DAAP: Privacy-Preserving Model Accuracy Estimation on Unlabeled Datasets Through Distribution-Aware Adversarial Perturbation

Guodong Cao, Wuhan University; Zhibo Wang, Zhejiang University; Yunhe Feng, University of North Texas; Xiaowei Dong, Wuhan University

https://www.usenix.org/conference/usenixsecurity24/presentation/cao-guodong
DAAP: Privacy-Preserving Model Accuracy Estimation on Unlabeled Datasets Through Distribution-Aware Adversarial Perturbation

Guodong Cao† Zhibo Wang‡,∗ Yunhe Feng† Xiaowei Dong†
†School of Cyber Science and Engineering, Wuhan University, P. R. China
‡The State Key Laboratory of Blockchain and Data Security, Zhejiang University, P. R. China
∗School of Cyber Science and Technology, Zhejiang University, P. R. China
†Department of Computer Science and Engineering, University of North Texas, USA
{guodongcao, xwdong}@whu.edu.cn, zhibowang@zju.edu.cn, yunhe.feng@unt.edu

Abstract
In the dynamic field of deep learning, accurately estimating model performance while ensuring data privacy against diverse and unlabeled test datasets presents a critical challenge. This is primarily due to the significant distributional shifts between training and test datasets, which complicates model evaluation. Traditional methods for assessing model accuracy often require direct access to the entire test dataset, posing significant risks of data leakage and model theft. To address these issues, we propose a novel approach: Distribution-Aware Adversarial Perturbation (DAAP). This method is designed to estimate the accuracy of deep learning models on unlabeled test datasets without compromising privacy. Specifically, DAAP leverages a publicly available dataset as an intermediary to bridge the gap between the model and the test data, effectively circumventing direct interaction and mitigating privacy concerns. By strategically applying adversarial perturbations, DAAP minimizes the distributional discrepancies between datasets, enabling precise estimation of model performance on unseen test data. We present two specialized strategies for white-box and black-box model contexts: the former focuses on reducing output entropy disparities, while the latter manipulates distribution discriminators. Overall, the DAAP introduces a novel framework for privacy-preserving accuracy estimation in model evaluation. This novel approach not only addresses critical challenges related to data privacy and distributional shifts but also enhances the reliability and integrity of model performance assessments. Our extensive evaluation on the CIFAR-10-C, CIFAR-100-C, and CelebA datasets demonstrates the effectiveness of DAAP in accurately estimating performance while safeguarding both data and model privacy.

1 INTRODUCTION
Deep neural networks (DNNs) have been widely deployed to solve various tasks, such as object detection [1], smart medical care [2], and advertisement recommendation [3], showcasing their ability to autonomously infer decision-making logic for inputting samples [4]. Yet, when deployed in real-world scenarios, these models encounter diverse and complex environments. The inputting data distributions could significantly diverge from those seen during training. For instance, a facial recognition system would process face data with different levels of brightness caused by weather change [5]; autonomous vehicles would discern various traffic signs due to regional variations [6]; smart healthcare systems would process the diversified data collected by different medical sensors [7], and so on. The inputting/test data exhibit varied characteristics and distribution probabilities, influenced by changing external factors such as weather and geography. This variability introduces substantial distributional disparities between training and test datasets, which results in different performances of a model between the test data and the training data [8].

To measure and estimate a model’s performance on a test dataset, traditional methods usually require knowing the ground truth (e.g., annotated labels) for comparative analysis. Unfortunately, the availability of annotated test data is a rarity in real life, complicating accuracy estimation for unlabeled datasets. Existing methods for estimating model accuracy on unlabeled datasets fall into three primary categories: annotation-based [9, 10], confidence-based [11, 12], and distribution-based [13–16] approaches. However, annotating significant volumes of test data is both time-consuming and labor-intensive, and a limited number of annotations may not fully capture the diversity of test dataset distributions. Both confidence-based and distribution-based strategies require the model to have complete access to the test dataset, raising significant privacy concerns about both test models and test data. For instance, if the test data owners have malicious intentions, they may steal the test model’s weights and even its training data information [17] via the output, which poses a significant threat to the model’s security [18, 19]. Additionally, malicious test models may infer sensitive and private information of the data when scanning the entire test dataset [20, 21]. Therefore, there is an imperative need for a
methodology that can estimate model accuracy on unlabeled test datasets, while safeguarding the privacy of both the data and the model.

In this paper, we aim to estimate model accuracy on unlabeled test datasets while preserving the privacy of both the test data and the model. To achieve this goal, we face the following challenges: (1) How to estimate the model’s accuracy in the absence of labels in the test data; (2) How to protect the privacy of test data and the test model during the accuracy estimation. To solve these challenges, we propose a novel and privacy-preserving method, named Distribution-Aware Adversarial Perturbation (DAAP), to estimate the model accuracy on unlabeled test datasets. This approach obviates the need for direct interaction between the test data and the model by utilizing a publicly available dataset as an intermediary, as illustrated in Figure 1, thus preserving the data and model privacy. This public dataset can be constructed either by manually labeling data or by choosing random samples from an existing labeled dataset, noting these samples are excluded from the model’s training process. To estimate the model accuracy on the unlabeled test dataset, we minimize the distribution differences (e.g., feature space change [22] and the field of out-of-distribution (OOD) [23]) between the unlabeled test and public datasets by adding distribution-aware adversarial perturbation on the public dataset. Based on the assumption that data are independent and identically distributed [24], a model’s performance is consistent across datasets with similar distributions [25, 26]. Therefore, we could estimate the model’s accuracy on the test dataset by predicting model accuracy on the perturbed public dataset. Furthermore, we have tailored distinct methodologies for estimating the accuracy of white-box and black-box models, leveraging our proposed method to achieve precise estimations.

In white-box settings, where the test data owner has full knowledge and access to the test model, model privacy leakage is not a concern. However, deploying the model locally for testing may not be feasible for test data owners due to software and hardware complexities. This necessitates submitting test data to the model for evaluation, but malicious models could steal sensitive information through member or property inference attacks [20, 21]. Therefore, minimizing the number of data samples submitted is crucial to mitigate privacy risks. Existing methods typically require processing the entire test dataset because they rely on individual sample predictions (e.g., logits), thus exposing the entire dataset to potential privacy threats [9–16]. Our approach, in contrast, leverages a small subset of unlabeled test data and an equivalent amount of publicly available labeled data to achieve accurate estimation. We achieve this by generating distribution-aware adversarial perturbations that will be deployed on public data, focusing on minimizing the overall data distribution discrepancy between the public and unlabeled datasets. To be more specific, the adversarial perturbation is guided and crafted by minimizing the model output entropy difference through an adversarial generator. This allows us to approximate the model’s accuracy on the test set by evaluating its performance using the public dataset, eliminating the need to access the entire unlabeled test dataset.

Traditional black-box scenarios keep test data owners in the dark about the inner workings of the model. They can only observe the model’s outputs for specific test inputs they provide. Our approach takes privacy a step further: it eliminates the need for even these outputs. This protects the privacy of both the test data and the model itself. In essence, test
data owners do not need to interact with the model at all. Instead, we leverage a publicly available labeled dataset and craft special modifications, called distribution-aware adversarial perturbations, that make it resemble the unseen test data’s distribution. Inspired by adversarial training [25], we use two components: a perturbation generator and a distribution discriminator. The discriminator’s goal is to identify the difference in data distributions between the public and unlabeled datasets. Meanwhile, the generator strives to create perturbations that fool the discriminator by making the public data appear like test data. After this training process, the generator produces perturbations that effectively align the distributions of both datasets. This allows us to estimate the model’s accuracy on the test set based on its performance on the perturbed public dataset, eliminating the need to access the raw test data itself. Importantly, the perturbation generator trains independently of the specific test model, granting the generated perturbations superior generalization capabilities. Consequently, multiple models can leverage this perturbed public data to estimate their accuracy on the test set with privacy protection of test data and test models.

The main contributions of this paper in the field of privacy-preserving model accuracy estimation for unlabeled test data can be summarized as follows:

- The paper proposes Distribution-Aware Adversarial Perturbation (DAAP), a novel privacy-preserving approach that achieves high test model accuracy on unlabeled test data and eliminates privacy risks that arise from direct interaction between the model and test data. The publicly available labeled datasets, loaded with the distribution-aware adversarial perturbation, serve as an intermediary to estimate the accuracy of deep learning models on unlabeled test datasets.

- DAAP accommodates both white-box and black-box implementations to meet the demands of real-world applications. Unlike existing methods that require individual sample assessments, our white-box design leverages the aggregated data distribution information from a small subset of test data to generate robust, distribution-aware perturbations for the public dataset. In contrast to existing approaches, our black-box strategy does not require interaction with the test model but can still generate effective distribution-aware perturbations to reduce the distribution discrepancy between the test and public datasets. This, in turn, strengthens the privacy protection for both the test model and data.

- The effectiveness of the DAAP method has been validated through comprehensive experiments conducted on the CIFAR-10-C, CIFAR-100-C, and CelebA datasets. The results demonstrate that the DAAP is an effective and generalizable method for estimating model accuracy on unlabeled test datasets while ensuring the privacy of both the test model and data.

## 2 RELATED WORK

The estimation of model accuracy on unlabeled datasets can be divided into three distinct categories, i.e., annotation-based, confidence-based, and distribution-based methods.

For the annotation-based, some researchers were inspired by the concept of test selection in the software engineering community [9, 10] and annotated part of the test data [27]. They annotated a subset to minimize the cross-entropy across the overall test data, enabling the subset’s accuracy to reflect that of all unlabeled test data. For example, the Aries [28] estimated model accuracy by matching the output distribution between partially annotated data and unlabeled test data. While annotation-based approaches could estimate model accuracy on unlabeled datasets, they require considerable human effort and incur substantial annotation expenses.

The confidence-based estimation approach measures model classification confidence in individual samples as a means to predict its accuracy on the entire test dataset. The Average Confidence (AC) is calculated as the average of the highest softmax scores across all examples in the test dataset. The samples above the average confidence are considered correctly classified. In contrast to AC, the Average Thresholded Confidence (ATC) approach introduces a predefined threshold. This threshold is determined based on a validation set originating from the source distribution, such as the training data, to ascertain the accuracy of classifications. Samples whose confidence scores surpass this threshold are categorized as accurately classified, offering a nuanced approach to evaluating model performance [12]. The Difference of Confidences (DoC) method estimates a model’s accuracy on test datasets by calculating the difference in model confidence between known (training) and unknown (test) datasets. This discrepancy serves as a predictive metric to assess increases or decreases in accuracy when the model classifies the test data [11]. Given that confidence-based methods rely on assessing the model’s confidence in each sample, their effectiveness and precision in results increase with the number of samples analyzed. However, this also raises the likelihood of potential data privacy breaches, as the extensive processing of samples may inadvertently expose sensitive information.

The distribution-based methods quantify the data difference between known (training) and unknown (test) datasets, predicting the model performance on the unknown dataset [29, 30]. Such data difference could be measured through first and second-order statistics of the mean vector of image feature representations [13, 14], e.g., Frechet Distance (FD) [15] or the Maximum Mean Difference (MMD) [16]. These methods explore the linear relationship between dataset distribution differences and model accuracy. However, they require the creation of numerous datasets, which is not only time-consuming...
but also demands substantial storage space. Moreover, when the test data contains outlier feature vectors or different class probabilities from the training data, the accuracy estimation of these methods becomes unreliable.

To address the constraints associated with the need for annotations, privacy concerns, and substantial storage requirements inherent in current methodologies, we introduce the Distribution-Aware Adversarial Perturbation (DAAP) method. This approach aims to accurately estimate model performance on test datasets while ensuring the privacy of both the model and the test data. DAAP employs a publicly available, labeled dataset as an intermediary to eliminate direct interaction between the test data and the models. By applying adversarial perturbation, it minimizes the distributional variance between the public dataset and the test dataset. As a result, the model’s accuracy on the unlabeled test dataset can be deduced from its performance on the public dataset, thereby providing a method that not only preserves privacy but also yields high accuracy in its estimations.

3 PRELIMINARY

This section briefly introduces the concepts of model training, accuracy estimation, and information entropy. These foundational notations will be used in the remainder of the paper.

3.1 Model Training

The model, denoted as \( M(\theta) \), is trained through the function \( M(x; \theta) : \mathcal{X} \rightarrow \mathcal{Y} \), where \( \mathcal{X} \) represents the input space, \( \mathcal{Y} \) signifies the label space, and \( \theta \) is the model parameter. This training process enables the model to learn complex patterns and features from a labeled training dataset. The training dataset is denoted as \( \mathcal{D}_{train} = \{(x_i, y_i)\}_{i=1}^{N} \), where \( x_i \) is the \( i \)-th sample out of \( N \) training samples, and \( y_i \in \{0, 1, \ldots, c-1\} \) represents the true label of \( x_i \), with \( c \) being the total number of classes. The objective of the training process is to minimize the discrepancy between the predicted labels and the true labels of the training samples. The training loss function can be written as:

\[
\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{J}(\hat{y}_i, y_i), \quad \text{s.t. } \hat{y}_i = M(x_i; \theta),
\]

where \( \mathcal{J}(\cdot, \cdot) \) denotes the cross-entropy function [31], used to quantify the disparity between the predicted labels and the ground truth. The model parameters are updated by minimizing this disparity during the model training, with the objective of enabling precise classification of input samples [32].

3.2 Model Estimation

Following the completion of training, the model’s generalization capabilities are assessed using a test dataset consisting of new, unseen data. This assessment ensures the model’s predictions are robust and applicable across a broad spectrum of datasets. The test dataset is defined as \( \mathcal{D}_{test} = \{(x_t, y_t)\}_{t=1}^{N_{test}} \), where \( N_{test} \) represents the total number of test samples. The accuracy of the model on the test dataset, denoted as \( \text{ACC}_{test} \), is determined by comparing the predicted labels \( \hat{y}_t \) with the actual ground truth labels \( y_t \). The formula for the model’s accuracy on the test dataset is expressed as follows:

\[
\text{ACC}_{test} = \frac{\sum_{t=1}^{N_{test}} \mathbb{1}[\hat{y}_t = y_t]}{N_{test}}
\]

where \( \mathbb{1}[\cdot] \) denotes an indicator function. In a classification model, this function returns 1 if the model’s prediction aligns with the ground truth labels of the samples, and 0 otherwise. However, most test data in real-life scenarios are not labeled. The unlabeled test dataset is designated as \( \mathcal{D}_{test} = \{x_t\}_{t=1}^{N_{test}} \).

The objective of this paper is to estimate the model’s accuracy on these unlabeled test datasets, i.e., minimizing the model estimation accuracy error between the actual accuracy \( \text{ACC}_{test} \) and the estimated accuracy \( \hat{\text{ACC}}_{test} \), which could be formalized as follows:

\[
\min \text{Error} = |\text{ACC}_{test} - \hat{\text{ACC}}_{test}|.
\]

3.3 Information Entropy

Information entropy quantifies the uncertainty of a random variable in information theory [33]. Within the framework of deep learning, this concept is utilized to describe models as conditional probability distributions, \( P(Y|X) \) [34]. For instance, given an input \( X \), the model outputs a variable \( Y \), comprising the probabilities \( P(y_i) \) associated with classifying the sample into the category \( y_i \). The entropy measures the amount of information conveyed by the model’s output, essentially measuring the expected uncertainty in the model’s predictions. The formula for calculating entropy is articulated as follows:

\[
\mathcal{H}(Y) = -\sum_{i=1}^{c} P(y_i) \log_2 P(y_i),
\]

where \( c \) is the number of categories. When the probability distribution of the model output is more uniform, the entropy value is higher. This higher entropy indicates that the model’s classification prediction is less certain or more disordered. Conversely, a more deterministic probability distribution results in a lower entropy value, signifying that the model’s classification prediction is more certain and reliable.

4 PROBLEM STATEMENT

In this section, we analyze privacy threats associated with the model estimation process for both the test data and the test
model, and we articulate the protection objectives to safeguard their privacy.

4.1 Analysis of Privacy Threats

After the training phase of the model \( M(\theta) \) concludes, it can be deployed in complex and unpredictable scenarios, where it may encounter unlabeled test datasets. Evaluating the model’s performance on these datasets is crucial for understanding its potential effectiveness in real-world applications and determining the necessity for further adjustments. Traditionally, accuracy estimation methods necessitate processing the entirety of the test data by the model and disclosing its predictions to achieve high estimation precision. These approaches, however, introduce significant privacy concerns, particularly in scenarios where either the test data owner or the model may have malicious intents. This paper delves into the privacy risks associated with estimating the accuracy of models, covering both white-box and black-box paradigms.

Privacy Risks in Test Data: Test datasets often contain sensitive details, such as data distributions and feature characteristics. Direct interaction between the model and test data during evaluation exacerbates the difficulty of protecting this sensitive information. Malicious models conducting extensive analysis of the test dataset can not only unveil data distributions but also heighten the risk of privacy violations [35]. These models might leverage such exposure to infer confidential attributes through property inference attacks [21], membership inference attacks [20], or even reconstruct the original test data [36]. While data encryption emerges as a viable countermeasure for enhancing privacy [37], it demands substantial computational resources.

Privacy Risks in Test Model: The outputs of a test model divulge considerable information about its functionality, tasks, and predictions. During performance estimation on unlabeled datasets, these outputs become accessible, revealing insights about the test model. Attackers may deduce the model’s algorithms, hierarchical structures, and parameter configurations from these outputs through extraction attacks [38], gaining an understanding of the model’s internal mechanisms and potentially reconstituting its parameters. More alarmingly, with the information gleaned, attackers could manipulate or compromise model decisions via adversarial sample attacks [39]. This represents a significant security threat to the model.

4.2 Protection Objectives

Privacy concerns primarily stem from the direct accessibility between the model and the test data. To address this, we introduce the Distribution-Aware Adversarial Perturbation (DAAP) method. This strategy facilitates the estimation of models on unlabeled datasets without necessitating interaction between the model and test data, thereby safeguarding privacy. Specifically, we employ a small, publicly available dataset, \( D_{pub} \), to act as a mediator between the test model and the test data. The perturbation generator then produces a distribution-aware adversarial perturbation, designed to align the distribution of \( D_{pub} \) with that of the test dataset. Consequently, we assess the model’s accuracy on the unlabeled test dataset by estimating its performance on \( D_{pub} \), which ensures the privacy of both the test model and the test data. Meanwhile, we formulate specific protection objectives for both white-box and black-box model estimations.

**White-box Model Estimation.** In the white-box testing scenario, where the test data owner possesses complete knowledge and access to the model, there is no concern about model privacy leakage. Our primary aim is the protection of test data privacy. Traditional model evaluation methods often require extensive iterations over all the test data to yield accurate estimates. However, our approach significantly reduces the exposure of test data by employing only a limited subset of publicly available datasets, thus minimizing privacy risks. Specifically, by leveraging knowledge about the model, such as its outputs, we train a perturbation generator \( G(\omega) \), which is steered by the entropy difference in outputs between the public and test datasets. The goal is to use a smaller, public dataset that can effectively approximate the test dataset as closely as possible. Consequently, the model’s accuracy on the unlabeled test dataset can be inferred from its performance on the labeled public dataset with applied perturbations. This strategy eliminates the need for the model to process the entire test dataset, thereby enhancing the privacy of the unlabeled test data. The process is formalized in the following equation:

\[
\begin{align*}
\min \ G(\mathcal{P}(M(D_{pub};\theta)), P(M(D_{test};\theta))) \\
\text{s.t. } D_{pub} = G(D_{pub};\omega) + D_{pub},
\end{align*}
\] (5)

where \( D_{pub} \) is the perturbed public dataset, \( P(\cdot) \) represents the entropy of the test model output, \( G(\cdot;\omega) \) is a function measuring the difference in model performance, \( \theta \) denotes test model parameters, and \( \omega \) is the generator parameter.

**Black-box Model Estimation.** In black-box settings, the internal mechanics of the test model remain undisclosed to the owners of the test data. Our approach emphasizes not only the privacy protection of the test data but also the security of the test models. The primary source of privacy threats originates from the direct interaction between test data and test models. To mitigate this risk, we utilize a publicly available labeled dataset and introduce distribution-aware adversarial perturbations generated by the perturbation generator \( G(\omega) \), aiming to align the distribution of the public dataset \( D_{pub} \) closely with that of the test dataset \( D_{test} \). The model’s accuracy on the test dataset is mirrored on the perturbed public dataset. The optimization objective for the black-box model can be formalized as follows:

\[
\begin{align*}
\min_{\omega} \ G(\mathcal{P}(D_{pub}'), \mathcal{P}(D_{test})) \\
\text{s.t. } D_{pub}' = G(D_{pub};\omega) + D_{pub},
\end{align*}
\] (6)
where $\Phi(\cdot)$ is the dataset distribution. Notably, the test model does not interact with the test data during the training of the generator or during the estimation phase, thereby safeguarding the privacy of both the test data and the model.

5 METHODOLOGY

In this section, we begin with a comprehensive overview of the proposed Distribution-Aware Adversarial Perturbation (DAAP) method. Then, we detail the designs for both white-box and black-box model estimations within the DAAP framework. Finally, we describe the intricate process of training perturbation generators specifically tailored for white-box and black-box model estimations.

5.1 Overview of DAAP

It is straightforward that a model, when trained on a labeled dataset, effectively learns the correlations between data features and their corresponding ground truth labels, thereby exhibiting strong performance on the observed dataset. However, in real-world applications, the test dataset often presents a different distribution compared to the training dataset, a divergence primarily attributed to variations in deployment environments. Moreover, when test data remain unlabeled, it becomes impractical to evaluate the accuracy of the model by comparing its predictions to ground truth labels. While current methodologies enable the assessment of model accuracy on unlabeled datasets [40], they introduce a significant privacy risk due to the requisite direct interactions with the unlabeled test data.

To accurately estimate model performance on unlabeled test datasets, we propose the Distribution-Aware Adversarial Perturbation method (DAAP). This method bridges the research gap between model accuracy estimation and avoiding the test data and model privacy leakages. The key idea of DAAP involves generating the distribution-aware adversarial perturbation that is applied to publicly available datasets, modifying their distribution to closely resemble that of the private test dataset. The accuracy of the model on the perturbed public datasets serves as a proxy for its accuracy on the unlabeled test dataset ensuring that both test data and models remain isolated during the evaluation process and thus preserving their privacy. In addition, we develop two accuracy estimation strategies based on DAAP for both white-box and black-box model assessments, demonstrating the method’s applicability and effectiveness.

5.2 White-box Model Estimation

In the white-box scenario, the data owner supplies a sample input to the test model, allowing the perturbation generator to access the model’s output. We take advantage of the model’s output entropy $\mathcal{H}$ (see Section 3.3 for more details), as an effective indicator for gauging model performance, even in the absence of annotated data. For a given sample $x$, the entropy measures the probability and uncertainty associated with the model’s outputs. A low entropy value indicates that the model’s predictions are precise and confident, whereas a high entropy value implies that the model’s predictions are less certain or more ambiguous.

The overview of the white-box model estimation is shown in Figure 2. It includes a perturbation generator $G(\omega)$ and a test model $M(\theta)$. We utilize a publicly available dataset as an intermediary, with a sample size significantly smaller than that of the unlabeled test dataset. Specifically, an equal number of samples from both the test and public datasets are input into the test model, avoiding the need for the test model to traverse the entire test dataset. We quantify the model performance divergence using the model’s output entropy difference across these two datasets [41]. Simultaneously, the perturbation generator $G(\omega)$ creates distribution-aware adversarial perturbations to minimize the model performance disparities. The key distinction between our approach and existing methods is that DAAP concentrates on minimizing the overall data distribution discrepancy between the public and unlabeled datasets, rather than focusing on individual samples. The model’s accuracy on the public dataset serves as a proxy for its performance on the test dataset. Consequently, the model accuracy on the public dataset can be used to estimate its accuracy on the unlabeled test dataset. This approach eliminates the need for the test model to process the entire test dataset, protecting the privacy of the test data.
5.3 Black-box Model Estimation

In the black-box scenario, the test model does not have direct access to the test data, and the generator cannot acquire any information about the model. In other words, we cannot evaluate differences in model performance across datasets through the analysis of output entropy. Inspired by adversarial learning, we design a perturbation generator \( G(\omega) \) and a distribution discriminator \( D(\theta) \). The generator creates adversarial perturbations and applies them to a public dataset, aiming to induce the discriminator to misclassify the modified public dataset data as test dataset data. Conversely, the discriminator strives to accurately differentiate the distributions of the public and test data, as shown in Figure 3. In addition, we minimize the discriminator’s output entropy \( H(D(x_{p}^{'}, \theta)) \) and \( H(D(x_{t}, \theta)) \) during the adversarial learning. This addition enables the discriminator to identify the source of the data with high confidence, thereby enhancing the effectiveness of adversarial training. The training generator and discriminator loss can be written as:

\[
\begin{align*}
\min_{\omega} L_G &= \mathbb{E}_{x_{p} \in \mathcal{D}_{pub}} [J(D(x_{p}^{'}, \theta)) + J(D(x, \theta))], \\
\min_{\theta} L_D &= \mathbb{E}_{x_{t} \in \mathcal{D}_{test}} [J(D(x_{t}, \theta))] + \mathbb{E}_{x_{p} \in \mathcal{D}_{pub}} [J(D(x_{p}^{'}, \theta))], \\
&\text{s.t. } \min_{\theta} H(D(x_{p}^{'}, \theta)), x_{p} \in \mathcal{D}_{pub} \quad \text{(8)} \\
&\min_{\theta} H(D(x_{t}, \theta)), x_{t} \in \mathcal{D}_{test} \\
&x_{p}^{'} = G(x_{p}, \omega) + x_{p}, \quad \|G(x_{p}, \omega)\|_{l} \leq \epsilon,
\end{align*}
\]

where \( J(\cdot) \) is the cross-entropy loss, and \( y_{dis} \) is the label of data source. When the data comes from the public dataset, its value is 1, and when the data comes from the test dataset, its value is -1. The loss \( L_G \) of the generator and \( L_D \) of the discriminator are calculated based on the source labels. These losses are backpropagated to update their parameters, aiming to minimize the distribution difference between the public and test datasets. It is noteworthy that the model does not participate in the generator training and accuracy estimating. The generated perturbation is independent of the test model, ensuring that the accuracy of any test models on the unlabeled test dataset can be predicted based on their accuracy on the public dataset.

5.4 DAAP Implementation

In this part, we provide a detailed description of the joint learning process and the accuracy estimation algorithm for the white-box model and black-box model.

5.4.1 White-box Model Implementation

In the estimation process for the white-box model, the test data owner is granted access to the model’s outputs. To mitigate privacy leakage of the test data, a limited and labeled publicly dataset is employed as an intermediary. This intermediary facilitates interactions between the model and the test data. Meanwhile, the generator \( G(\omega) \) is responsible for creating perturbations that are applied to the public dataset. These perturbations are designed to align the model’s performance across the test and public datasets by minimizing the loss function, denoted as \( L_G \) in Equation 7. After the training phase, the model’s accuracy on the public dataset is expected to closely match its accuracy on the test dataset. This methodology not only estimates the model’s accuracy on unlabeled test data but also ensures the preservation of data privacy throughout the evaluation process. Detailed training and estimation algorithms for DAAP can be found in Algorithm 1.
Algorithm 1 Training of DAAP for estimating the white-box model accuracy.

Input: The perturbation generator $G(;\omega)$ and the test model $M(;\theta)$ have parameters $\omega$ and $\theta$, respectively. $x_p$ and $y_p$ denote a public sample and its corresponding label, while $x_t$ represents a test sample. The variable $\hat{y}$ denotes the test model’s prediction, and $y$ is the ground truth label. A publicly available and labeled dataset is $D_{pub}$, and an unlabeled test dataset is $D_{test}$. The total number of public data points is $N_{pub}$. $Num$ denotes the number of epochs, and the learning rate is $\eta$.

Output: The model’s estimated accuracy, $\overline{ACC}_{test}$, on the unlabeled test dataset.

1: Initialize the generator $G$, the test model $M$.
2: for epoch $1$ to $Num$ do
3: for batch $x_p, x_t$ in $D_{pub}, D_{test}$ do
4: $x'_p = x_p + G(x_p; \omega)$
5: $\hat{y}_p = M(x'_p; \theta)$
6: $\hat{y}_t = M(x_t; \theta)$
7: # Calculate the loss of entropy difference.
8: $\min L_G = \sum_{i=1}^{N_{pub}} \|H(\hat{y}_p) - H(\hat{y}_t)\|_1$
9: Update $\omega \leftarrow \omega - \eta_\omega \nabla _{\omega} L_G$
10: end for
11: end for
12: # The model accuracy estimation.
13: $D'_p = D_{pub} + G(D_{pub}, \omega)$
14: $\hat{y}'_p = M(x'_p, \theta), (x'_p, y'_p) \in D'_p$
15: $ACC_{pub} = \frac{\sum_{i=1}^{N_{pub}} I[\hat{y}'_p = y'_p]}{N_{pub}}$
16: return $ACC_{pub}$

5.4.2 Black-box Model Implementation

In the estimation process of a black-box model, the model does not have access to the test data, and its outputs are not available. Drawing inspiration from adversarial learning, we introduce a distribution discriminator $D(\theta)$ and a perturbation generator $G(\omega)$. The discriminator aims to accurately identify the source of input samples by minimizing the loss function $L_D$. The objective of the generator is to create perturbations and deceive the discriminator as effectively as possible by minimizing $L_G$. The adversarial training process alternates between the perturbation generator and the distribution discriminator. This iterative procedure updates the generator $G(\omega)$ through the adversarial learning losses $L_G$ and $L_D$ in Equation 8. The training process ends when the discriminator classifies the perturbed public samples as test samples. This approach enables us to estimate the model’s accuracy on the unlabeled test dataset by approximating its performance on the perturbed public dataset. Detailed descriptions of the training and estimation algorithms are provided in Algorithm 2.

Algorithm 2 Training of DAAP for estimating the black-box model accuracy.

Input: The discriminator $D(;\theta)$, the perturbation generator $G(;\omega)$, and the test model $M(;\theta)$ have parameters $\omega$, $\theta$, and $\theta$, respectively. The cross-entropy loss is $J(\cdot)$. $x_p$ and $y_p$ represent a public sample and its corresponding label, while $x_t$ represents a test sample. The label of the data distribution, $y_{dis}$, can take the value of 1 or -1. The publicly available dataset with ground truth labels is $D_{pub}$, and the test dataset is $D_{test}$. The test model’s prediction is $\hat{y}$, and the discriminator’s prediction is $\hat{y}^D$. The total number of public data points is $N_{pub}$, $Num$ denotes the number of epochs, and $\eta$ is the learning rate.

Output: The model’s estimated accuracy, $\overline{ACC}_{test}$, on the unlabeled test dataset.

1: Initialize the generator $G$, the discriminator $D$.
2: for epoch $1$ to $Num$ do
3: for batch $x_p, x_t$ in $D_{pub}, D_{test}$ do
4: $x'_p = x_p + G(x_p; \omega)$
5: $\hat{y}_p = M(x'_p; \theta)$
6: $\hat{y}_t = M(x_t; \theta)$
7: # Calculate the generator loss.
8: $\min L_G = \sum_{i=1}^{N_{pub}} J(\hat{y}_p, -1)$
9: Update $\omega \leftarrow \omega - \eta_\omega \nabla _{\omega} L_G$
10: # Calculate the discriminator loss.
11: $\min L_D = \sum_{i=1}^{N_{pub}} (J(\hat{y}'_p, y_{dis}) + J(\hat{y}'_p, y_{dis}) + H(\hat{y}'_p) + H(\hat{y}'_p))$
12: Update $\theta \leftarrow \theta - \eta_\theta \nabla _{\theta} L_D$
13: end for
14: end for
15: # The model accuracy estimation.
16: $D'_p = D_{pub} + G(D_{pub}, \omega)$
17: $\hat{y}'_p = M(x'_p, \theta), (x'_p, y'_p) \in D'_p$
18: $ACC_{pub} = \frac{\sum_{i=1}^{N_{pub}} I[\hat{y}'_p = y'_p]}{N_{pub}}$
19: return $ACC_{pub}$

6 EXPERIMENTS

In this section, our proposed method (DAAP) is compared comprehensively with existing methods with multiple datasets and models.
Table 1: Results of the estimation model accuracy on the CIFAR-10-C. The best results are highlighted in bold.

(a) Results on the test dataset with Snow.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>75.87%</td>
<td>79.45%</td>
<td>77.55%</td>
<td>77.08%</td>
<td>74.84%</td>
<td>76.16%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>3.58%</td>
<td>1.68%</td>
<td>1.21%</td>
<td>1.03%</td>
<td><strong>0.29%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td>87.11%</td>
<td>89.62%</td>
<td><strong>88.67%</strong></td>
<td>88.14%</td>
<td>85.63%</td>
<td>88.09%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.51%</td>
<td>1.56%</td>
<td>1.03%</td>
<td>1.48%</td>
<td><strong>0.98%</strong></td>
</tr>
</tbody>
</table>

(b) Results on the test dataset with Fog.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>81.50%</td>
<td>82.97%</td>
<td>80.44%</td>
<td>80.31%</td>
<td>80.51%</td>
<td>80.66%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.47%</td>
<td>1.06%</td>
<td>1.19%</td>
<td>0.99%</td>
<td><strong>0.84%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td>92.53%</td>
<td>93.84%</td>
<td>93.79%</td>
<td>93.66%</td>
<td>90.91%</td>
<td>92.15%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.31%</td>
<td>1.26%</td>
<td>1.13%</td>
<td>1.62%</td>
<td><strong>0.38%</strong></td>
</tr>
</tbody>
</table>

(c) Results on the test dataset with Frost.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>78.48%</td>
<td>79.71%</td>
<td>77.60%</td>
<td>79.04%</td>
<td>77.74%</td>
<td>78.07%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.23%</td>
<td>0.88%</td>
<td>0.56%</td>
<td>0.74%</td>
<td><strong>0.41%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td>88.26%</td>
<td>89.98%</td>
<td>89.32%</td>
<td>89.81%</td>
<td>87.07%</td>
<td>88.63%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.72%</td>
<td>1.06%</td>
<td>1.55%</td>
<td>1.19%</td>
<td><strong>0.37%</strong></td>
</tr>
</tbody>
</table>

(d) Results on the test dataset with Gaussian Noise.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>70.28%</td>
<td>72.98%</td>
<td>71.69%</td>
<td>71.31%</td>
<td>71.63%</td>
<td><strong>69.93%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.70%</td>
<td>1.41%</td>
<td>1.03%</td>
<td>1.35%</td>
<td><strong>0.35%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td>76.23%</td>
<td>78.60%</td>
<td>78.99%</td>
<td>78.42%</td>
<td>77.84%</td>
<td><strong>77.06%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.37%</td>
<td>2.76%</td>
<td>2.19%</td>
<td>1.61%</td>
<td><strong>0.83%</strong></td>
</tr>
</tbody>
</table>

(e) Results on the test dataset with Brightness.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>81.13%</td>
<td>82.86%</td>
<td>82.27%</td>
<td>82.19%</td>
<td>79.89%</td>
<td><strong>81.33%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.73%</td>
<td>1.14%</td>
<td>1.06%</td>
<td>1.24%</td>
<td><strong>0.20%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td>92.76%</td>
<td>90.06%</td>
<td>90.96%</td>
<td>93.89%</td>
<td>91.34%</td>
<td><strong>93.54%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.70%</td>
<td>1.80%</td>
<td>1.13%</td>
<td>1.42%</td>
<td><strong>0.78%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shuff-Net</td>
<td>81.91%</td>
<td>85.04%</td>
<td>82.95%</td>
<td>83.76%</td>
<td>78.70%</td>
<td>81.03%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>3.85%</td>
<td>1.76%</td>
<td>2.57%</td>
<td>2.49%</td>
<td><strong>0.16%</strong></td>
</tr>
<tr>
<td>VGG-16</td>
<td>88.63%</td>
<td>86.65%</td>
<td>90.15%</td>
<td>90.12%</td>
<td>87.21%</td>
<td>88.15%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.98%</td>
<td>1.52%</td>
<td>1.49%</td>
<td>1.42%</td>
<td><strong>0.48%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td>81.13%</td>
<td>82.86%</td>
<td>82.27%</td>
<td>82.19%</td>
<td>79.89%</td>
<td><strong>81.33%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.73%</td>
<td>1.14%</td>
<td>1.06%</td>
<td>1.24%</td>
<td><strong>0.20%</strong></td>
</tr>
<tr>
<td>VGG-16</td>
<td>93.33%</td>
<td>95.97%</td>
<td>94.61%</td>
<td>94.42%</td>
<td>91.34%</td>
<td>93.01%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.64%</td>
<td>1.28%</td>
<td>1.09%</td>
<td>1.99%</td>
<td><strong>0.32%</strong></td>
</tr>
</tbody>
</table>

6.1 Experimental Setup

**Test Models.** We trained four prevalent architectural models on the CIFAR-10 and CIFAR-100 datasets [42]: EfficientNet (Eff) [43], ShuffleNet (Shuff) [44], ResNet-18 (Res-18) [45], and VGG-16 (Vgg-16) [46]. In addition, we also trained the VGG-16 and ResNet-18 models on the CelebA dataset [47].

These models are extensively employed in the field of artificial intelligence. EfficientNet optimizes both efficiency and accuracy by adjusting network depth and width, representing state-of-the-art architecture. ShuffleNet, designed for mobile and embedded devices, exemplifies a lightweight architecture. ResNet-18, featuring skip connections among convolutional...
layers, typifies residual architectures. VGG-16, a foundational model for various vision tasks, represents classic deep convolutional network architectures. Collectively, these models span a spectrum from lightweight to deep architectures, offering a comprehensive evaluation of different network types across diverse tasks.

**Test Datasets.** For the test dataset, we utilize CIFAR-10-C, CIFAR-100-C and transformed CelebA, which encompass 15 types of various perturbations and corruptions. These variants are derived from the CIFAR-10, CIFAR-100 and CelebA test datasets through the addition of modifications, such as changes in brightness, blurring of images, and other alterations. The distribution of the modified datasets differs significantly from that of the original datasets due to these corruptions. Following the methodology outlined in [28], we select five specific modified datasets from both CIFAR-10-C and CIFAR-100-C for detailed analysis. We randomly select 1,000 samples from CIFAR-10, CIFAR-100, and CelebA as public datasets, respectively, when evaluating the model’s accuracy on the CIFAR-10-C, CIFAR-100-C, and transformed CelebA datasets. These samples are excluded from model training.

**Baselines and Metric.** To demonstrate the performance superiority of our proposed Distribution-Aware Adversarial Perturbation (DAAP) method, we employed several prevalent accuracy estimation techniques: Average Confidence (AC), Average Thresholded Confidence (ATC), and Difference of Confidence (DoC). These methods are designed based on confidence measures. Additionally, the Aries method estimates model accuracy by matching the output distribution on annotated data with that on unlabeled test data. We confined the perturbations to within adversarial sample guidelines, i.e., limiting perturbations to $\frac{\delta}{255}$. Furthermore, we utilized two widely acknowledged metrics, Maximum Mean Discrepancy (MMD) and Earth Mover’s Distance (EMD), to quantify the distribution disparity between the test dataset and the publicly available dataset. The two metrics can be used to illustrate the efficacy of our approach using the distribution difference function. All experiments were conducted on a server equipped with an NVIDIA Tesla V100 16GB GPU.

### 6.2 White-box Model Estimation Results

To demonstrate the effectiveness of our proposed Distribution-Aware Adversarial Perturbation (DAAP) in white-box model
estimation, we evaluated the test model accuracy on five transformed datasets from CIFAR-10-C, derived from CIFAR-10. It is important to note that traditional methods such as AC, ATC, DoC, and Aries require processing the entire test dataset, potentially leading to privacy breaches of the test data. In contrast, DAAP employs a publicly available dataset as a buffer between the unlabeled test dataset and the test model, utilizing only 10% of the original test dataset’s size. This approach significantly reduces the need to directly process all test data, thereby enhancing privacy protection for the test data.

The experimental results on the CIFAR-10-C dataset are detailed in Table 1. Across multiple datasets, the average estimation errors for different methods are as follows: 2.28% for AC, 1.50% for ATC, 1.36% for DoC, and 1.42% for Aries. In comparison, our DAAP method demonstrates a significantly lower average estimation error of 0.53%. Notably, DAAP improves the average estimation accuracy by 0.83% compared to DoC, the previously best-performing method.

Our approach utilizes a small public dataset as an intermediary between the test dataset and the model, effectively achieving superior estimation accuracy on the CIFAR-10-C test dataset compared to existing methods. We can obtain similar conclusions on the CIFAR-100-C and CelebA datasets (see Section A for more details). Moreover, our method (DAAP) substantially reduces the risk of privacy breaches by avoiding the need for direct access to the entire test datasets.

6.3 Black-box Model Estimation Results

For black-box model estimation, the test data owner does not have any information about the test model and the generator training does not rely on the test model. Additionally, there is no direct interaction between the test model and the test data during the model accuracy estimation phases. Achieving this is difficult with existing methods. However, we demonstrate the effectiveness of our method (DAAP) for black-box model estimation using various transformed datasets.

The experimental results for the CIFAR-10-C dataset are detailed in Table 2(a). For the CIFAR-10-C-Snow dataset, the estimation errors for the EfficientNet, ShuffleNet, ResNet-18, and VGG-16 models are 1.68%, 0.33%, 0.28%, and 0.31%, respectively. Our method (DAAP) achieves an average estimation error of 1.30%, representing an average improvement of 0.66% in accuracy compared to the methods listed in Table 1(a). For the CIFAR-10-C-Fog dataset, our method records an average estimation error of 0.74%, improving estimation accuracy by 0.77% on average relative to the methods in Table 1(b). Similarly, for the CIFAR-10-C-Frost dataset, DAAP’s average estimation error is 1.13%, which is an improvement of 0.53%, 0.07%, and 0.13% over the AC, ATC, and DoC methods, respectively, as recorded in Table 1(c). On the CIFAR-10-C-GN dataset, our method’s average estimation error is 1.17%, with a 0.63% average improvement in accuracy compared to the methods in Table 1(d). Lastly, for the CIFAR-10-C-Brightness dataset, the average estimation error with our method is only 0.96%, improving accuracy by 0.69% on average against the methods in Table 1(e).

We also obtained similar conclusions on the CIFAR-100-C and CelebA datasets, as detailed in Table 2(b) and Table 2(c). In summary, our method demonstrates robust performance on both the CIFAR-10-C, CIFAR-100-C and CelebA test datasets for black-box model estimation. DAAP achieves
superior results on several datasets. Importantly, our approach provides accurate model estimation on unlabeled test datasets without requiring interaction between the test data and the models. This method significantly enhances the privacy and security of both the test data and the test models.

6.4 Quantitative Assessment of DAAP with Different Perturbation Ranges

To quantitatively assess the performance across various adversarial perturbation ranges, we generate adversarial perturbations at different values of \( \varepsilon \). As shown in the first row of Figure 4, the EfficientNet model achieves an accuracy of 81.03% on the natural public dataset (where \( \varepsilon = 0 \)). When \( \varepsilon \leq \frac{2}{255} \), model accuracy on the public dataset increases, illustrating that minor perturbations can reduce overfitting to the training dataset. As the value of \( \varepsilon \) increases, the model accuracy approaches that of the test dataset. Notably, at \( \varepsilon = \frac{9}{255} \), the estimation error of DAAP is only 0.29%, demonstrating better performance compared to existing estimation methods. However, with \( \varepsilon \geq \frac{9}{255} \), excessive perturbation impairs the information content of the samples, resulting in a significant decline in model accuracy on the public dataset. The estimated accuracy also diverges significantly from the true accuracy on the test datasets. Similar trends are observed on the other dataset in the second row of Figure 4.

6.5 Quantitative Assessment of DAAP with Different Public Data Amounts

To quantitatively analyze the impact of varying public data volumes on model accuracy, we generate distribution-aware adversarial perturbations and apply them to different amounts of public datasets. We use ResNet-18 as the architecture for the test model, and the test datasets include CIFAR10-C-Snow, CIFAR10-C-Fog, CIFAR100-C-Snow, and CIFAR100-C-Fog. As shown in Figure 5, the first row presents the accuracy estimation results in a white-box scenario, while the second row shows the results in a black-box scenario.

For the ten-category task model, our DAAP method achieves high evaluation accuracy, even with a relatively small dataset volume. This effectiveness is due to DAAP’s strategic use of a minimal amount of public data that includes all categories in the test dataset. Consequently, the perturbation generator benefits from access to samples from all categories during the training phase. This significantly enhances the ability of the generated distribution-aware adversarial perturbations to modify the distribution of categories involved in generator training.

For more complex classification tasks, a small amount of public data often fails to encompass all categories in the test dataset. The perturbation generator lacks access to samples from all categories during training, which limits the ability of the generated distribution-aware adversarial perturbations to alter the distribution of categories unrepresented in the generator training. Consequently, the models exhibit low estimation accuracy with limited public data. However, as the volume of public data increases, the accuracy of model estimation improves. This improvement demonstrates that for tasks with a large number of categories, the estimation accuracy on unlabeled test datasets can be enhanced by increasing the public data volume to include all sample categories. These experimental findings highlight the impact of varying amounts of public data on model estimation accuracy, demonstrating that
with more public data, the estimation precision of our method on unlabeled test datasets becomes increasingly accurate.

### 6.6 Comparison with Unlearned Distribution Functions

In this paper, our objective is to align the distribution of the publicly available dataset with that of the test dataset. This alignment enables us to gauge model accuracy on the publicly available dataset and extrapolate accuracy on the unlabeled test dataset. To illustrate the efficacy of the distribution difference functions, we utilize two widely acknowledged metrics: Maximum Mean Discrepancy (MMD) [48] and Earth Mover’s Distance (EMD) [49]. These metrics quantify the distribution disparity between the test dataset and the publicly available dataset and guide perturbation generator training. In this section, the test datasets are derived from the conversion of CIFAR-10.

The MMD quantifies the average discrepancy between the distances of two distributions within a feature space. This metric is especially valuable for evaluating the similarity. The EMD, also known as the Wasserstein distance, calculates the minimal cost required to transform one distribution into another. It is an effective tool for comparing the similarity between two images, with a particular focus on structural measures. SSIM is an index that measures the similarity between two images, with a particular focus on structural measures.

#### 6.7 Member Inference Attack Analysis

DAAP maintains a strict separation between the public and test datasets and applies only minimal perturbations to the public dataset. These perturbations are carefully controlled to ensure that they do not significantly alter individual samples or cause them to closely resemble those in the test dataset. Moreover, our method primarily aims to minimize performance discrepancies across the entire public and test datasets, rather than on an individual sample basis. Consequently, our method effectively preserves the privacy of the test data against Member Inference Attacks (MIA), as DAAP does not generate sample-specific adversarial perturbations.

To demonstrate that individual samples are not significantly altered and that the risk of MIA is effectively mitigated, we measure the similarity between the perturbed public dataset and the test dataset using the Structural Similarity Index (SSIM) [50]. SSIM is an index that measures the similarity between two images, with a particular focus on structural information. The SSIM formula is as follows:

\[
SSIM(x, y) = \frac{(2\mu_x\mu_y + C1)(2\sigma_{xy} + C2)}{({\mu_x^2 + \mu_y^2 + C1})({\sigma_x^2 + \sigma_y^2 + C2})},
\]  

\[\text{(9)}\]
new approach to address privacy concerns. Our extensive evaluation on the CIFAR-10-C, CIFAR-100-C, and CelebA datasets showcases the efficacy of DAAP, highlighting its robustness, high precision, and security for the estimation of model accuracy across various application domains.

7 CONCLUSION

In this paper, we have introduced DAAP, a novel Distribution-Aware Adversarial Perturbation method designed to estimate the accuracy of deep learning models on unlabeled test datasets while ensuring the privacy of both data and models. The key idea of DAAP involves generating distribution-aware adversarial perturbations that are applied to publicly available datasets, modifying their distribution to closely resemble that of the private test dataset. The model's accuracy on the perturbed public datasets serves as a proxy for its accuracy on the unlabeled test dataset. The DAAP eliminates the need for direct interaction between the model and test data, thereby mitigating potential privacy risks. Our extensive evaluation on the CIFAR-10-C, CIFAR-100-C, and CelebA datasets showcases the efficacy of DAAP, highlighting its robustness, high precision, and security for the estimation of model accuracy across various application domains.

ACKNOWLEDGMENTS

This work was supported by Key R&D Program of Zhejiang (Grant No. 2024C01164, 2022C01018), National Key R&D Program of China (Grant No. 2021ZD0112803), and National Natural Science Foundation of China (Grants No. 62122066, U20A20182).

References


A Results of White-box Estimation

To demonstrate the generalization and flexibility of DAAP, we conducted estimation experiments across various architectures and tasks, such as CIFAR-100-C and CelebA.

A.1 Results on the CelebA

For the white-box model estimation, the accuracy of the smile detection model is evaluated, as shown in Table 5. For example, the ResNet-18 model on the Frost dataset records an actual accuracy of 90.10%. DAAP estimates the model’s accuracy at 90.68%, resulting in a minimal estimation error of 0.58%, the lowest among the compared existing estimation methods. We conducted assessments across various datasets, yielding the following average estimation errors: AC at 2.16%, ATC at 1.30%, and DoC at 1.13%. The Aries method averages an error of 1.09%. Our DAAP method outperforms all others with a lower average error of 0.65%. Notably, DAAP surpasses the top-performing DoC method, further improving estimation accuracy by an additional 0.44% on average.

A.2 Results on the CIFAR-100-C

To illustrate the efficacy of our method across complex datasets, we trained four architectural models on CIFAR-100 and subsequently estimated their accuracies on five derived datasets, each transformed based on CIFAR-100. The experimental outcomes for the CIFAR-100-C dataset are documented in Table 6. For the EfficientNet model, the observed accuracy on the Snow dataset is 38.66%, whereas our method estimated the accuracy to be 38.21%, resulting in an estimation error of 0.45%. This represents an improvement of 0.44% in estimation accuracy on average compared to the best-performing Degree of Certainty (DoC) method. Similarly, the ShuffleNet model exhibited an actual accuracy of 48.18% on the same dataset, with our method estimating it at 47.64%, resulting in an estimation error of 0.54%. Our method surpasses the Aries method, improving the estimation accuracy by an average of 0.48%. For the ResNet-18 model, the accuracy was recorded at 53.28%, with our method estimating it at 53.06%, corresponding to an estimation error of 0.17%. Compared to DoC, the accuracy further improves by 0.42% on average. Lastly, the VGG-16 model achieved an

<table>
<thead>
<tr>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-18</td>
<td>Est</td>
<td>86.90%</td>
<td>89.02%</td>
<td>88.34%</td>
<td>87.65%</td>
<td>87.72%</td>
<td>87.13%</td>
<td>VGG-16</td>
<td>Est</td>
<td>84.30%</td>
<td>86.95%</td>
<td>82.61%</td>
<td>83.24%</td>
</tr>
<tr>
<td></td>
<td>Err</td>
<td>0.00%</td>
<td>2.12%</td>
<td>1.44%</td>
<td>0.75%</td>
<td>0.82%</td>
<td>0.23%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-18</td>
<td>Est</td>
<td>83.84%</td>
<td>86.45%</td>
<td>85.12%</td>
<td>82.24%</td>
<td>82.81%</td>
<td>84.71%</td>
<td>VGG-16</td>
<td>Est</td>
<td>91.98%</td>
<td>94.06%</td>
<td>93.11%</td>
<td>90.17%</td>
</tr>
<tr>
<td></td>
<td>Err</td>
<td>0.00%</td>
<td>2.61%</td>
<td>1.28%</td>
<td>1.60%</td>
<td>1.03%</td>
<td>0.87%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-18</td>
<td>Est</td>
<td>90.10%</td>
<td>92.93%</td>
<td>91.12%</td>
<td>91.01%</td>
<td>91.27%</td>
<td>90.68%</td>
<td>VGG-16</td>
<td>Est</td>
<td>92.38%</td>
<td>94.77%</td>
<td>93.49%</td>
<td>93.41%</td>
</tr>
<tr>
<td></td>
<td>Err</td>
<td>0.00%</td>
<td>2.83%</td>
<td>1.02%</td>
<td>0.91%</td>
<td>1.17%</td>
<td>0.58%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-18</td>
<td>Est</td>
<td>90.28%</td>
<td>89.03%</td>
<td>91.20%</td>
<td>91.41%</td>
<td>91.65%</td>
<td>91.04%</td>
<td>VGG-16</td>
<td>Est</td>
<td>90.46%</td>
<td>92.39%</td>
<td>91.68%</td>
<td>92.00%</td>
</tr>
<tr>
<td></td>
<td>Err</td>
<td>0.00%</td>
<td>1.25%</td>
<td>0.92%</td>
<td>1.13%</td>
<td>1.37%</td>
<td>0.76%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
<th>Test_model</th>
<th>R_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Res-18</td>
<td>Est</td>
<td>82.82%</td>
<td>84.79%</td>
<td>84.64%</td>
<td>84.08%</td>
<td>83.88%</td>
<td>82.01%</td>
<td>VGG-16</td>
<td>Est</td>
<td>88.15%</td>
<td>90.10%</td>
<td>89.72%</td>
<td>90.17%</td>
</tr>
<tr>
<td></td>
<td>Err</td>
<td>0.00%</td>
<td>1.97%</td>
<td>1.82%</td>
<td>1.26%</td>
<td>1.06%</td>
<td>0.81%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 6: Results of the estimation model accuracy on the CIFAR-100-C. The best results are highlighted in bold.

(a) Results on the test dataset with Snow.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>38.66%</td>
<td>40.49%</td>
<td>39.73%</td>
<td>39.87%</td>
<td>39.57%</td>
<td>38.21%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.83%</td>
<td>1.07%</td>
<td>1.21%</td>
<td>0.91%</td>
<td><strong>0.45%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>53.23%</td>
<td>51.49%</td>
<td>51.71%</td>
<td>52.16%</td>
<td>53.89%</td>
<td>53.06%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.74%</td>
<td>1.52%</td>
<td>1.07%</td>
<td>0.66%</td>
<td><strong>0.17%</strong></td>
</tr>
</tbody>
</table>

(b) Results on the test dataset with Fog.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>45.34%</td>
<td>47.26%</td>
<td>46.34%</td>
<td>46.16%</td>
<td>45.02%</td>
<td><strong>32%</strong></td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.92%</td>
<td>1.16%</td>
<td>1.00%</td>
<td>0.82%</td>
<td></td>
</tr>
<tr>
<td>Res-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>57.33%</td>
<td>55.43%</td>
<td>55.98%</td>
<td>56.27%</td>
<td>54.89%</td>
<td>56.41%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.90%</td>
<td>1.35%</td>
<td>1.06%</td>
<td>0.62%</td>
<td></td>
</tr>
</tbody>
</table>

(c) Results on the test dataset with Frost.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>39.96%</td>
<td>38.12%</td>
<td>41.38%</td>
<td>41.20%</td>
<td>39.08%</td>
<td>39.56%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.84%</td>
<td>1.42%</td>
<td>1.24%</td>
<td>0.88%</td>
<td><strong>0.40%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>51.66%</td>
<td>53.70%</td>
<td>53.06%</td>
<td>50.77%</td>
<td>50.28%</td>
<td>50.95%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.04%</td>
<td>1.40%</td>
<td>0.89%</td>
<td>1.38%</td>
<td><strong>0.71%</strong></td>
</tr>
</tbody>
</table>

(d) Results on the test dataset with Gaussian Noise.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>32.39%</td>
<td>34.39%</td>
<td>33.69%</td>
<td>31.47%</td>
<td>33.55%</td>
<td>33.06%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.00%</td>
<td>1.30%</td>
<td>0.92%</td>
<td>1.16%</td>
<td><strong>0.67%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>38.06%</td>
<td>39.80%</td>
<td>40.66%</td>
<td>39.65%</td>
<td>39.52%</td>
<td>39.03%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.74%</td>
<td>2.60%</td>
<td>1.59%</td>
<td>1.46%</td>
<td><strong>0.97%</strong></td>
</tr>
</tbody>
</table>

(e) Results on the test dataset with Brightness.

<table>
<thead>
<tr>
<th>Test_model</th>
<th>Real_ACC</th>
<th>AC</th>
<th>ATC</th>
<th>DoC</th>
<th>Aries</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eff-Net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>45.56%</td>
<td>43.07%</td>
<td>44.10%</td>
<td>45.15%</td>
<td>46.78%</td>
<td>46.31%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>2.49%</td>
<td>1.46%</td>
<td>0.41%</td>
<td>1.22%</td>
<td><strong>0.75%</strong></td>
</tr>
<tr>
<td>Res-18</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Est</td>
<td>62.18%</td>
<td>60.66%</td>
<td>63.10%</td>
<td>63.33%</td>
<td>61.07%</td>
<td>61.32%</td>
</tr>
<tr>
<td>Err</td>
<td>0.00%</td>
<td>1.52%</td>
<td>0.92%</td>
<td>1.15%</td>
<td>1.11%</td>
<td><strong>0.86%</strong></td>
</tr>
</tbody>
</table>

accuracy of 45.97% on the Snow dataset, with our estimated accuracy at 46.59%, resulting in an estimation error of 0.62%. This estimation is more accurate by 0.28% on average when compared with the better-performing Aries method.