
Knowledge Distillation Detection for Open-weights Models

Qin Shi^{*1}, Amber Yijia Zheng^{*2}, Qifan Song¹, Raymond A. Yeh²

¹Department of Statistics, Purdue University

²Department of Computer Science, Purdue University
{shi622, zheng709, qfsong, rayyeh}@purdue.edu

^{*}Equal contribution.

Abstract

We propose the task of *knowledge distillation detection*, which aims to determine whether a student model has been distilled from a given teacher, under a practical setting where only the student’s weights and the teacher’s API are available. This problem is motivated by growing concerns about model provenance and unauthorized replication through distillation. To address this task, we introduce a model-agnostic framework that combines data-free input synthesis and statistical score computation for detecting distillation. Our approach is applicable to both classification and generative models. Experiments on diverse architectures for image classification and text-to-image generation show that our method improves detection accuracy over the strongest baselines by 59.6% on CIFAR-10, 71.2% on ImageNet, and 20.0% for text-to-image generation. The code is available at https://github.com/shqi11j/distillation_detection.

1 Introduction

Knowledge distillation [20] transfers learned knowledge from a larger teacher model to a smaller student model to reduce the computational and storage requirements. As opposed to directly training the smaller model from scratch alone, knowledge distillation leads to improved generalization of the student model and has found success in several areas, including image classification [8, 20, 35, 43, 63], large language models [16, 40, 49], and text-to-image generation [23, 28, 45, 56].

While the original intent of knowledge distillation is to build efficient models, distillation techniques lead to an unintended consequence that one could potentially clone proprietary teacher models without permission. This raises concerns about the violation of intellectual property and how to attribute the source of machine learning models [33, 41, 50]. In this work, we aim to study whether it is possible to *detect* if a model has been distilled from a given teacher model, in short, *knowledge distillation detection*. Specifically, we consider the realistic scenario of detecting distillation for open-weight student models, without making any assumptions about the distillation method, training data, or requiring the weights of the teacher model.

We also consider formulating the problem in a multiple-choice setting: given a student model and a set of candidate teachers, predict which teacher was used for distillation. This setting has two advantages for the ease of experiment setup. First, it avoids the need to calibrate an absolute threshold, as the decision is based on comparing across candidates, which is more common in real-world cases. Second, it produces results that are easier to interpret, as one can directly identify the most likely teacher. Note, the detection (binary) setting can be recovered from this formulation by applying a calibrated score function and threshold.

Prior works have considered related problems such as membership inference [5, 22] and out-of-distribution detection [29]. These works demonstrate that models may retain statistical traces of their training data or source model, even after distillation or fine-tuning. In particular, recent work on auditing data usage [22] and memorization in generative models [6, 47] shows that it is possible to identify some aspects of model origin through probing-based methods. However, these approaches typically focus on detecting training data membership, rely on access to the teacher model, or assume architectural similarity. They **do not** directly tackle the problem of determining whether a model is distilled from another model that we are interested in.

To address knowledge distillation detection, we propose a model-agnostic approach that only requires access to the student model’s weights, without requiring the teacher model’s weights or training data. Our approach consists of three stages: input construction, score computation, and decision making. In the first stage, we generate synthetic queries that probe model behavior, using data-free synthesis techniques tailored to the model modality. In the second stage, we extract statistical scores from the model’s responses, capturing either point-wise discrepancies or set-level distributional alignment. Finally, we apply a decision rule to determine whether the model has been distilled, based on these scores. This design is general and thus supports a wide range of models and tasks.

We conduct extensive experiments on both image classification and text-to-image generation tasks. For classification, we test a variety of architectures (ResNet18, DLA, DPN, DenseNet, ResNet50, and ResNeXt50) and distillation methods (KD, RKD, OFA-KD). For text-to-image generation, we report on student models distilled from SD-v2.1, SDXL, and PixArt, including BK-SDM, AMD, DMD2, and SDXL-Lightning. Across both domains, our method improves detection accuracy over the strongest baselines by 59.6% on CIFAR-10, 71.2% on ImageNet, and 20.0% for text-to-image generation.

Our contributions are summarized as follows:

- We introduce the task of knowledge distillation detection of open-weights student models, without access to the teacher model or training data.
- We propose a model-agnostic approach consisting of input construction, score computation, and decision rule stages, adaptable across both classification and generative settings.
- Empirical results on both classification and text-to-image tasks shows the effectiveness and generality of our method.

2 Related Work

Knowledge distillation (KD) is an effective technique for compressing information from large pre-trained models into lightweight student models by aligning either the outputs or intermediate representations of the teacher and student [8, 37, 43, 48, 61, 63, 68]. The concept was first introduced by Buciluă et al. [3] and later popularized by Hinton et al. [20]. For classification tasks, knowledge distillation methods are commonly grouped into three categories: logit-based [20], feature-based [18, 43], and relation-based approaches [10, 35].

While classification-focused KD emphasizes improving accuracy or reducing model size, distillation for diffusion models primarily aims to accelerate generation [23, 28, 31, 32, 42, 44–46, 53, 56]. A typical approach is to train a student model to approximate the teacher’s sampling trajectory, often formulated as an ODE, with significantly fewer inference steps. For instance, AMD [45] distills generative features from the teacher; BK-SDM [23] uses block pruning and feature distillation; and DMD [57] matches the output distribution directly via KL divergence without step-wise supervision. Unlike prior work that focuses on improving the performance of distillation, we study the inverse problem to detect whether one model is distilled from the other.

Data-free quantization and distillation. We review two different tasks that use sample synthesis to train a quantized/student model. One task is data-free model quantization [52], which targets to quantize weights, activations, and even gradients to low-precision, to yield highly compact models without access to the training data. Data-free knowledge distillation [30] is a technique in which a smaller student model is trained to replicate the behavior of a teacher model without access to the original training data. Both tasks share the goal of transferring knowledge from a pre-trained teacher model to a target model using synthetic data. Common approaches include directly learning a fixed set of representative images [4, 27] or training a generator to produce synthetic

inputs [2, 11, 38, 52, 55, 59]. In our work, we also synthesize inputs as we do not assume access to the training data; however, our focused task is different. Rather than aiming for a quantized or distilled model, we generate these samples to collect statistics from the student/teacher model to perform knowledge distillation detection.

AI for good. Open-weight models can be freely shared and modified, amplifying both benefits and risks, which makes safety, transparency, and accountability crucial. Prior work on model immunization [64–67] develops defenses that reduce vulnerabilities of open-sourced image classification and diffusion models to harmful adaptations. Other efforts focus on safety interventions for large language models, such as red-teaming and watermarking [13, 24, 36], to mitigate misuse and enable provenance tracking. Our work is complementary: instead of altering model behavior, we address accountability by detecting whether a student has been distilled from a teacher, providing a mechanism for provenance and trust in open-weights generative models.

3 Background

Knowledge distillation. Let $f : \mathcal{X} \rightarrow \mathcal{Y}$ be the teacher model and $g_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ be the student model, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces. Given a training dataset $\mathcal{D} = \{(\mathbf{x}, \mathbf{y})\}$, knowledge distillation trains the student model by considering both the ground-truth supervision and the knowledge from the teacher model. The training objective can be formulated as

$$\mathcal{L}_{\text{KD}}(\theta) = \sum_{(\mathbf{x}, \mathbf{y}) \in \mathcal{D}} (1 - \lambda) \cdot \ell_{\text{hard}}(g_\theta(\mathbf{x}), \mathbf{y}) + \lambda \cdot \ell_{\text{soft}}(g_\theta(\mathbf{x}), f(\mathbf{x})), \quad (1)$$

where $\lambda \in [0, 1]$ balances the loss term, ℓ_{hard} denotes the loss with respect to the true label $\mathbf{y}$, and ℓ_{soft} compares the student’s and teacher’s outputs, *e.g.*, using KL divergence over softened logits.

4 Knowledge Distillation Detection

Task formulation. Given an open-weight distilled student model g_θ , the goal of distillation detection is to identify which teacher model, from a set $\mathcal{F} \triangleq \{f^{(1)}, f^{(2)}, \dots, f^{(K)}\}$, was used to distill the student. For a realistic setting, we assume the following at prediction time: (a) that there is API access to each of the teacher models, (b) the student model is open-weight, (c) the knowledge distillation algorithm is unknown, and (d) the data used to train or distill the models are unavailable.

More formally, this task can be formulated as a multiple-choice problem for the K teacher candidates, *i.e.*, we aim to develop a detection algorithm

$$\mathcal{A} : \{g_\theta : \mathcal{X} \rightarrow \mathcal{Y}\} \rightarrow \{1, \dots, K\} \quad (2)$$

that maps from a given student model to the index of the teacher model. That is, if the algorithm returns k , then it is most likely that the teacher f_k is used to distill g_θ . Compared with the task of deciding whether a student is distilled from a specific teacher, the multiple-choice setup is easier to interpret. It compares the student against a set of candidate teachers and identifies the most likely source without requiring an explicit threshold; see Sec. 4.1. Moreover, our method naturally extends to the binary setting: one can test whether a student is distilled from a given teacher by applying a threshold to the score. Results of this pairwise detection are discussed in Sec. 5.3.

4.1 Approach

At a high level, determining whether a student model has been distilled from a teacher model involves comparing their outputs. In our setting, however, no data is available; thus, the detection algorithm must first construct synthetic inputs. Once such inputs are obtained, both the teacher and student models are used to produce outputs, which are then compared to compute point-wise scores. These scores are then aggregated to make a final prediction. We illustrate our approach in Fig. 1. As the method’s implementation may vary depending on the task, we first present the general framework, followed by task-specific choices in Sec. 4.2 and Sec. 4.3.

Prediction. Our proposed detection algorithm $\mathcal{A}$ is based on score maximization to identify the teacher model most likely used for distillation. Specifically, the predicted teacher index is given by

$$k^* = \arg \max_{k \in \{1, \dots, K\}} S(g_\theta, f^{(k)}, \mathcal{P}), \quad (3)$$

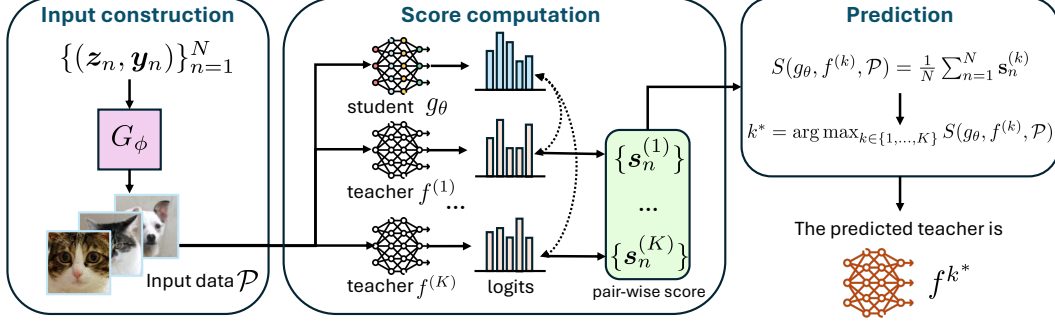


Figure 1: **Knowledge distillation detection pipeline.** The framework consists of three stages: input construction via a generator G , score computation between the student and candidate teachers, and prediction by selecting the teacher with the highest aggregated score.

where S denotes a general score function that quantifies the “alignment” between the student model g_θ and the k^{th} teacher model $f^{(k)}$, evaluated over a constructed input set $\mathcal{P} = \{\mathbf{x}_n\}_{n=1}^N$.

Score computation. The score S plays a central role in our method, which reflects how well the student aligns with each candidate teacher. We consider two types of score functions that capture this alignment at different levels:

- *Point-wise score* S_{point} computes the discrepancy between the student and teacher outputs independently on each input $\mathbf{x}_n \in \mathcal{P}$. Let $\delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0}$ denote a general distance or divergence measure, e.g., KL divergence. For each input $\mathbf{x}_n$, we compute a pairwise score $s_n^{(k)}$ between the student and the k^{th} teacher defined as:

$$s_n^{(k)} = \frac{1}{\delta(g_\theta(\mathbf{x}_n), f^{(k)}(\mathbf{x}_n)) + \epsilon}, \quad (4)$$

where $\epsilon > 0$ is a small constant to ensure numerical stability. The overall point-wise score is then given by the average over all N constructed inputs:

$$S_{\text{point}}(g_\theta, f^{(k)}, \mathcal{P}) = \frac{1}{N} \sum_{n=1}^N s_n^{(k)}. \quad (5)$$

- *Set-level score* S_{set} measure the *population* discrepancy between g_θ and each $f^{(k)}$ by comparing their outputs across the entire input set $\mathcal{P}$. The score takes on the following form:

$$S_{\text{set}}(g_\theta, f^{(k)}, \mathcal{P}) = \Gamma \left(\left\{ \left(g_\theta(\mathbf{x}_n), f^{(k)}(\mathbf{x}_n) \right) \right\}_{\mathbf{x}_n \in \mathcal{P}} \right), \quad (6)$$

where $\Gamma : (\mathcal{Y} \times \mathcal{Y})^N \rightarrow \mathbb{R}$ is a function that operates on the joint set of outputs to capture alignment patterns over the set, such as closeness in the distribution.

Input construction. As discussed, a set of inputs $\mathcal{P} = \{\mathbf{x}_n\}_{n=1}^N$ is required to collect output statistics from both the teacher and student models. Intuitively, the method would benefit from inputs $\mathbf{x}_n$ that yield scores which are “separable”, i.e., they can distinguish the true teacher model used for distillation from other candidate teachers. For example, inputs that are overfitted by both the teacher and student models may be a strong indicator of a distillation relationship. Hence, our input construction strategy is designed to favor inputs with confident predictions from the student model, which tend to improve the discriminability of the subsequent computed scores.

We will next describe the task-specific details for detecting distillation in image classification and text-to-image generation.

4.2 Image Classification Specific Details

In the classification setting, both the teachers $\{f^{(k)}\}$ and the student g_θ are multi-class predictors, i.e., $\mathcal{X} \rightarrow \Delta^C$, where Δ^C denotes the probability simplex over C classes.

For *point-wise score*, we choose the pairwise score $s_n^{(k)}$ to be computed using the KL divergence between the teacher and student’s output. For a generated input $\mathbf{x}$, we compute the discrepancy as $D_{\text{KL}}(g_\theta(\mathbf{x}) \| f^{(k)}(\mathbf{x}))$, and convert it into an aggregated point-wise score S_{point} using the inverse formulation in Eq. (5).

For *set-level score*, we choose Γ to be Aligned Cosine Similarity (ACS) [17], which computes the average cosine similarity between linearly projected student and teacher logits across a set of inputs:

$$\text{ACS}(\mathbf{G}, \mathbf{F}) = \frac{1}{N} \sum_{n=1}^N \frac{\langle (\mathbf{H}\mathbf{G})_{n\cdot}, (\mathbf{H}\mathbf{F}\mathbf{R})_{n\cdot} \rangle}{\|(\mathbf{H}\mathbf{G})_{n\cdot}\|_2 \|(\mathbf{H}\mathbf{F}\mathbf{R})_{n\cdot}\|_2}, \quad (7)$$

where $\mathbf{G} = [g_\theta(\mathbf{x}_1), \dots, g_\theta(\mathbf{x}_N)]^\top$ and $\mathbf{F} = [f^{(k)}(\mathbf{x}_1), \dots, f^{(k)}(\mathbf{x}_N)]^\top$ are matrices of student and teacher logits, $\mathbf{H} = \mathbf{I}_N - \frac{1}{N} \mathbf{1}_N \mathbf{1}_N^\top$ is the centering function and

$$\mathbf{R} = \arg \min_{\mathbf{R} \in \{\mathbf{R}^\top \mid \mathbf{R} = \mathbf{I}\}} \|\mathbf{H}\mathbf{G} - \mathbf{H}\mathbf{F}\mathbf{R}\|_F \quad (8)$$

is a projector for mapping centered $\mathbf{F}$ to centered $\mathbf{G}$. Here, we use an orthogonal transformation obtained by performing singular value decomposition (SVD). This set-level score captures distribution-level alignment between the two models in a compact and representation-aware manner.

To construct the input set $\mathcal{P}$, we aim to generate in-distribution inputs. We adopt a mixup-based synthesis strategy [11, 60], where each generated input is a label-conditioned combination of latent representations from a generator G_ϕ , i.e.,

$$\hat{\mathbf{x}} = G_\phi \left(\sum_{i=1}^C \mathbf{w}_i \cdot e(\mathbf{z}_i, \mathbf{y}_i) \right). \quad (9)$$

Here, the generator G_ϕ is parametrized with a deep net, with $\{\mathbf{z}_i\}_{i=1}^C \sim \mathcal{N}(0, \mathbf{I})$, labels $\{\mathbf{y}_i\}_{i=1}^C \subset \mathcal{Y}$, interpolation weights $\{\mathbf{w}_i\}_{i=1}^C \sim \text{Dir}(\mathbf{1})$ for mixup, and $e(\mathbf{z}_i, \mathbf{y}_i)$ denoting one label-conditioned latent encoding, which is a linear layer that takes in the concatenation of $\mathbf{z}_i$ and $\mathbf{y}_i$.

The generator G_ϕ is optimized to produce data $\hat{\mathbf{x}}$ that has high confidence from g_θ as follows,

$$\min_{\phi} \sum_{i=1}^C \mathbf{w}_i \cdot \mathcal{L}_{\text{hard}}(g_\theta(\hat{\mathbf{x}}(\phi)), \mathbf{y}_i) + \mathcal{L}_{\text{BNS}}(g_\theta, \hat{\mathbf{x}}(\phi)), \quad (10)$$

where we use the cross-entropy loss for $\mathcal{L}_{\text{hard}}$, and $\mathcal{L}_{\text{BNS}}$ is the batch normalization statistics (BNS) alignment loss [55]. Specifically, the BNS loss encourages the generated samples to match the internal activation statistics of real data by aligning the mean and variance at each batch norm layer:

$$\mathcal{L}_{\text{BNS}}(G_\phi, g_\theta) = \sum_{l=1}^L \|\boldsymbol{\mu}_l^r - \boldsymbol{\mu}_l\|_2^2 + \|\boldsymbol{\sigma}_l^r - \boldsymbol{\sigma}_l\|_2^2, \quad (11)$$

where $\boldsymbol{\mu}_l^r$ and $\boldsymbol{\sigma}_l^r$ are the batch mean and variance of the generated data from G_ϕ at the l -th batch norm layer, and $\boldsymbol{\mu}_l, \boldsymbol{\sigma}_l$ are the corresponding running statistics stored in the pre-trained model g_θ .

After the generator G_ϕ is trained, we generate class-conditional synthetic samples for each label $\mathbf{y}$ without mixup by sampling new noise $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ to construct the input set

$$\mathcal{P} = \left\{ G_\phi(e(\mathbf{z}, \mathbf{y})) \mid \mathbf{z} \sim \mathcal{N}(0, \mathbf{I}) \forall \mathbf{y} \right\}. \quad (12)$$

4.3 Text-to-image Generation Specific Details

In the text-to-image setting, both the teachers $\{f^{(k)}\}$ and the student model g_θ are conditional generative models that map a text prompt to an image, i.e., $\mathcal{X} \rightarrow \mathcal{I}$, where $\mathcal{X}$ is the text input space and $\mathcal{I}$ is the image space.

For *point-wise score* S_{point} , we choose to compute pairwise score $s_n^{(k)}$ using LPIPS [62], which measures the semantic similarity between images generated by the student and teacher models for the same prompt and the same noise latent.

For the *set-level score* S_{set} , we choose to apply Centered Kernel Alignment (CKA) [25] with RBF kernels as a similarity index to measure the global alignment between student and teacher models. Concretely, we use a pre-trained CLIP image encoder [39] to extract representation embeddings from images generated by both the student and teacher models, and apply CKA to compare the two distributions of embeddings. Compared to the ACS used for the classification task, the CKA is flexible to complex distributions and robust to high dimensionality, thus it is more suitable for the text-to-image task.

Given two sets of image embeddings $\mathbf{X}$ and $\mathbf{Y}$, each extracted from images generated from the student and teacher models, we first compute the RBF kernel matrices:

$$K_{\mathbf{X}} = \exp\left(-\frac{1}{2\sigma^2}D_{\mathbf{X}}\right), \quad D_{\mathbf{X}} = \text{diag}(\mathbf{X}\mathbf{X}^\top)\mathbf{1}^\top + \mathbf{1}\text{diag}(\mathbf{X}\mathbf{X}^\top)^\top - 2\mathbf{X}\mathbf{X}^\top, \quad (13)$$

and similar for $K_{\mathbf{Y}}$. With the centering function $H = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top$, the CKA score is defined as

$$\text{CKA}(\mathbf{X}, \mathbf{Y}) = \frac{\text{Tr}(\tilde{K}_{\mathbf{X}}\tilde{K}_{\mathbf{Y}})}{\|\tilde{K}_{\mathbf{X}}\|_F \cdot \|\tilde{K}_{\mathbf{Y}}\|_F} \quad \text{where } \tilde{K}_{\mathbf{X}} = HK_{\mathbf{X}}, \tilde{K}_{\mathbf{Y}} = HK_{\mathbf{Y}}. \quad (14)$$

Lastly, to generate the set of outputs from the teacher and student models, we need an input set $\mathcal{P}$. In the case of a text-to-image model, the input requires text prompts to specify what to generate. Here, we simply use the empty string as the prompt. This choice ensures that the input remains in distribution to the model trained for classifier-free guidance. The training involves randomly omitting (replacing with an empty string) the text condition, usually with a drop rate of 0.05, to enable unconditional generation. As a result, the model learns to treat empty prompts as valid inputs, which makes them stable and suitable for evaluating the model’s behavior in distillation detection.

5 Experiments

To evaluate the effectiveness and generalizability of our approach, we conduct experiments on detecting knowledge distillation in two tasks: image classification and text-to-image generation.

Evaluation metrics. As knowledge distillation is formulated as a multi-class classification problem, we consider the following metrics:

1. *Accuracy (Acc.)*: For all student models, we compute the proportion for which the predicted teacher matches the true teacher. That is, we report how often our method correctly identifies the teacher model used for distillation.
2. *Area Under the Curve (AUC)*: For each student model, we view the prediction as a one-vs-rest binary classification by treating the true teacher as the positive class and all others as negative. We compute the area under the ROC curve and report the average AUC across all student models. This metric reflects how well the scores rank the true teacher higher than the incorrect ones.

5.1 Distillation Detection on Image Classification

Datasets and model architectures. For image classification, we use two standard datasets: CIFAR-10 and ImageNet [14]. On CIFAR-10, we select four popular network architectures: ResNet-18 [19], DLA [58], DPN-92 [9], and DenseNet-121 [21]. For each architecture, we first train a teacher model from scratch. Then, for each teacher model, we distill its knowledge into student models of all four architectures, resulting in 16 student models for each knowledge distillation method.

On ImageNet, we consider three architectures: ResNet-18 [19], ResNet-50 [19], and ResNeXt-50 [51]. For each architecture, we use a teacher model pre-trained on ImageNet-1K and distill student models on ImageNet-100 into all three architectures, resulting in 9 student models for each knowledge distillation method. If the logit-matching is required during distillation, we only match the logits corresponding to the ImageNet-100 labels.

Model distillation methods. In image classification tasks, we consider the original knowledge distillation (KD) [20], rational knowledge distillation (RKD) [35], and one-for-all KD (OFA) [18]. Each student model is distilled using one of these approaches from a given teacher model.

Baselines. We explore several different baselines following the general framework of our approach, drawing inspiration from the scores used in related works, such as membership inference attacks,

Table 1: Detecting distillation on CIFAR-10. Each cell reports the mean and standard deviation of Acc./AUC over 10 random seeds. Bold indicates the highest Accuracy or AUC in each column (excluding Oracle). Note that when $N = 1$, the set-level scores cannot be computed.

Method	Input Size N					Average
	1	5	10	50	100	
Oracle	0.45 $\pm$ 0.10/0.64 $\pm$ 0.07	0.58 $\pm$ 0.10/0.77 $\pm$ 0.06	0.65 $\pm$ 0.07/0.82 $\pm$ 0.06	0.87 $\pm$ 0.04/0.96 $\pm$ 0.01	0.95 $\pm$ 0.02/0.99 $\pm$ 0.01	0.70 $\pm$ 0.07/0.84 $\pm$ 0.04
MIA Filter + KL	0.43 $\pm$ 0.06/0.66 $\pm$ 0.04	0.49 $\pm$ 0.05/0.74 $\pm$ 0.02	0.51 $\pm$ 0.04/0.76 $\pm$ 0.02	0.54 $\pm$ 0.03/0.79 $\pm$ 0.01	0.55 $\pm$ 0.02/0.80 $\pm$ 0.01	0.51 $\pm$ 0.04/0.75 $\pm$ 0.02
OOD Filter + KL	0.42 $\pm$ 0.06/0.68 $\pm$ 0.06	0.48 $\pm$ 0.03/0.73 $\pm$ 0.03	0.50 $\pm$ 0.04/0.75 $\pm$ 0.03	0.55 $\pm$ 0.03/0.79 $\pm$ 0.01	0.54 $\pm$ 0.03/0.80 $\pm$ 0.01	0.50 $\pm$ 0.04/0.75 $\pm$ 0.03
MIA Filter + CKA	-	0.32 $\pm$ 0.05/0.58 $\pm$ 0.05	0.38 $\pm$ 0.11/0.64 $\pm$ 0.06	0.64 $\pm$ 0.06/0.85 $\pm$ 0.04	0.72 $\pm$ 0.06/0.90 $\pm$ 0.04	0.52 $\pm$ 0.07/0.74 $\pm$ 0.05
OOD Filter + CKA	-	0.33 $\pm$ 0.09/0.59 $\pm$ 0.06	0.39 $\pm$ 0.06/0.65 $\pm$ 0.06	0.61 $\pm$ 0.08/0.84 $\pm$ 0.04	0.72 $\pm$ 0.05/0.91 $\pm$ 0.02	0.51 $\pm$ 0.07/0.75 $\pm$ 0.05
MIA Filter + MMD	-	0.40 $\pm$ 0.08/0.67 $\pm$ 0.05	0.45 $\pm$ 0.06/0.70 $\pm$ 0.04	0.36 $\pm$ 0.03/0.62 $\pm$ 0.03	0.29 $\pm$ 0.02/0.58 $\pm$ 0.03	0.38 $\pm$ 0.04/0.64 $\pm$ 0.03
OOD Filter + MMD	-	0.47 $\pm$ 0.07/0.70 $\pm$ 0.03	0.47 $\pm$ 0.05/0.71 $\pm$ 0.05	0.38 $\pm$ 0.03/0.65 $\pm$ 0.04	0.28 $\pm$ 0.02/0.59 $\pm$ 0.02	0.40 $\pm$ 0.04/0.66 $\pm$ 0.03
Ours (KL)	0.62$\pm$0.05/0.75$\pm$0.06	0.82$\pm$0.05/0.89$\pm$0.04	0.86$\pm$0.04/0.92$\pm$0.02	0.87 $\pm$ 0.03/0.94 $\pm$ 0.01	0.87 $\pm$ 0.02/0.94 $\pm$ 0.01	0.81 $\pm$ 0.05/0.89 $\pm$ 0.03
Ours (ACS)	-	0.75 $\pm$ 0.04/0.72 $\pm$ 0.01	0.82 $\pm$ 0.04/0.74 $\pm$ 0.01	0.88$\pm$0.03/0.77$\pm$0.01	0.89$\pm$0.04/0.77$\pm$0.01	0.83$\pm$0.04/0.75$\pm$0.01

Table 2: Detecting distillation on ImageNet. Each cell reports Acc./AUC. Bold indicates the best value per column (excluding Oracle).

Method	Input Size N					Average
	1	5	10	50	100	
Oracle	0.56 $\pm$ 0.22/0.71 $\pm$ 0.22	0.67 $\pm$ 0.16/0.88 $\pm$ 0.14	0.77 $\pm$ 0.15/0.94 $\pm$ 0.06	0.84 $\pm$ 0.12/0.99 $\pm$ 0.02	0.91 $\pm$ 0.08/1.00 $\pm$ 0.00	0.75 $\pm$ 0.15/0.90 $\pm$ 0.09
MIA Filter + KL	0.34 $\pm$ 0.03/0.75 $\pm$ 0.05	0.33 $\pm$ 0.01/0.76 $\pm$ 0.03	0.33 $\pm$ 0.00/0.76 $\pm$ 0.02	0.33 $\pm$ 0.00/0.76 $\pm$ 0.01	0.33 $\pm$ 0.00/0.76 $\pm$ 0.01	0.33 $\pm$ 0.01/0.76 $\pm$ 0.03
OOD Filter + KL	0.34 $\pm$ 0.05/0.62 $\pm$ 0.06	0.33 $\pm$ 0.03/0.62 $\pm$ 0.02	0.32 $\pm$ 0.03/0.62 $\pm$ 0.03	0.32 $\pm$ 0.02/0.63 $\pm$ 0.01	0.31 $\pm$ 0.02/0.63 $\pm$ 0.01	0.33 $\pm$ 0.03/0.62 $\pm$ 0.03
MIA Filter + CKA	-	0.41 $\pm$ 0.06/0.56 $\pm$ 0.05	0.33 $\pm$ 0.06/0.51 $\pm$ 0.06	0.38 $\pm$ 0.08/0.54 $\pm$ 0.09	0.42 $\pm$ 0.10/0.55 $\pm$ 0.08	0.39 $\pm$ 0.07/0.54 $\pm$ 0.07
OOD Filter + CKA	-	0.36 $\pm$ 0.09/0.55 $\pm$ 0.10	0.34 $\pm$ 0.11/0.48 $\pm$ 0.09	0.36 $\pm$ 0.09/0.49 $\pm$ 0.12	0.36 $\pm$ 0.07/0.57 $\pm$ 0.05	0.36 $\pm$ 0.09/0.52 $\pm$ 0.09
MIA Filter + MMD	-	0.34 $\pm$ 0.01/0.50 $\pm$ 0.04	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.50 $\pm$ 0.01
OOD Filter + MMD	-	0.34 $\pm$ 0.01/0.47 $\pm$ 0.04	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.50 $\pm$ 0.00	0.33 $\pm$ 0.00/0.49 $\pm$ 0.01
Ours (KL)	0.47$\pm$0.13/0.68$\pm$0.13	0.64$\pm$0.09/0.88$\pm$0.04	0.66$\pm$0.07/0.89$\pm$0.04	0.74$\pm$0.04/0.93$\pm$0.01	0.75 $\pm$ 0.03/0.92 $\pm$ 0.01	0.65 $\pm$ 0.07/0.86 $\pm$ 0.05
Ours (ACS)	-	0.58 $\pm$ 0.08/0.66 $\pm$ 0.07	0.60 $\pm$ 0.09/0.66 $\pm$ 0.04	0.70 $\pm$ 0.07/0.81 $\pm$ 0.03	0.79$\pm$0.02/0.85$\pm$0.02	0.67$\pm$0.06/0.74$\pm$0.04

although our application differs. Each baseline alters one component of our method while keeping the other elements unchanged. Baselines 1-2 focus on the input construction process, while baselines 3-4 concentrate on the score design. Please see the Appendix for more details.

1. *Membership inference attack filter (MIA filter)* [12, 54]: This baseline constructs the input set by identifying samples that resemble training data. We use random noise as input and split the samples into two disjoint subsets. For the first subset, we collect the student model’s output logits and randomly assign Train and Test labels to them. A binary classifier is then trained to distinguish between the two. On the second subset, we collect logits from the student model and pass them through the trained classifier. Inputs classified as Train are retained for detection, while those classified as Test are discarded. KL divergence is used as the score function in the subsequent stage.
2. *Out-of-distribution detector filter (OOD filter)* [29]: This baseline constructs the input set by filtering for in-distribution samples. It computes an energy score from the model’s logits for random noise inputs. Samples with low energy (high confidence) are treated as in-distribution and retained, while high-energy (low-confidence) samples are filtered out. As in our method, the KL divergence is used for scoring.
3. *MMD-FUSE* [1]: It computes the normalized log-sum-exp of Maximum Mean Discrepancy (MMD) statistics between student and teacher logits outputs under permutations.
4. *CKA* [25]: It computes the representational similarity via centered kernel alignment. RBF kernels are applied to the logits of the student and teacher, followed by centering and computation of the Hilbert-Schmidt Independence Criterion (HSIC).

Besides the baselines above, we also introduce **Oracle**, where we directly sample real inputs from the training distribution in the input construction stage and use KL divergence as the score function. This setup illustrates the performance when the original training data is available.

Results. In Tab. 1, we present the results of distillation detection on CIFAR-10 across varying numbers of synthesized inputs $N = |\mathcal{P}|$. Set-level scores cannot be computed when $N = 1$. For point-level score, even with a single input, our method using KL divergence achieves strong performance, with an accuracy of 0.62 and an AUC of 0.75, consistently outperforming all baselines.

As N increases, performance improves for both our methods and the baselines, reflecting the benefit of aggregating more input samples. By $N = 50$, our KL and ACS variants reach AUCs of 0.94 and 0.77, respectively, and accuracy continues to improve. In contrast, baselines tend to plateau around

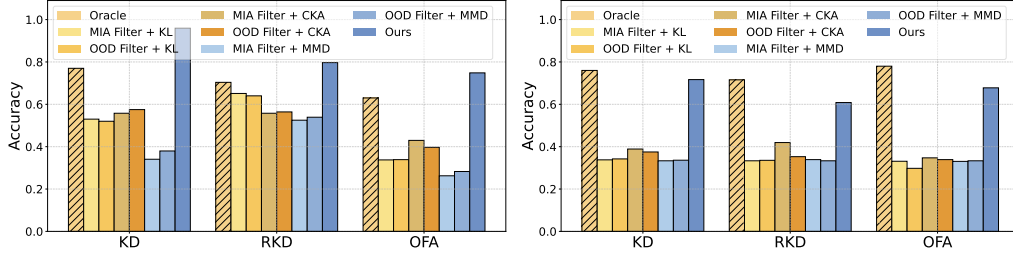


Figure 2: Accuracy of knowledge distillation detection across different distillation methods. CIFAR-10 (left) and ImageNet (right).

0.50–0.55 in accuracy and 0.73–0.80 in AUC. Notably, our methods exhibit greater stability and higher average performance across all settings, even outperforming the Oracle that uses real data.

Tab. 2 presents distillation detection results on ImageNet. As we can observe, our methods consistently outperform the baselines. Notably, with just 5 inputs, our KL-based method achieves an accuracy of 0.64 and AUC of 0.88, which is significantly higher than all baseline combinations, whose accuracy remains around 0.33 and AUC ranges from 0.47 to 0.76. As the number of inputs increases, our performance remains stable and continues to improve, reaching 0.75 accuracy and 0.92 AUC with 100 inputs. On average, our KL variant achieves 0.65/0.86, and ACS achieves 0.67/0.74, both substantially higher than the baselines, which mostly remain around 0.33–0.36 in accuracy.

In Fig. 2, we present the accuracy of detection on CIFAR10 and ImageNet across different distillation methods. Our method consistently achieves the highest accuracy across all distillation types, outperforming all baselines and even the Oracle in CIFAR-10. In addition, our method achieves the best performance under the standard KD setup compared to RKD and OFAKD, suggesting that detection is more effective when the student model is trained using vanilla knowledge distillation.

5.2 Detect Distillation on Text-to-image Models

Model distillation methods and architectures. For the text-to-image models, we use 10 publicly available pre-trained student models distilled from larger teacher models using the following:

1. AMD [45]: a one-step student model distilled from Stable Diffusion 2.1 and a one-step student model distilled from PixArt.
2. DMD2 [56]: four-step distilled models derived from Stable Diffusion XL (SDXL).
3. SDXL-Lightning [28]: student models with 1, 2, 4, and 8 sampling steps, all distilled from Stable Diffusion XL (SDXL).
4. BK-SDM-v2 [23]: improved variants of BK-SDM (base, small, and tiny), distilled from Stable Diffusion 2.1.

Baselines. For detection on text-to-image models, we consider two categories of baselines: captioning-based input construction stage and alignment-based set-level scores. Baseline 1-2 focuses on comparing the input construction process, and 3-6 focuses on the score design.

1. *BLIP-Base* [26]: We use the BLIP-Base model to generate captions for the unconditional generation of the student model. These captions are then used as input for the detection pipeline.
2. *GPT-2* [34]: Similar to BLIP-Base, we use the ViT-GPT2 model to caption for the unconditional generation of the student model, and feed the resulting text for detection.
3. *MMD-FUSE* [25]: As in the classification setting, we calculate p -value as the score to measure the alignment between the student and teacher. Here, it is applied to the generated images instead of the logits.
4. *ACS* [17]: This baseline computes the average cosine similarity between the projected logits of the student and teacher across a set of generated samples.
5. *CLIP* [39]: We compute the cosine similarity between the CLIP image embeddings of student and teacher generations.
6. *DINO* [7]: We extract features from generated images using a pre-trained DINO backbone and compute the cosine similarity between the resulting embeddings of the student and teacher.

Table 3: Detecting distillation in text-to-image generation model. Each cell reports Acc./AUC.

Method	Input Size N					Average
	1	5	10	50	100	
Blip-Base + CLIP	$0.71 \pm 0.10 / 0.78 \pm 0.07$	$0.77 \pm 0.09 / 0.91 \pm 0.04$	$0.81 \pm 0.08 / 0.93 \pm 0.04$	$0.84 \pm 0.05 / 0.96 \pm 0.01$	$0.83 \pm 0.05 / 0.96 \pm 0.01$	$0.79 \pm 0.07 / 0.91 \pm 0.03$
Blip-Base + DINO	$0.76 \pm 0.11 / 0.83 \pm 0.06$	$0.79 \pm 0.09 / 0.93 \pm 0.03$	$0.80 \pm 0.08 / 0.94 \pm 0.03$	$0.80 \pm 0.04 / 0.95 \pm 0.01$	$0.81 \pm 0.05 / 0.95 \pm 0.01$	$0.79 \pm 0.08 / 0.92 \pm 0.03$
Blip-Base + ACS	-	$0.56 \pm 0.08 / 0.52 \pm 0.16$	$0.58 \pm 0.09 / 0.63 \pm 0.17$	$0.59 \pm 0.07 / 0.98 \pm 0.03$	$0.57 \pm 0.06 / 1.00 \pm 0.00$	$0.57 \pm 0.08 / 0.78 \pm 0.09$
Blip-Base + MMD	-	$0.67 \pm 0.10 / 0.96 \pm 0.02$	$0.73 \pm 0.08 / 0.97 \pm 0.01$	$0.58 \pm 0.06 / 0.89 \pm 0.05$	$0.50 \pm 0.00 / 0.67 \pm 0.02$	$0.62 \pm 0.06 / 0.87 \pm 0.03$
GPT-2 + CLIP	$0.70 \pm 0.13 / 0.81 \pm 0.09$	$0.73 \pm 0.06 / 0.89 \pm 0.05$	$0.81 \pm 0.07 / 0.93 \pm 0.02$	$0.81 \pm 0.03 / 0.95 \pm 0.01$	$0.80 \pm 0.00 / 0.95 \pm 0.01$	$0.77 \pm 0.06 / 0.91 \pm 0.04$
GPT-2 + DINO	$0.81 \pm 0.13 / 0.87 \pm 0.09$	$0.79 \pm 0.09 / 0.91 \pm 0.04$	$0.80 \pm 0.06 / 0.94 \pm 0.03$	$0.80 \pm 0.04 / 0.95 \pm 0.02$	$0.80 \pm 0.00 / 0.95 \pm 0.01$	$0.80 \pm 0.07 / 0.92 \pm 0.04$
GPT-2 + ACS	-	$0.51 \pm 0.09 / 0.47 \pm 0.17$	$0.57 \pm 0.11 / 0.58 \pm 0.19$	$0.52 \pm 0.04 / 0.97 \pm 0.04$	$0.54 \pm 0.05 / 1.00 \pm 0.01$	$0.54 \pm 0.07 / 0.75 \pm 0.10$
GPT-2 + MMD	-	$0.64 \pm 0.09 / 0.95 \pm 0.01$	$0.68 \pm 0.06 / 0.96 \pm 0.01$	$0.51 \pm 0.03 / 0.89 \pm 0.03$	$0.50 \pm 0.00 / 0.66 \pm 0.00$	$0.58 \pm 0.05 / 0.87 \pm 0.01$
Ours (LPIPS)	$0.89 \pm 0.05 / 1.00 \pm 0.00$	$0.94 \pm 0.05 / 1.00 \pm 0.00$	$0.97 \pm 0.05 / 1.00 \pm 0.00$	$1.00 \pm 0.00 / 1.00 \pm 0.00$	$1.00 \pm 0.00 / 0.99 \pm 0.01$	$0.96 \pm 0.03 / 1.00 \pm 0.00$
Ours (CKA)	-	$0.67 \pm 0.23 / 0.73 \pm 0.15$	$0.72 \pm 0.15 / 0.83 \pm 0.17$	$0.78 \pm 0.09 / 0.99 \pm 0.02$	$0.80 \pm 0.06 / 1.00 \pm 0.00$	$0.74 \pm 0.13 / 0.89 \pm 0.08$

Table 4: Ablation study on input construction and score computation on classification.

Setting	CIFAR-10	ImageNet	Average
OOD filter + ACS	$0.56 \pm 0.05 / 0.77 \pm 0.04$	$0.37 \pm 0.07 / 0.52 \pm 0.08$	$0.47 \pm 0.06 / 0.65 \pm 0.06$
Synthetic Data + CKA	$0.82 \pm 0.03 / 0.77 \pm 0.01$	$0.62 \pm 0.07 / 0.71 \pm 0.04$	$0.72 \pm 0.05 / 0.74 \pm 0.03$
Synthetic Data + ACS (Ours)	$0.83 \pm 0.04 / 0.75 \pm 0.01$	$0.67 \pm 0.06 / 0.74 \pm 0.04$	$0.75 \pm 0.05 / 0.75 \pm 0.03$

Results. Tab. 3 shows the performance of distillation detection for text-to-image generation models under varying numbers of synthesized inputs. Our methods consistently outperform all baselines in both accuracy and AUC across all input sizes, especially when we apply a point-wise score function. With just a single input, it achieves 0.89 accuracy and 1.00 AUC, already exceeding all competing methods. The strong performance is partly due to the use of CFG during distillation, where the unconditional generation is explicitly modeled, *i.e.*, the empty string is used in the distillation process.

Next, as the number of inputs increases, performance quickly saturates, reaching 1.00 accuracy and 1.00 AUC from $N = 10$ onward. This trend is maintained across all settings, with an average of 0.96/1.00, indicating strong robustness and effectiveness. The best-performing baselines, such as GPT-2 + DINO and Blip-Base + DINO, had around 0.80 accuracy, even with large input sizes.

For set-level scoring approaches, our method using CKA demonstrates clear improvements over other set-based baselines. Its accuracy and AUC increase steadily with N , reaching 0.80/0.99 at $N = 100$. In comparison, other baselines using set-level score ACS or MMD either stagnate or degrade when $N > 10$, indicating limited scalability. These results highlight the effectiveness of our design in leveraging larger input sets for improved detection.

5.3 Ablation and Discussion

Pairwise detection. To show that our multiple-choice setup can be extended to a binary detection, we implement the HSIC test [15] to obtain a p-value as the score for the text-to-image generation task with 3 teacher models and 10 student models. Each of the student models is paired with its true teacher (positive) and two random candidate teachers (negative). A pair is predicted as distillation if the p-value is below 0.05. As shown in Tab. 5, from the 30 teacher-student pairs, accuracy and F1 score are computed from these binary predictions, and AUC is based on the scores across different thresholds for p-value. This method can achieve 0.86 accuracy and 0.91 AUC, which shows that generalizing to the pairwise testing setup is possible. We view this as a natural extension and a valuable direction for future work.

Ablate inputs and scores. We ablate input construction and set-level scoring in Tab. 4 for classification, reporting both accuracy and AUC. Using OOD-filtered inputs with ACS yields the weakest results, especially on ImageNet with 0.37 accuracy and 0.52 AUC, indicating poor alignment with student model behavior. Replacing the input with our synthetic samples and using CKA improves performance significantly, reaching 0.72 accuracy and 0.74 AUC on average. Our full method

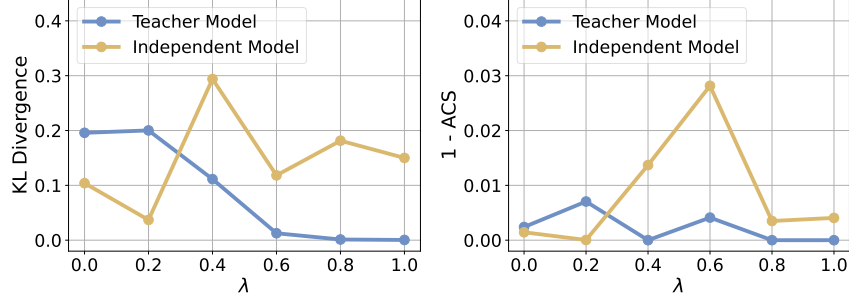


Figure 3: KL divergence (left) and 1 - ACS (right) between the student and teacher (or independent model) outputs as a function of the distillation weight λ . The total loss used to train the student is defined in Eq. (1), combining a hard cross-entropy term and a soft KL divergence term.

Table 5: Performance of distillation detection using the HSIC test on the text-to-image task.

Input Size N	Acc.	AUC	F1
10	0.74 ± 0.05	0.76 ± 0.07	0.43 ± 0.09
50	0.86 ± 0.04	0.91 ± 0.07	0.80 ± 0.06
100	0.80 ± 0.04	0.94 ± 0.04	0.75 ± 0.05

achieves the best overall results 0.75/0.75, outperforming the strongest ablation by 0.03 in accuracy while maintaining robust AUC. These results confirm the importance of both input and score design.

Impact of knowledge transfer strength. Intuitively, the quality/strength of the distillation should impact the effectiveness of our approach. In Fig. 3, we show how our detection method depends on the strength of knowledge transfer during distillation. The parameter λ controls the contribution of ℓ_{soft} in the distillation loss defined in Eq. (1), with higher λ placing greater weights on matching the teacher. We use KL divergence and 1 - ACS as the scores measuring the distance between the two models. We compute the scores between the outputs of the student model and either the teacher model or an independently trained model with the same architecture as the student. When $\lambda < 0.5$, the scores between the student and its teacher could be larger/comparable to those between the student and an independent model with the same architecture, and our detection rule will fail in such cases. As λ increases, the student is more strongly regularized to align with the teacher, resulting in reliably smaller scores and improved detection performance.

6 Conclusion

We introduce the task of knowledge distillation detection that aims to identify whether a student model has been distilled from a given teacher from only the student’s weights and the teacher’s API access. Our model-agnostic framework combines data-free input synthesis with score maximization to make a prediction. The proposed method works on both classification and text-to-image generation. Experiments across diverse architectures and distillation methods show that our approach consistently outperforms strong baselines without requiring access to training data or teachers’ weights. While detection may be less reliable when the student inherits only limited influence from the teacher, performance remains strong in typical distillation scenarios. Looking ahead, extending the framework to free-form inputs in diffusion and large language models represents a promising next step, supported by the model-agnostic nature of our approach.

Acknowledgment

This project is supported in part by an NSF Award #2420724 and the Ross-Lynn Research Scholar Grant. Song’s research is partially supported by the NVIDIA Academic Grant Program Award.

References

- [1] Felix Biggs, Antonin Schrab, and Arthur Gretton. Mmd-fuse: Learning and combining kernels for two-sample testing without data splitting. In *Proc. NeurIPS*, 2023. 7
- [2] Kuluhan Binici, Shivam Aggarwal, Nam Trung Pham, Karianto Leman, and Tulika Mitra. Robust and resource-efficient data-free knowledge distillation by generative pseudo replay. In *Proc. AAAI*, 2022. 3
- [3] Cristian Buciluă, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proc. ACM SIGKDD*, 2006. 2
- [4] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Proc. CVPR*, 2020. 2
- [5] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Membership inference attacks from first principles. In *Proc. S&P*, 2021. 2
- [6] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Nicholas Carlini, Matthew Jagielski, and Nicholas Carlini. Extracting training data from diffusion models. In *Proc. S&P*, 2023. 2
- [7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proc. ICCV*, 2021. 8
- [8] Defang Chen, Jian-Ping Mei, Hailin Zhang, Can Wang, Yan Feng, and Chun Chen. Knowledge distillation with the reused teacher classifier. In *Proc. CVPR*, 2022. 1, 2
- [9] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. Dual path networks. In *Proc. NeurIPS*, 2017. 6
- [10] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *Proc. AAAI*, 2018. 2
- [11] Kanghyun Choi, Deokki Hong, Noseong Park, Youngsok Kim, and Jinho Lee. Qimera: Data-free quantization with synthetic boundary supporting samples. In *Proc. NeurIPS*, 2021. 3, 5
- [12] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *Proc. ICML*, 2021. 7
- [13] Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 2024. 3
- [14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009. 6
- [15] Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. In *Proc. NeurIPS*, 2007. 9
- [16] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *Proc. ICLR*, 2024. 1
- [17] William L Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic word embeddings reveal statistical laws of semantic change. In *Proc. ACL*, 2016. 5, 8
- [18] Zhiwei Hao, Jianyuan Guo, Kai Han, Yehui Tang, Han Hu, Yunhe Wang, and Chang Xu. One-for-all: Bridge the gap between heterogeneous architectures in knowledge distillation. In *Proc. NeurIPS*, 2023. 2, 6
- [19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 6
- [20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 2, 6

- [21] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proc. CVPR*, 2017. 6
- [22] Robin Jia, Shiori Sagawa, Sang Michael Xie, Haoran Zhang, and Percy Liang Wang. Auditing data provenance in text-generation models. In *Proc. NeurIPS*, 2022. 2
- [23] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: A lightweight, fast, and cheap version of stable diffusion. In *Proc. ECCV*. Springer, 2024. 1, 2, 8
- [24] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. *Proceedings of Machine Learning Research (PMLR)*, 2023. 3
- [25] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proc. ICML*, 2019. 6, 7, 8
- [26] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proc. ICML*, 2022. 8
- [27] Yuhang Li, Feng Zhu, Ruihao Gong, Mingzhu Shen, Xin Dong, Fengwei Yu, Shaoqing Lu, and Shi Gu. Mixmix: All you need for data-free compression are feature and data mixing. In *Proc. ICCV*, 2021. 2
- [28] Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024. 1, 2, 8
- [29] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. In *Proc. NeurIPS*, 2020. 2, 7
- [30] Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. In *Proc. AISTATS*, 2018. 2
- [31] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021. 2
- [32] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proc. CVPR*, 2023. 2
- [33] Cade Metz. Openai believes deepseek distilled its data for training: Here’s what to know about the technique. *The New York Times*, 2025. Accessed: 2025-04-01. 1
- [34] Nlpconnect. Vit-gpt2 image captioning. <https://huggingface.co/nlpconnect/vit-gpt2-image-captioning>, 2022. 8
- [35] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proc. CVPR*, 2019. 1, 2, 6
- [36] Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proc. EMNLP*, 2022. 3
- [37] Mary Phuong and Christoph H Lampert. Distillation-based training for multi-exit architectures. In *Proc. ICCV*, 2019. 2
- [38] Biao Qian, Yang Wang, Richang Hong, and Meng Wang. Adaptive data-free quantization. In *Proc. CVPR*, 2023. 3
- [39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proc. ICML*, 2021. 6, 8
- [40] Leonardo Ranaldi and Andre Freitas. Aligning large and small language models via chain-of-thought reasoning. In *Proc. ACL*, 2024. 1
- [41] Siladitya Ray. OpenAI Believes DeepSeek ‘Distilled’ Its Data For Training—Here’s What To Know About The Technique. *Forbes*, 2025. Accessed: 2025-04-01. 1
- [42] Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. In *Proc. NeurIPS*, 2024. 2
- [43] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014. 1, 2

- [44] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *Proc. ICLR*, 2022. 2
- [45] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. In *SIGGRAPH Asia 2024 Conference Papers*, 2024. 1, 2, 8
- [46] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proc. ICML*, 2023. 2
- [47] Om Thakkar, Nicholas Carlini, Matthew Jagielski, and Florian Tramer. Understanding and controlling memorization in diffusion models. *arXiv preprint arXiv:2212.03860*, 2022. 2
- [48] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proc. ICCV*, 2019. 2
- [49] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proc. ACL*, 2023. 1
- [50] Winston & Strawn LLP. Is ai distillation by deepseek ip theft? <https://www.winston.com/en/insights-news/is-ai-distillation-by-deepseek-ip-theft>, 2024. Accessed: 2025-04-01. 1
- [51] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proc. CVPR*, 2017. 6
- [52] Shoukai Xu, Haokun Li, Bohan Zhuang, Jing Liu, Jiezhong Cao, Chuangrun Liang, and Minghui Tan. Generative low-bitwidth data free quantization. In *Proc. ECCV*, 2020. 2, 3
- [53] Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. Perflow: Piecewise rectified flow as universal plug-and-play accelerator. In *Proc. NeurIPS*, 2024. 2
- [54] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *Proc. CSF*, 2018. 7
- [55] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proc. CVPR*, 2020. 3, 5
- [56] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. In *Proc. NeurIPS*, 2024. 1, 2, 8
- [57] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proc. CVPR*, 2024. 2
- [58] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proc. CVPR*, 2018. 6
- [59] Shikang Yu, Jiachen Chen, Hu Han, and Shuqiang Jiang. Data-free knowledge distillation via feature exchange and activation region constraint. In *Proc. CVPR*, 2023. 3
- [60] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proc. ICLR*, 2018. 5
- [61] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proc. ICCV*, 2019. 2
- [62] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, 2018. 5
- [63] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proc. CVPR*, 2022. 1, 2
- [64] Amber Yijia Zheng and Raymond A Yeh. Imma: Immunizing text-to-image models against malicious adaptation. In *Proc. ECCV*, 2024. 3
- [65] Amber Yijia Zheng and Raymond A Yeh. Multi-concept model immunization through differentiable model merging. In *Proc. AAAI*, 2025.

- [66] Amber Yijia Zheng, Chiao-An Yang, and Raymond A Yeh. Learning to obstruct few-shot image classification over restricted classes. In *Proc. ECCV*, 2024.
- [67] Amber Yijia Zheng, Site Bai, Brian Bullins, and Raymond A. Yeh. Model immunization from a condition number perspective. In *Forty-second International Conference on Machine Learning*, 2025. 3
- [68] Xiatian Zhu, Shaogang Gong, et al. Knowledge distillation by on-the-fly native ensemble. In *Proc. NeurIPS*, 2018. 2

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The paper clearly defines the task of knowledge distillation detection and proposes a modular, model-agnostic framework applicable to both classification and text-to-image models. These claims are supported by detailed methodology and comprehensive experiments.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We provide the limitation that detection may be less reliable when the student inherits only limited influence from the teacher in the conclusion section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the implementation details in Sec. A2 to reproduce the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the implementation details in Sec. A2 to reproduce the results and will release the source code upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the experimental settings in Sec. 5 and more details can be found in Sec. A2.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All results reported in the tables are averaged over 10 runs with different random seeds. Both the mean and standard deviation are provided to reflect the variability and statistical reliability of the experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide the compute resources in Sec. A2 for the generator training, inference, and the score computation.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We confirm that the research conducted with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Please find broader impact in Sec. A4.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We do not foresee such risks for this paper.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cited all the existing assets in the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: We introduce new methods and code as part of this work. While they are not publicly released at submission time to preserve anonymity, we commit to releasing them upon acceptance. This will include the training and inference code of the generator and the code for computing all the scores.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: the core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

Appendix

The appendix is organized as follows:

- In Sec. A1, we report additional quantitative results.
- In Sec. A2, we provide implementation details of our approach for reproducibility. The code will be released.
- In Sec. A3, we document the details of the compared baselines.
- In Sec. A4, we discuss broader impacts.

A1 Additional Results

Table A1: Point-wise scores between the student models and each candidate models using KD method on CIFAR-10. The reported values are the mean $\pm$ standard deviation across runs.

Teacher	Student	Ours (Method)				MIA Filter + KL (Method)			
		RN18 (Candidate)	DLA(Candidate)	DPN92(Candidate)	DN121(Candidate)	RN18(Candidate)	DLA(Candidate)	DPN92(Candidate)	DN121(Candidate)
RN18	RN18	0.00 $\pm$ 0.00	0.27 $\pm$ 0.17	0.05 $\pm$ 0.02	0.05 $\pm$ 0.03	0.13 $\pm$ 0.04	0.21 $\pm$ 0.06	1.03 $\pm$ 0.19	0.13 $\pm$ 0.04
	DLA	0.05 $\pm$ 0.05	0.96 $\pm$ 0.13	0.76 $\pm$ 0.25	0.88 $\pm$ 0.29	0.11 $\pm$ 0.08	0.03 $\pm$ 0.03	2.97 $\pm$ 0.50	0.24 $\pm$ 0.19
	DPN92	0.00 $\pm$ 0.00	0.13 $\pm$ 0.11	0.02 $\pm$ 0.04	0.00 $\pm$ 0.00	0.05 $\pm$ 0.02	0.06 $\pm$ 0.03	2.29 $\pm$ 0.44	0.14 $\pm$ 0.04
	DN121	0.00 $\pm$ 0.00	0.28 $\pm$ 0.15	0.02 $\pm$ 0.01	0.02 $\pm$ 0.02	0.19 $\pm$ 0.10	0.03 $\pm$ 0.02	3.59 $\pm$ 0.42	0.22 $\pm$ 0.04
DLA	RN18	0.01 $\pm$ 0.00	0.00 $\pm$ 0.00	0.09 $\pm$ 0.06	0.04 $\pm$ 0.02	0.12 $\pm$ 0.06	0.02 $\pm$ 0.01	2.19 $\pm$ 0.30	0.11 $\pm$ 0.03
	DLA	0.07 $\pm$ 0.03	0.00 $\pm$ 0.00	0.07 $\pm$ 0.04	0.06 $\pm$ 0.02	0.35 $\pm$ 0.14	0.02 $\pm$ 0.02	4.43 $\pm$ 0.45	0.45 $\pm$ 0.19
	DPN92	0.02 $\pm$ 0.03	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	0.01 $\pm$ 0.00	0.11 $\pm$ 0.10	0.02 $\pm$ 0.00	2.15 $\pm$ 0.27	0.10 $\pm$ 0.04
	DN121	0.19 $\pm$ 0.06	0.00 $\pm$ 0.00	0.33 $\pm$ 0.28	0.02 $\pm$ 0.03	0.15 $\pm$ 0.06	0.03 $\pm$ 0.01	2.23 $\pm$ 0.17	0.12 $\pm$ 0.07
DPN92	RN18	1.06 $\pm$ 0.21	2.16 $\pm$ 0.31	0.04 $\pm$ 0.05	0.64 $\pm$ 0.21	0.21 $\pm$ 0.03	0.31 $\pm$ 0.04	1.00 $\pm$ 0.07	0.21 $\pm$ 0.03
	DLA	1.30 $\pm$ 0.43	1.24 $\pm$ 0.22	0.00 $\pm$ 0.00	0.37 $\pm$ 0.14	0.38 $\pm$ 0.25	0.42 $\pm$ 0.24	1.14 $\pm$ 0.29	0.31 $\pm$ 0.16
	DPN92	1.83 $\pm$ 0.19	1.66 $\pm$ 0.20	0.00 $\pm$ 0.00	0.28 $\pm$ 0.19	0.34 $\pm$ 0.08	0.46 $\pm$ 0.08	0.58 $\pm$ 0.07	0.32 $\pm$ 0.05
	DN121	0.07 $\pm$ 0.09	0.19 $\pm$ 0.08	0.00 $\pm$ 0.01	0.03 $\pm$ 0.02	0.13 $\pm$ 0.06	0.05 $\pm$ 0.03	1.98 $\pm$ 0.36	0.07 $\pm$ 0.03
DN121	RN18	0.03 $\pm$ 0.01	0.14 $\pm$ 0.06	0.02 $\pm$ 0.02	0.00 $\pm$ 0.00	0.09 $\pm$ 0.05	0.06 $\pm$ 0.03	1.64 $\pm$ 0.15	0.05 $\pm$ 0.03
	DLA	0.03 $\pm$ 0.02	0.05 $\pm$ 0.04	1.02 $\pm$ 0.21	0.00 $\pm$ 0.00	0.12 $\pm$ 0.04	0.13 $\pm$ 0.03	1.27 $\pm$ 0.17	0.09 $\pm$ 0.07
	DPN92	0.61 $\pm$ 0.28	0.19 $\pm$ 0.13	0.14 $\pm$ 0.20	0.00 $\pm$ 0.00	0.10 $\pm$ 0.05	0.11 $\pm$ 0.03	1.09 $\pm$ 0.13	0.05 $\pm$ 0.04
	DN121	0.11 $\pm$ 0.07	0.17 $\pm$ 0.11	0.02 $\pm$ 0.02	0.00 $\pm$ 0.00	0.14 $\pm$ 0.06	0.17 $\pm$ 0.04	1.21 $\pm$ 0.17	0.10 $\pm$ 0.04

In Tab. A1, we compare the scores of our method and the best baseline across all student models on CIFAR-10. According to Tab. A1, we observe that some student models are easier to identify than others. For example, students distilled from RN18 are relatively easy cases, where both our method and the baseline achieve correct detection with large score gaps between the true teacher and other candidates. In contrast, harder cases such as RN18 students distilled from DN121 show much smaller differences, sometimes as low as 0.01 between the true teacher and the next closest candidate. Intuitively, the architectural similarity between the student and teacher models may influence the difficulty of detection.

Table A2: Point-wise scores between the student models and each candidate models using KD method. The reported values are the mean $\pm$ standard deviation across runs.

Teacher	Student	Ours (Method)			MIA Filter + KL (Method)		
		RN18 (Candidate)	RN50 (Candidate)	RNXt50 (Candidate)	RN18 (Candidate)	RN50 (Candidate)	RNXt50 (Candidate)
RN18	RN18	2.41 $\pm$ 1.07	7.01 $\pm$ 1.28	6.88 $\pm$ 1.18	0.13 $\pm$ 0.01	0.29 $\pm$ 0.02	0.70 $\pm$ 0.02
	RN50	0.60 $\pm$ 0.40	5.12 $\pm$ 2.17	4.43 $\pm$ 1.01	0.15 $\pm$ 0.01	0.32 $\pm$ 0.01	0.72 $\pm$ 0.03
	RNXt50	0.29 $\pm$ 0.25	1.90 $\pm$ 1.07	2.61 $\pm$ 0.95	0.18 $\pm$ 0.01	0.27 $\pm$ 0.01	0.68 $\pm$ 0.02
RN50	RN18	8.99 $\pm$ 1.22	9.45 $\pm$ 0.42	10.53 $\pm$ 1.47	0.36 $\pm$ 0.01	0.51 $\pm$ 0.04	0.92 $\pm$ 1.47
	RN50	8.31 $\pm$ 2.19	5.48 $\pm$ 2.17	6.69 $\pm$ 2.08	0.26 $\pm$ 0.01	0.45 $\pm$ 0.02	0.79 $\pm$ 0.03
	RNXt50	5.21 $\pm$ 1.72	3.13 $\pm$ 1.84	3.67 $\pm$ 1.47	0.34 $\pm$ 0.01	0.51 $\pm$ 0.03	0.78 $\pm$ 0.04
RNXt50	RN18	9.91 $\pm$ 1.33	11.22 $\pm$ 1.07	10.32 $\pm$ 1.26	0.45 $\pm$ 0.03	0.78 $\pm$ 0.05	1.01 $\pm$ 0.07
	RN50	9.85 $\pm$ 1.48	10.45 $\pm$ 1.75	8.17 $\pm$ 0.95	0.36 $\pm$ 0.02	0.42 $\pm$ 0.02	0.76 $\pm$ 0.03
	RNXt50	8.65 $\pm$ 1.69	9.39 $\pm$ 1.30	6.98 $\pm$ 1.98	0.52 $\pm$ 0.01	0.68 $\pm$ 0.02	0.82 $\pm$ 0.06

A similar conclusion can be drawn from Tab. A2 on ImageNet. For easy cases, such as students distilled from RN18, the true teacher can be accurately identified with large score gaps over other candidates. However, for harder cases, such as RN18 students distilled from RN50 or RNXt50, both our method and the baseline tend to incorrectly identify the teacher as RN18. These hard cases often involve significant architectural differences between the student and teacher models, suggesting that architectural mismatch contributes to detection difficulty.

In Tab. A3, we report the scores of our method and the best baseline across all student models on the text-to-image generation task. We use 1 - LPIPS in our method, which shows that the similarity

Table A3: Point-wise scores between the student models and each candidate models for text-to-image models. The reported values are the mean $\pm$ standard deviation across runs.

Teacher	Student	Ours (Method)			GPT-2 + DINO (Method)		
		SD-v2.1 (Candidate)	SDXL (Candidate)	PixArt (Candidate)	SD-v2.1 (Candidate)	SDXL (Candidate)	PixArt (Candidate)
SD-v2.1	BK-SDM-v2-b	0.69 $\pm$ 0.01	0.28 $\pm$ 0.00	0.19 $\pm$ 0.00	1.00 $\pm$ 0.00	0.98 $\pm$ 0.00	0.97 $\pm$ 0.00
	BK-SDM-v2-s	0.68 $\pm$ 0.01	0.28 $\pm$ 0.00	0.19 $\pm$ 0.00	1.00 $\pm$ 0.00	0.98 $\pm$ 0.00	0.97 $\pm$ 0.00
	BK-SDM-v2-t	0.70 $\pm$ 0.01	0.27 $\pm$ 0.00	0.18 $\pm$ 0.00	1.00 $\pm$ 0.00	0.98 $\pm$ 0.00	0.97 $\pm$ 0.00
	AMD	0.56 $\pm$ 0.01	0.23 $\pm$ 0.01	0.15 $\pm$ 0.01	1.00 $\pm$ 0.00	0.98 $\pm$ 0.00	0.97 $\pm$ 0.00
SDXL	DMD2	0.24 $\pm$ 0.01	0.32 $\pm$ 0.01	0.20 $\pm$ 0.00	0.98 $\pm$ 0.00	0.99 $\pm$ 0.00	0.98 $\pm$ 0.00
	SDXL-L-1	0.19 $\pm$ 0.00	0.24 $\pm$ 0.01	0.22 $\pm$ 0.00	0.93 $\pm$ 0.00	0.96 $\pm$ 0.00	0.97 $\pm$ 0.00
	SDXL-L-2	0.25 $\pm$ 0.01	0.41 $\pm$ 0.01	0.18 $\pm$ 0.00	0.99 $\pm$ 0.00	0.98 $\pm$ 0.00	0.98 $\pm$ 0.00
	SDXL-L-4	0.26 $\pm$ 0.00	0.48 $\pm$ 0.01	0.21 $\pm$ 0.00	0.98 $\pm$ 0.00	0.99 $\pm$ 0.00	0.98 $\pm$ 0.00
	SDXL-L-8	0.23 $\pm$ 0.01	0.55 $\pm$ 0.01	0.21 $\pm$ 0.00	0.97 $\pm$ 0.00	0.99 $\pm$ 0.00	0.98 $\pm$ 0.00
PixArt	AMD-PixArt	0.19 $\pm$ 0.00	0.37 $\pm$ 0.00	0.43 $\pm$ 0.00	0.98 $\pm$ 0.00	0.98 $\pm$ 0.00	0.99 $\pm$ 0.00

between each student and their true teacher is the highest. For instance, BK-SDM-v2-b has 0.69 ± 0.01 to the teacher SD-v2.1, but much lower values to SDXL and PixArt. However, the baseline scores are nearly saturated across all candidates, typically 0.97–1.00 and making teacher identification infeasible. This saturation indicates a lack of discrimination in the embedding space used by the baseline.

A2 Implementation details

Teacher and student models in classification task. We use ResNet-18, DLA, DPN92, and DenseNet121 as the backbone of the teacher and student models on CIFAR-10. We train the teacher/student models from scratch with these architectures first. Here we apply the same training strategy: stochastic gradient descent (SGD) with a learning rate of 0.01, momentum of 0.9, and weight decay of $5e-4$; training length of 40 epochs with a batch size of 64. We apply the OneCycle learning rate scheduler, with a maximum learning rate of 0.1, computed over 45000/64 steps per epoch and updated at every training step.

On ImageNet, since the teacher models are available on torchvision, we only need to train student models. The models are trained for 200 epochs with a batch size of 256. We use SGD with a learning rate of 0.1, momentum of 0.9, and weight decay of $1e-4$. A cosine annealing learning rate scheduler is applied, with the minimum learning rate set to 0.0 and a total decay period over the whole epoch.

For the teacher model training, we use only the standard cross-entropy loss for classification. For student models trained with knowledge distillation (KD), only the KL divergence loss is used. In particular, for the RKD and OFA methods, we remove the hard-label supervision entirely and replace it with KL loss, so that the student models are trained solely under the guidance of the teacher.

In the RKD method, the total loss is a weighted combination of 0.3 KL loss and 0.7 RKD loss. Following the original RKD paper, we set the weight of the distance loss to 1 and the weight of the angle loss to 2. RKD loss is computed using the features after the final average pooling and flattening. To ensure dimensional compatibility between teacher and student models, we insert a fully connected projection layer for student models whose feature dimensions do not match those of the teacher. In the OFA method, the loss is a weighted combination of 0.3 KL loss and 0.7 OFA loss. For models with different architectures, we manually divide the hidden representations into four segments and select after-activation latents for computing the OFA loss.

Input construction in classification task. Our generator takes as input a noise vector and a class label, and produces a synthesized image conditioned on the label. Note that this label is first embedded using a learnable embedding layer. The normalized embedding is passed through a linear projection layer before being fed into the image generation pipeline.

On CIFAR-10, the projected embedding is transformed via a fully connected layer into a spatial feature map of shape $128 \times 8 \times 8$. This feature map is processed through three convolutional blocks, with bilinear upsampling applied between the blocks to restore spatial resolution:

- **Block 0:** A single BatchNorm layer.
- **Block 1:** A convolutional layer with 128 channels, followed by BatchNorm and LeakyReLU.

- **Block 2:** Two convolutional layers: one reducing to 64 channels and another producing the final image with the desired number of channels, followed by Tanh and BatchNorm.

On ImageNet, the input is also projected through a linear layer before being mapped to spatial features. The resulting vector is then passed through a fully connected layer and reshaped into a $128 \times 56 \times 56$ feature map. The convolutional backbone consists of three blocks, with bilinear upsampling applied between them:

- **Block 0:** Categorical Conditional BatchNorm (CCBN) is applied.
- **Block 1:** A convolutional layer with 128 filters, followed by CCBN and LeakyReLU activation.
- **Block 2:** One convolution with 128 channels that outputs the image channels, followed by CCBN, LeakyReLU. Followed by applying another convolutional layer, reducing to 64 channels, followed by a Tanh activation, and a final BatchNorm.

During training, the generator supports mixing noise latents and labels through weighted projection. The probability of applying mixup is set to 0.4. The number of inputs used for mixing is 2 for CIFAR-10 and 10 for ImageNet.

For CIFAR-10, we train for 400 epochs using a batch size of 200. The optimizer is Adam with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The label embedding dimension is 64. The learning rate is set to 0.001, using a multi-step schedule that decays by a factor of 0.1 at epochs 100, 200, and 300.

For ImageNet, we also train for 400 epochs with a batch size of 16. Adam is used with the same β_1 and β_2 values as in CIFAR-10. The label embedding dimension is set to 256 for ResNet18 students and 512 for ResNet50 and ResNeXt50 students. The learning rate is 0.001, with the same multi-step decay schedule.

Hardware for experiments. For training teacher and student models on CIFAR-10, we use a single A30 GPU. For ImageNet, student models are trained using two L40S GPUs.

In the first stage of our knowledge distillation detection pipeline, we train a generator to produce synthesized images in classification tasks using a single L40S GPU.

For score computation and prediction, we only perform inference with the student and teacher models. This stage uses a single A30 GPU for CIFAR-10 classification models, and a single L40S GPU for ImageNet classification and text-to-image generation models.

A3 Baseline details

Most of the baselines have already been introduced in Sec. 5. Here, we formally describe MMD-FUSE in this section. MMD-FUSE is a kernel-based two-sample test. In our setting, the null and alternative hypotheses are:

$$H_0 : g_\theta \sim \mathcal{D}_{\text{distill}}(f) \quad \text{vs.} \quad H_1 : g_\theta \sim \mathcal{D}_{\text{indep}}, \quad (\text{A15})$$

Let the student model outputs be denoted by $\{x_i\}_{i=1}^n$ and the teacher outputs by $\{y_i\}_{i=1}^n$. MMD-FUSE adaptively combines multiple kernels into a single test statistic. For a finite set of candidate kernels $\{k\}_{m=1}^M$, it computes each normalized MMD value and then takes a softmax over them:

$$\text{MMD_FUSE} = \frac{1}{\beta} \log \left(\sum_{m=1}^M \exp(\beta \widehat{\text{MMD}}_m^2) \right),$$

where $\widehat{\text{MMD}}_m^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k_m(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k_m(y_i, y_j) - \frac{2}{n^2} \sum_{i=1}^n \sum_{j=1}^n k_m(x_i, y_j)$. To compute the p-value of the test, MMD-FUSE uses a permutation procedure. A smaller p-value indicates stronger evidence against the null hypothesis, suggesting that the student model is not a distilled version of the teacher.

In our experiment, we use two base kernels: the Laplace kernel and the RBF kernel. For each kernel, we select 10 different bandwidths. The number of permutations used for estimating the p-value is set to 1000, and the temperature scalar for the softmax aggregation is set to $\beta = 1$.

A4 Broader impacts

This work contributes to model accountability by introducing a method for detecting whether a model has been trained via knowledge distillation from a specific teacher. This has positive implications for auditing model provenance, protecting intellectual property, and improving transparency in the deployment of machine learning systems. Our detection framework can help identify unauthorized reuse of proprietary models, especially in scenarios where model weights are publicly released but training procedures are not disclosed. At the same time, detection tools may influence how models are shared or reused, and like any statistical method, they may produce imperfect predictions. We suggest using such tools with awareness of their limitations and context.