Distilling the Undistillable: Learning from a Nasty Teacher

Surgan Jandial[1], Yash Khasbage[2], Arghya Pal[3], Vineeth N Balasubramanian[2], and Balaji Krishnamurthy[1]

[1] Adobe MDSR Labs
[2] Indian Institute of Technology, Hyderabad
[3] Dept. Of Psychiatry and Radiology, Harvard

Abstract. The inadvertent stealing of private/sensitive information using Knowledge Distillation (KD) has been getting significant attention recently and has guided subsequent defense efforts considering its critical nature. Recent work *Nasty Teacher* proposed to develop teachers which can not be distilled or imitated by models attacking it. However, the promise of confidentiality offered by a nasty teacher is not well studied, and as a further step to strengthen against such loopholes, we attempt to bypass its defense and steal (or extract) information in its presence successfully. Specifically, we analyze *Nasty Teacher* from two different directions and subsequently leverage them carefully to develop simple yet efficient methodologies, named as *HTC* and *SCM*, which increase the learning from Nasty Teacher by upto 68.63% on standard datasets. Additionally, we also explore an improvised defense method based on our insights of stealing. Our detailed set of experiments and ablations on diverse models/settings demonstrate the efficacy of our approach.

Keywords: Knowledge Distillation, Model Stealing, Privacy.

1 Introduction

Knowledge Distillation utilizes the outputs of a pre-trained model (i.e teacher) to train a generally smaller model (i.e student). Typically, KD methods are used to compress models that are wide, deep and require significant computational resources and pose challenges to model deployment. Over the years, KD methods have seen success in various settings beyond model compression including few-shot learning [29], continual learning [6], and adversarial robustness [11], to name a few – highlighting its importance in training DNN models. However, recently, there has been a growing concern of misusing KD methods as a means to steal the implicit model knowledge of a teacher model that could be proprietary and confidential to an organization. KD methods provide an inadvertent pathway for leak of intellectual property that could potentially be a threat for science and society. Surprisingly, the importance of defending against such KD-based stealing was only recently explored in [22, 19], making this a timely and important topic.

In particular, [22] recently proposed a defense mechanism to protect such KD-based stealing of intellectual property using a training strategy called the

'Nasty Teacher'. This strategy attempts to transform the original teacher into a model that is '*undistillable*', i.e., any student model that attempts to learn from such a teacher gets significantly degraded performance. This method maximally disturbs incorrect class logits (a significant source of model knowledge), which produces confusing outputs devoid of clear, meaningful information. This method showed promising results in defending against such KD-based stealing from DNN models. However, any security-related technology development requires simultaneous progress of both attacks and defenses for sturdy progress of the field, and eventually lead to the development of robust models. In this work, we seek to test the extent of the defense obtained by the 'Nasty Teacher' [22], and show that it is possible to recover model knowledge despite this defense using the logit outputs of such a teacher. Subsequently, we leverage the garnered insights and propose a simple yet effective defense strategy, which significantly improves defense against KD-based stealing.

To this end, we ask two key questions: (i) can we transform the outputs of the Nasty Teacher to reduce the extent of confusion, and thus be able to steal despite is defense? and (ii) can we transform the outputs of the Nasty Teacher to recover hidden essential relationships between the class logits? To answer these two questions, we propose two approaches – High-Temperature Composition (HTC) which systematically reduces confusion in the logits and Sequence of Contrastive Model (SCM) which systematically recovers relationships between the logits. These approaches result in performance improvement of KD, thereby highlighting the continued vulnerability of DNN models to KD-based stealing. Because of their generic formulation and simplicity, we believe our proposed ideas could apply well to similar approaches that may be developed in future along the same lines as the Nasty Teacher. To summarize, this work analyzes key attributes of output scores (which capture the strength and clarity of model knowledge) that could stimulate knowledge stealing and thereby leverages those to strengthen defenses against such attacks too. Our key contributions are summarized as follows:

- We draw attention to the recently identified vulnerability of KD methods in model-stealing, and analyze the first defense method in this direction, i.e. Nasty Teacher, from two perspectives: (i) reducing the extent of confusion in the class logit outputs; and (ii) extracting essential relationship information from the class logit outputs. We develop two simple yet effective strategies – High Temperature Composition (HTC) and Sequence of Contrastive Model (SCM) – which can undo the defense of the Nasty Teacher, pointing to the need for better defenses in this domain.
- Secondly, we leverage our obtained insights and propose an extension of Nasty Teacher, which outperforms the earlier defense under similar settings.
- We conduct exhaustive experiments and ablation studies on standard benchmark datasets and models to demonstrate the effectiveness of our approaches.

We hope that our efforts in this work will provide important insights and encourage further investigation on a critical problem with DNN models in contemporary times where privacy and confidentiality are increasingly valued.

2 Related Work

We discuss prior work both from perspectives of Knowledge Distillation (KD) as well as its use in model-stealing below.

Knowledge Distillation: KD methods transfer knowledge from a larger network (referred to as *teacher*) to a smaller network (referred to as *student*) by enforcing students to match the teacher's output. With seminal works [4, 14] laying the foundation, KD has gained wide popularity in recent years. The initial techniques for KD mainly focused on distilling knowledge from logits or probabilities. This idea got further extended to distilling features in [31, 40, 36, 28], and many others. In all such methods, KD is used to improve the performance of the student model in various settings. More detailed surveys on KD can be found in [12, 35, 21]. Our focus in this work, however, is on recent works [22, 37, 19], which have discussed how KD can unintentionally expose threats to Intellectual Property (IP) and private content of the underlying DNN models and data, thereby motivating a new, important direction in KD methods.

Model Stealing and KD: Model stealing involves stealing any information from a DNN model that is desired to be inaccessible to an adversary/end-user. Such stealing can happen in multiple ways: *(1) Model Extraction as a Black Box.* An adversary could query existing model-based software, and with just its outputs clones the knowledge into a model of their own; *(2) Using Data Inputs.* An adversary may potentially access similar/same data as the victim, which can be used to extract knowledge/IP; or *Using Model Architecture/Parameters.* An adversary may attempt to extract critical model information – such as the architecture type or the entire model file – through unintentional leaks, academic publications or other means. There have been a few disparate efforts in the past to protect against model/IP stealing in different contexts such as watermark-based methods [34, 41], passport-based methods [8, 42], dataset inference [25], and so on. These methods focused on verifying ownership, while other methods such as [17, 15] focused on defending against few model extraction attacks. However, the focus of these efforts was different from the one discussed herein. In this work, we specifically explore the recently highlighted problem of KD-based model stealing [22, 19]. As noted in [22, 19], most existing verification and defense methods do not address KD-based stealing, leaving this rather critical problem vulnerable. Our work helps analyze the first defense for KD-based stealing [22], identifies loopholes using simple strategies and also leverages them to propose a newer defense to this problem. We believe our findings will accelerate further efforts in this important space. The work closest to ours is one that has been recently published – Skeptical Student [19] – which probes the confidentiality of [22] by appropriately designing the student (or hacker) architecture. Our approach in this work is different, and focuses on mechanisms of student training, without changing the architecture. [4]

[4] Code available at https://github.com/surgan12/NastyAttacks.

3 Learning from a Nasty Teacher

3.1 Background

Knowledge Distillation (KD): KD methods train a smaller student network, θ_s, with the outputs of a typically large pre-trained teacher network, θ_t alongside the ground-truth labels. Given an input image $\mathbf{x}$, the output logits of student given by $\mathbf{z}_s = \theta_s(\mathbf{x})$ and teacher logits given by $\mathbf{z}_t = \theta_t(\mathbf{x})$, a temperature parameter τ is used to soften the logits and obtain a transformed output probability vector using the softmax function:

$$\mathbf{y}_s = softmax(\mathbf{z}_s/\tau), \mathbf{y}_t = softmax(\mathbf{z}_t/\tau) \tag{1}$$

where $\mathbf{y}_s$ and $\mathbf{y}_t$ are the new output probability vectors of the student and teacher, respectively. The final loss function used to train the student model is given by:

$$\mathcal{L} = \alpha \cdot \lambda \cdot KL(y_s, y_t) + (1 - \alpha) \cdot \mathcal{L}_{CE} \tag{2}$$

where KL stands for Kullback-Leibler divergence, ($\mathcal{L}_{CE}$) represents standard cross-entropy loss, and λ, α are two hyperparameters to control the importance of the loss function terms ($\lambda = \tau^2$ generally).

KD-based Stealing: Given a stealer (or student) model, denoted by θ_s, and a victim (or teacher) θ_t, the stealer is said to succeed in stealing knowledge using KD if by using the input-output information of the victim, it can grasp some additional knowledge which is not accessible in the victim's absence. As stated in [22], this phenomenon can be measured in terms of difference in maximum accuracy of stealer with and without stealing from victim. Formally, stealing is said to happen if:

$$Acc_w(KD(\theta_s, \theta_t)) > Acc_{wo}(\theta_s) \tag{3}$$

where the left expression refers to the accuracy with stealing, and the right one refers to accuracy without stealing.

Defense against KD based Stealing: Following [22], we consider a method M as defense, if it degrades the student's tendency (or accuracy) of stealing. Formally, considering the accuracy of stealer without defense M as $Acc_w(KD(\theta_s, \theta_t))$ and with defense as $Acc_{wm}(KD(M(\theta_t, \theta_t)))$, M is said to be a defense if:

$$Acc_{wm}(KD(M(\theta_s, \theta_t))) < Acc_w(KD(\theta_s, \theta_t)) \tag{4}$$

Nasty Teacher(NT)[22]: The Nasty Teacher methodology transforms the original model to a model which has accuracy as high as the original model (to ensure model usability) but whose output distribution (or logits) significantly camouflages the meaningful information.

Formally, given a teacher model θ_t, they output a nasty teacher model θ_n trained by minimizing cross-entropy loss $\mathcal{L}_{CE}$ with target labels y (to ensure high accuracy) and also by maximizing KL-Divergence $\mathcal{L}_{KL}$ with the outputs of the original teacher (to maximally contrast or disturb from the original and create a confusing distribution). This can be written as:

$$\mathcal{L}_n(\mathbf{x}, y) = \mathcal{L}_{ce}(\theta_n(\mathbf{x}), y) - \omega \cdot \tau_A^2 \cdot \mathcal{L}_{KL}(\theta_n(\mathbf{x}), \theta_t(\mathbf{x})) \tag{5}$$

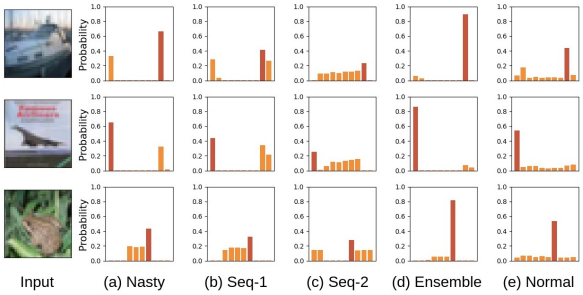

Fig. 1: Softmax Outputs of: (a) Nasty Teacher; (b) Seq-1 (Intermediate model S^i in Sec 3.4); (c) Seq-2 (Similar to Seq-1); (d) Ensemble of Seq-1 and Seq-2 as used in SCM ; and (e) Normal Teacher. Note that the class output distribution of the ensemble is similar to the original (normal) teacher. Maroon = Target class; Orange = other classes.

where ω is a weighting parameter, and τ_A^2 is a temperature parameter. Figure 1(a) provides a visual illustration of Nasty Teacher's outputs after softmax. We see that when compared to a normal teacher model in Figure 1(e), it maintains correct class assignments but significantly changes the semantic information contained in the class distribution.

3.2 Feasibility of KD Based Stealing

As discussed earlier, standard KD techniques [14, 12, 35, 21] learn well with just the outputs of the teacher and hence are well-suited for stealing models released as APIs, MLaaS, so on (known as the black-box setting). However, it is also true that the performance of KD methods relies on factors such as training data, architecture choice and the amount of information revealed in the outputs. Thus, one might argue that we can permanently restrict attackers' access to these and prevent KD-based stealing attacks. We now discuss each of these and illustrate the feasibility of KD-based attacks. **(1) Restricting Access to Training Data:** While developers try their best to protect such IP assets, as discussed by [19], there can continue to be numerous reasons for concern: (i) The developers might have bought the data from a vendor who can potentially sell it to others; (ii) Intentionally or not, there is a distinct possibility for data leaks; (iii) Many datasets are either similar to or subsets of large-scale public datasets (ImageNet[32], BDD100k[38], so on), which can be effectively used as proxies; or (iv) Model inversion techniques can be used to recover training data from a pre-trained model [37, 43, 16, 30, 3] both in white-box and black box settings. Such methods [16], in fact, do not even require the soft outputs, just the hard predicted label from the model suffices. Thus, as these methods evolve, we can only expect an adversary to become capable of obtaining training data of sufficient quantity and quality to allow such KD-based stealing. **(2) Restricting Access to Architecture:** While developers may not reveal the architecture information completely, common development practices still pose concerns for

safety: (i) Most applications simply utilize architectures from existing model hubs [1, 2], sometimes even with the same pre-trained weights (transfer learning), which narrow down the options an adversary needs to try; (ii) Availability of additional tools such as AutoML [27, 10], Neural Architecture Search (NAS) [7, 20] can help attackers search for architectures that match a specific criteria. When combined with the increasingly available compute power (in terms of TPUs, GPUs), this can make models significantly vulnerable to attacks; (iii) With advancements in KD methods, it has been shown that knowledge can be distilled from any architecture to the other: [9] shows distillation with same capacity networks, [39] shows distillation from even poorly trained networks, and so on. Besides, KD has also witnessed techniques that require no data information and still achieve distillation (i.e. data-free distillation [5, 37]). Hence, more opportunities for an attacker to carry out KD-based stealing. **(3) Releasing Incomplete or Randomly Noised Teacher Outputs:** The degree of exposition (i.e. the number of classes revealed) and the clarity of scores (i.e. ease of understanding and inferring from scores) play a vital role in ensuring the quality of knowledge transfer. Hence, one might argue that releasing only top-K class scores can help contain attackers; however, such an approach is infeasible in many use cases where it is necessary to fetch the entire score map. One might also consider adding random noise to make teacher outputs undistillable. This approach generally lowers model performance but also make its failure tracking difficult.

The above discussion motivates the need to develop fundamentally sound strategies to protect against stealing. To this end, we analyze "Nasty Teacher" [22] from two different directions and subsequently leverage insights from KD literature to propose simple yet effective approaches to show that it is still vulnerable to knowledge stealing. We name our attack methods as: *"High Temperature Composition (HTC)"* and *"Sequence of Contrastive Models (SCM)"*, and describe them below. We explain our methods using the Nasty Teacher as the pivot, primarily because it is the only known KD-stealing defense method at this time. Our strategies however are general, and can be applied to any KD based stealing method.

3.3 High Temperature Composition: HTC

Motivation: Nasty Teacher Creates Confusing Signals. In Figure 1, we observe that the original teacher θ_t emulates a single-peak distribution and consistently has low scores for incorrect classes. Now, because the Nasty Teacher is trained to contrast with the original teacher, it produces high scores for few incorrect classes, and thus results in a multi-peak distribution (see Fig 1b). In particular, a few incorrect classes score almost as high as the correct class while other incorrect classes score almost a zero, which, as discussed in [22] introduces confusion in the outputs. We attribute this confusion to two key aspects: (i) some low-scoring incorrect classes getting ignored, and (ii) some high-scoring incorrect classes behaving as importantly as the correct class. Fig 2c shows a visualization of this observation. Since the student model is now forced to learn

these incorrect peaks as equally important as the correct class, it gets a false signal and diverges while training.

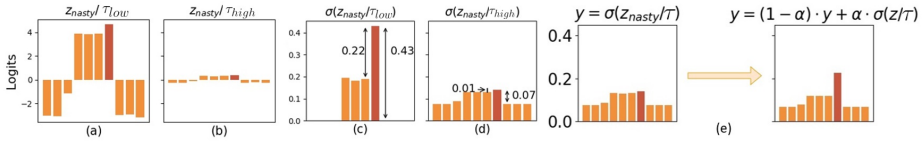

Fig. 2: The effect of temperature in HTC. We demonstrate the logits and probabilities at low temperature (as generally used in KD) and high temperature (as used in HTC). **(a)** Logits at low temperature $\tau_{low} = 4$, **(b)** Logits at high temperature $\tau_{high} = 50$, **(c)** Probabilities at τ_{low} , **(d)** Probabilities at τ_{high}, **(e)** Composing with one-hot to increase the peak.

Proposition. *Transform the output to reduce the degree of confusion in them.*
Method. We hypothesize that distillation from defenses such as Nasty Teacher can be improved by increasing the relative importance of low-scoring incorrect classes and including their presence in the output. Increasing the importance of low-scoring incorrect classes makes the high-scoring incorrect classes lesser important, thus reducing confusion. We note that this idea can be used generically, even independent of the Nasty Teacher's defense. To this end, we first soften the teacher's outputs with a high temperature τ ($\tau > 50$ in our case). Figure 2b shows how this reduces the relative disparity among the scores and brings them closer, thus helping reduce confusion. From the softmax outputs in Figure 2d, we further see that this not only allows the other incorrect classes to be viewed in the output but also gives rise to relationships (or variations) which were earlier not visible. Formally, the above operation can be written as:

$$\mathbf{y}_{nasty} = softmax(\mathbf{z}_{nasty}/\tau) \tag{6}$$

Although we get a much more informative output, the above transformation does also reduce the relative peak of the correct class, which for distillation may not be ideal. We overcome this by using a convex combination of $\mathbf{y}_{nasty}$ with the one-hot target vector to obtain the final output $\mathbf{y}_{net}$, which makes this strategy more meaningful (see Figure 2)):

$$\mathbf{y}_{net} = (1-\alpha)\cdot\mathbf{y} + \alpha\cdot\mathbf{y}_{nasty} \tag{7}$$

In the above discussion, we propose to create our own training targets which satisfy the two properties to learn despite the Nasty Teacher's defense: (i) has a high peak for the correct class; and (ii) has the rich semantic class score information. Finally, to learn from this teacher and match its distributions, we minimize the cross-entropy lpss between student probabilities ($\mathbf{s} = softmax(\mathbf{z}_{student})$) and HTC teacher ($\mathbf{y}_{net}$) targets as:

$$\mathcal{L}_{\mathcal{HTC}} = -\sum_i \mathbf{y}_{net,i}\cdot\log\mathbf{s}_i$$

Combining the above with Eqn 7, $\mathcal{L}_{CE}$ as cross-entropy loss and $\mathcal{L}_{KL}$ as KL-Divergence, we can now write the above as:

$$\begin{aligned}\mathcal{L}_{\mathcal{HTC}} &= -\sum_i ((1-\alpha)\cdot \mathbf{y}_i + \alpha \cdot \mathbf{y}_{nasty,i}) \log \mathbf{s}_i \\ &= -(1-\alpha)\cdot \sum_i \mathbf{y}_i \cdot \log \mathbf{s}_i - \alpha \cdot \sum_i \mathbf{y}_{nasty,i} \cdot \log \mathbf{s}_i \\ &= (1-\alpha)\cdot \mathcal{L}_{CE}(\mathbf{s}, \mathbf{y}) + \alpha \cdot \mathcal{L}_{CE}(\mathbf{s}, \mathbf{y}_{nasty})\end{aligned} \tag{8}$$

$$\mathcal{L}_{\mathcal{HTC}} = (1-\alpha)\cdot \mathcal{L}_{CE}(\mathbf{s}, \mathbf{y}) + \alpha \cdot m \cdot \mathcal{L}_{KL}(\mathbf{s}, \mathbf{y}_{nasty}) \tag{9}$$

In Equation 9, we see that $\mathcal{L}_{CE}$ is replaced with $\mathcal{L}_{KL}$. Here, we use the fact that cross-entropy loss and KL-divergence differ by a term $\sum_i \mathbf{y}_{nasty,i} \log \mathbf{y}_{nasty,i}$, which remains a constant for student gradient computation because of fixed teacher outputs $\mathbf{y}_{nasty}$. $\mathcal{L}_{\mathcal{HTC}}$ is thus a KD-loss similar to what was discussed in Sec 3.1. To adjust the extent of knowledge transfer, we finally re-weight the KL-divergence term with a hyperparameter multiplier m. Conceptually, m does not make a difference to the idea, but we observe this to be useful while training. Figure 3a provides a visualization for this approach.

3.4 Sequence of Contrastive Models: SCM

Motivation: Nasty Hides Essential Information. As Nasty Teacher selectively causes some incorrect classes to exhibit peaks while inhibiting scores for others, it hides certain inter-class relationships to protect against KD-based attacks.

Proposition.: *Transform the output to extract/recover the essential class relationships or possibly the entire original teacher distribution*

Method.: To begin with, we ask the question - what would happen if we used Eqn 5 as is with the given nasty teacher θ_n? We would expect to see a model with accuracy as high as the teacher but with presence of class distribution peaks different from it. If we perform this operation for "k" such sequential steps, we can expect to obtain "k" potentially different output distributions. Building on this thought, we introduce a Sequential Contrastive Training strategy, wherein we form a sequence of "k" contrastive models by training each model to contrast with the model just before it. Formally, taking the nasty teacher θ_n as the starting point of the sequence, G as the method for generating the next contrastive item in the sequence, which in our approach is the same as the one described in Eqn 5, the sequence S^k with k as the sequence length can be written as:

$$S^k = \begin{cases} S^i = \theta_n & i = 1 \\ S^i = G(S^{i-1}) & 1 < i \leq k \end{cases} \tag{10}$$

Thus, while maximising KL-divergence between models, each model learns a different probability distribution and has its own unique set of confusing class

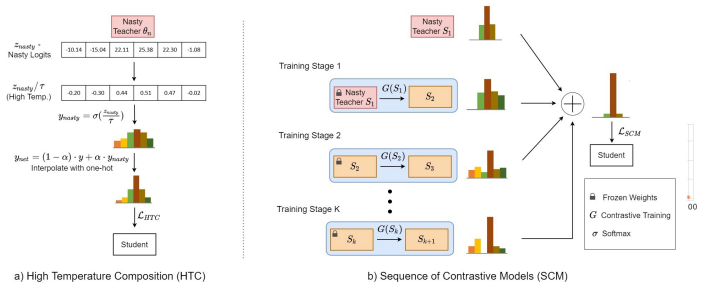

Fig. 3: *Illustration of our Methods*: High Temperature Composition (HTC) and Sequence of Contrastive Models (SCM). *HTC* learns from Nasty Teacher (NT) outputs by reducing their confusion, while *SCM* learns by extracting semantic information from them.

relationships. We then take an ensemble of their outputs, which is denoted as $\mathbf{z}_{ens}$ and use this alongside the ground truth labels y to train the student (or stealer) model, i.e.

$$\mathcal{L}_{\mathcal{SCM}} = (1-\alpha)\cdot\mathcal{L}_{CE}(softmax(\mathbf{z}_s), \mathbf{y}) + \alpha\cdot m\cdot\mathcal{L}_{KL}(softmax(\mathbf{z}_s/\tau), softmax(\mathbf{z}_{ens}/\tau)) \quad (11)$$

where hyperparameters m, α, τ have the same meaning as in Eqn 9. The core intuition behind SCM is that a relationship may be important in its own distribution but only the essential relationships will be important across distributions. Therefore, by taking the ensemble, we not only capture such relationships but also obtain a distribution closer to the original teacher. This idea is visualized in Figure 1 where (b) and (c) represent two successive checkpoints in the sequence, and (d) represents their ensemble. It can be clearly noted that different items in the sequence consistently output diversity in class relationships, and further, their ensemble illustrates a class distribution closer to the original teacher (Figure 1e).

4 Experiments and Results

4.1 Experimental Setup

For all the experiments, we follow the same models and training configurations as used in the code provided by [22][5].

Datasets and Network Baselines: For datasets, we use CIFAR-10 (C10), CIFAR-100 (C100) and TinyImageNet (TIN) datasets in our evaluation. For teacher (or victim) networks, we use ResNet18 [13] for CIFAR-10, both Resnet18

[5] https://github.com/VITA-Group/Nasty-Teacher

and ResNet50 for CIFAR-100 and TinyImageNet. For student (or stealer) networks, we use MobileNetv2 [33], ShuffleNetV2 [23], and Plain CNNs [26] in CIFAR-10 and CIFAR-100 and MobileNetV2, ShuffleNetV2 in TinyImageNet. For baselines, Vanilla in Table 1 refers to the cross entropy training, *Normal KD*/*Nasty KD* refer to the distillation (or stealing) with original/nasty teacher, *Skeptical* refers to the recent work [19], and *HTC, SCM* refers to our approaches.
Training Baselines: We follow the parameter choice from [22]. For generating Nasty Teacher in Eq. 5, τ_A is set to 4 for CIFAR-10 and 20 for CIFAR-100/TinyImageNet, correspondingly ω to 0.04, 0.005, 0.01 for CIFAR-10, CIFAR-100, TinyImageNet respectively. For both Nasty KD and Normal KD, (α, τ) parameters of distillation in Eq. 2 are set to (0.9, 4) in plain CNNs and (0.9, 20) in MobileNetV2, ShuffleNetV2. Plain CNNs are optimized with Adam[18] (LR=1e-3) for 100 epochs while MobileNetV2, ShuffleNetV2 are optimized with SGD (momentum=0.9, weight decay=5e-4, initial LR=0.1) for 160 epochs with LR decay of 10 at epoch [80, 120] in CIFAR-10 and for 200 epochs with LR decay of 5 at epoch [60, 120, 160] in CIFAR-100. Training settings in TinyImageNet are same as used in CIFAR-100. Moreover, all the experiments use batch size of 128 with standard image augmentations.
Training HTC and SCM : For both HTC and SCM, we search m in {1, 5, 10, 50}, τ in {4, 20, 50, 100}, α in {0.1,....0.9}. For TinyImageNet, we include $\tau = 200$ also in the earlier set of τ. While any number of sequence models can conceptually be chosen for SCM , our experiments only leverage a max. of 4 such Sequence Contrastive models. Note, for all these we choose the search space rather intuitively and finally present their impact in Sec. 4.3.

4.2 Quantitative Results

We report the results in Table 1 and clearly observe the degradation of student performance while learning from Nasty teacher. Subsequently, our approaches: HTC and SCM increase learning from Nasty by upto 58.75% in CIFAR-10, 68.63% in CIFAR-100 and 60.16% in TinyImageNet. We significantly outperform the recent state of the art Skeptical [19] and unlike them we also consistently outperform the Vanilla training. Moreover, many times we achieve very close performances and other few times even better performance than Normal-Teacher (i.e stealing from unprotected victim model), see the † marked cells in table 1. Further, we combine our training methods HTC and SCM with the novel stealing architecture Skeptical Student [19] and report improvements in Table 1.

4.3 Ablation Studies

Choice of τ, m in HTC: HTC (Sec 3.3) depends on softening temperature (τ) which adjusts the optimal representation, and multiplier m which adjusts the optimal weight to transfer this knowledge. Thus, we now study each of these parameters separately. We also note that α primarily controls the ground truth signal, hence, we omit varying α here and set it to 0.9. We vary the τ as 10, 20, 50,

Dataset	Tch.	Stu.	Vanilla	Normal KD	Nasty KD	Skep. (NeurIPS'21)	HTC	SCM	Skep. +HTC	Skep. +SCM
C10	Res18	CNN	86.31	87.83	82.27	86.71	87.38	**87.85**	87.17	87.04
		Mob	89.58	89.30	31.73	90.53	90.03†	90.48†	91.45†	**91.55**†
		Shuf	91.03	91.17	79.73	91.34	91.61†	91.93†	92.45†	**92.76**†
C100	Res18	CNN	58.38	62.35	58.62	58.38	61.21	**61.31**	59.64	59.17
		Mob	68.90	72.75	3.15	66.89	71.01	71.06	71.48	**71.74**
		Shuf	71.43	74.43	63.67	70.00	74.04	**75.23**†	73.78	74.23
C100	Res50	CNN	58.38	61.84	58.93	59.15	**61.24**	59.58	59.48	59.17
		Mob	68.90	72.22	3.03	66.65	70.49	**71.66**	71.54	71.16
		Shuf	71.43	73.91	62.8	70.02	72.37	73.25	73.60	**74.73**†
TIN	Res18	Mob	55.69	61.00	0.85	47.37	56.28	**61.01**†	59.05	58.88
		Shuf	60.30	63.45	23.78	54.78	60.46	62.09	**63.05**	62.28
TIN	Res50	Mob	55.89	57.84	1.10	48.21	56.00	**59.46**†	58.56†	58.88†
		Shuf	60.30	62.02	24.27	56.08	60.80	61.55	**63.14**†	61.92

Table 1: *Accuracy (higher is better)* of HTC and SCM against baselines. **bold** represents best performance, underline the second best in learning from Nasty Teacher. † *represents instances* that even outperform the Normal KD. *Abbreviations* – Tch : Teacher, Stu : Student, Skep : Skeptical [19], Res50 : ResNet50, Res18 : ResNet18, Mob : MobileNetV2, Shuf : ShuffleNetV2.

	Vanilla	Nasty KD	Ours
CNN	58.38	58.62	**61.25**
ShuffleNetV2	71.43	63.67	**72.71**

(a) Effect of architecture choice.

	2	3	4
CNN	60.99	61.24	**61.31**
ShuffleNetV2	74.45	74.24	**75.23**

(b) Effect of sequence length.

Table 2: Analysing hyperparameter choice for SCM.

100 with $m = 50$ on ShuffleNetV2 and present results in Fig. 4 (a),(b) to demonstrate the effect of setting temperature to an optimal higher value. We then vary m keeping the τ at 50 to observe the motivation of its careful selection in 4(d).

Choice of Architecture and k in SCM : Given a Nasty Teacher, we currently use the same architecture type to generate $S^1, S^2, ..S^k$. However, we now explore if SCM generalizes to different architecture choices in Table 2(a). Rather than obtaining seq. items as ResNet18 → ResNet18 i.e ${S^i}_{ResNet18} = G({S^{i-1}}_{ResNet18})$, we hereby do ${S^i}_{ShuffleNetV2} =$

	CIFAR10 ResNet18	CIFAR100 ResNet18	CIFAR100 ResNet50
Nasty Teacher	0.4446	14.6363	10.4892
SCM Ensemble	**0.2471**	**5.8398**	**5.3165**

Table 3: KL-Divergence scores of a given model (col 1) with the original teacher.

$G(S^{i-1}{}_{ResNet18})$, and finally ensemble generated ShuffleNetV2 (S^2) and original ResNet18 (S^1) to train the students (or stealers). Table 2 further shows that even with different architecture, SCM can extract information from Nasty Teacher. Moving ahead, we now ablate on *sequence length* k. From Table 2(b), we observe improvement in performance as we increase number of sequence items. This can be intuitively linked to getting better estimate of essential relationships with more number of models, hence improved distillation.

Similarity of SCM and Original Teacher: As with SCM we seek to potentially recover the original teacher distribution, thus in addition to the visualization we present in 1 we hereby conduct this study to quantitatively estimates the closeness of the SCM ensemble and normal teacher. We use KL-Divergence as our metric and present results in Table 3. It can can be clearly seen that compared to Nasty Teacher, SCM ensemble consistently results in low KL-Divergence, thereby lending support to our motivation in recovering original teacher.

4.4 Limited and No Data Setting

In this section, we consider settings to learn from the nasty teacher by using only a part of the data. Specifically, we experiment three settings and for the sake of simplicity use HTC to investigate learning from nasty under these settings.
No Label Available: Here, we consider no access to the labels for data used in 4.1. Table 5 includes the results against this setting.
Limited Data Available: Here, we evaluate the learning by varying the percent of data used [10%, 30%, 50%, 70%, 90%]. From Fig. 4(c), we observe to better extract knowledge in all data-subset fractions. Specifically, performance difference becomes even more notable when very low fraction is used (like in 10% setting, we perform approx. 15% better than Nasty Teacher). In addition, we also observe that we always remains significantly closer to the original teacher.

Network	Nasty	HTC
CNN	16.38	**26.03**
MobileNetV2	41.79	**52.96**
ShuffleNetV2	70.04	**76.29**

Table 4: No Data Available Results for CIFAR10.

Dataset	Network	Nasty	HTC
CIFAR-10	CNN	82.64	**87.48**
CIFAR-100	Shufflenetv2	64.41	**74.17**

Table 5: No Label Available results. Using images without their ground truth.

No Data Available: Here, we consider the setting where no data is available. More precisely, we take the existing data-free distillation method [5], and add our method to it. Table 4 demonstrates consistent improvement while learning from nasty in this setting.

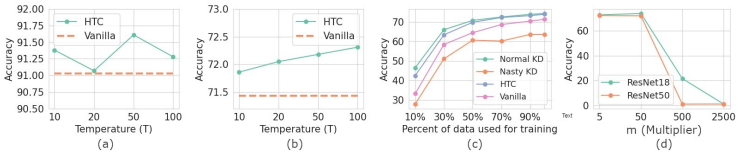

Fig. 4: Effect of Temperature: **(a)** C10, Student : ShuffleNetV2, Teacher : ResNet18 and **(b)** C100, Student : ShuffleNetV2, Teacher : ResNet50. **(c)** Effect of percentage of data. **(d)** Effect of multiplier m.

4.5 Leveraging SCM for an improved defense

Previously, we discuss the vulnerabilities of Nasty Teacher [22]. We now discuss this section in regards to improve "Nasty Teacher" defense.

Nasty Teacher derives its efficacy from the peaks created for the incorrect classes (Fig. 1). Along the lines that each in-correct peak adds to the confusion while learning, we hypothesize that the number of in-correct peaks in the output distribution affects the effectiveness of the defense. In general, more number of peaks can be correlated with the increased confusion, hence a better defense. However, in case of Nasty we often see the number of peaks to be not many in comparison to the total number of classes (see Fig. 1(a) for illustration), thereby creating a chance to improve Nasty's defense via increasing the number of in-correct peaks. In lieu of this observation, we propose to use our previously proposed Sequence of Contrastive Model Training for an improved defense. As model obtained with each SCM's step displays diverse set of peaks, we expect them to exhibit more number of peaks, largely because they were trained in a way to eventually contrast from nasty teacher (i.e low number of peaks). Fig. 1 illustrates this via columns (b),(c) where Seq-1/Seq-2 to have more number of peaks than Nasty Teacher. Thus, we choose one of these intermediate sequence models ($S^1, S^2 ... S^k$) for teacher and further show our results (denoted by **Ours**) in Table 6. For ease of our exploration, we typically test out with first or second model from our sequence in 3.4 as our defense teacher. Though our model incurs a small drop in accuracy ($< 2\%$), the significant improvement (sometimes, as much as 10%) it produces while defending makes it attractive for application. Diving deep into the results, one may consider this

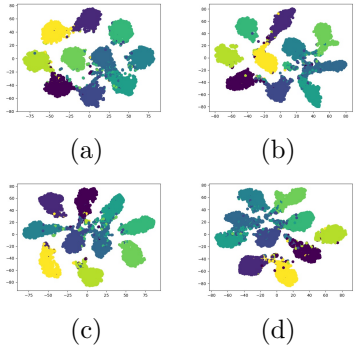

Fig. 5: t-SNE plots of features before FC layers for SCM on CIFAR-10. Model : ResNet-18, **a)** Normal Teacher **b)** Nasty Teacher **c)** Seq-1 **d)** Seq-2

CIFAR-10 ResNet18	Normal Teacher (Acc=95.09)	Nasty KD (Acc=94.28)	Nasty KD ES (Acc=93.02)	Ours (Acc=92.99)
CNN	85.99	82.27	84.62	**80.13**
MobileNetV2	89.58	31.73	28.58	**22.12**
ShuffleNetV2	91.03	79.73	87.81	**73.23**
CIFAR-100 ResNet50	Normal Teacher (Acc=77.96)	Nasty KD (Acc=77.4)	Nasty KD ES (Acc=76.2)	Ours (Acc=75.44)
CNN	58.38	58.93	58.45	**54.26**
MobileNetV2	71.43	3.03	2.16	**1.4**
ShuffleNetV2	68.90	63.16	62.95	**54.00**

Table 6: Results of our defense, Nasty KD and Nasty KD ES. Training hyper-parameters are same as discussed in section 4.1.

degradation in KD related to the decrease in accuracy. We now evaluate this dimension for a better understanding. Specifically, we obtain a model with an accuracy similar to ours by training a Nasty Teacher (refer 3.1) followed by early stopping it at the desired accuracy. We dub this approach Nasty KD ES (Early Stopped) and include the results in Table 6. We observe that for a similar accuracy Nasty KD ES has a significantly poor defense against **our method**, and in some cases it even defends poorly compared to original Nasty KD. Moreover, we visualize the features of intermediate sequence models in Fig. 5 t-SNE [24] plots and infer the intermediates i.e Seq-1 and Seq-2 to possess a similar class separation as the normal teacher 5(a) while maintaining the desired defense.

5 Conclusion

In this work, we focus on the threat of KD-based model stealing; specifically, the recent work Nasty Teacher [22] which proposes a defense against such stealing. We study [22] from two different directions and systematically show that we can still extract knowledge from Nasty Teacher with our approaches: HTC and SCM. Extensive experiments demonstrate our efficacy to extract knowledge from Nasty. Leveraging the insights we gain in our approaches, we finally also discuss an extension of Nasty Teacher that serves as a better defense. Concretely, we highlight a few dimensions that affect the defense against KD-based stealing to facilitate subsequent efforts in this direction. As our future work, we intend to improvise the existing defense and simultaneously explore such stealing in a relatively white-box setting, wherein the goal will be to defend even if the adversary gets hold of the model features/parameters.

Acknowledgements. This work was partly supported by the Department of Science and Technology, India through the DST ICPS Data Science Cluster program. We also thank the anonymous reviewers for their valuable feedback in improving the presentation of this paper.

References

1. Pytorch model hub (last accessed on 8 march 2022). https://pytorch.org/hub/ (2022) 6
2. Tensorflow model hub (last accessed on 8 march 2022). https://www.tensorflow.org/hub (2022) 6
3. Aïvodji, U., Gambs, S., Ther, T.: Gamin: An adversarial approach to black-box model inversion. ArXiv (2019) 5
4. Bucilua, C., Caruana, R., Niculescu-Mizil, A.: Model compression. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 535–541. KDD '06, Association for Computing Machinery (2006) 3
5. Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., Tian, Q.: Dafl: Data-free learning of student networks. In: ICCV (2019) 6, 12
6. Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021) 1
7. Elsken, T., Metzen, J.H., Hutter, F.: Simple and efficient architecture search for convolutional neural networks (2018) 6
8. Fan, L., Ng, K.W., Chan, C.S.: Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In: Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019) 3
9. Furlanello, T., Lipton, Z.C., Tschannen, M., Itti, L., Anandkumar, A.: Born-again neural networks. In: International Conference on Machine Learning, ICML 2018. vol. 80, pp. 1602–1611 (2018) 6
10. Gaudel, R., Sebag, M.: Feature selection as a one-player game. p. 359–366. ICML'10, Omnipress, Madison, WI, USA (2010) 6
11. Goldblum, M., Fowl, L., Feizi, S., Goldstein, T.: Adversarially robust distillation. Proceedings of the AAAI Conference on Artificial Intelligence **34**(04), 3996–4003 (Apr 2020) 1
12. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. International Journal of Computer Vision **129**(6), 1789–1819 (Jun 2021) 3, 5
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016) 9
14. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop (2015) 3, 5
15. Juuti, M., Szyller, S., Dmitrenko, A., Marchal, S., Asokan, N.: Prada: Protecting against dnn model stealing attacks. 2019 IEEE European Symposium on Security and Privacy (EuroS&P) pp. 512–527 (2019) 3
16. Kahla, M., Chen, S., Just, H.A., Jia, R.: Label-only model inversion attacks via boundary repulsion (To Appear CVPR 2022) 5
17. Kariyappa, S., Qureshi, M.K.: Defending against model stealing attacks with adaptive misinformation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 3
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015) 10

19. Kundu, S., Sun, Q., FU, Y., Pedram, M., Beerel, P.A.: Analyzing the confidentiality of undistillable teachers in knowledge distillation. In: 35th Neural Information Processing Systems (2021) 1, 3, 5, 10, 11
20. Liu, H., Simonyan, K., Yang, Y.: DARTS: Differentiable architecture search. In: International Conference on Learning Representations (2019) 6
21. Liu, Y., Zhang, W., Wang, J., Wang, J.: Data-free knowledge transfer: A survey (2021) 3, 5
22. Ma, H., Chen, T., Hu, T.K., You, C., Xie, X., Wang, Z.: Undistillable: Making a nasty teacher that {cannot} teach students. In: International Conference on Learning Representations (2021) 1, 2, 3, 4, 6, 9, 10, 13, 14
23. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV) (September 2018) 10
24. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9**, 2579–2605 (2008) 14
25. Maini, P., Yaghini, M., Papernot, N.: Dataset inference: Ownership resolution in machine learning. In: International Conference on Learning Representations (2021) 3
26. Mirzadeh, S., Farajtabar, M., Li, A., Levine, N., Matsukawa, A., Ghasemzadeh, H.: Improved knowledge distillation via teacher assistant. Proceedings of the AAAI Conference on Artificial Intelligence **34**, 5191–5198 (04 2020) 10
27. Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E.B., Turaga, D.: Learning feature engineering for classification. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. p. 2529–2535. IJCAI'17, AAAI Press (2017) 6
28. Park, W., Kim, D., Lu, Y., Cho, M.: Relational knowledge distillation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3967–3976 (2019) 3
29. Rajasegaran, J., Khan, S., Hayat, M., Khan, F.S., Shah, M.: Self-supervised knowledge distillation for few-shot learning. https://arxiv.org/abs/2006.09785 (2020) 1
30. Razzhigaev, A., Kireev, K., Kaziakhmedov, E., Tursynbek, N., Petiushko, A.: Black-box face recovery from identity features. In: ECCV Workshops (2020) 5
31. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. In: Bengio, Y., LeCun, Y. (eds.) International Conference on Learning Representations, ICLR 2015 3
32. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015) 5
33. Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 4510–4520 (2018) 10
34. Uchida, Y., Nagai, Y., Sakazawa, S., Satoh, S.: Embedding watermarks into deep neural networks. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval. p. 269–277 (2017) 3
35. Wang, L., Yoon, K.: Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. IEEE Transactions on Pattern Analysis Machine Intelligence (01), 1–1 (5555) 3, 5
36. Yim, J., Joo, D., Bae, J., Kim, J.: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7130–7138 (2017) 3

37. Yin, H., Molchanov, P., Alvarez, J.M., Li, Z., Mallya, A., Hoiem, D., Jha, N.K., Kautz, J.: Dreaming to distill: Data-free knowledge transfer via deepinversion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 3, 5, 6
38. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 5
39. Yuan, L., Tay, F.E., Li, G., Wang, T., Feng, J.: Revisiting knowledge distillation via label smoothing regularization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3903–3911 (2020) 6
40. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In: International Conference on Learning Representations, ICLR 2017 3
41. Zhang, J., Chen, D., Liao, J., Fang, H., Zhang, W., Zhou, W., Cui, H., Yu, N.: Model watermarking for image processing networks. Proceedings of the AAAI Conference on Artificial Intelligence **34**(07), 12805–12812 (Apr 2020) 3
42. Zhang, J., Chen, D., Liao, J., Zhang, W., Hua, G., Yu, N.: Passport-aware normalization for deep model protection. In: Advances in Neural Information Processing Systems (NeurIPS) (2020) 3
43. Zhang, Y., Jia, R., Pei, H., Wang, W., Li, B., Song, D.: The secret revealer: Generative model-inversion attacks against deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 5