A. Background

A.1. Machine Learning

We consider supervised classification tasks (Murphy, 2012; Shalev-Shwartz & Ben-David, 2014), wherein a model is trained to predict some class label y, given input data x. Commonly, x may be an image or sentence and y is then the corresponding label, e.g., a digit 0-9 or a text sentiment.

We focus our study on neural networks (Bengio et al., 2017): functions composed as a series of linear-transformation layers, each followed by a non-linear activation. The overall layer structure is called the model's *architecture* and the learnable parameters of the linear transformations are the *weights*. For a classification problem with K-classes, the last layer of a neural network outputs a vector $\mathbf{v}$ of K values (often called logits). The *softmax* function is typically used to convert the logits into normalized confidence scores:[7]. $\text{softmax}(\mathbf{v})_i := e^{v_i}/\sum_{i=1}^{K} e^{v_i} \in [0,1]$. For a model h, we define the model's output $h(x)$ as the vector of softmax values. The model's predicted label is the class with highest confidence, i.e., $\text{argmax}_i\ h(x)_i$.

A.1.1. Data Augmentation

Augmentations are natural transformations of existing data points that preserve class semantics (e.g., small translations of an image), which are used to improve the generalization of a classifier (Cubuk et al., 2018; Sohn et al., 2020; Taylor & Nitschke, 2018). They are commonly used on state-of-the-art models (He et al., 2015; Cubuk et al., 2018; Perez & Wang, 2017) to increase the diversity of the finite training set, without the need to acquire more labeled data (in a costly process). Augmentations are especially important in low-data regimes (Sajjad et al., 2019; Fadaee et al., 2017; Cui et al., 2015) and are domain-specific: they apply to a certain type of input, (e.g., images or text).

We focus on image classifiers, where the main types of augmentations are affine transformations (rotations, reflections, scaling, and shifts), contrast adjustments, cutout (DeVries & Taylor, 2017), and blurring (adding noise). By synthesizing a new data sample as an augmentation of an existing data sample, $x' = \text{augment}(x)$, the model can learn a more semantically-meaningful set of features. Data augmentation can potentially teach the machine learning model to become invariant to the augmentation (e.g., rotationally or translationally invariant).

[7]While it is common to refer to the output of a softmax as a "probability vector" because its components are in the range $[0,1]$ and sum to 1, we refrain from using this terminology given that the scores output by a softmax cannot be rigorously interpreted as probabilities (Gal, 2016)

A.1.2. Transfer Learning

Transfer learning is a common technique used to improve generalization in low-data regimes (Tan et al., 2018). By leveraging data from a *source task*, it is possible to transfer knowledge to a *target task*. Commonly, a model is trained on the data of the source task and then fine-tuned on data from the output task. In the case of neural networks, it is common to fine-tune either the entire model or just the last layer.

A.2. Membership Inference

Membership inference attacks (Shokri et al., 2016) are a form of privacy leakage that identify if a given data sample was in a machine learning model's training dataset. Given a sample x and access to a trained model h, the adversary uses a classifier or decision rule f_h to compute a membership *prediction* $f(x;h) \in \{0,1\}$, with the goal that $f(x;h) = 1$ whenever x is a training point. The main challenge in mounting a membership inference attack is creating the classifier f, under various assumptions about the adversary's knowledge of h and its training data distribution.

Prior work assumes that an adversary has only black-box access to the trained model h, via a query interface that on input x returns part or all of the confidence vector $h(x)$.

Shadow Models The original membership inference attack of Shokri et al. (Shokri et al., 2016) creates a membership classifier $f(x;h)$, tuned on a number of local "shadow" (or, source) models. Assuming the adversary has access to data from the same (or similar) distribution as h's training data, the shadow model approach trains the auxiliary source models $\hat{h}_i$ on this data. Since $\hat{h}_i$ is trained by the adversary, they know whether or not any data point was in the training set, and can thus construct a dataset of confidence vectors $\hat{h}_i$ with an associated membership label $m \in \{0,1\}$. The adversary trains a classifier f to predict m given $\hat{h}_i(x)$. Finally, the adversary queries the targeted model h to obtain $h(x)$ and uses f to predict the membership of x in h's training data.

Salem et al. (Salem et al., 2018) later showed that this attack strategy can succeed even without data from the same distribution as h, and only with data from a similar task (e.g., a different vision task). They also showed that training shadow models is unnecessary: applying a simple threshold predicting $f(x;h) = 1$ (x is a member) when the max prediction confidence, $\max_i h(x)$, is above a tuned threshold, suffices.

Towards Label-only Approaches Yeom et al. (Yeom et al., 2018) propose a simple baseline attack: the adversary predicts a data point x as being a member of the training set when h classifies x correctly. The accuracy of this baseline

attack directly reflects the gap in the model's train and test accuracy: if h overfits (i.e., obtains higher accuracy) on its training data , this baseline attack will achieve non-trivial membership inference. We call this the gap attack. If the adversary's target points are equally likely to be members or non-members of the training set (see Appendix B.2) , this attack achieves an accuracy of

$$1/2 + (\text{acc}_{\text{train}} - \text{acc}_{\text{test}})/2 \ ,$$

where $\text{acc}_{\text{train}}, \text{acc}_{\text{test}} \in [0, 1]$ are the target model's accuracy on training data and held out data respectively.

To the best of our knowledge, this is the only attack proposed in prior work that makes use of only the model's predicted label, $y = \text{argmax}_i \, h(x)_i$. Our goal is to investigate how this simple baseline can be surpassed to achieve label-only membership inference attacks that perform on par with attacks that use access to the model's confidence scores.

Indirect Membership Inference The work of Long et al. (Long et al., 2018) investigates membership inference through *indirect access*, wherein the adversary only queries h on inputs x' that are related to x, but not x directly. Our label-only attacks similarly make use of information gleaned from querying h on data points related to x (specifically, perturbed versions of x).

The main difference is that we focus on label-only attacks, whereas the work of Long et al. (Long et al., 2018) assumes adversarial access to the model's confidence scores. Our attacks will also be allowed to query and obtain the label at the chosen point x.

Adversarial Examples and Membership Inference Song et al. (Song et al., 2019) also make use of adversarial examples to infer membership. Their approach crucially differs from ours in two aspects: (1) they assume access to and predict membership using the confidence scores, and (2) they target models that were explicitly trained to be robust to adversarial examples. In this sense, (2) bares some similarities with our attacks on models trained with data augmentation (see Section 6, where we also find that a model's invariance to some perturbations can leak additional membership signal).

Defenses Defenses against membership inference broadly fall into two categories.

First, standard regularization techniques, such as L2 weight normalization (Shokri et al., 2016; Jia et al., 2019; Truex et al., 2018; Nasr et al., 2018a), dropout (Jia et al., 2019), or differential privacy have been proposed to address the role that overfitting plays in a membership inference attack's success rate (Shokri et al., 2016). Heavy regularization has been shown to limit overfitting and to effectively defend against membership inference, but may result in a significant degradation in the model's accuracy. Moreover, Yeom et al. (Yeom et al., 2018) show that overfitting is sufficient, but not necessary, for membership inference to be possible.

Second, defenses may reduce the information contained in a model's confidences, e.g., by truncating them to a lower precision (Shokri et al., 2016), reducing the dimensionality of the confidence-vector to only some top k scores (Shokri et al., 2016; Truex et al., 2018), or perturbing confidences via an adversary-aware "minimax" approach (Nasr et al., 2018a; Yang et al., 2020; Jia et al., 2019). These defenses modify either the model's training or inference procedure to produce minimally perturbed confidence vectors that thwart existing membership inference attacks. We refer to these defenses as "confidence-masking" defenses.

Outliers in Membership Inference Most membership inference research is focused on protecting the average-case user's privacy: the success of a membership inference attack is evaluated over a large dataset. Long et al. (Long et al., 2018) focus on understanding the vulnerability of *outliers* to membership inference. They show that some (< 100) outlier data points can be targeted and have their membership inferred to high (up to 90%) precision (Long et al., 2017; 2018). Recent work explores how overfitting impacts membership leakage from a defender's (white-box) perspective, with complete access to the model (Leino & Fredrikson, 2019).

B. Evaluation Setup

Because our main goal is to show that label-only attacks can match the success of prior attacks, we consider a similar threat model that matches prior work–except that we restrict the adversary to label-only queries.

As in prior work (Shokri et al., 2016), we assume that the adversary has: (1) full knowledge of the task; (2) knowledge of the target model's architecture and training setup; (3) partial data knowledge, i.e., access to a disjoint partition of data samples from the same distribution as the target model's training data (see below for more details); and (4) knowledge of the targeted points' labels, y.

B.1. Our Threat Model

Generating Membership Data Some works have explored generating data samples x for which to perform membership inference on, which assumes the least data knowledge (Shokri et al., 2016; Fredrikson et al., 2015). These cases work best with minimal numbers of features or binary features because they can take many queries (Shokri et al., 2016). Other works assumes access to the confidence vectors (Fredrikson et al., 2015). Our work assumes that

candidate samples have already been found by the adversary. We leave to future work the efficient discovery of these samples on high-dimensionality data using a label-only query interface.

In our threat model, we always use a disjoint, non-overlapping (i.e., no data points are shared) set of samples for training and test data for the target model. The source model uses another two separate subsets of the task's total data pool. Due to the balanced priors we assume, all subsets (i.e., the target model training and test sets, and the source model training and test sets) are always of the same size. In the case of CIFAR100, we use the target models training dataset (members) as the source models test dataset (non-members), and vice versa.

Model Architectures For computer vision tasks, we use two representative model architectures, a standard convolutional neural network (CNN) and a ResNet (He et al., 2015). Our CNN has four convolution layers with ReLU activations. The first two 3×3 convolutions have 32 filters and the second two have 64 filters, with a max-pool in between the two. To compute logits we feed the output through a fully-connected layer with 512 neurons. This model has 1.2 million parameters. Our ResNet-28 is a standard Wide ResNet-28 taken directly from (Sohn et al., 2020) with 1.4 million parameters. On Purchase-100, we use a fully connected neural network with one hidden layer of size 128 and the $Tanh$ activation function, exactly as in (Shokri et al., 2016). For Texas-100, Adult, and Locations we mimic this model but add a second hidden layer matching the first.

For the attacks from prior work based on confidence vectors, and our new label-only attacks based on data augmentations, we use shallow neural networks as membership predictor models f. Specifically, for augmentations, we use two layers of 10 neurons and LeakyReLU activations (Maas et al., 2013). The confidence-vector attack models use a single hidden layer of 64 neurons, as originally proposed by Shokri et al. (Shokri et al., 2016). We train a separate prediction model for each class We observe minimal changes in attack performance by changing the architecture, or by replacing the predictor model f by a simple thresholding rule. Our combined boundary distance and augmentation attack uses neural networks as well. For simplicity, our decision boundary distance attacks use a single global thresholding rule, $2,500$ queries, and the L2 distance metric. See Section 3.4 for more details.

B.2. On Measuring Success

Some recent works have questioned the use of (balanced) accuracy as a measure of attack success and proposed other measures more suited for imbalanced priors: where any data point targeted by the adversary is a-priori unlikely to be a training point (Jayaraman et al., 2020). As our main goal is to study the effect of the model's *query interface* on the ability to perform membership inference, we focus here on the same balanced setting considered in most prior work. We also note that the assumption that the adversary has a (near-) balanced prior need not be unrealistic in practice: For example, the adversary might have query access to models from two different medical studies (trained on patients with two different conditions) and might know a-priori that some targeted user participated in one of these studies, without knowing which.

C. Threat Model

The goal of a membership inference attack is to determine whether or not a candidate data point was used to train a given model. In Table 3, we summarize different sets of assumptions made in prior work about the adversary's knowledge and query access to the model.

C.1. Adversarial Knowledge

The membership inference threat model originally introduced by Shokri et al. (Shokri et al., 2016), and used in many subsequent works (Long et al., 2017; Truex et al., 2018; Salem et al., 2018; Song et al., 2019; Nasr et al., 2018b), assumes that the adversary has *black-box* access to the model h (i.e., they can only query the model for its prediction and confidence but not inspect its learned parameters). Our work also assumes black-box model access, with the extra restriction (see Section C.2 for more details) that the model only returns (hard) labels to queries. Though studying membership inference attacks with white-box model access (Leino & Fredrikson, 2019) has merits (e.g., for upper-bounding the membership leakage), our label-only restriction inherently presumes a black-box setting (as otherwise, the adversary could just run h locally to obtain confidence scores). Although we are focused on the label-only domain, our attack methodologies can be applied for analysis in the white-box domain.

Assuming a black-box query interface, there are a number of other dimensions to the adversary's assumed knowledge of the trained model:

Task Knowledge refers to global information about the model's prediction task and, therefore, of its prediction API. Examples of task knowledge include the total number of classes, the class-labels (dog, cat, etc.), and the input format (32×32 RGB or grayscale images, etc.). Task knowledge is always assumed to be known to the adversary, as it is necessary for the classifier service to be useful to a user.

Training Knowledge refers to knowledge about the *model architecture* (e.g., the type of neural network, its number of layers, etc.) and how it was trained (the training

Table 3. **Survey of membership inference threat models**. L is the model's loss function, τ is a calibration term reflecting the difficulty of the sample, θ are the model parameters centered around θ^*, θ_0 are the parameters on all other datapoints (other than x), $\text{aug}(x)$ is a data augmentation of x (e.g., image translation), x' is an adversarial-example of x, and $\text{dist}_h(x, y)$ is the distance from x to the decision boundary. Train, data, label, and model knowledge mean, respectively, that the adversary (1) knows the model's architecture and training algorithm, (2) has access to other samples from the training distribution, (3) knows the true label, y for a given x, and (4) knows the model parameter values.

Query Interface	Attack Feature	Knowledge	Source
confidence vector	$h(x), y$	train, data, label	(Shokri et al., 2016)
confidence vector	$h(x)$	train, data	(Long et al., 2017)
confidence vector	$h(x)$	–	(Salem et al., 2018)
confidence vector	$\mathcal{L}\left(h\left(x\right),\ y\right)$	label	(Yeom et al., 2018)
confidence vector	$\mathcal{L}\left(h\left(x\right),\ y\right) + \tau(x)$	label	(Sablayrolles et al., 2019)
confidence vector	$-(\theta - \theta_0^*)^T \nabla_\theta \mathcal{L}\left(h\left(x\right),\ y\right)$	train, data, label, model	(Sablayrolles et al., 2019)
confidence vector	h(x'), y	train, data, label	(Song et al., 2019)
label-only	$\text{argmax}\, h(x), y$	label	(Yeom et al., 2018)
label-only	$\text{argmax}\, h(\text{aug}(x)), y$	train, data, label	ours
label-only	$\text{dist}_h(x, y)$	train, data, label	ours
label-only	$\text{dist}_h(\text{aug}(x), y)$	train, data, label	ours

algorithm, training dataset size, etc). This information could be publicly available or inferable from a model extraction attack (Tramèr et al., 2016; Wang & Gong, 2018).

Data Knowledge constitutes knowledge about the data that was used to train the target model. Full knowledge of the training data renders membership inference trivial because the training members are already known. Partial knowledge may consist in having access to (or the ability to generate) samples from the same or a related data distribution.

Label Knowledge refers to knowledge of the *true* label y for each point x for which the adversary is predicting membership. Whether knowledge of a data point implies knowledge of its true label depends on the application scenario. Salem et al. (Salem et al., 2018) show that attacks that rely on knowledge of query labels can often be matched by attacks that do not.

C.2. Query Interface

Our paper studies a different query interface than most prior membership inference work. The choice of query interface ultimately depends on the application needs where the target model is deployed. We define two types of query interfaces, with different levels of response granularity:

Full confidence vectors On a query x, the adversary receives the full vector of confidence scores $h(x)$ from the classifier. In a multi-class scenario, each value in this vector corresponds to an estimated confidence that this class is the correct label. Restricting access to only part of the confidence vector has little effect on the adversary's success (Shokri et al., 2016; Truex et al., 2018; Salem et al., 2018).

Label-only Here, the adversary only obtains the predicted label $y = \text{argmax}_i\, h(x)_i$, with no confidence scores. This is the minimal piece of information that any query-able machine learning model must provide and is thus the most restrictive query interface for the adversary. Such a query interface is also realistic, as the adversary may only get *indirect* access to a deployed model in many settings. For example, the model may be part of a larger system taking actions based on the model's predictions—the adversary can only observe the system's actions but not the internal model's confidence scores.

In this work, we focus exclusively on the above label-only regime. Thus, in contrast to prior research (Shokri et al., 2016; Hayes et al., 2019; Truex et al., 2018; Salem et al., 2018), our attacks can be mounted against *any* machine learning service, regardless of the granularity provided by the query interface.

D. Confidence-Masking Defense Descriptions

MemGuard This defense solves a constrained optimization problem to compute a defended confidence-vector $h^{\text{defense}}(x) = h(x) + n$, where n is an adversarial noise vector that satisfies the following constraints: (1) the model still outputs a vector of "probabilities", i.e., $h^{\text{defense}}(x) \in [0, 1]^K$ and $\|h^{\text{defense}}(x)\|_1 = 1$; (2) the model's predictions are unchanged, i.e., $\text{argmax}\, h^{\text{defense}}(x) = \text{argmax}\, h(x)$; and (3) the noisy confidence vector "fools" existing membership inference attacks. To enforce the third constraint, the defender

locally creates a membership attack predictor f, and then optimizes the noise n to cause f to mis-predict membership.

Prediction Purification Prediction purification (Yang et al., 2020) is a similar defense. It trains a purifier model, G, that is applied to the output vector of the target model. That is, on a query x, the adversary receives $G(h(x))$. The purifier model G is trained so as to minimize the information content in the confidence vector, whilst preserving model accuracy. While the defense does not guarantee that the model's labels are preserved at all points, the defense is by design incapable of preventing the baseline gap attack, and it is likely that our stronger label-only attacks would similarly be unaffected (intuitively, $G(h(x))$ is just another deterministic classifier, so the membership leakage from a point's distance to the decision boundary should not be expected to change).

Adversarial Regularization This defense trains the target model in tandem with a defensive membership classifier. This defensive membership classifier is a neural network that accepts both the confidence-vector, $h(x)$, of the target model, and the true label, y, that is one-hot encoded. Following the input $h(x)$ there are four fully connected layers of sizes 100, 1024, 512, 64. Following the input y, there are three fully connected layers of sizes 100, 512, 64. The two 64 neuron layers are concatenated (to make a layer of size 128), and passed through three more fully connected layers of sizes 256, 64, and the output layer of size 1. ReLU activations are used after every layer except the output, which uses a sigmoid activation.

The defensive membership classifier and the target model are trained in tandem. First the target model is trained a few (here, 3) epochs. Then for k steps, the defensive membership classifier is trained using an equal batch on members and non-members (which should be different from the held-out set for the target model). After, the target model is trained on one batch of training data. The target model's loss function is modified to include a regularization term using the output of the defensive classifier on the training data. This regularization term is weighted by λ.

E. Description of Common Regularizers

Dropout (Srivastava et al., 2014) is a simple regularization technique, wherein a fraction $\rho \in (0, 1)$ of weights are randomly "dropped" (i.e., set to zero) in each training step. Intuitively, dropout samples a new random neural network at each step, thereby preventing groups of weights from overfitting. At test time, the model is deterministic and uses all the learned weights. We experiment with different dropout probabilities ρ.

L1 and L2 regularization simply add an additional term of the form $\lambda \cdot ||w||$ to the model's training loss, where w is a vector containing all of the model's weights, the norm is either L1 or L2, and $\lambda > 0$ is a hyper-parameter governing the scale of the regularization relative to the learning objective. Strong regularization (i.e., large λ) reduces the complexity of the learned model (i.e., it forces the model to learn smaller weights). We experiment with different regularization constants λ.

Differential privacy guarantees that any output from a (randomized) algorithm on some dataset D, would have also been output with roughly the same probability (up to a multiplicative e^{ϵ} factor) if one point in D were arbitrarily modified. For differential privacy, we use DP-SGD (Abadi et al., 2016), a private version of stochastic gradient descent that clips per-example gradients to an L2 norm of τ, and adds Gaussian noise $\mathcal{N}(0, c^2\tau^2)$ to each batch's gradient. We train target models with fixed parameters $c = 0.5$ and $\tau = 2$. We train for a varied number of steps, to achieve provable differential privacy guarantees for $10 \leq \epsilon \leq 250$.

F. Additional Figures

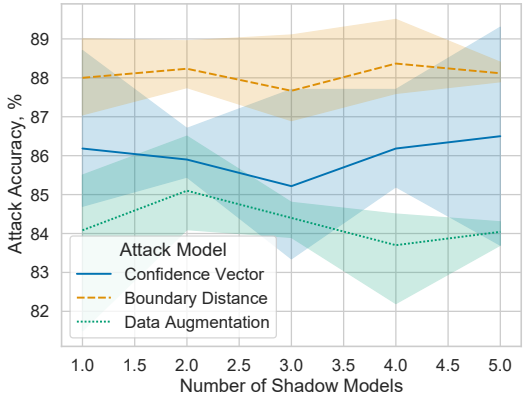

Figure 7. **Attack accuracy of our label-only attacks for various numbers of shadow models.** Target and source models are trained on 1000 data points from CIFAR-10. The number of shadow models **does not** have a significant impact on the attack accuracy.

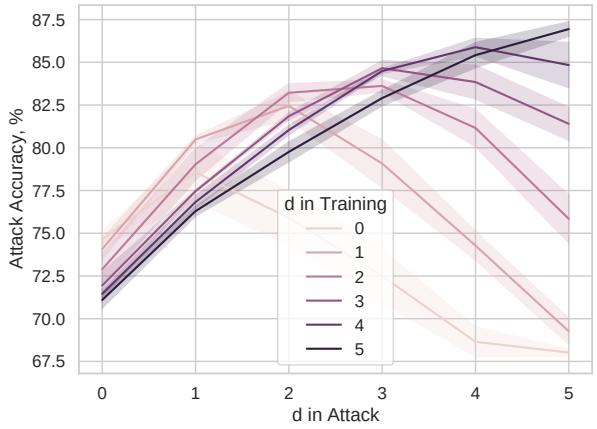

Figure 8. **Attack accuracy of our translation attack for various choices of d.** Target models are trained on 2500 data points from CIFAR-10 with varied sizes of translation augmentations. The attack's accuracy is maximized when it evaluates the same size d of translations as used for training.

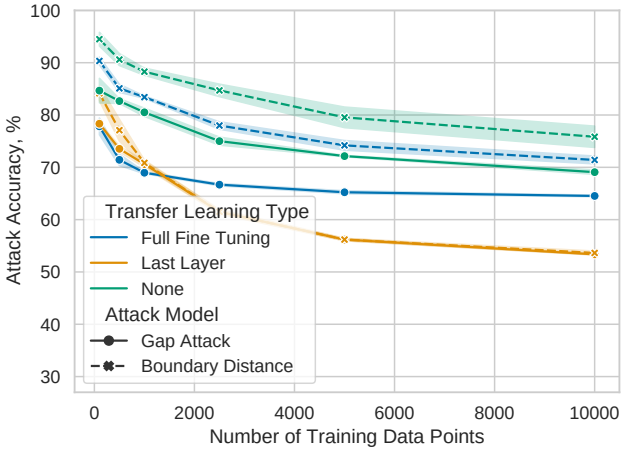

Figure 10. **Accuracy of membership inference attacks on CIFAR-10 models trained with transfer learning.** The source model for transfer learning is trained on all of CIFAR-100. Models are tuned on subsets of CIFAR-10.

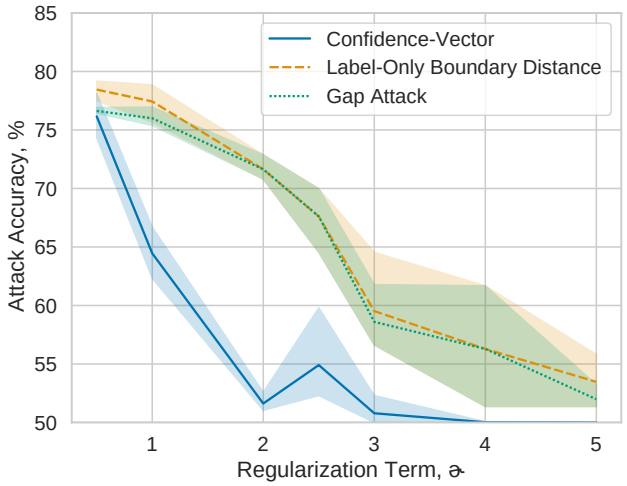

Figure 9. **Accuracy of membership inference attacks on CIFAR-10 models protected with Adversarial Regularization (Nasr et al., 2018a).** Target models are trained on a subset of 2500 images. We test several values of k, the ratio of maximization to minimization steps and find that setting $k = 1$ enabled the target model to converge to a defended state. We report results as we vary the second hyper-parameter, λ, which balances the two training objectives (low training error and low membership leakage). This defense strategy does not explicitly aim to reduce the train-test gap and thus does not protect against label-only attacks. However, we find that this defense prevents attacks from exploiting beyond 3 percentage points of the gap attack. Test accuracy ranges from 45% to 20%, where $\lambda \geq 3$ had a test accuracy below 35%.

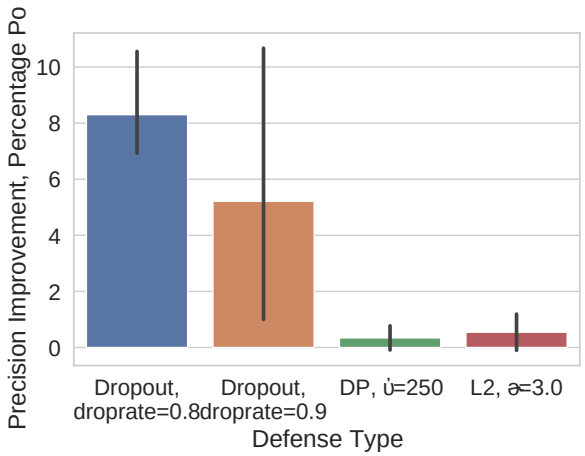

Figure 11. **Outlier membership inference attacks on defended models.** Target and source models are trained on a subset of 2500 points from CIFAR-10. $\beta = 2\%$ outliers are identified with less than $\gamma = 10$ neighbors. We show precision-improvement from the undefended model, using our label-only boundary distance attack.