# FACE-AUDITOR: Data Auditing in Facial Recognition Systems

Min Chen[1] Zhikun Zhang[1,2*] Tianhao Wang[3] Michael Backes[1] Yang Zhang[1*]

[1]*CISPA Helmholtz Center for Information Security*
[2]*Stanford University* [3]*University of Virginia*

## Abstract

Few-shot-based facial recognition systems have gained increasing attention due to their scalability and ability to work with a few face images during the model deployment phase. However, the power of facial recognition systems enables entities with moderate resources to canvas the Internet and build well-performed facial recognition models without people's awareness and consent. To prevent the face images from being misused, one straightforward approach is to modify the raw face images before sharing them, which inevitably destroys the semantic information, increases the difficulty of retroactivity, and is still prone to adaptive attacks. Therefore, an auditing method that does not interfere with the facial recognition model's utility and cannot be quickly bypassed is urgently needed.

In this paper, we formulate the auditing process as a user-level membership inference problem and propose a complete toolkit FACE-AUDITOR that can carefully choose the probing set to query the few-shot-based facial recognition model and determine whether any of a user's face images is used in training the model. We further propose to use the similarity scores between the original face images as reference information to improve the auditing performance. Extensive experiments on multiple real-world face image datasets show that FACE-AUDITOR can achieve auditing accuracy of up to 99%. Finally, we show that FACE-AUDITOR is robust in the presence of several perturbation mechanisms to the training images or the target models.[1]

## 1 Introduction

Facial recognition is widely used to perform identification [19, 39, 46, 50]. Modern facial recognition system utilizes machine learning models to determine whether a face image being verified belongs to the authorized users (the complete system also includes other components like face detection [56], liveness detection [36], etc). In the training phase, a facial recognition model takes in multiple images for each user (e.g., from different angles) in advance. In the identification phase, the facial recognition model compares the image being examined with the pre-existing pictures to determine whether it belongs to the authorized users and (if yes) to which authorized user this image belongs. More recently, few-shot learning [16, 18, 22, 43, 45] dominate the traditional learning in facial recognition systems because it requires only *a few* "anchor" face images from the authorized users.

With the power of facial recognition systems, entities with moderate resources can canvas the Internet for face images and build well-performed facial recognition models without people's awareness and consent. For example, clearview.ai reveals that a private company has collected 3 billion online face images and trained a powerful model capable of recognizing millions of citizens. Such kinds of misuse of facial recognition systems are potentially disastrous [34] and infringe the privacy laws such as European Union's General Data Protection Regulation (GDPR). GDPR states that *the personal data must only be processed if the individual has given explicit consent* (Article 6(1)(a)), and *the processing of personal data must be lawful, fair, and transparent* (Article 5(1)(a)) [1]. This means that if the third parties want to use the data owner's face images, they need to obtain consent from the data owner and inform the data owner how their face images are processed. Sharing personal data online typically implies that the data owners are willing to share their data with the public for social or promotion purposes. However, this does not grant others the right to misuse the data for unconsent purposes, particularly in commercial activities.

To prevent face images from being misused, one straightforward method is to modify the raw face images before uploading them to the Internet, such as distorting the face images [27], producing adversarial patches [48], or adding imperceptible pixel-level cloaks [40]. However, these approaches inevitably destroy the semantic information of the face images and also increase the difficulty of retroactivity.

*Corresponding authors.

[1]The source code of our experiments can be found at https://github.com/MinChen00/Face-Auditor.

Also, researchers have argued that such defenses can be bypassed by newer technologies [37], which leads to an endless arms race between the attacker and defender.

## 1.1 Our Contributions

In this paper, we take a different angle by advocating a responsible *auditing* approach that enables normal users to detect whether their private face images are being used to train a facial recognition system. This approach provides users with evidence in claiming proprietary of their face images. Furthermore, it complies with data privacy protection regulations such as GDPR, which gives users the right to know how their data is processed. If data owners do not want any entity to use their face images to train the facial recognition system, they can use FACE-AUDITOR to audit if their face images are being used. If they find their face images were used without their consent, the data owners can take legal action against the model developer in accordance with GDPR regulations.

Concretely, we propose FACE-AUDITOR to determine whether a target user's face images were used to train a facial recognition model. The underlying problem can be formalized as *user-level* membership inference. Different from classic *sample-level* membership inference, which detects whether a specific sample was used for training the target model, user-level membership inference aims to determine whether any of a target user's data was used to train the model. Here, the auditor has a set of samples (images) of a target user, and these samples are not necessarily used to train the target facial recognition model to claim/predict membership.

**Methodology.** We discuss the details of the technical challenges and provide a systematic analysis of how we address each of them in Section 4.5. Briefly, to obtain an auditing model that works broadly, we adopt the well-established yet comprehensive *shadow model paradigm* that aims to mimic the behavior of the target facial recognition model: We assume the auditor can access an auxiliary dataset to train the shadow model. In a more restricted and practical scenario, the auxiliary data does not need to share the same distribution as the target model's training dataset. To achieve the goal of user-level auditing, FACE-AUDITOR accepts a set of target face images as input and outputs a binary indicator of a member user or non-member user. To cope with the few-shot learning paradigm, given the target face images, we supplement a set of anchor face images to form a *probing set*. We then use the probing set to query the target facial recognition system and generate "posteriors" (a sequence of similarity scores from the target face to the anchor faces) as the features for FACE-AUDITOR. Here, the target face image is not necessarily used to train the target model. To further improve the auditing performance, we propose using reference information to strengthen the auditing feature, which can be calculated by comparing the original target face and anchor images.

**Evaluation.** We conduct experiments on three representative few-shot learning algorithms and four human face datasets to illustrate the effectiveness of FACE-AUDITOR. The results show that FACE-AUDITOR can achieve up to 99% auditing accuracy on the SiameseNet model trained on the UMDFaces and VGGFace2 datasets. We observe that when the target model has high representation capability, it is more difficult to audit. Furthermore, we conduct experiments to validate that adding the reference information of the original image can effectively improve the auditing performance. For instance, after introducing reference information to audit the RelationNet model, we achieve 72% accuracy improvement.

**Robustness.** In practice, the target model might be equipped with different obfuscation techniques to preserve the privacy of the training data [5, 21, 27, 40, 48]. Therefore, we conduct experiments to validate the robustness of FACE-AUDITOR when the training images or the target models are protected. Concretely, we consider three representative privacy-preserving mechanisms in a general ML model pipeline: Input perturbation (perturb the training images), training perturbation (perturb the training gradient by enforcing differential privacy), and output perturbation (perturb the output similarity scores). We also show the robustness of FACE-AUDITOR under an adaptive attack setting, where the target model's output is perturbed specifically to evade the auditing from FACE-AUDITOR. We observe that the performance of FACE-AUDITOR only slightly drops, which indicates the robustness of FACE-AUDITOR.

In summary, our contributions are four-fold:

- We take the first step to investigating the auditing approach that enables ordinary users to detect whether their private face images are being used to train a facial recognition system when only similarity metric information is accessible.
- We carefully design the probing set for querying the target facial recognition model and propose using multiple metrics to construct the reference feature to enhance the auditing performance.
- We systematically evaluate the factors that affect the auditing performance and highlight some design oracles for an effective auditor.
- In practice, an advanced adversary might be aware of the existence of the auditor and try to evade the detection of their misuse. Therefore, we investigate the robustness of FACE-AUDITOR when the training images or the target models are protected by different defense mechanisms.

## 2 Related Work

**Privacy of Facial Recognition System.** With the proliferation of facial recognition systems, their privacy issues have attracted increasing attention [9, 47, 54]. To protect users'

privacy, one strategy is to make the face images difficult for a facial recognition system to recognize by relying on adversarial examples [9, 15]. Sharif et al. [41] show that adding specially printed glasses can cause the wearer to be misidentified. Komkov et al. [23] propose to add carefully computed adversarial stickers on a hat to reduce its wearer's likelihood of being recognized. Others propose to add adversarial patches to make it difficult for facial recognition systems to recognize the user as a person in an image [48, 55]. An alternative is to evade the facial recognition models by poisoning their training samples. One representative method is Fawkes [40]. However, these approaches can inevitably destroy the semantic information of the face images and are still vulnerable to advanced adversaries [37].

**Sample-level Membership Inference.** Previous studies on membership inference attacks mainly focus on sample-level membership inference [10, 32, 38, 42, 44]. The first membership inference attack was proposed by Shokri et al. [42], which uses shadow models to mimic the target model's behavior and generate training data for the attack model. Salem et al. [38] gradually removed the assumptions of [42] by proposing three different attack methods. More recently, membership inference has been extensively investigated in various ML models and tasks, such as federated learning [32], natural language processing [44], and neural architecture search [20].

To mitigate the threat of membership inference, a plethora of defense mechanisms have been proposed. These defenses can be classified into three categories: Reducing overfitting, perturbing posteriors, and adversarial training. There are several ways to reduce overfitting in the machine learning field, such as $\ell_2$-regularization [42], dropout [38], and model stacking [38]. In [26], the authors proposed to explicitly reduce the overfitting by adding to the training loss function a regularization term, which is defined as the difference between the output distributions of the training set and the validation set. Jia et al. [21] proposed a posterior perturbation method inspired by adversarial examples. Nasr et al. [35] proposed an adversarial training defense to train a secure target classifier. During the training of the target model, a defender's attack model is trained simultaneously to launch the membership inference attack. The optimization objective of the target model is to reduce the prediction loss while minimizing the membership inference attack accuracy.

**User-level Membership Inference.** Compared to the sample-level membership inference, the user-level inference is less investigated. The first user-level membership inference was proposed by Song et al. [44] for the natural language models, including next-word prediction, neural machine translation, and dialog generation. They design and evaluate a black-box auditing method that can detect, with very few queries to a model, if a particular user's texts were used to train it. Miao et al. [33] then investigate the user-level membership inference against the automatic speech recognition model. Note that Audio-Auditor feeds multiple audios of the target user to the target model independently and obtains multiple transcriptions. The auditing features are constructed by combining the input audio, input transcriptions, output transcriptions, and their statistical information. On the other hand, FACE-AUDITOR needs to carefully design the probing set and combine the similarity scores as the auditing feature. We also introduce image-level similarity as reference information to improve auditing performance. Furthermore, FACE-AUDITOR does not need access to the exact face images used to train the target model; instead, it only needs to take a few new face images of the target user. While Audio-Auditor does not have this property (at least in their experiments).

The most relevant study with FACE-AUDITOR is Li et al. [25]. While we both focus on user-level membership inference against metric learning models and share the same intuition that the images from a member user tend to be closer to each other in the latent space. However, our work differs from Li et al. [25] in multiple aspects. First, the **threat model is different**. Li et al. assume the adversary can access the embeddings of the target model, while FACE-AUDITOR can only obtain the similarity scores, which is more practical in real-world facial recognition systems and more challenging to design the inference/auditing model. Second, the **feature design is different**. Li et al. feed all the face images of the target user to the embedding extractor and use the embedding distances of the input images as the attack feature. On the other hand, FACE-AUDITOR carefully designs the probing set to query the target model and uses the similarity score as the basic auditing feature. We further discover that the raw image similarity can serve as a decisive reference information that significantly increases the auditing performance. Finally, the **application range is different**. FACE-AUDITOR achieves good auditing performance for both simple models such as SiameseNet and complex models such as ProtoNet and RelationNet, while Li et al. only achieve acceptable performance for SiameseNet. We refer the readers to the detailed experiments in Appendix A.

**Attacks for AI System Auditing.** Using attacks against machine learning as an auditing tool has been a growing trend in trustworthy AI [29, 44]. "Desirable attacks" [6] against ML, as an example, can be used for legitimate concerns like human rights and civil liberties. Determining whether a given image is present in a facial recognition database [17] can help individuals determine whether they can bring a court case against the service provider. Model inversion [17] can detect potential bias decision-making in credit risk evaluation systems. Adversarial examples can be used as an obfuscation tool to make users less likely to be tracked [2] or re-identified [40]. A similar notion of "subversive AI" adopts human-centered enhanced adversarial machine learning to evade algorithmic surveillance before publishing content online. Protective Optimization Technologies (POTs) [24] offer a more general terminology for repurposing the original system to enhance privacy, evade discrimination, or avoid surveillance.

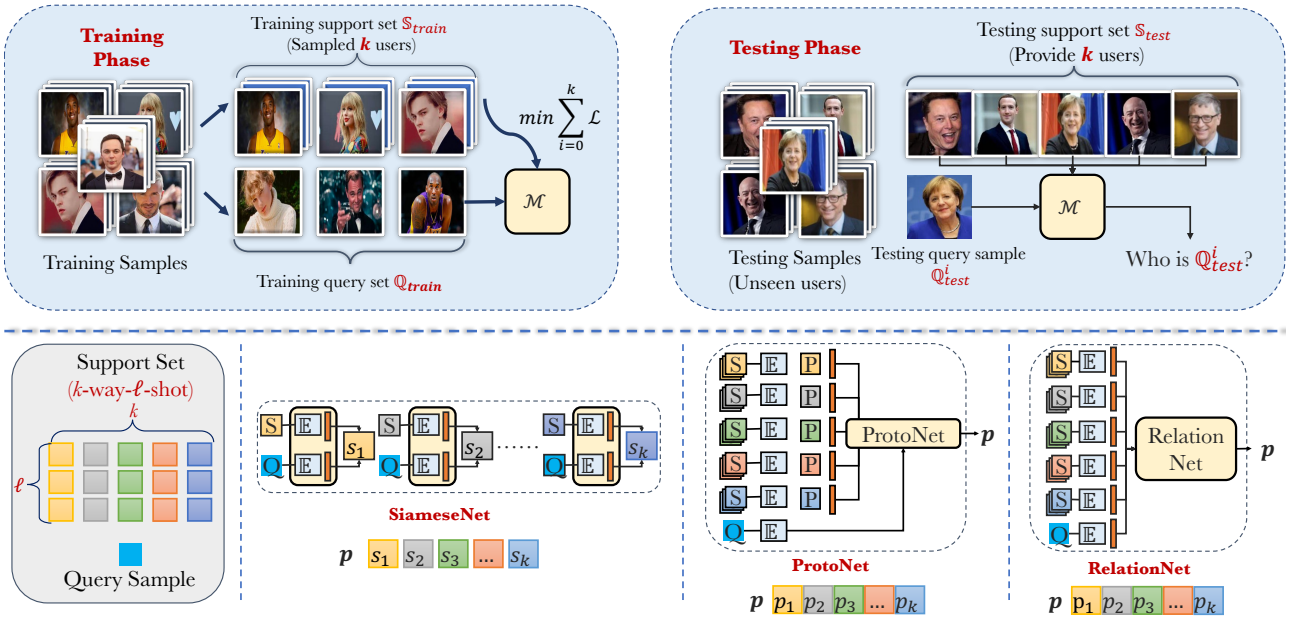



**Figure 1: Illustration of the metric-based few-shot facial recognition models. The algorithm consists of both the training phase and the testing phase. The objective of the training phase is to train a feature extractor $\mathcal{E}$ to make the images in the query set $\mathbb{Q}$ close (in terms of the embeddings) to the images in the support set $\mathbb{S}$ if they come from the same user, and make them far away when they are from different users. In the testing phase, given a query image, we predict its identity label as the closest user in the support set. A SiameseNet takes image pairs as input and outputs a similarity score. Both ProtoNet and RelationNet takes as input the support set simultaneously and outputs a posteriors vector. ProtoNet measures the similarity by Euclidean distance while RelationNet explicitly learns a trainable relation module.**

# 3 Preliminaries

## 3.1 Facial Recognition System

The objective of the facial recognition system is to identify face images. Formally, there is a pre-defined set of persons, which we call *authorized users*. They each contribute multiple face images (which we call *anchor images*) for the system to "memorize" them so that when a new face image comes, the system knows which (if any) authorized user this image corresponds to. A straightforward approach is to train a classification model with the anchor face images. However, the classification model oftentimes requires a large amount of training data, while it is difficult to collect many face images from each authorized user in practice. Furthermore, the set of authorized users often changes over time, for example, when new colleagues join or leave a company. The classification model needs to be retrained when the set of authorized users changes. To address these challenges and improve the scalability of facial recognition systems, companies have turned to using few-shot learning techniques [52].

## 3.2 Few-shot Learning for Facial Recognition

Few-shot learning is a machine learning paradigm that aims to obtain good learning performance given limited supervised information in the training set [12]. The high-level idea of few-shot learning is to exploit prior knowledge to help train, thus reducing the size of the actual training set. An important branch of commonly used few-shot learning algorithms is based on *metric learning*, which learns the similarity/relation (measured by some metric) among the images instead of (in the traditional classification problem) learning the mapping from an image to a specific label.

Figure 1 illustrates the general pipeline of metric-based few-shot learning algorithms, which consists of training and testing phases (also called the deployment phase in facial recognition systems). The training phase takes a large, publicly available training dataset $\mathcal{D}_{train}$ (which consists of samples for many classes) and runs in multiple iterations. In each iteration, we construct a *support set* $\mathbb{S}_{train}$, which consists of randomly selected $k$ classes, each with $\ell$ samples, from $\mathcal{D}_{train}$ (this is referred to as $k$-way-$\ell$-shot few-shot learning). We also construct a *query set* $\mathbb{Q}_{train}$ similar to $\mathbb{S}_{train}$ by sampling from the same classes (note that *query set* might be a bit confusing when used in training, but this is the standard terminology in few-shot learning). Our goal is to train a feature extractor $\mathcal{E}$ so that the features (embeddings) of the images from $\mathbb{S}_{train}$ and $\mathbb{Q}_{train}$ are optimized to be similar/close (in terms of some metric) if they belong to the same class, and dissimilar if they are from different classes. In the testing/deployment phase,

we have a new support set $\mathbb{S}_{test}$ (anchor images from authorized users). Since $\mathcal{E}$ has already been trained to perform well in distinguishing samples from different classes; given one query image, we predict it as the closest class in $\mathbb{S}_{test}$. As $\mathbb{S}_{test}$ is taken as input in the testing case, and we only care about similarities, it is easy to add/remove authorized users in few-shot learning. Different metric-based few-shot algorithms vary in their strategies for making predictions conditioned on the support set. In the following, we introduce several representative metric-based few-shot algorithms [22, 46, 50]:

**Siamese Network (SiameseNet) [28].** The most simple yet commonly used few-shot learning algorithm relies on the Siamese network, which inputs a pair of images and outputs their similarity score. It consists of a feature extractor $\mathcal{E}$ that learns the embedding of each image and a similarity metric (e.g., cosine similarity) that compares any two embeddings. The objective is to train $\mathcal{E}$ so that the image pairs with the same label (positive pairs) have high similarity scores (in the embedding space), and image pairs with different labels (negative pairs) have low similarity scores.

As the SiameseNet is designed to learn the similarity between two images, it can be easily adapted to deal with the few-shot learning tasks. Concretely, in the training phase, we pair the images from the support set $\mathbb{S}_{train}$ and the query set $\mathbb{Q}_{train}$ one by one. If the query image and the support image come from the same user, they form a positive pair; otherwise, they form a negative pair. In the testing phase, given a query image $\mathbb{Q}^i_{test}$, we compare it with all the images in the testing support set $\mathbb{S}_{test}$. If the largest similarity score exceeds a predefined threshold, the query image belongs to the corresponding user; otherwise, the target image does not belong to any user.

**Prototypical Network (ProtoNet) [43].** ProtoNet is specially designed for few-shot learning tasks. It also contains a feature extractor $\mathcal{E}$ that transforms the images into embeddings. Different from SiameseNet which takes pairs, ProtoNet takes all the samples from the support set $\mathbb{S}_{train}$ simultaneously and compares the similarity between the query images in $\mathbb{Q}_{train}$ and the support images in $\mathbb{S}_{train}$. For each class in the support set $\mathbb{S}_{train}$, we calculate the mean of the embeddings and generate a "prototype". The objective is to train $\mathcal{E}$ to make the images in the query set $\mathbb{Q}_{train}$ close to the prototype with the same user and far from the prototypes with different users. The distance between the query embedding and the prototype is measured by *Euclidean distance*.

**Relation Network (RelationNet) [45].** RelationNet shares a similar paradigm with ProtoNet, which consists of a feature extractor $\mathcal{E}$ to transform the support set $\mathbb{S}$ into prototypes and the query set $\mathbb{Q}$ into image embeddings. It also aims to make the images in the query set $\mathbb{Q}_{train}$ close to the prototype from the same user, and far from the prototypes from different users. The main difference from ProtoNet is that, instead of using Euclidean distance to measure the distance between query embedding and the prototype, it explicitly learns a trainable relation module, which typically consists of multiple stacked fully connected layers.

# 4 Auditing Methodology

## 4.1 Problem Statement

**Auditing Goal.** We aim to determine whether any of the *target user $u$*'s face images were used to train a target facial recognition model $\mathcal{M}_T$ (*target model* for short). We formulate this auditing process as a *user-level* membership inference problem. Formally, assume the target user $u$ has a set of face images $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$, the user-level membership inference aims to distinguish between $\mathcal{U} \cap \mathcal{D}^T_{train} \neq \varnothing$ (member user) and $\mathcal{U} \cap \mathcal{D}^T_{train} = \varnothing$ (non-member user), where $\mathcal{D}^T_{train}$ is the training dataset of $\mathcal{M}_T$. This is different from the classic *sample-level* membership inference that aims to determine whether a specific face image was used to train the target model, i.e., $u_i \in \mathcal{D}^T_{train}$ (member sample) or $u_i \notin \mathcal{D}^T_{train}$ (non-member sample).

**Auditing Scenario.** Facial recognition systems are often trained by computer vision companies and sold to individual users or other companies for deployment. The model developer of the facial recognition system might collect face images from the Internet and misuse these face images without the data owners' consent. Users who want to audit potential misuse of their face images could use FACE-AUDITOR as a privacy-auditing tool.

Note that FACE-AUDITOR is unnecessary to be trained by individuals. Alternatively, a third party with legal access to facial images (such as a qualified Auditing-ML-as-a-Service company, law enforcement, or government agency) can purchase the well-known facial recognition systems in the market and provides (free or charged) auditing services to individuals. By doing so, the third-party entity can ensure auditing accuracy and efficiency, making it more convenient for users who want to audit their face images. Individuals can quickly check if their face images are being used without their consent and take appropriate actions, such as reporting to the authorities or suing the model developer per the protection of privacy regulations [1, 3, 4].

**Auditor's Capabilities.** The auditor has a basic knowledge of the facial recognition model, such as metric scores, input format, etc. To mimic the real-world application, we consider the most challenging setting where the auditor only has *black-box* access to the target model. We assume the auditor can obtain an auxiliary face image dataset. Note that the auxiliary dataset does not need to contain face images from the same set of users or even the same distribution as the target model; thus, the auditor can utilize some online public datasets to build FACE-AUDITOR, which is practical in real-world applications. In the auditing phase, the auditor does not need access to the

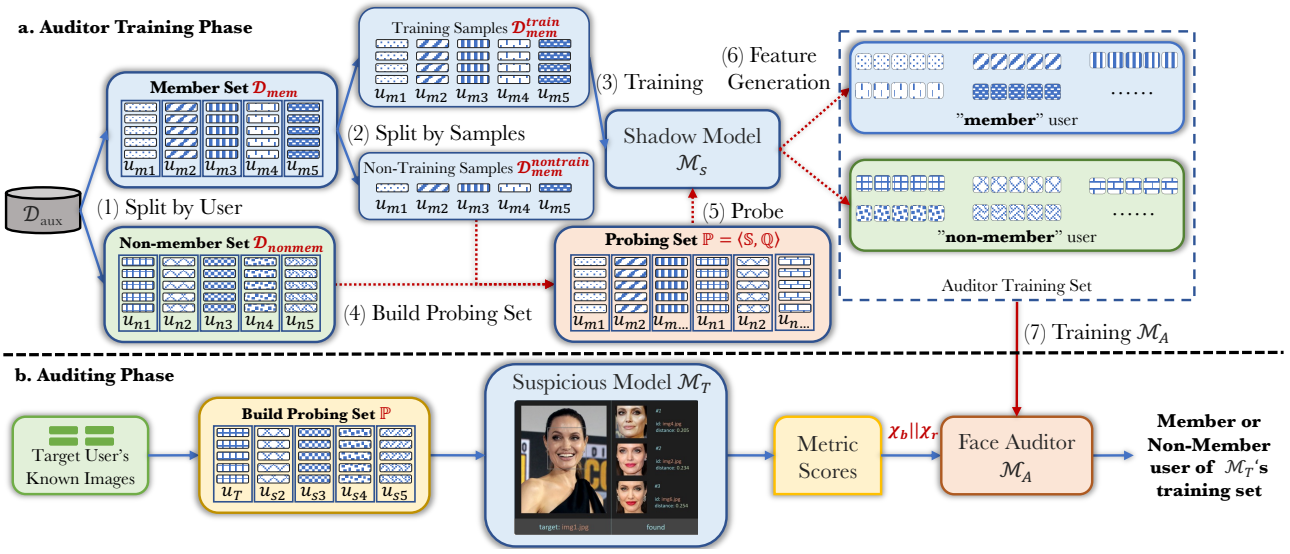



**Figure 2: Overview of FACE-AUDITOR. There are two phases, training and auditing. The *training phase* is composed of seven steps: (1) The auditor splits its auxiliary dataset into the disjoint member and nonmember set by the user. (2) The member set is further split into training and non-training samples. (3) The training samples are used to train a shadow model. (4) The non-training samples and testing set are to form a probing set. (5) We use the probing set to query the shadow model and collect the model outputs. (6) The outputs are labeled by member or non-member, depending on whether the input is from $\mathcal{D}_{mem}^{nontrain}$ or $\mathcal{D}_{nonmem}$. (7) We train a supervised binary classifier as our auditing model. In *auditing phase*, the auditor builds a probing set with known images from the target users (users to be audited) and then queries a suspicious facial recognition model $\mathcal{M}_T$ and collects the corresponding outputs (i.e., similarity scores) as the auditing feature. Feeding these feature vectors into FACE-AUDITOR, the auditor gives a prediction of member or non-member user.**

specific face images used to train the target model; instead, it only needs to take a few available face images of the target user. Furthermore, the auditor can design their own support set (legitimate users) and query set to audit the target model.

## 4.2 Overview

Figure 2 illustrates the overall workflow of FACE-AUDITOR. Generally, there are two phases, *auditor training* and *target user auditing*. The auditor training phase aims to train a binary classifier that can distinguish between member users and non-member users. The general idea is to use the auxiliary dataset $\mathcal{D}_{aux}$ to train a *shadow model* that mimics the behavior of the target model. We then design a *probing set* (consisting of support set and query set) to query the shadow model and generate a set of *similarity scores* (between support set and query set), which serves as features to train the auditor model $\mathcal{M}_A$. While most of the existing studies on membership inference are based on the shadow model paradigm [33,38,42,44], the main challenge lies in constructing the attack/audit features for the attack model. For the sample-level membership inference against classification models, the attack features are constructed by feeding the target samples to the target model independently and using the output posteriors as attack features. On the other hand, in the user-level few-shot setting, the auditor *does not have the exact images* that are used to train the target model. Thus, we need to carefully design a probing set to query the target model and combine the similarity scores as audit features.

In the auditing phase, the auditor collects a set of new face images from the target user and builds a probing set to query the target model. The auditor then collects the similarity scores returned by the target model as the auditing features and feeds them to $\mathcal{M}_A$, which gives a prediction of the member or non-member user.

## 4.3 Auditor Training Phase

**Training the Shadow Model.** Assume the auxiliary dataset $\mathcal{D}_{aux}$ contains face images of $U$ users, and each user has $I$ face images. We first split $\mathcal{D}_{aux}$ into two disjoint datasets by users, namely member dataset $\mathcal{D}_{mem}$ and nonmember dataset $\mathcal{D}_{nonmem}$. Recall that, for member users, FACE-AUDITOR does not need to have access to the specific images used to train the auditor model; thus, for the member dataset $\mathcal{D}_{mem}$, we further split it (by sample) into two disjoint parts, $\mathcal{D}_{mem}^{train}$ and $\mathcal{D}_{mem}^{nontrain}$. We use $\mathcal{D}_{mem}^{train}$ to train the shadow model and use $\mathcal{D}_{mem}^{nontrain}$ and $\mathcal{D}_{nonmem}$to construct the probing set. To be more clear, all users in $\mathcal{D}_{mem}$ are the member users, while images in $\mathcal{D}_{mem}^{train}$ are member samples, and images in $\mathcal{D}_{mem}^{nontrain}$ are non-member samples. We follow the procedure described in Section 3.2 to construct the support and query set to train

the shadow model $\mathcal{M}_s$.

**Constructing the Probing Dataset.** Unlike classical classification models that take a single image as input and output posteriors, few-shot learning models require a support set $\mathbb{S}$ and a query set $\mathbb{Q}$ as input and output a sequence of similarity scores (as described in Section 3.2). Consequently, generating an auditing feature for few-shot learning is more complex than traditional membership inference attacks against classification models. To improve auditing performance, the auditor must carefully design the support set $\mathbb{S}$ and query set $\mathbb{Q}$, rather than directly feeding the training and testing datasets to the shadow model $\mathcal{M}_s$ to obtain posteriors. For ease of presentation, we call the combination of the support set and query set as probing set $\mathbb{P} = \langle \mathbb{S}, \mathbb{Q} \rangle$.

Since the architecture of SiameseNet is slightly different from ProtoNet and RelationNet, we need to design different probing sets for them.

- **SiameseNet.** As discussed in Section 3.2, the SiameseNet model processes the support set separately, which leads to its probing set consisting of a 1-way-$\ell$-shot support set and multiple query images. For each probe, we set both the support set and the query images from the same target user (the user to be audited, who may be a member from $\mathcal{D}_{mem}^{nontrain}$ or a non-member from $\mathcal{D}_{nonmem}$).
- **ProtoNet & RelationNet.** Unlike SiameseNet, both ProtoNet and RelationNet takes the support set and query images together as input, forming a $k$-way-$\ell$-shot support set. We assign the first class of the support set to the target user and select the query images from that user. The remaining classes in the support set can be selected from any user, as the similarity scores between these classes and the query images are not used to generate the auditing features.

**Generating the Auditing Feature.** We use the *similarity scores* between the query image and the support set returned by the shadow model as the *basic auditing feature* $\chi_b$. We use $q$ images in the query set $\mathbb{Q}$, resulting in an auditing feature vector of length $q$.

To further improve the auditing performance, we consider using the image-level similarity between the query image and the support set as additional reference information, referred to as *reference auditing feature* $\chi_r$. In summary, the auditing feature $\chi$ is a concatenation of the basic auditing feature and the reference auditing feature, i.e., $\chi = \chi_b || \chi_r$. In this paper, we consider three types of image-level similarity metrics: Directly compare the similarity between image pixels (MSE and CosSim), compare the structural similarity between images (SSIM), and use a deep neural network to compare (LPIPS). Denote the pixel matrix of two images as $X$ and $Y$, and four metrics can be described as follows.

- **MSE (Mean Square Error).** We first represent the image pair as two pixel vectors $X$ and $Y$, the MSE of these two images is calculated as $\frac{1}{N}\sum_{i=1}^{N}(X_i - Y_i)^2$, where $N$ is the total number of pixels. A smaller MSE indicates higher similarity.
- **CosSim (Cosine Similarity).** For two pixel vectors $X$ and $Y$, the CosSim is calculated as $\frac{X \cdot Y}{||X|| ||Y||}$, where $\cdot$ represents an inner product of two vectors and $||\cdot||$ presents the cardinality of a vector. The values of CosSim are in the range of [-1, 1]. A larger CosSim indicates higher similarity.
- **SSIM (Structural Similarity Index Measure) [53].** It compares two images by considering luminance, contrast, and structure. Formally, $SSIM(X,Y) = \ell(X,Y)^{\alpha} \cdot c(X,Y)^{\beta} \cdot s(X,Y)^{\gamma}$, where $\ell(\cdot)$, $c(\cdot)$, $s(\cdot)$ represent luminance, contrast, structure respectively, and $\alpha, \beta, \gamma$ are weight parameters. A larger SSIM value indicates higher similarity.
- **LPIPS (Learned Perceptual Image Patch Similarity) [60].** The general idea is to use a pretrained convolutional model to transform the two images $X$ and $Y$ into embeddings, normalize the activations in the channel dimension, and take the $\ell_2$ distance. We then average across spatial dimensions and across all layers. A larger LPIPS indicates higher similarity.

We conduct empirical experiments in Section 5.3 to show that the reference information can effectively improve the auditing performance, and cosine similarity achieves relatively better performance in most settings.

**Training the Auditing Model.** For all the few-shot learning models, we use $\mathcal{D}_{train}^{nonmem}$ and $\mathcal{D}_{test}$ to construct the probing set for member users and non-member users, respectively. We use a three-layer multi-layer perceptron (MLP) with 100 hidden neurons as the auditing model.

## 4.4 Auditing Phase

To determine whether a target user's face images are used to train the target model, the auditor only needs to take multiple face images from the target user. Note that these face images are not necessarily used to train the target model. The auditor then uses the same strategy as the training phase to construct the probing set $\mathbb{P}$ and generate the auditing feature $\chi$. Finally, the auditing feature is fed to the auditing model to determine the membership status of the target user.

## 4.5 Discussion

Here we highlight the technical challenges of FACE-AUDITOR and discuss how we address them in this paper.

**Mapping Behaviors Differs in Few-shot-based Facial Recognition Models.** In traditional ML models, the inputs and outputs are directly mapped. The model outputs either posteriors to known classes or corresponding labels, which enriches the information for a successful membership inference. However, few-shot-based facial recognition models do not

directly map the training data into the corresponding labels (users); instead, they only learn a similarity metric between images. Even though a class (user) is not seen in the training phase, a facial recognition model could generate an accurate similarity score due to the structural uniqueness of the human face. Therefore, we cannot determine the membership status of the target users by seeing if the target model can recognize them. To solve this challenge, FACE-AUDITOR relies on the fact that images of a member user tend to have higher similarity scores than those of a non-member. To construct the auditing feature, we concatenate the similarity scores of multiple shots and fix one way in the support set as the target user, which maximizes the similarity of the images from the same user and can be easily implemented in the $k$-way-$\ell$-shot input manner.

**Black-box Auditing under Domain Shift.** This paper identifies a more practical but strict scenario in which the auditor does not know the underlying training data distribution, which brings more challenges to the shadow model paradigm. On the one hand, the auditor cannot train a perfect model to mimic the behavior of a targeting facial recognition model. On the other hand, a black-box auditor can only design the query set to interact with the target model and maximize the difference between member and non-member users. We trained FACE-AUDITOR slightly differently from the previous shadow model paradigm to solve the domain shift problem. Concretely, we do not limit our auditor model to the known images but collect unknown images from the training users of the shadow model, increasing the generalization ability of FACE-AUDITOR when disjoint users exist.

**Well-generalized Models are More Difficult to Audit than the Overfitted Ones.** A well-generalized model often has a low overfitting level and can behave similarly well on unseen samples from both member and non-member users. It is more favorable in practice but more challenging to unify a metric to differentiate between member and non-member users. As a result, the typical overfitting intuition that guides a successful membership inference attack in the classification model does not apply to few-shot learning settings. As shown in Figure 4, the overfitting level is low in three few-shot facial recognition models. Thanks to the reference auditing feature, we can build a ground for the anchor images and gradually compare the difference between member and non-member users. We empirically validate the contribution of reference information in Section 5.3.

# 5 Evaluation

In this section, we first describe the experimental setup in Section 5.1 and evaluate the overall auditing performance in Section 5.2. We then validate the effectiveness of the reference information and investigate the effectiveness of different image-level similarity metrics in Section 5.3. Finally, we show the transferability of FACE-AUDITOR in Section 5.4. We further evaluate the impact of different hyperparameters on the auditing performance in Appendix B and investigate the robustness of FACE-AUDITOR when four defense mechanisms are introduced to protect the training images or the target models in Appendix C.

## 5.1 Experimental Setup

**Face Datasets.** We perform experiments on four widely used real-world face image datasets: UMDFaces [7], WebFace [58], VGGFace2 [8], and CelebA [30]. The details of the dataset are as follows.

- **UMDFaces [7].** The original dataset contains 367,888 face images for 8,277 identities. The labels of all the face images are annotated either by human annotators or deep neural networks. The number of samples for each identity varies from 7 to 203.
- **WebFace [58].** This is a human face dataset that contains 494,414 face images of 10,575 identities collected from the IMDb website. The number of samples for each identity varies from 2 to 769.
- **VGGFace2 [8].** It is a human face dataset containing 3.31 million images of 9131 identities, which are collected from the Google image search engine. These face images show large variations in pose (yaw, pitch, and roll), age, race, lighting, and background. The number of samples for each identity varies from 87 to 825.
- **CelebA [30].** The original dataset contains 202,599 images of 10,177 identities. The number of samples for each identity varies from 1 to 35.

Since the number of face images for all users is highly unbalanced, to make the experimental results comparable, we filter out the users with a number of images less than 100 (except for CelebA). For the users having more than 100 images, we randomly sample 100 images for them. We resize all images to $96 \times 96$ and evaluate the performance of both the target model and the auditing model. Under the setting of Figure 2, which indicates half of the images from 40% users are used to train the shadow/target models, we randomly select 10% images from the 40% to generate member labels and 10% testing images for generating non-member labels, they use the data to train and evaluate the performance of FACE-AUDITOR. We summarize the dataset split in Table 1.

**Target Models.** We experiment on three facial recognition system architectures as introduced in Section 3.2, all with the default configurations.

- **SiameseNet.** Following the setting of [22], we implement the SiameseNet with a four-convolution-layer feature extractor with a ReLU and Max-Pooling for each convolution

**Table 1: Dataset split in detail. The dataset was split into two halves for the shadow model and target model, with users being divided equally. We allocated 80% of the users for $\mathcal{D}_{mem}$ and the remaining 20% for $\mathcal{D}_{nonmem}$. Within the training set, each user's images were split into two equal parts. One part (50% as $\mathcal{D}_{mem}^{train}$) was used to train the shadow/target model, while the other part (50% as $\mathcal{D}_{mem}^{nontrain}$) was used to generate the member labels. This split ensured sufficient training data for a well-performed shadow/target model. We keep member and nonmember labels balanced for a fair and accurate evaluation of FACE-AUDITOR.**

| Dataset after Prepossessing | | | Target/Shadow Model | | Auditing Model | |
|---|---|---|---|---|---|---|
| Dataset ($\mathcal{D}$) | #. Users ($U$) | #. Images per User ($I$) | #. Training Images $(40\%*U)*(50\%*I)$ | #. Testing Images $(10\%*U)*(50\%*I)$ | #. Training Images $(10\%*U)*(50\%*I)$ | #. Testing Images $(10\%*U)*(50\%*I)$ |
| UMDFaces | 200 | 100 | 4,000 | 1,000 | 1,000 | 1,000 |
| Webface | 827 | 100 | 16,520 | 4,130 | 4,130 | 4,130 |
| VggFace2 | 5,257 | 100 | 105,140 | 26,285 | 26,285 | 26,285 |
| CelebA | 6,348 | 20 | 25,392 | 6,348 | 6,348 | 6,348 |

layer to learn complex patterns in the data. The target model is trained with BCE loss and Adam optimizer.

- **ProtoNet.** Following the setting of [43], we implement the ProtoNet with a four-convolution-layer feature extractor with batch normalization and ReLU activation function for each convolution layer. The target model is trained with cross-entropy loss and an SGD optimizer with a step scheduler.
- **RelationNet.** Following the setting of [45], we implement the RelationNet with a four-convolution-layer feature extractor and a two-convolution-layer relation network. The feature extractor and the RelationNet are trained with Adam optimizer with a step scheduler. We use MSE loss to train the metric parameters of RelationNet.

**Metrics.** We use the following four metrics to evaluate the performance of FACE-AUDITOR.

- **Accuracy.** We use accuracy to measure the auditing success rate. Concretely, accuracy measures the correctly predicted probing sets to the total probing sets. Higher accuracy means better performance.
- **AUC.** For a binary classification model (our attack model), AUC (the Area Under the Curve) is the measure of the ability of a classifier to distinguish between classes when the decision threshold varies. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes. AUC equals 1 indicates perfect prediction, while 0.5 indicates random guessing.
- **F1 Score.** F1 Score is a harmonic mean of *precision* (the proportion of true positive cases to the member classes) and *recall* (the proportion of true positive cases to all correctly predicted classes), which can provide a better measure of the incorrectly classified cases than the accuracy metric. A higher F1 Score indicates better auditing performance.
- **False Positive Rate (FPR).** The False Positive Rate (FPR) evaluates the proportion of incorrect ownership claims to the total cases. In practice, a higher false positive rate may degrade the credibility of FACE-AUDITOR and cause unnecessary lawsuits. In our case, a lower FPR indicates better auditing performance.

**Experimental Settings.** Following the classical setting of shadow model-based membership inference [33, 38, 42, 44], we equally split each dataset by users into two disjoint parts, the target set $\mathcal{D}_T$ and auxiliary set $\mathcal{D}_{aux}$. We then split both the target set $\mathcal{D}_T$ and the auxiliary set $\mathcal{D}_{aux}$ as in Section 4.3. We train the auditing model on $\mathcal{D}_{aux}$ and evaluate the auditing model on $\mathcal{D}_T$. We evaluate 5-way-5-shot with 5 queries by default and explore the impacts of different parameters in Appendix B.

**Implementation.** We implement all the target models and the auditing model with Python 3.7 and PyTorch 1.7. All experiments are run on an NVIDIA DGX-A100 server with 2 TB memory and Ubuntu 18.04 LTS system. All the experiments are run 10 times with mean and standard deviation reported.

## 5.2 Overall Auditing Performance

**Target Model Performance.** We first investigate the performance of the target models. Table 2 illustrates the training accuracy, the testing accuracy, and the overfitting (accuracy gap between training and testing datasets) of three target models trained on four face image datasets. We first observe that the overfitting level varies across different models but keeps low in most settings. Besides, the RelationNet achieves the best testing accuracy, indicating that RelationNet has the best representation power.

**Auditing Performance.** We then evaluate the overall auditing performance of FACE-AUDITOR. We conduct experiments on three target models trained on four face image datasets and report the auditing performance with four metrics in Figure 3.

In general, we observe that FACE-AUDITOR achieves good auditing performance for all the target models and datasets. For instance, SiameseNet, ProtoNet, and RelationNet trained on the UMDFaces dataset achieve up to 1.0, 0.80, and 0.85 auditing accuracy, respectively. We further observe that the auditing performance varies on three different target models. We achieve the best auditing performance on SiameseNet and the worst on ProtoNet. This is due to the different memorization power of the target models. The memorization power of different models can be explained by the fact that member users' similarity between face images is optimized in the

**Table 2: Target model performance. Higher test accuracy means better representation power and higher overfitting indicates a worse generalization ability of the target model.**

| Dataset | UmdFaces | | | WebFace | | | VGGFace2 | | | CelebA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mathcal{M}_{Target}$ | $\mathcal{M}_{SiameseNet}$ | $\mathcal{M}_{ProtoNet}$ | $\mathcal{M}_{RelationNet}$ | $\mathcal{M}_{SiameseNet}$ | $\mathcal{M}_{ProtoNet}$ | $\mathcal{M}_{RelationNet}$ | $\mathcal{M}_{SiameseNet}$ | $\mathcal{M}_{ProtoNet}$ | $\mathcal{M}_{RelationNet}$ | $\mathcal{M}_{SiameseNet}$ | $\mathcal{M}_{ProtoNet}$ | $\mathcal{M}_{RelationNet}$ |
| **Train Acc.** | 0.775 | 0.960 | 1.000 | 0.650 | 0.748 | 0.800 | 0.818 | 0.951 | 1.000 | 0.647 | 0.818 | 0.940 |
| **Test Acc.** | 0.500 | 0.794 | **0.852** | 0.460 | 0.670 | **0.757** | 0.767 | 0.868 | **0.943** | 0.603 | 0.802 | **0.867** |
| **Overfitting** | **0.275** | 0.166 | 0.148 | **0.190** | 0.078 | 0.043 | 0.051 | **0.083** | 0.057 | 0.044 | 0.016 | **0.073** |

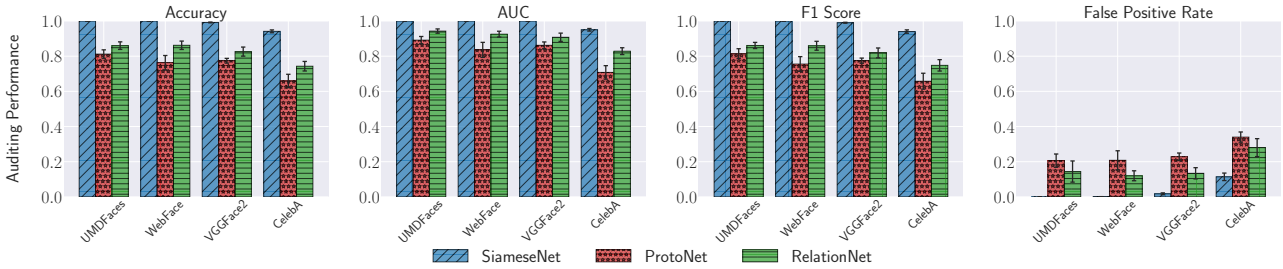
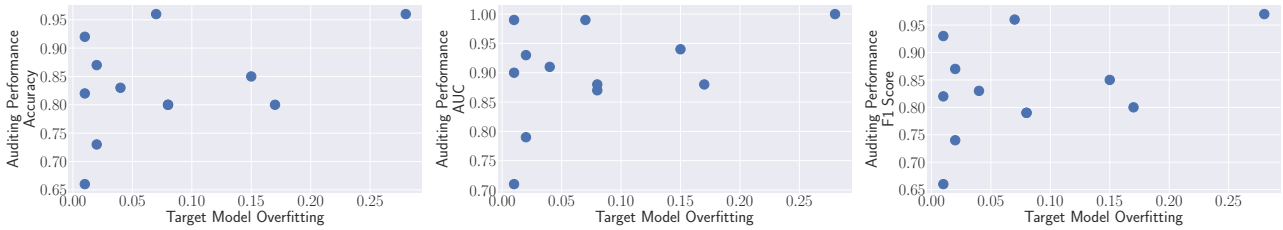



**Figure 3: Overall auditing performance for four evaluation metrics. We evaluate three model architectures grouped by dataset. We list the auditing performance over four different evaluation metrics in each subfigure.**




**(a) Accuracy** **(b) AUC** **(c) F1 Score**

**Figure 4: Relation between target model overfitting and auditing performance. Twelve dots in each subfigure represent the combination of three target model architectures and four datasets. The Pearson correlation values between auditing performance (for accuracy, AUC, and F1 Score) and overfitting level are** 0.412**,** 0.406**, and** 0.412**, respectively.**

training process. SiameseNet has the highest memorization power since images of the member users are separately optimized in the training process. In contrast, that of ProtoNet and RelationNet are optimized together with other classes. Comparing ProtoNet and RelationNet, since RelationNet uses a trainable relation module to compute the similarity scores while ProtoNet directly computes the Euclidean distance; thus RelationNet has higher memorization power than ProtoNet.

Comparing different datasets, we observe the best auditing performance on UMDFaces and the worst on CelebA. This is because UMDFaces has the least users (i.e., 200 users in our experiment), and CelebA has the most users (i.e., 6348 users in our experiment). Besides, CelebA only contains 20 images for each user; the samples used to represent a user are much fewer than the other three datasets, thus further increasing the challenge for auditing.

While we observe that **it is easier to infer a target user's membership status when there are fewer users and each user has more samples in the dataset**. In practice, it is unnecessary to train few-shot learning-based facial recognition models on face datasets of more than $3k$ users since the objective of the few-shot learning is to learn the similarity information between classes, and $3k$ users are enough for learning a few-shot learning model.

**Impact of Overfitting.** Previous studies have shown that overfitting plays a crucial role in launching a successful membership inference [38, 57]. To investigate the impact of overfitting, we provide a scatter plot showing the relation between the overfitting and the auditing performance in Figure 4. We observe that higher overfitting indeed leads to better auditing performance. Unlike classical sample-level membership inference requiring relatively high overfitting to achieve satisfying inference performance, **FACE-AUDITOR can achieve good auditing performance even when the overfitting level is low**. For instance, when the overfitting level is 0.02, FACE-AUDITOR can achieve 0.93 auditing accuracy. On the other hand, the classical sample-level membership inference can only achieve 0.6 accuracy when the overfitting is 0.02 (see Figure 2 in [38]).

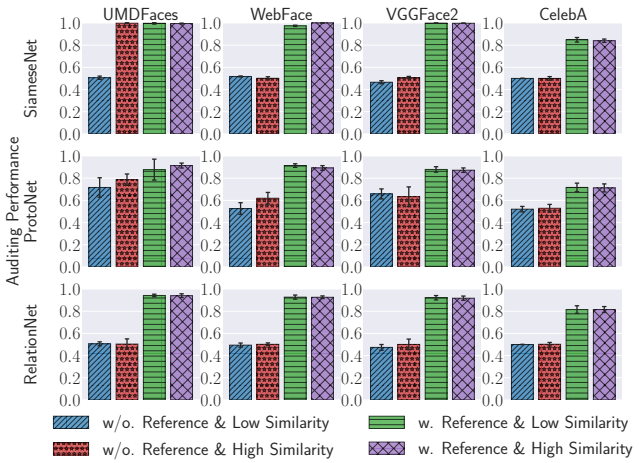



**Figure 5: Impact of the reference information and the similarity selection. We use AUC to measure auditing performance and put the performance evaluated on other metrics in [11].**

## 5.3 Effectiveness of Reference Information

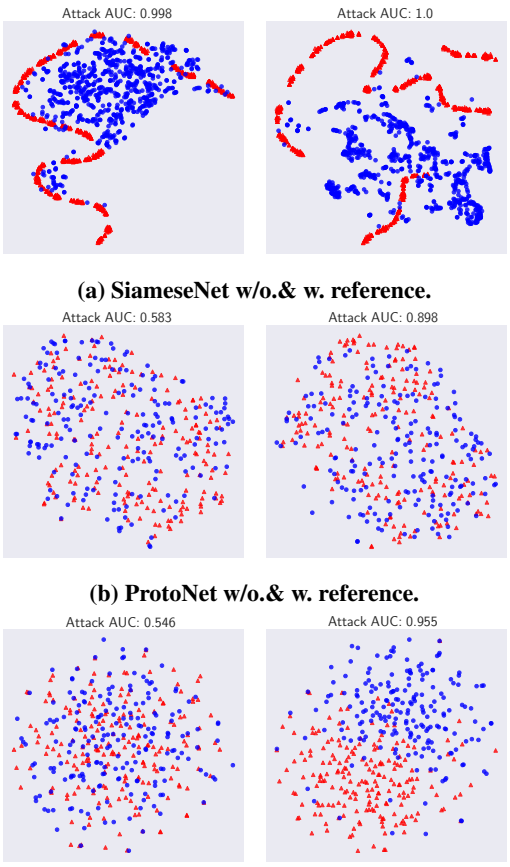
As discussed in Section 4, the reference information helps to improve auditing performance. In this subsection, we first validate the effectiveness of the reference information, then investigate the impact of different similarity metrics. We use AUC to measure auditing performance and put the performance evaluated on other metrics in [11].

**Effectiveness.** We conduct experiments on four face image datasets and three target models to validate the effectiveness of the reference information. The experimental results in Figure 5 illustrate that **exploiting reference information can significantly improve the auditing performance in most of the settings** (by comparing the "w/o." and "w." bars).

We further explore why the reference information can improve the auditing performance using a t-SNE plot in Figure 6. Specifically, by comparing Figure 6c left and right subfigures, we observe that the member and non-member are much further from each other after exploiting the reference information. We also observe different effects of reference information on the three target models. We suspect the reason is that the improvement level by reference information is positively correlated to the memorization power of different models, and ProtoNet has the lowest memorization power in terms of user-level membership inference as discussed in Section 5.2.

**Impact of Similarity.** Given the reference information of the raw face images, another question is whether to choose query images with high similarity or low similarity to the support set. To answer this, we compare the auditing performance when choosing the five highest-similarity query images and the five lowest-similarity query images from the testing dataset. The experimental results are shown in Figure 5.

By comparing the "Low Similarity" and "High Similarity" bars, we observe that the original similarity between the




**(a) SiameseNet w/o.& w. reference.**

**(b) ProtoNet w/o.& w. reference.**

**(c) RelationNet w/o.& w. reference.**

**Figure 6: T-SNE visualization on the impact of reference information. Each red triangle is a member sample, and each blue circle is a non-member sample of the UMDFaces dataset.**

query images and the support set only slightly impacts the auditing performance on ProtoNet and RelationNet. When auditing the SiameseNet model, high similarity pairs can enhance the auditing performance. Take the SiameseNet trained on CelebA as an example, the "Low Similarity" query images can achieve 0.758 auditing accuracy, while the "High Similarity" query images can achieve 0.858 auditing accuracy.

In summary, randomly choosing query images from the testing dataset when constructing the probing set can make the auditing model work well in most cases, but **to achieve the best auditing performance, "High Similarity" images are recommended.**

**Choice of Similarity Metrics.** We can adopt multiple metrics to measure the similarity between the target image and the support set as discussed in Section 4.3. Figure 7 illustrates the auditing performance when using different similarity metrics to generate the reference information. We first observe that all four metrics can achieve relatively high auditing performance on SiameseNet. Regarding the auditing performance on ProtoNet and RelationNet, the performance variance increases

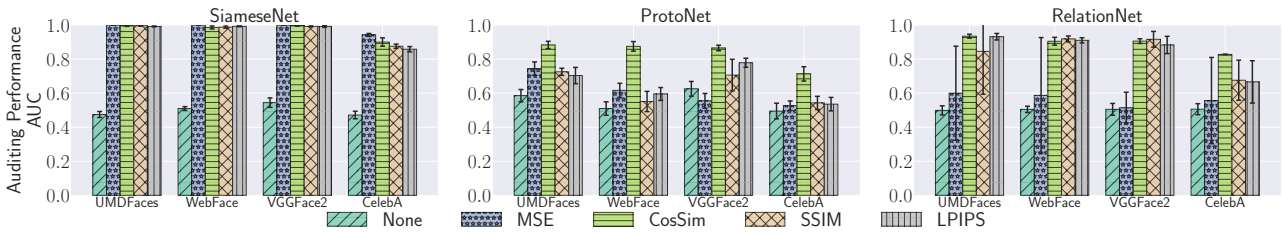
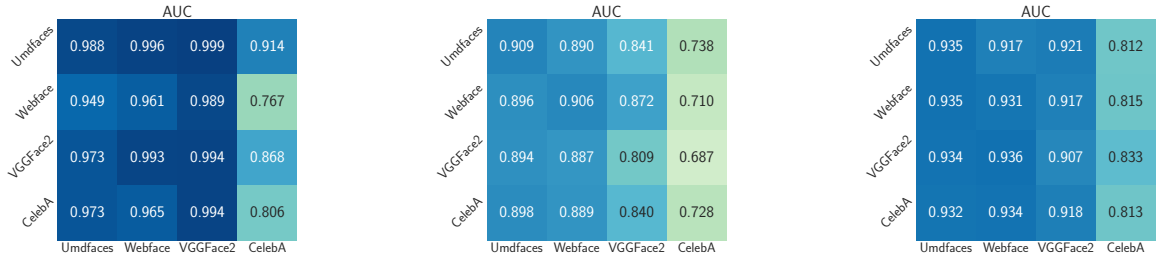



**Figure 7: Auditing performance (measured by AUC) when using different similarity metrics to generate the reference information.**




**(a) SiameseNet** **(b) ProtoNet** **(c) RelationNet**

**Figure 8: Auditing performance (measured by AUC) under datasets transfer. The x-axis is the dataset used to train the shadow models and probe the target/shadow models. The y-axis is the dataset used to train the target models.**

among different datasets. **In general, CosSim can achieve the best and the most stable performance in most of the settings.** We posit the reason is that it generates a bounded value (-1 to 1), which tends to be consistent with the normalized input feature of FACE-AUDITOR.

## 5.4 Auditor Transferability

In practice, the auditor might not be aware of the target model's architecture or its training data distribution. Thus, in this section, we aim to evaluate the transferability of FACE-AUDITOR. We first evaluate the dataset transferability when the training data of the shadow model comes from a different distribution than the target model, and then evaluate the model transferability when the architecture of the shadow model is different from the target model.

**Dataset Transferability.** We conduct experiments on three target models. For each target model, we use one dataset as the auxiliary dataset and the other three datasets as target datasets. In total, we have 16 combinations. We report the experimental results for the AUC metric in Figure 8. In general, we observe that FACE-AUDITOR maintains a good performance when the target dataset and the auxiliary dataset come from different distributions in most of the cases. For instance, when the auxiliary dataset is VGGFace2, and the target dataset is WebFace, we can achieve up to 0.954 auditing accuracy, only 0.029 lower than the same distribution auxiliary dataset.

Two reasons can explain the high auditing performance under dataset transfer settings. On the one hand, the uniqueness of human faces does not change substantially. Once a user's image is seen during the training process of the target model, it is easy to distinguish it from those never seen before. Which also shows the severe privacy risks of facial recognition models. On the other hand, FACE-AUDITOR is trained on a shadow dataset with no user overlap as the target model's training dataset. **The disjoint classes split forces the auditing model to not rely on the overfitting intuition to determine the membership status but learn to discriminate from the metric scores' internal correlations.**

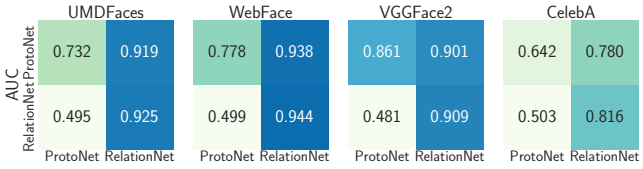



**Figure 9: Auditing performance (measured by AUC) under model transfer. In each subfigure, the x-axis represents the target model, and the y-axis represents the shadow model.**

**Model Transferability.** We conduct experiments between RelationNet and ProtoNet due to the fact that they share the same input data format and report the experimental results in Figure 9. We observe that the auditing performance slightly decreases when the architecture of the shadow model is different from the target model. The drop is significant when using RelationNet as the shadow model to audit ProtoNet. On the contrary, using ProtoNet as the shadow model to audit RelationNet can achieve better performance. We suspect the reason is that the linear Euclidean metric of ProtoNet is a particular case of non-linear metric, which supports a pre-trained linear model still work in most cases.

## 6 Discussion

**Practical Impacts of FACE-AUDITOR.** FACE-AUDITOR can serve as a complementary tool for existing privacy-protective actions. Governments and regulators can use FACE-AUDITOR as a tool for enforcing privacy regulations by determining if models are misusing data and violating individuals' privacy rights. FACE-AUDITOR can be used by individuals as an auditing tool to detect potential misuse of face data. If a misuse happens, they can take legal actions to correct or withdraw their data (according to GDPR Articles 15, 16, 17, 18). FACE-AUDITOR can also be employed by model developers to conduct self-inspection and ensure that their models are compliant with privacy regulations while demonstrating transparency in their data processing practices.

**Potential Risks of Using FACE-AUDITOR.** While FACE-AUDITOR has the potential to increase transparency for users contributing their data to train a model, it also poses a threat to the intellectual property of the model provider. A malefactor could exploit FACE-AUDITOR to launch user-level membership inference attacks against models with sensitive training data and use FACE-AUDITOR as a stepping stone for other malicious activities, such as attribute inference attacks. Knowing that a user is in a sensitive facial recognition-based system's authorized zone could also allow an attacker to design adversarial examples to become an authorized user. On a facial recognition-based disease diagnosis system, a known member user might also expose to the privacy of having a particular disease. Although model developers can introduce protective mechanisms, our evaluation of the robustness of FACE-AUDITOR in Appendix C demonstrates that its inference performance remains high even when subjected to four perturbations. In situations where private information is at risk of being inferred by malicious users, criminal laws such as the UK's Data Protection Act 2018 (sections 170 and 171) can deter such activities from occurring [4].

**Extend to Other Data Domain.** Our paper mainly focuses on facial recognition models, so we use the term "user-level" instead of "class-level". We believe our method can be adapted to other objects as well, and the key challenge lies in choosing the appropriate reference information. For instance, when dealing with text data, a better reference might be the frequency of rare words rather than sentence-level or word-level similarity [44].

## 7 Conclusion

In this paper, we proposed FACE-AUDITOR to determine whether a target user's face images were used to train a few-shot-based facial recognition system relying on the user-level membership inference. We carefully designed the probing set to query the few-shot-based facial recognition system. We further proposed to use the similarity scores between the raw face images as reference information to improve the auditing performance. We showed that FACE-AUDITOR is robust when the users' face images or the target models are equipped with different defense mechanisms. In the end, we discuss the practical implications and potential risks of using FACE-AUDITOR.

## Acknowledgments

We thank all anonymous reviewers and our shepherd for their constructive comments. This work is partially funded by the Helmholtz Association within the project "Trustworthy Federated Data Analytics" (TFDA) (funding number ZT-I-OO1 4), by the European Health and Digital Executive Agency (HADEA) within the project "Understanding the individual host response against Hepatitis D Virus to develop a personalized approach for the management of hepatitis D" (D-Solve) (grant agreement number 101057917), and by NSF grant number 2217071, 2220433 and 2213700.

## References

[1] https://gdpr-info.eu/.

[2] https://equalais.media.mit.edu/.

[3] https://oag.ca.gov/privacy/ccpa.

[4] https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted.

[5] Martin Abadi, Andy Chu, Ian Goodfellow, Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *CCS*, pages 308–318, 2016.

[6] Kendra Albert, Jonathon Penney, Bruce Schneier, and Ram Shankar Siva Kumar. Politics of Adversarial Machine Learning. *CoRR abs/2002.05648*, 2020.

[7] Ankan Bansal, Anirudh Nanduri, Carlos Domingo Castillo, Rajeev Ranjan, and Rama Chellappa. UMDFaces: An Annotated Face Dataset for Training Deep Networks. In *IJCB*, pages 464–473, 2017.

[8] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. VGGFace2: A Dataset for Recognising Faces across Pose and Age. In *FG*, pages 67–74, 2018.

[9] Varun Chandrasekaran, Chuhan Gao, Brian Tang, Kassem Fawaz, Somesh Jha, and Suman Banerjee. Face-Off: Adversarial Face Obfuscation. *Privacy Enhancing Technologies Symposium*, 2021.

[10] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When Machine Unlearning Jeopardizes Privacy. In *CCS*, pages 896–911, 2021.

[11] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, and Yang Zhang. FACE-AUDITOR: Data Auditing in Facial Recognition Systems. *CoRR abs/2303.04729*, 2023.

[12] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A Closer Look at Few-shot Classification. In *ICLR*, 2019.

[13] Linkang Du, Zhikun Zhang, Shaojie Bai, Changchang Liu, Shouling Ji, Peng Cheng, and Jiming Chen. AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy. In *CCS*, pages 1266–1288, 2021.

[14] Yuntao Du, Yujia Hu, Zhikun Zhang, Ziquan Fang, Lu Chen, Baihua Zheng, and Yunjun Gao. LDPTrace: Locally Differentially Private Trajectory Synthesis. In *VLDB*, 2023.

[15] Ivan Evtimov, Pascal Sturmfels, and Tadayoshi Kohno. FoggySight: A Scheme for Facial Lookup Privacy. *Privacy Enhancing Technologies Symposium*, 2021.

[16] Masoud Faraki, Xiang Yu, Yi-Hsuan Tsai, Yumin Suh, and Manmohan Chandraker. Cross-Domain Similarity Learning for Face Recognition in Unseen Domains. In *CVPR*, pages 15292–15301, 2021.

[17] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *CCS*, pages 1322–1333, 2015.

[18] Jianzhu Guo, Xiangyu Zhu, Chenxu Zhao, Dong Cao, Zhen Lei, and Stan Z. Li. Learning Meta Face Recognition in Unseen Domains. In *CVPR*, pages 6162–6171, 2020.

[19] Yaron Gurovich, Yair Hanani, Omri Bar, Guy Nadav, Nicole Fleischer, Dekel Gelbman, Lina Basel-Salmon, Peter M. Krawitz, Susanne B. Kamphausen, Martin Zenker, Lynne M. Bird, and Karen W. Gripp. Identifying Facial Phenotypes of Genetic Disorders Using Deep Learning. *Nature Medicine*, 2019.

[20] Hai Huang, Zhikun Zhang, Yun Shen, Michael Backes, Qi Li, and Yang Zhang. On the Privacy Risks of Cell-Based NAS Architectures. In *CCS*, pages 1427–1441, 2022.

[21] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *CCS*, pages 259–274, 2019.

[22] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese Neural Networks for One-Shot Image Recognition. In *DL*, 2015.

[23] Stepan Komkov and Aleksandr Petiushko. AdvHat: Real-World Adversarial Attack on ArcFace Face ID System. In *ICPR*, pages 819–826, 2020.

[24] Bogdan Kulynych, Rebekah Overdorf, Carmela Troncoso, and Seda F. Gùrses. POTs: Protective Optimization Technologies. In *FAT*, pages 177–188, 2020.

[25] Guoyao Li, Shahbaz Rezaei, and Xin Liu. User-Level Membership Inference Attack against Metric Embedding Learning. In *PAIR2Struct*, 2022.

[26] Jiacheng Li, Ninghui Li, and Bruno Ribeiro. Membership Inference Attacks and Defenses in Classification Models. In *CODASPY*, pages 5–16, 2021.

[27] Tao Li and Lei Lin. AnonymousNet: Natural Face De-Identification With Measurable Privacy. In *CVPRW*, pages 56–65, 2019.

[28] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. SphereFace: Deep Hypersphere Embedding for Face Recognition. In *CVPR*, pages 6738–6746, 2017.

[29] 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*, 2022.

[30] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *ICCV*, pages 3730–3738, 2015.

[31] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning Differentially Private Recurrent Language Models. In *ICLR*, 2018.

[32] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *S&P*, pages 497–512, 2019.

[33] Yuantian Miao, Minhui Xue, Chao Chen, Lei Pan, Jun Zhang, Benjamin Zi Hao Zhao, Dali Kaafar, and Yang Xiang. The Audio Auditor: User-Level Membership Inference in Internet of Things Voice Services. *Privacy Enhancing Technologies Symposium*, 2021.

[34] Thiago Guimaraes Moraes, Eduarda Costa Almeida, and José Renato Laranjeira de Pereira. Smile, You are being Identified! Risks and Measures for the Use of Facial Recognition in (Semi-)public Spaces. *AI Ethics*, 2021.

[35] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine Learning with Membership Privacy using Adversarial Regularization. In *CCS*, pages 634–646, 2018.

[36] Rodrigo Frassetto Nogueira, Roberto de Alencar Lotufo, and Rubens Campos Machado. Fingerprint Liveness Detection Using Convolutional Neural Networks. *IEEE Transactions on Information Forensics and Security*, 2016.

[37] Evani Radiya-Dixit, Sanghyun Hong, Nicholas Carlini, and Florian Tramér. Data Poisoning Won't Save You From Facial Recognition. In *AdvML*, 2022.

[38] 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 *NDSS*, 2019.

[39] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *CVPR*, pages 815–823, 2015.

[40] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y. Zhao. Fawkes: Protecting Privacy against Unauthorized Deep Learning Models. In *USENIX Security*, pages 1589–1604, 2020.

[41] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. In *CCS*, pages 1528–1540, 2016.

[42] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *S&P*, pages 3–18, 2017.

[43] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical Networks for Few-shot Learning. In *NIPS*, pages 4077–4087, 2017.

[44] Congzheng Song and Vitaly Shmatikov. Auditing Data Provenance in Text-Generation Models. In *KDD*, pages 196–206, 2019.

[45] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to Compare: Relation Network for Few-Shot Learning. In *CVPR*, pages 1199–1208, 2018.

[46] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *CVPR*, pages 1701–1708, 2014.

[47] Mingtian Tan, Zhe Zhou, and Zhou Li. The Many-faced God: Attacking Face Verification System with Embedding and Image Recovery. In *ACSAC*, pages 17–30, 2021.

[48] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. Fooling Automated Surveillance Cameras: Adversarial Patches to Attack Person Detection. In *CVPR*, pages 49–55, 2019.

[49] Haiming Wang, Zhikun Zhang, Tianhao Wang, Shibo He, Michael Backes, Jiming Chen, and Yang Zhang. PrivTrace: Differentially Private Trajectory Synthesis by Adaptive Markov Model. In *USENIX Security*, 2023.

[50] Mei Wang and Weihong Deng. Deep Face Recognition: A Survey. *Neurocomputing*, 2021.

[51] Tianhao Wang, Joann Qiongna Chen, Zhikun Zhang, Dong Su, Yueqiang Cheng, Zhou Li, Ninghui Li, and Somesh Jha. Continuous Release of Data Streams under both Centralized and Local Differential Privacy. In *CCS*, pages 1237–1253, 2021.

[52] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a Few Examples: A Survey on Few-shot Learning. *ACM Computing Surveys*, 2020.

[53] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image Quality Assessment: from Error Visibility to Structural Similarity. *IEEE Transactions on Image Process*, 2004.

[54] Emily Wenger, Shawn Shan, Haitao Zheng, and Ben Y. Zhao. SoK: Anti-Facial Recognition Technology. In *S&P*, 2022.

[55] Zuxuan Wu, Ser-Nam Lim, Larry S. Davis, and Tom Goldstein. Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors. In *ECCV*, pages 1–17, 2020.

[56] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. WIDER FACE: A Face Detection Benchmark. In *CVPR*, pages 5525–5533, 2016.

[57] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting. In *CSF*, pages 268–282, 2018.

[58] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning Face Representation from Scratch. *CoRR abs/1411.7923*, 2014.

[59] Quan Yuan, Zhikun Zhang, Linkang Du, Min Chen, Peng Cheng, and Mingyang Sun. PrivGraph: Differentially Private Graph Data Publication by Exploiting Community Information. In *USENIX Security*, 2023.

[60] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*, pages 586–595, 2018.

[61] Zhikun Zhang, Tianhao Wang, Jean Honorio, Ninghui Li, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. PrivSyn: Differentially Private Data Synthesis. In *USENIX Security*, pages 929–946, 2021.

[62] Zhikun Zhang, Tianhao Wang, Ninghui Li, Shibo He, and Jiming Chen. CALM: Consistent Adaptive Local Marginal for Marginal Release under Local Differential Privacy. In *CCS*, pages 212–229, 2018.

# A Comparison with Li et al. [25]

Recently, Li et al. [25] propose a user-level membership inference attack against embedding metric models when the adversary can access the embedding of the target model and leverage an auxiliary dataset to train multiple shadow models. They use two distance-based features to perform the attack: The average distance of the target user's images to their centroid ($C_u$) and the average pairwise distances of the target user's images ($P_u$). The authors propose to combine the two *scalar values* as the attack feature.[2] We next compare the performance of Li et al. and FACE-AUDITOR.

**Observation.** Table 3 illustrates the experimental results, and it shows that FACE-AUDITOR outperforms the method of Li et al. in most cases, indicating the features of FACE-AUDITOR are more informative. Specifically, the method of Li et al. can work on SiameseNet (0.6-0.7 accuracy). This is consistent with the results reported in their original paper (note that we use different datasets; thus, the results are slightly different). However, the performance on ProtoNet and RelationNet is close to random guess. We suspect the reason is that ProtoNet and RelationNet learn the relative distance information between different classes. The method of Li et al. only considers intra-class correlation but neglects the inter-class correlation. While FACE-AUDITOR utilizes the posterior (in ProtoNet and RelationNet) corresponding to the target user, which inherently considers inter-class correlation. Also, with the help of reference information, FACE-AUDITOR can better capture the slight difference between the original image distances and the similarity scores.

[2]The authors do not open-source their code. We thus implement the method of Li et al. by ourselves and conduct experiments on four datasets and three models.

# B Hyperparameter Study

Recall that we need carefully design a probing set $\mathbb{P} = \langle \mathbb{S}, \mathbb{Q} \rangle$ to achieve an optimal auditing performance. We have three important hyperparameters in the probing set i.e., the number of ways $k$, the number of shots $l$, and the number of queries $q$. We investigate their impacts on auditing performance in Appendix B.1, Appendix B.2, and Appendix B.3. We use AUC to measure the auditing performance, the results on other metrics are in Appendix B of [11]. We also investigate the impact of image size in Appendix B.4 and the impact of the embedding extractor in Appendix B.5 of [11].

## B.1 The Number of Ways $k$

In Figure 10, we observe only a slight decrease in the auditing accuracy (less than 4%) as we increase the number of ways $k$ in the support set for all three model architectures. This parameter affects the search space of the target model (we observe a worse target model performance in more ways of the support set), but it does not significantly affect the auditing model, as we only use the largest similarity score to form the auditing feature. This also explains why FACE-AUDITOR can work when thousands of users are in the training set of the target model.

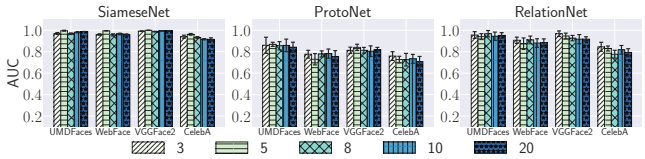



**Figure 10: Number of ways ($k$) in the support set.**

## B.2 The Number of Shots $l$

The results in Figure 11 show that increasing the number of shots in the support set leads to a more precise description of a user, as reflected by better target model performance on ProtoNet and RelationNet. However, since SiameseNet only takes image pairs as input, the target model performance is unaffected by the number of shots. Interestingly, we found that the auditing performance consistently improved as the number of shots increased for the SiameseNet. We believe this is because ProtoNet and RelationNet represent each user's multiple images as a whole and calculate inter-class distances to discriminate between multiple users. When generating the posteriors, ProtoNet and RelationNet already consider

Table 3: Comparison with Li et al. [25]. For their method, we use the $(C_u, P_u)$ as the default feature of the auditing model.

| Model | Dataset | Accuracy | | AUC | | F1 Score | | False Positive Rate | |
|---|---|---|---|---|---|---|---|---|---|
| | | Li et al. | FACE-AUDITOR | Li et al. | FACE-AUDITOR | Li et al. | FACE-AUDITOR | Li et al. | FACE-AUDITOR |
| SiameseNet | UMDFaces | $65.00 \pm 10.68$ | $100.00 \pm 0.00$ | $68.35 \pm 3.24$ | $100.00 \pm 0.00$ | $64.49 \pm 1.18$ | $100.00 \pm 0.00$ | $35.10 \pm 3.58$ | $0.00 \pm 0.00$ |
| | Webface | $63.00 \pm 8.04$ | $100.00 \pm 0.00$ | $64.47 \pm 3.71$ | $100.00 \pm 0.10$ | $61.20 \pm 3.56$ | $100.00 \pm 0.05$ | $38.30 \pm 2.44$ | $0.05 \pm 0.10$ |
| | VggFace2 | $60.05 \pm 6.32$ | $99.17 \pm 0.23$ | $63.31 \pm 5.20$ | $99.01 \pm 0.51$ | $60.61 \pm 7.32$ | $97.64 \pm 0.63$ | $37.04 \pm 3.97$ | $1.75 \pm 0.52$ |
| | CelebA | $57.72 \pm 1.37$ | $94.13 \pm 0.81$ | $59.85 \pm 4.18$ | $95.00 \pm 0.74$ | $59.33 \pm 3.81$ | $94.01 \pm 0.95$ | $44.55 \pm 1.48$ | $11.40 \pm 2.13$ |
| ProtoNet | UMDFaces | $50.00 \pm 1.00$ | $81.20 \pm 2.40$ | $49.92 \pm 6.98$ | $89.13 \pm 2.16$ | $53.33 \pm 26.67$ | $81.47 \pm 3.12$ | $80.00 \pm 40.00$ | $20.60 \pm 3.78$ |
| | Webface | $50.00 \pm 0.20$ | $76.30 \pm 4.08$ | $49.33 \pm 3.20$ | $83.59 \pm 4.27$ | $53.33 \pm 26.67$ | $75.54 \pm 4.20$ | $80.00 \pm 40.00$ | $20.80 \pm 5.42$ |
| | VggFace2 | $50.00 \pm 2.40$ | $77.40 \pm 1.36$ | $48.48 \pm 6.17$ | $86.19 \pm 1.90$ | $40.00 \pm 32.66$ | $77.43 \pm 1.48$ | $60.00 \pm 48.99$ | $22.80 \pm 2.14$ |
| | CelebA | $50.00 \pm 0.25$ | $65.90 \pm 3.80$ | $48.86 \pm 5.51$ | $70.77 \pm 3.85$ | $53.33 \pm 26.67$ | $65.76 \pm 4.53$ | $80.00 \pm 40.00$ | $34.00 \pm 2.90$ |
| RelationNet | UMDFaces | $50.20 \pm 0.25$ | $86.00 \pm 2.14$ | $55.57 \pm 4.78$ | $94.28 \pm 1.21$ | $44.45 \pm 28.17$ | $86.09 \pm 1.75$ | $63.40 \pm 45.83$ | $14.40 \pm 6.02$ |
| | Webface | $50.00 \pm 0.00$ | $86.20 \pm 2.38$ | $53.39 \pm 3.48$ | $92.53 \pm 1.66$ | $48.50 \pm 24.49$ | $85.92 \pm 2.53$ | $39.60 \pm 48.50$ | $12.00 \pm 2.83$ |
| | VggFace2 | $49.70 \pm 0.60$ | $82.60 \pm 2.58$ | $51.68 \pm 4.09$ | $90.75 \pm 2.35$ | $27.82 \pm 31.74$ | $81.85 \pm 2.80$ | $41.20 \pm 48.03$ | $13.40 \pm 3.20$ |
| | CelebA | $49.80 \pm 0.40$ | $74.30 \pm 2.71$ | $49.16 \pm 2.10$ | $82.86 \pm 1.93$ | $50.42 \pm 12.49$ | $74.79 \pm 3.24$ | $86.00 \pm 28.00$ | $28.00 \pm 5.10$ |

the influence of multiple shots, resulting in each user being represented as a single vector for comparison, regardless of the number of shots in the support set. On the other hand, SiameseNet takes image pairs per probe, and more shots indicate more diverse probes from a single user. This allows for capturing a user's character from multiple probes, leading to an increase in auditing performance.

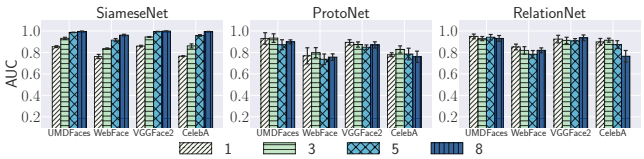



**Figure 11: Number of shots ($l$) in the support set.**

## B.3 The Number of Query Images $q$

We investigated the impact of the number of query images $q$ on three datasets with 100 images per user in their preprocessed dataset, providing a wide range of $q$ values to explore. Our results, shown in Figure 12, demonstrate that auditing performance improves and the false positive rate decreases as the number of query images increases. The rationale is that more query images lead to a broader auditing feature that captures more information about the user, and more images of a user can help distinguish them from other users. The increasing trend is more pronounced for RelationNet and ProtoNet, suggesting that more diverse queries can reveal more information about the underlying training data of the few-shot facial recognition models, especially when the model has a higher memorization ability.

# C Robustness of FACE-AUDITOR

In this section, we investigate the robustness of FACE-AUDITOR when the target models' pipeline is perturbed to

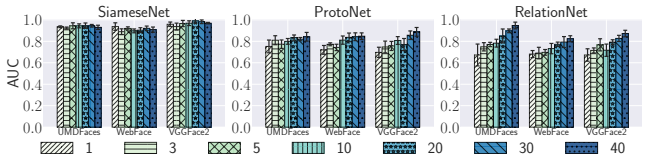



**Figure 12: Number of query images ($q$).**

evade auditing. Concretely, we consider four defense mechanisms: *Input perturbation* in Appendix C.1 (perturb the training images), *training perturbation* in Appendix C.2 (perturb the training gradient by enforcing differential privacy), and *output perturbation* in Appendix C.3 (perturb the similarity scores returned by the target models). In the end, we also explore an adaptive adversary scenario in Appendix C.4.

## C.1 Input Perturbation

**Methodology.** Multiple techniques are proposed to perturb the face images before training the facial recognition models [9, 15] and prevent the face images from being misused. In our experiments, we consider a recently proposed technique called Fawkes [40]. The general idea of Fawkes is to add imperceptible noise to the target images that drive the embeddings of the face images to deviate from that of the raw face images. According to its homepage, it has been downloaded more than 840,000 times and used in various applications.

**Visualization of Adversarial Perturbation.** We visualize training images under three input adversarial perturbation levels by the Fawkes [40] in Figure 13.

**Evaluation.** The open-sourced Fawkes implementation[3] allows us to choose three perturbation levels: Low, middle, and high. We experiment on the UMDFaces dataset and three target model architectures. Concretely, we first use Fawkes with three perturbation levels to prepossess all the images in UMDFaces, and then use the same pipeline introduced in Section 4.3 to build our shadow model and auditing model.

[3] https://github.com/Shawn-Shan/fawkes

**Table 4: Auditing performance under input perturbation on UMDFaces. The higher the perturbation level, the better the privacy-preserving level. To give a direct impression of the perturbation, we show a visualization of different perturbation levels in Figure 13.**

| Target Model | SiameseNet | | ProtoNet | | RelationNet | |
|---|---|---|---|---|---|---|
| Perturbation Level | $\mathcal{M}_{Target}$ Acc. ($\Delta$) | $\mathcal{M}_{Auditor}$ Acc. | $\mathcal{M}_{Target}$ Acc. ($\Delta$) | $\mathcal{M}_{Auditor}$ Acc. | $\mathcal{M}_{Target}$ Acc. ($\Delta$) | $\mathcal{M}_{Auditor}$ Acc. |
| **Original** | **0.500** | **0.991 ± 0.000** | **0.782** | **0.879 ± 0.000** | **0.847** | **0.961 ± 0.000** |
| Low | 0.485 (-0.015) | 1.000 ± 0.001 | 0.803 (+0.021) | 0.785 ± 0.073 | 0.874 (+0.027) | 0.914 ± 0.019 |
| Middle | 0.496 (-0.004) | 0.993 ± 0.002 | 0.843 (+0.061) | 0.852 ± 0.032 | 0.877 (+0.030) | 0.903 ± 0.017 |
| High | 0.477 (-0.023) | 0.973 ± 0.004 | 0.777 (-0.005) | 0.843 ± 0.027 | 0.838 (-0.009) | 0.913 ± 0.021 |

**Table 5: Auditing performance under training perturbation. We report the target model performance and auditing performance for three different privacy-preserving levels, i.e., Low, Middle, and High. Original means the target model without enforcing DP-SGD.**

| Dataset | Target Model Perturbation Level | SiameseNet | | ProtoNet | | RelationNet | |
|---|---|---|---|---|---|---|---|
| | | $\mathcal{M}_{Target}$ Acc. | $\mathcal{M}_{Auditor}$ AUC | $\mathcal{M}_{Target}$ Acc. | $\mathcal{M}_{Auditor}$ AUC | $\mathcal{M}_{Target}$ Acc. | $\mathcal{M}_{Auditor}$ AUC |
| VGGFace2 | **Original** | **0.575** | **0.996 ± 0.002** | **0.868** | **0.866 ± 0.016** | **0.943** | **0.906 ± 0.014** |
| | Low | 0.330 | 0.981 ± 0.004 | 0.433 | 0.929 ± 0.000 | 0.215 | 0.932 ± 0.011 |
| | Middle | 0.250 | 0.990 ± 0.007 | 0.425 | 0.877 ± 0.024 | 0.214 | 0.913 ± 0.019 |
| | High | 0.258 | 0.982 ± 0.005 | 0.405 | 0.885 ± 0.024 | 0.215 | 0.909 ± 0.012 |

**Figure 13: An illustration of images under different levels of adversarial noise perturbation.**

We report the target model performance and the auditing performance under different perturbation levels in Table 4. We observe a slight performance drop of the target model when applying high-level perturbation, indicating the perturbed face images (especially under high-level perturbation) are more difficult to train. Regarding the auditing performance, we only observe a slight drop (the drop percentage is less than 6%), which indicates that **FACE-AUDITOR is robust to input perturbation**.

## C.2 Training Perturbation

**Methodology.** A generic approach to protect users' data privacy is differential privacy (DP), which guarantees that any sample in the input dataset has a limited impact on the final output [13,14,49,51,59,61,62]. For machine learning models, the most representative DP algorithm is Differentially-Private Stochastic Gradient Descent (DP-SGD) [5]. In general, DP-SGD adds Gaussian noise to gradient $g$ during the target ML model's training process, i.e., $\tilde{g} = g + \mathcal{N}\left(0, \Delta_g^2 \sigma^2 \mathbf{I}\right)$. Note that there is no prior knowledge to determine the influence of a single training sample on the gradient $g$; thus, the sensitivity of $g$ cannot be directly computed. To address this problem, DP-SGD proposes to bound the $\ell_2$ norm of the gradient to $C$ by clipping $g$ to $g/\max\{1, ||g||_2/C\}$. This clipping ensures that if $||g||_2 \leq C$, $g$ is preserved; otherwise, it gets scaled down to be the norm of $C$. As such, the sensitivity of $g$ is bounded by $C$. Note that we aim to show the defensive performance of adding perturbation in the training process of the target model. Besides, existing user-level DP mainly focus on the federated learning setting [31,32]. They do not fit to few-shot learning paradigms.

**Evaluation.** We conduct experiments on four datasets and three target models. The experimental results in Table 5 show that **DP-SGD has a severe impact on the target model performance**. We further observe variations in the auditing performance across the three model architectures: SiameseNet is more sensitive to DP-SGD while ProtoNet and RelationNet are less sensitive. Take VGGFace2 as an example. Applying a high-level noise to the training phase of SiameseNet makes target model accuracy drop by 55%, while the auditing accuracy only drops 1.4%. The auditing accuracy of ProtoNet and RelationNet remains almost the same.

## C.3 Output Perturbation

**Methodology.** Another approach to protecting ML models from inference attacks is adding perturbations on the target models' outputs. In this subsection, we evaluate the robustness of our auditing model when the similarity scores returned by the target model are perturbed. We implement this defense by adding a zero-mean Laplace noise with a standard deviation $\delta$ to the target model's outputs.

**Evaluation.** We conduct experiments on all four datasets and three target models. The experimental results in Figure 14 show that **FACE-AUDITOR is robust to output perturbation**. Concretely, the auditing performance on SiameseNet and ProtoNet does not drop significantly, and the auditing performance drop on RelationNet is in the scope of 15%. We

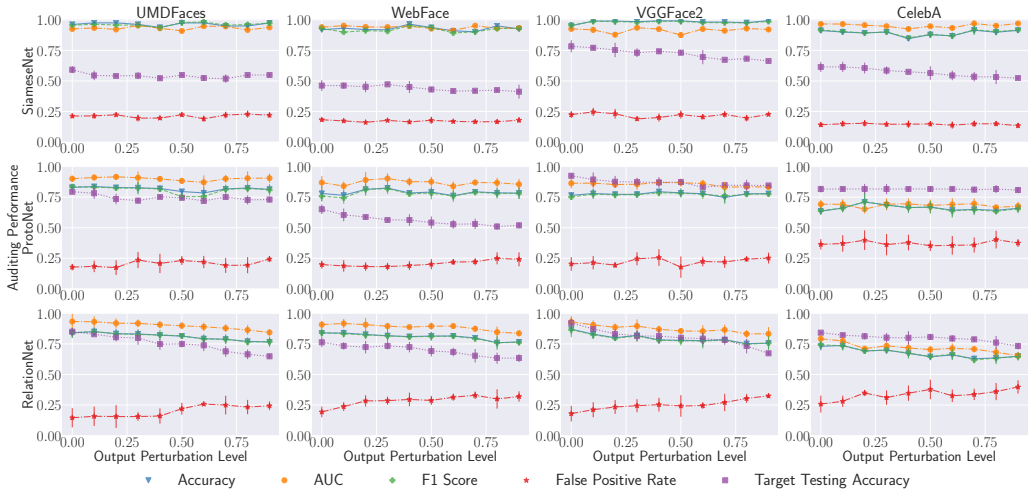



**Figure 14: Auditing performance under output perturbations. The x-axis represents different noise levels used to perturb the target model's outputs. Higher values mean a stronger perturbation degree. The y-axis represents the auditing performance.**

also observe a slight drop in the target model performance when the noise perturbation level increases, which indicates the robustness of FACE-AUDITOR.

## C.4 MemGuard

**Threat Model.** In practice, a malicious data collector might be aware of the existence of FACE-AUDITOR. They modify their facial recognition models in a way to evade auditing and gain financial benefits or avoid a lawsuit. We evaluate the performance of FACE-AUDITOR when the target model's output is perturbed to avoid membership auditing.

**Methodology.** We follow the design intuition of MemGuard [21] and perform adaptive attacks against FACE-AUDITOR. The general idea is to perturb the similarity scores (outputs of the target model) while achieving two objectives: Minimum label loss and maximum auditing confusion. The first goal guarantees the noisy posteriors do not change the predicted labels of the target model given any inputs. The second goal aims to make FACE-AUDITOR randomize its predictions of the user-level membership status given any face images to be audited. Concretely, to make FACE-AUDITOR unable to distinguish member and non-member users, the adaptive attacker aims to add the maximum noise on the similarity scores under the constraint of not affecting the corresponding label. To ensure that the final summation of the target model's output is valid (summing to one), after adding the maximum noise to the target similarity score, we apply a SoftMax function to the entire similarity score vector, generating the final perturbed score vector. Note that the adversary cannot perturb the reference information as it is prepared by FACE-AUDITOR and is a fixed value given any input images; thus, we concatenate the original reference information and the perturbed similarity scores as the auditing feature.

**Results.** Figure 15 illustrates the auditing performance of

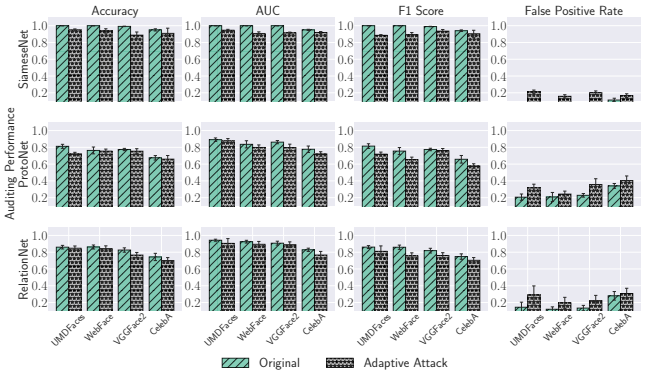



**Figure 15: Auditing performance comparison under an adaptive adversary against FACE-AUDITOR.**

FACE-AUDITOR under an adaptive attack. We observe that adaptive perturbation on the target model's outputs only slightly affects the auditing performance. The drop in auditing performance is less than 5%. This differs from the sample-level membership inference case, in which MemGuard leads to near-random guessing attack performance. There are three reasons. First, MemGuard can only perturb one value of the auditing feature per query, while FACE-AUDITOR queries the target model multiple times and combine the similarity scores of multiple queries as the auditing feature. Second, in sample-level membership inference, an adaptive adversary can perturb the whole attack feature (the posterior of the target sample) simultaneously, but it can only perturb one value per query in our user-level membership inference setting. Third, the reference information helps maintain the relative correlation of the query images and captures the subtle difference between member and non-member users. Additionally, our experimental results (in Appendix C.3) show a limited impact of output perturbation even without the minimum label loss constraint. In summary, the similarity scores are more

difficult for an adaptive adversary to perturb than output perturbation due to the minimum label loss constraint, which keeps the predicted label of a query sample fixed for a given support set and leaves the adversary little room to perturb.