022. 4

[10] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *IEEE Symposium on Security and Privacy (S&P)*, pages 39–57. IEEE, 2017. 2, 3

[11] Laura Cervi. Exclusionary Populism and Islamophobia: A comparative analysis of Italy and Spain. *Religions*, 2020. 3

[12] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 343–362. ACM, 2020. 3, 9, 12

[13] Jianbo Chen, Michael I. Jordan, and Martin J. Wainwright. HopSkipJumpAttack: A Query-Efficient Decision-Based Attack. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1277–1294. IEEE, 2020. 3

[14] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When Machine Unlearning Jeopardizes Privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 896–911. ACM, 2021. 3, 9, 12

[15] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning. *CoRR abs/1712.05526*, 2017. 1

[16] Yufei Chen, Chao Shen, Yun Shen, Cong Wang, and Yang Zhang. Amplifying Membership Exposure via Data Poisoning. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2022. 13

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186. ACL, 2019. 1

[18] Cynthia Dwork and Aaron Roth. *The Algorithmic Foundations of Differential Privacy*. Now Publishers Inc., 2014. 9

[19] Wenjie Fu, Huandong Wang, Chen Gao, Guanghua Liu, Yong Li, and Tao Jiang. Practical Membership Inference Attacks against Fine-tuned Large Language Models via Self-prompt Calibration. *CoRR abs/2311.06062*, 2023. 7

[20] Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. Property Inference Attacks on Fully Connected Neural Networks using Permutation Invariant Representations. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 619–633. ACM, 2018. 4, 13

[21] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations (ICLR)*, 2015. 2, 3, 4, 12

[22] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Grag. Badnets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. *CoRR abs/1708.06733*, 2017. 1

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. IEEE, 2016. 2, 7

[24] Xinlei He, Zheng Li, Weilin Xu, Cory Cornelius, and Yang Zhang. Membership-Doctor: Comprehensive Assessment of Membership Inference Against Machine Learning Models. *CoRR abs/2208.10445*, 2022. 7

[25] Xinlei He and Yang Zhang. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In *ACM*

*SIGSAC Conference on Computer and Communications Security (CCS)*, pages 845–863. ACM, 2021. 9

[26] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269. IEEE, 2017. 7

[27] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1875–1885. ACL, 2018. 1, 2

[28] Bargav Jayaraman and David Evans. Evaluating Differentially Private Machine Learning in Practice. In *USENIX Security Symposium (USENIX Security)*, pages 1895–1912. USENIX, 2019. 9

[29] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 259–274. ACM, 2019. 3

[30] Ira Kemelmacher-Shlizerman, Steven M. Seitz, Daniel Miller, and Evan Brossard. The MegaFace Benchmark: 1 Million Faces for Recognition at Scale. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4873–4882. IEEE, 2016. 1

[31] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 7

[32] Konstantina Kourou, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, and Dimitrios I. Fotiadis. Machine Learning Applications in Cancer Prognosis and Prediction. *Computational and Structural Biotechnology Journal*, 2015. 1

[33] Klas Leino and Matt Fredrikson. Stolen Memories: Leveraging Model Memorization for Calibrated White-Box Membership Inference. In *USENIX Security Symposium (USENIX Security)*, pages 1605–1622. USENIX, 2020. 3, 9

[34] Huichen Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. QEBA: Query-Efficient Boundary-Based Blackbox Attack. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1218–1227. IEEE, 2020. 12

[35] Ninghui Li, Min Lyu, Dong Su, and Weining Yang. *Differential Privacy: From Theory to Practice*. Morgan & Claypool Publishers, 2016. 9

[36] Zheng Li, Yiyong Liu, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. Auditing Membership Leakages of Multi-Exit Networks. *CoRR abs/2208.11180*, 2022. 7

[37] Zheng Li, Yiyong Liu, Xinlei He, Ning Yu, Michael Backes, and Yang Zhang. Auditing Membership Leakages of Multi-Exit Networks. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1917–1931. ACM, 2022. 12

[38] Zheng Li and Yang Zhang. Membership Leakage in Label-Only Exposures. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 880–895. ACM, 2021. 1, 2, 3, 5, 12, 13

[39] Xiang Ling, Shouling Ji, Jiaxu Zou, Jiannan Wang, Chunming Wu, Bo Li, and Ting Wang. DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model. In *IEEE Symposium on Security and Privacy (S&P)*, pages 673–690. IEEE, 2019. 6

[40] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning Attack on Neural Networks. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2018. 1

[41] Yiyong Liu, Zhengyu Zhao, Michael Backes, and Yang Zhang. Membership Inference Attacks by Exploiting Loss Trajectory. *CoRR abs/2208.14933*, 2022. 7

[42] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, Michael Backes, Emiliano De Cristofaro, Mario Fritz, and Yang Zhang. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In *USENIX Security Symposium (USENIX Security)*, pages 4525–4542. USENIX, 2022. 1, 3, 6, 7, 9

[43] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738. IEEE, 2015. 7

[44] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 3, 5, 12

[45] Saeed Mahloujifar, Esha Ghosh, and Melissa Chase. Property Inference from Poisoning. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1120–1137. IEEE, 2022. 4

[46] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *IEEE Symposium on Security and Privacy (S&P)*, pages 497–512. IEEE, 2019. 2, 3, 4, 13

[47] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine Learning with Membership Privacy using Adversarial Regularization. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 634–646. ACM, 2018. 1, 2, 3, 7, 9

[48] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against

Centralized and Federated Learning. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1021–1035. IEEE, 2019. 3, 12

[49] Milad Nasr, Shuang Song, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlini. Adversary Instantiation: Lower Bounds for Differentially Private Machine Learning. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2021. 9

[50] Ren Pang, Zheng Zhang, Xiangshan Gao, Zhaohan Xi, Shouling Ji, Peng Cheng, and Ting Wang. TROJANZOO: Everything You Ever Wanted to Know about Neural Backdoors (But Were Afraid to Ask). *CoRR abs/2012.09302*, 2020. 6

[51] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. *CoRR abs/1610.00768*, 2018. 4, 6

[52] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings. In *IEEE European Symposium on Security and Privacy (Euro S&P)*, pages 372–387. IEEE, 2016. 2, 3

[53] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Semantically Equivalent Adversarial Rules for Debugging NLP models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 856–865. ACL, 2018. 1, 2

[54] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs Black-box: Bayes Optimal Strategies for Membership Inference. In *International Conference on Machine Learning (ICML)*, pages 5558–5567. PMLR, 2019. 3

[55] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. ML-Leaks: Model and Data Independent Membership Inference Attacks and Defenses on Machine Learning Models. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2019. 1, 2, 3, 7, 12

[56] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *IEEE Symposium on Security and Privacy (S&P)*, pages 3–18. IEEE, 2017. 1, 2, 3, 4, 7, 8, 12

[57] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 7

[58] Congzheng Song and Vitaly Shmatikov. Overlearning Reveals Sensitive Attributes. In *International Conference on Learning Representations (ICLR)*, 2020. 2, 3, 13

[59] Mary H. Stanfill, Margaret Williams, Susan H. Fenton, Robert A. Jenders, and William R. Hersh. A Systematic Literature Review of Automated Clinical Coding and Classification Systems. *J. Am. Medical Informatics Assoc.*, 2010. 1

[60] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing Properties of Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2014. 2, 12

[61] Yuxin Wen, Leo Marchyok, Sanghyun Hong, Jonas Geiping, Tom Goldstein, and Nicholas Carlini. Privacy Backdoors: Enhancing Membership Inference through Poisoning Pre-trained Models. *CoRR abs/2404.01231*, 2024. 13

[62] Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A. Gunter, and Bo Li. Detecting AI Trojans Using Meta Neural Analysis. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2021. 4

[63] Boyang Zhang, Zheng Li, Ziqing Yang, Xinlei He, Michael Backes, Mario Fritz, and Yang Zhang. SecurityNet: Assessing Machine Learning Vulnerabilities on Public Models. In *USENIX Security Symposium (USENIX Security)*. USENIX, 2024. 1, 7

[64] Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhumin Chen, Pengfei Hu, and Yang Zhang. Membership Inference Attacks Against Recommender Systems. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 864–879. ACM, 2021. 12

[65] Zixiao Zhang, Xiaoqiang Lu, Guojin Cao, Yuting Yang, Licheng Jiao, and Fang Liu. ViT-YOLO: Transformer-Based YOLO for Object Detection. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2799–2808. IEEE, 2021. 1

[66] Tianyue Zheng, Weihong Deng, and Jiani Hu. Cross-Age LFW: A Database for Studying Cross-Age Face Recognition in Unconstrained Environments. *CoRR abs/1708.08197*, 2017. 1

[67] Bolei Zhou, Àgata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 7

[68] Junhao Zhou, Yufei Chen, Chao Shen, and Yang Zhang. Property Inference Attacks Against GANs. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2022. 4, 6, 13

**Table 9: TPR @0.1% FPR of** ADV2MemInf**.**

| Model | Mode | CelebA | | CIFAR10 | | Places | |
|---|---|---|---|---|---|---|---|
| | | Origin | Composition | Origin | Composition | Origin | Composition |
| **DenseNet121** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.000 | 0.007 | 0.009 | 0.011 | 0.002 | 0.003 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.002 | 0.006 | 0.002 | 0.217 | 0.002 | 0.003 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.004 | 0.008 | 0.016 | 0.887 | 0.001 | 0.500 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.004 | 0.010 | 0.011 | 0.875 | 0.002 | 0.486 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.054 | 0.079 | 0.105 | 0.203 | 0.061 | 0.182 |
| **ResNet18** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.003 | 0.004 | 0.006 | 0.002 | 0.004 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.003 | 0.009 | 0.004 | 0.073 | 0.002 | 0.003 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.002 | 0.007 | 0.003 | 0.879 | 0.001 | 0.501 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.004 | 0.008 | 0.009 | 0.868 | 0.001 | 0.490 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.078 | 0.283 | 0.120 | 0.144 | 0.013 | 0.088 |
| **VGG19** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.006 | 0.002 | 0.074 | 0.002 | 0.004 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.011 | 0.003 | 0.239 | 0.002 | 0.009 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.008 | 0.016 | 0.902 | 0.001 | 0.500 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.009 | 0.002 | 0.899 | 0.001 | 0.494 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.007 | 0.034 | 0.180 | 0.278 | 0.026 | 0.056 |

**Table 10: TPR @0.1% FPR of** PropInf2MemInf**.**

| Model | Mode | CelebA | | CIFAR10 | | Places | |
|---|---|---|---|---|---|---|---|
| | | Origin | Composition | Origin | Composition | Origin | Composition |
| **DenseNet121** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.002 | 0.007 | 0.003 | 0.002 | 0.003 | 0.003 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.007 | 0.000 | 0.001 | 0.003 | 0.003 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.002 | 0.009 | 0.000 | 0.001 | 0.003 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.003 | 0.009 | 0.000 | 0.000 | 0.002 | 0.003 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.003 | 0.003 | 0.002 | 0.002 | 0.001 | 0.003 |
| **ResNet18** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.000 | 0.004 | 0.000 | 0.002 | 0.000 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.006 | 0.000 | 0.000 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.004 | 0.007 | 0.002 | 0.001 | 0.002 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.005 | 0.009 | 0.001 | 0.004 | 0.002 | 0.004 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.003 | 0.004 | 0.002 | 0.004 | 0.001 | 0.001 |
| **VGG19** | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.010 | 0.001 | 0.001 | 0.001 | 0.004 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.014 | 0.000 | 0.003 | 0.002 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.013 | 0.001 | 0.002 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.015 | 0.005 | 0.000 | 0.004 | 0.001 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\mathsf{aux}} \rangle$ | 0.001 | 0.001 | 0.001 | 0.001 | 0.000 | 0.000 |

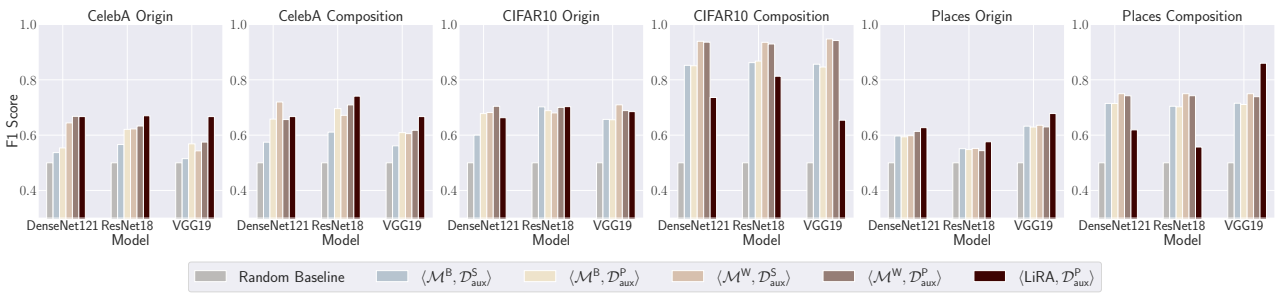


**Figure 8: F1 score of** ADV2MemInf **under different threat models, datasets, and target model architectures.**

**Table 11: TPR @0.1% FPR of PropInf2MemInf with adding DP.**

| | Mode | CelebA Origin | CelebA Composition | CIFAR10 Origin | CIFAR10 Composition | Places Origin | Places Composition |
|---|---|---|---|---|---|---|---|
| $\varepsilon = 10$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.002 | 0.004 | 0.001 | 0.001 | 0.001 | 0.004 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.002 | 0.002 | 0.004 | 0.000 | 0.001 | 0.005 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.003 | 0.005 | 0.002 | 0.001 | 0.000 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.001 | 0.001 | 0.000 | 0.001 | 0.002 | 0.003 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.000 | 0.000 | 0.000 | 0.002 | 0.002 |
| $\varepsilon = 20$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.001 | 0.005 | 0.004 | 0.012 | 0.001 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.002 | 0.001 | 0.002 | 0.007 | 0.001 | 0.005 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.005 | 0.004 | 0.008 | 0.005 | 0.005 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.000 | 0.005 | 0.016 | 0.010 | 0.000 | 0.009 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.001 | 0.000 | 0.002 | 0.003 | 0.003 |
| $\varepsilon = 50$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.005 | 0.009 | 0.009 | 0.035 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.004 | 0.004 | 0.005 | 0.007 | 0.000 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.001 | 0.001 | 0.004 | 0.016 | 0.002 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.003 | 0.002 | 0.004 | 0.011 | 0.001 | 0.001 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.003 | 0.012 | 0.003 | 0.002 | 0.001 | 0.002 |

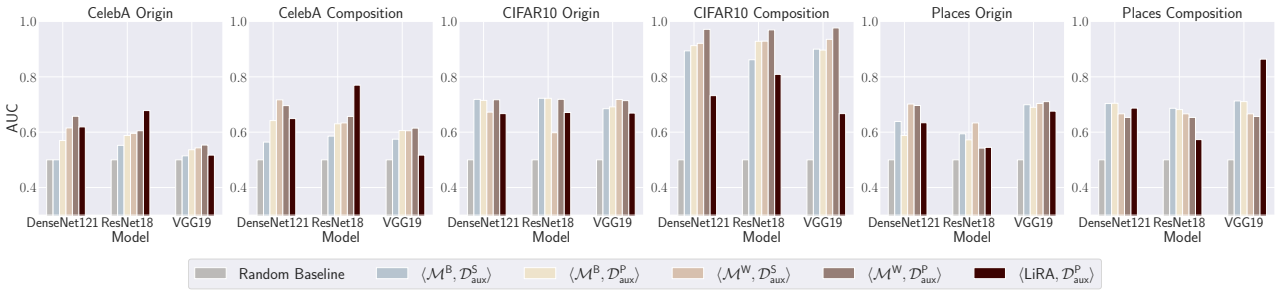


**Figure 9: AUC of ADV2MemInf under different threat models, datasets, and target model architectures.**

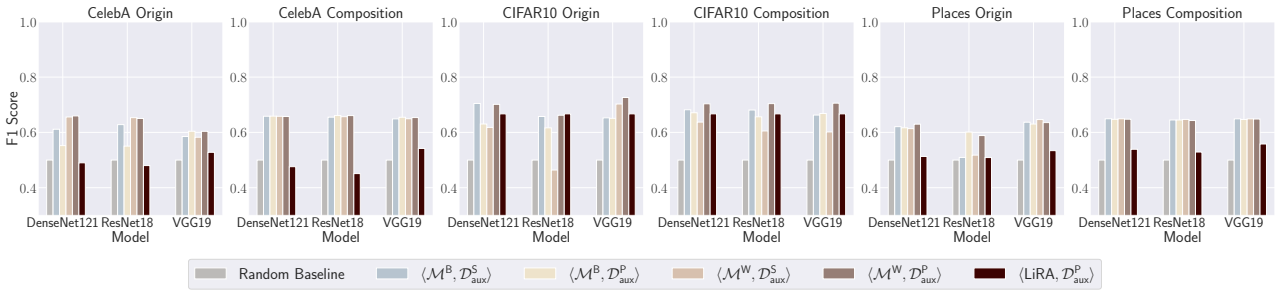


**Figure 10: F1 score of PropInf2MemInf under different threat models, datasets, and target model architectures.**

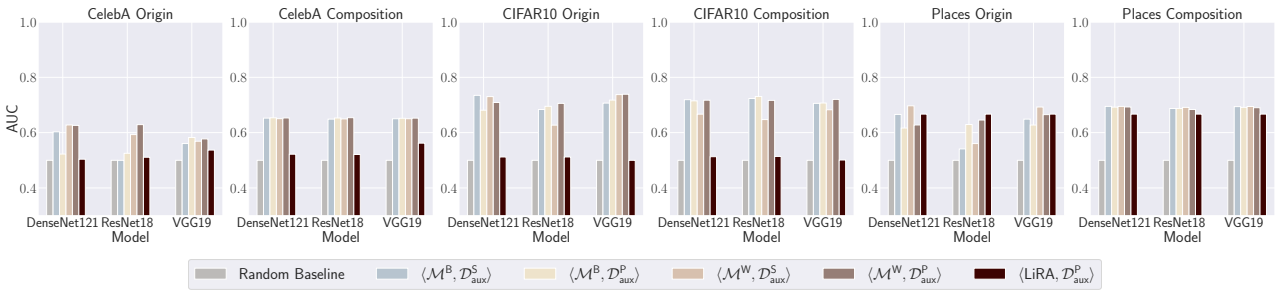


**Figure 11: AUC of PropInf2MemInf under different threat models, datasets, and target model architectures.**

**Table 12: TPR @0.1% FPR of ADV2MemInf with adding DP.**

| | Mode | CelebA | | CIFAR10 | | Places | |
|---|---|---|---|---|---|---|---|
| | | Origin | Composition | Origin | Composition | Origin | Composition |
| $\varepsilon = 10$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.001 | 0.019 | 0.035 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.003 | 0.001 | 0.002 | 0.014 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.002 | 0.001 | 0.001 | 0.016 | 0.002 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.001 | 0.003 | 0.002 | 0.020 | 0.001 | 0.005 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.000 | 0.008 | 0.000 | 0.002 | 0.003 | 0.000 |
| $\varepsilon = 20$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.001 | 0.001 | 0.002 | 0.021 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.003 | 0.002 | 0.006 | 0.009 | 0.002 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.002 | 0.001 | 0.003 | 0.016 | 0.002 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.001 | 0.002 | 0.006 | 0.011 | 0.002 | 0.002 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.002 | 0.006 | 0.001 | 0.001 | 0.004 | 0.005 |
| $\varepsilon = 50$ | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.005 | 0.009 | 0.009 | 0.035 | 0.001 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{B}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.004 | 0.004 | 0.005 | 0.007 | 0.000 | 0.001 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.001 | 0.001 | 0.004 | 0.016 | 0.002 | 0.002 |
| | $\langle \mathcal{M}^{\mathsf{W}}, \mathcal{D}^{\mathsf{P}}_{\text{aux}} \rangle$ | 0.003 | 0.002 | 0.004 | 0.011 | 0.001 | 0.001 |
| | $\langle \mathsf{LiRA}, \mathcal{D}^{\mathsf{S}}_{\text{aux}} \rangle$ | 0.003 | 0.012 | 0.003 | 0.002 | 0.001 | 0.002 |

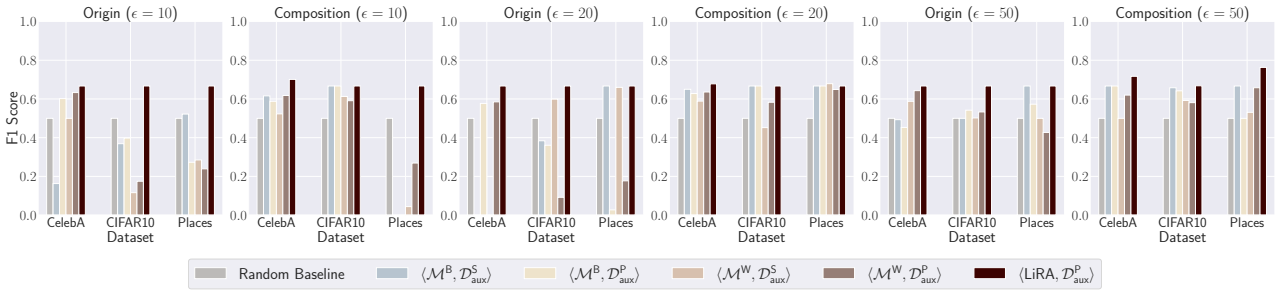


**Figure 12: F1 score of ADV2MemInf under different threat models, datasets, and target model architectures with adding DP.**

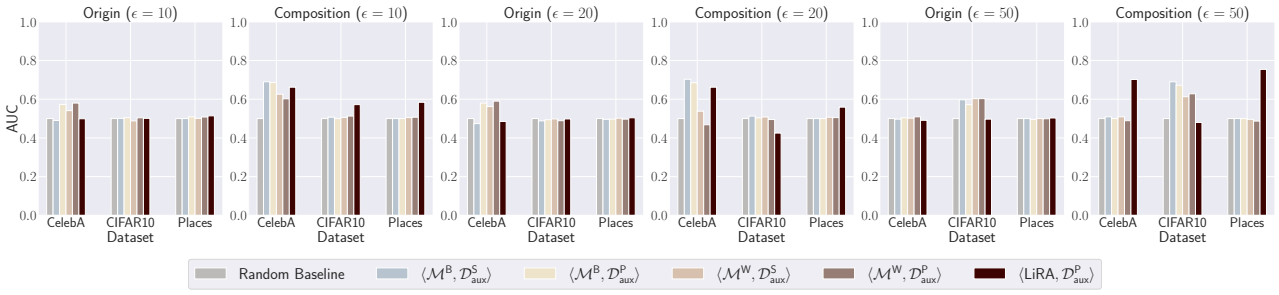


**Figure 13: AUC of ADV2MemInf under different threat models, datasets, and target model architectures with adding DP.**

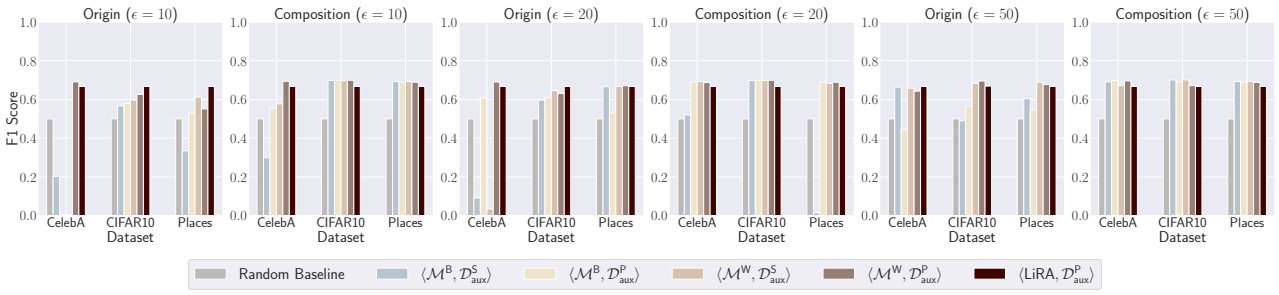


**Figure 14: F1 score of PropInf2MemInf under different threat models, datasets, and target model architectures with adding DP.**

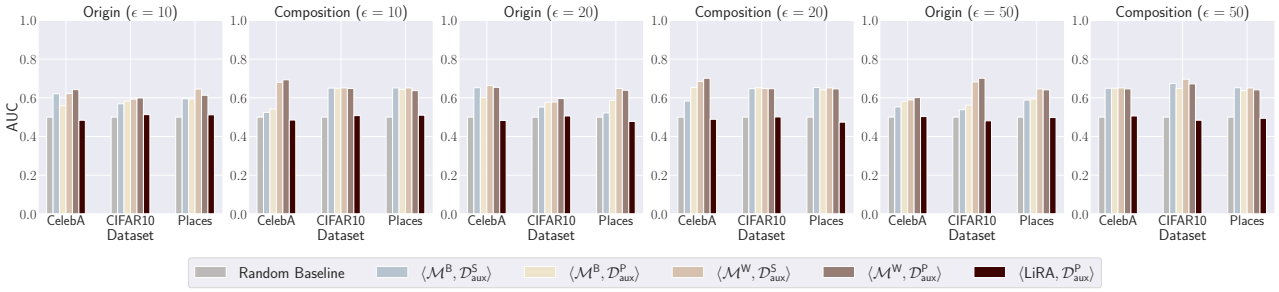


**Figure 15: AUC of** PropInf2MemInf **under different threat models, datasets, and target model architectures with adding DP.**

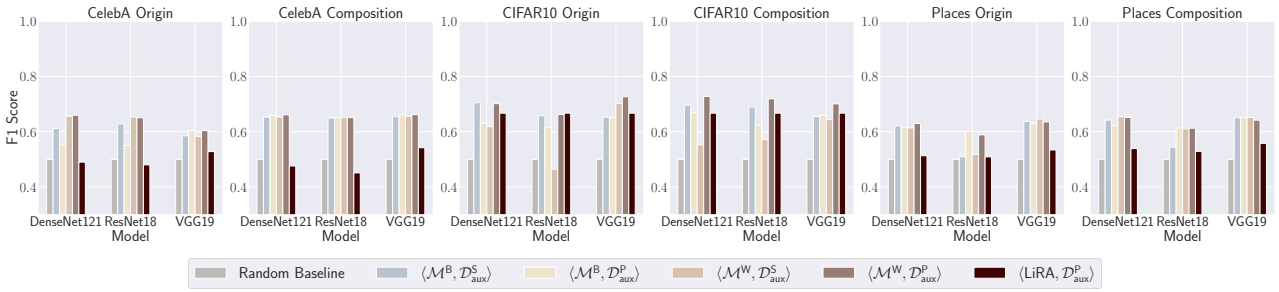


**Figure 16: F1 Score of** ADV2PropInf2MemInf **under different threat models, datasets, and target model architectures.**

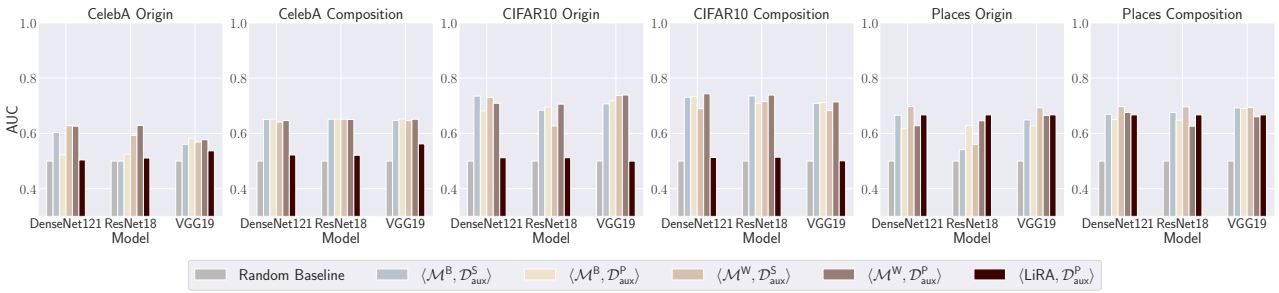


**Figure 17: AUC of** ADV2PropInf2MemInf **under different threat models, datasets, and target model architectures.**