2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) | 978-1-6654-4899-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CVPRW53098.2021.00362

Renofeation: A Simple Transfer Learning Method for Improved Adversarial Robustness

Ting-Wu Chin[1], Cha Zhang[2], Diana Marculescu[13]
Carnegie Mellon University[1], Microsoft Cloud and AI[2], The University of Texas at Austin[3]
tingwuc@andrew.cmu.edu, chazhang@mirosoft.com, dianam@utexas.edu

Abstract

Fine-tuning through knowledge transfer from a pre-trained model on a large-scale dataset is a widely spread approach to effectively build models on small-scale datasets. In this work, we show that a recent adversarial attack designed for transfer learning via re-training the last linear layer can successfully deceive models trained with transfer learning via end-to-end fine-tuning. This raises security concerns for many industrial applications. In contrast, models trained with random initialization without transfer are much more robust to such attacks, although these models often exhibit much lower accuracy. To this end, we propose noisy feature distillation*, a new transfer learning method that trains a network from random initialization while achieving clean-data performance competitive with fine-tuning.*

1. Introduction

Transfer learning is an important approach that enables training deep neural networks faster and with relatively less data than training from scratch without any prior knowledge. Specifically, we consider the setting where we want to maximize the performance on a target task assuming the availability of a pre-trained model trained on a source task. This setting has various applications and has led to state-of-the-art performance in several image classification tasks [4]. Moreover, this setting is also considered in industry in the form of machine-as-a-service, such as Google's Cloud AutoML [10] and Microsoft's Custom Vision service [19] where users can upload custom data to fine-tune a pre-trained model. We refer to this setting as transfer learning throughout this paper.

Transfer learning for ConvNets has received great attention due to its effectiveness in achieving high accuracy. It has been shown [30] that the pre-trained model trained on a large-scale dataset (such as ImageNet) acts as an effective feature extractor that supersedes hand-crafted feature extractors. Subsequent work [41, 6] has found that inheriting the pre-trained weights and starting learning from there (often referred to as "fine-tuning") can result in even larger performance improvements. Fine-tuning has then been adopted in various tasks to achieve state-of-the-art results. Besides fine-tuning, several prior methods have relied on fine-tuning with an explicit regularization loss to further enhance the performance of transfer learning [39, 18]. While prior art has demonstrated that fine-tuning might not necessarily outperform training from random initialization for some tasks, such as classifying medical images [25] and object detection and semantic segmentation with sufficient training data [11], it is important to note that fine-tuning is the state-of-the-art method for small and visually similar datasets such as the Caltech-UCSD Bird 200 datasets [37].

Very recently it has been demonstrated [27] that models transferred by re-learning the last linear layer are vulnerable to adversarial examples crafted solely based on the pre-trained model. In other words, an adversary can attack a pre-trained model available on open repositories, *e.g.*, TorchVision, and use the adversarial image to deceive the transferred models. In this paper, we show that such an attack can also deceive models transferred with end-to-end fine-tuning. This finding raises security concerns for the widely-adopted fine-tuning mechanism, which is also used in industrial applications such as Google's AutoML [10] and Microsoft's Custom Vision [19]. In this work, we take a first step toward alleviating this problem. Intuitively, the vulnerability to such an attack stems from the similarity between the pre-trained and the transferred models. However, we find that models transferred with existing fine-tuning methods are similar to the pre-trained ones, which in turn makes them vulnerable to the attack developed by Rezaei *et al.* [27]. In contrast, models trained with random initialization are much more robust to such attacks, with the caveat that these models often exhibit much lower accuracy compared to fine-tuning. As an alternative to prior methods, we propose re-training with noisy feature distillation (or **Renofeation** for short), which achieves clean-data performance similar to fine-tuning and the robustness of train-

978-1-6654-4899-4/21/$31.00 ©2021 IEEE
DOI 10.1109/CVPRW53098.2021.00362

ing with random initialization. Overall, our contributions are as follows:

- We show that the attack proposed in prior work is suitable not only for transfer learning by re-training the last linear layer, but also for transfer learning with end-to-end fine-tuning, which raises security concerns for the widely-adopted fine-tuning paradigm.
- We propose Renofeation, a new transfer learning method that results in competitive clean-data performance compared to fine-tuning with significant better robustness. **Compared to previous transfer learning methods, ours is the first that argues for "reinitializing the weights"**.
- We conduct extensive experiments on four networks and five datasets with hyper-parameter tuning and ablation studies to empirically demonstrate the effectiveness of the proposed method.

2. Background

2.1. Transfer learning

It is known that deep neural networks trained on large-scale datasets such as ImageNet learn surprisingly transferable features [30]. That is, one can re-purpose a pre-trained network to other classification tasks by simply learning a linear classifier on top of the features from the penultimate layer. Later, researchers have found that when the entire pre-trained model is optimized with a small learning rate, performance can be even better [17, 6, 26, 11, 41], and this scheme is also known as "fine-tuning". With the desire of not forgetting the useful features learned from the large-scale dataset, explicit regularization was proposed to further improve fine-tuning. Specifically, L2SP [39, 16] imposes regularization to avoid weights deviating from the pre-trained weights in a ℓ_2 sense. Similarly, DELTA [18, 14, 34] imposes regularization to avoid representations deviating from the pre-trained representations. Besides these transfer learning methods, training from random initialization is often considered as the baseline for transfer learning, which does not leverage the information learned from the pre-trained model.

Transfer learning using extra information or architectural changes have also been investigated in the literature. Ge *et al.* [7] developed a method to improve fine-tuning by leveraging additional training data obtained from large-scale datasets. Cui *et al.* [4] used Earth Mover's Distance to measure domain similarity between datasets and showed that pre-training on similar domains results in better transfer. Wang *et al.* [36] discovered that increasing the model capacity (wider or deeper) improves the effectiveness of fine-tuning.

2.2. Adversarial examples

Adversarial examples [32] for deep learning models have received growing attention due to their potential impact on machine learning systems. According to different threat models, there are various types of attacks. In a white-box threat model, where the adversary knows all the information regarding a model, fast gradient sign method (FGSM) [9], projected gradient descent, and CW [1] have been shown to be strong attacks. Counteracting these attacks, adversarial training [21] is the dominant approach for robusifying deep networks. On the other hand, there are also methods targeting a black-box threat model where the adversary can only query the model and obtain the probability vector [20, 23, 2].

In this work, our threat model assumes that the adversary has access to the *model weights* and *model architecture* for the *pre-trained model*. The adversary *does not* have access to the *task-specific transferred model*. This threat model aligns with practical usage of deep learning models where researchers use pre-trained models on large datasets (like ImageNet) and fine-tune them for other tasks. Based on this threat model, prior art [27] has proposed an attack that successfully compromises the task-specific transferred models, which raises security concerns for transfer learning. In this work, we find that such an attack not only successfully deceives transfer learning by re-training the last linear layer, but also works for end-to-end fine-tuning. We further propose an algorithm to improve the robustness of the transferred model under this particular threat model. On a different threat model, Shafahi *et al.* [29] have proposed to improve the adversarial robustness of the transferred model in a white-box setting by transferring to the target model the robust features obtained through adversarial training. While in this work we use feature distillation to improve clean data performance, knowledge distillation has been explored to improve the robustness of the student model by distilling from a robust teacher [8].

To craft an adversarial example under our threat model, we adopt an attack from Rezaei *et al.* [27], which optimizes the following objective:

$$\begin{aligned} &\arg\min_{\delta} \ \|f_K(x+\delta,\theta_0)-t\|_2^2 \\ &s.t. \ \|\delta\|_\infty \leq B, \end{aligned} \tag{1}$$

where f_K is the output of the penultimate layer, t is a target vector that is set to a scalar m multiplied by a one-hot vector. m is chosen to be large and B denotes the perturbation budget. The pixel intensity in this formulation is normalized and constrained to $[0, 1]$. We optimize equation 1 via projected gradient descent (PGD). Intuitively, the objective is trying to find a small-norm perturbation such that the response of the penultimate layer of the pre-trained model is polarized. Once the perturbation δ for a specific input im-

age x is found, the perturbed image $x + \delta$ is used to attack a transferred model θ.

3. Motivation

We start with the following research question: "*Can the attack proposed by Rezaei et al. [27] compromise transfer learning that fine-tunes the entire model?*" This is unclear as such an attack was originally proposed to deceive a specific transfer learning method, *i.e.*, re-learning the last linear layer. Since end-to-end fine-tuning provides much better performance compared to only learning the last linear layer [18], fine-tuning is a widely adopted method for transfer learning. As a result, it would be less concerning if such an attack only works for re-learning the last linear layer but not end-to-end fine-tuning.

To answer this question, we consider five training methods with five transfer learning datasets. For training methods, we consider *Linear classifier* that only re-learn the last linear layer, *Fine-tuning* that trains all the parameters, *L2SP* [39] that trains all the parameters with weight regularization, *DELTA* that trains all the parameters with representation regularization, and a baseline *Re-training* that trains from random initialization using the target dataset without any transfer learning. We summarize the methods used in Table 1. As for datasets, we consider Stanford Dog [15], Caltech-UCSD Bird [37], Stanford Actions [40], MIT Indoor Scenes [24], and Flower [22]. We proceed by crafting adversarial examples by solving equation (1) using PGD. Then, we evaluate the *attack success rate (ASR)*, which is calculated by the conditional probability $P(\text{wrong with adversarial-data} \mid \text{correct with clean-data})$, for each method and dataset combination. To provide context, we also evaluate the top-1 image classification accuracy on clean images for each method and dataset combination.

As shown in Table 2, we find that the attack proposed by Rezaei *et al.* [27] can deceive models trained with end-to-end fine-tuning with high attack success rate. This raises security concerns for the widely-adopted fine-tuning paradigm. Besides confirming that DELTA is the best transfer learning method among the considered ones, we observe that although the clean data performance for re-training is less than ideal compared to transfer learning, it has low attack success rate. This observation leads us to the following question: "*Why are models trained with end-to-end fine-tuning vulnerable to such an attack while re-training are robust?*"

We conjecture that it is because the re-trained model has low similarity compared to the model under attack, *i.e.*, the pre-trained model, while the fine-tuned models are initialized with the pre-trained weights that lead to potential similarity. To verify our conjecture, we measure the correlation between the attack success rate and the ℓ_2 distance between the pre-trained and the transferred models. As for the distance measure, we look into two metrics. One is on the weight space, which measure the ℓ_2 distance between the pre-trained weights and the transferred weights. The other metric is on the feature space, where we compute the ℓ_2 feature distance averaged across different layers and training data (also known as the feature distillation loss). As shown in Figure 1, the distance between the transferred and the pre-trained models negatively correlates with attack success rate for both distance measures, which matches our conjecture.

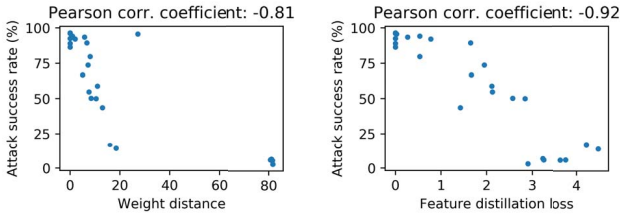

Figure 1: Robustness *vs.* distance between transferred and pre-trained models for the five baseline methods on five datasets discussed in Table 2

4. Methodology

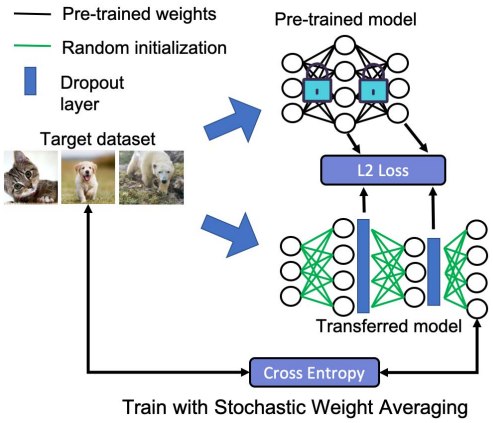

Figure 2: Schematic overview of Renofeation

To craft a defense mechanism based on our observation, the goal is to reduce model similarity without harming the benefits of transfer learning. To this end, we propose re-training with noisy feature distillation, or Renofeation for short. As shown in Fig. 2, Renofeation consists of two ingredients: (1) re-initialize the model with random initialization as opposed to inheriting the pre-trained weights and (2) train the model on the target dataset using noisy feature distillation. The first step removes the similarity with the pre-trained model that is embedded in the pre-trained weights. The second step uses feature distillation to encourage repre-

Table 1: Summary of different transfer learning methods. $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_0$ denote the weights for the transferred and pre-trained neural network, respectively. $f_l(\cdot,\cdot)$ denotes the output (feature) of the l^{th} layer.

	Re-training	Linear classifier	Fine-tuning	L2SP [39]	DELTA [18]	Renofeation (Ours)
Random init. (layer)	All	Last	Last	Last	Last	All
Variable (layer)	All	Last	All	All	All	All
Regularization	$\|\boldsymbol{\theta}\|_2^2$	$\|\boldsymbol{\theta}\|_2^2$	$\|\boldsymbol{\theta}\|_2^2$	$\|\boldsymbol{\theta}-\boldsymbol{\theta}_0\|_2^2$	$\sum_{l=1}^{L}\|f_l(x,\boldsymbol{\theta})-f_l(x,\boldsymbol{\theta}_0)\|_2^2$	$\sum_{l=1}^{L}\|f_l(x,\boldsymbol{\theta})-f_l(x,\boldsymbol{\theta}_0)\|_2^2$ Dropout Stochastic Weight Averaging
Transfer Mechanism	N/A	Weights	Weights	Weights	Weights and Features	Features
Defense Mechanism	Random init.	N/A	N/A	N/A	N/A	Random init. Feature regularization

Table 2: Robustness evaluation for the baseline transfer learning methods for ResNet18. ASR denotes attack success rate, which is computed as $P(\text{wrong with adversarial-data} \mid \text{correct with clean-data})$ (the lower the more robust). Clean denotes the Top-1 accuracy for clean-data.

		Dog	Bird	Action	Indoor	Flower
Linear classifier	clean	84.22	67.02	73.64	72.54	88.52
	ASR	96.06	96.47	92.49	88.95	86.40
Fine-tuning	clean	81.84	77.67	77.19	75.37	95.71
	ASR	89.36	50.33	73.75	54.75	14.07
L2SP	clean	83.82	77.51	77.22	75.15	95.63
	ASR	94.08	50.08	92.16	66.73	16.75
DELTA	clean	**84.39**	**78.75**	**77.69**	**78.36**	**95.90**
	ASR	95.65	58.83	93.51	79.71	43.65
Re-training	clean	70.77	69.76	51.90	59.93	87.38
	ASR	**5.99**	**6.14**	**5.82**	**6.73**	**3.00**

sentation similarity to the pre-trained model for improving clean data performance while using a noisy process to discourage over-fitting to the pre-trained representation. As for the implementation for "noisy" feature distillation, we adopt two regularization methods: spatial dropout [33] during training and stochastic weight averaging [13].

Dropout Dropout was proposed to avoid co-adaptation among neurons by randomly dropping out features during training [12]. In this work, we consider spatial-dropout [33] for convolutional layers. Dropout has been used as a successful defense during evaluation time [35] which suggests that the adversarial features are highly co-adapted. As a result, instead of plain feature distillation that tries to mimic all the features of the pre-trained network, we propose to match the randomly dropped features to reduce the possibility of learning vulnerable features. We note that we do not randomly drop features during the evaluation time.

Stochastic Weight Averaging (SWA) SWA has shown great promise in improving the generalization performance for deep neural networks [13]. The core idea is to average numerous local optima to form the final solution. It has been demonstrated empirically that SWA improves generalization while increasing the training loss. The rationale behind adopting SWA is that SWA is shown to be a successful technique that trades training loss for testing loss, which is exactly our goal: increasing the feature distillation loss without hurting the prediction performance.

5. Experiments

5.1. Datasets and implementation detail

In this work, we consider five datasets to transfer to and models trained on ImageNet as pre-trained models. The datasets under consideration are shown in Table 3.

For training, we use a batch size of 64 and stochastic gradient descent with momentum following prior art [18, 39]. For the experiments using fine-tuning, *i.e.*, those that start with pre-trained weights, we use 30,000 iterations to make sure the loss converges. Additionally, we tune the learning rate, weight decay, and momentum for fine-tuning each dataset according to prior art [17]. Specifically, we tune learning rate $\in \{0.01, 0.005\}$, momentum $\in \{0, 0.9\}$, and weight decay $\in \{0, 10^{-4}\}$ using grid search. For re-training, the hyper-parameters are set throughout the experiments across datasets without tuning. We use 90,000 iterations, learning rate 0.01, momentum 0.9, and weight decay 0.005. Also, weight decay for the last linear layer is set to 0.01 across all the experiments following [18, 39]. We use cosine learning rate decay for all the experiments. For fine-tuning methods that come with hyper-parameters such as L2SP and DELTA, we tune λ to obtain the best transferred results according to prior art [39, 18].

We apply SWA by training with half of the learning rate, *i.e.*, 0.005, as suggested in prior art [13]. SWA training considered has constant learning rate with 30,000 iterations. We average the models every 500 iterations. We insert the dropout layer after those that are used for the feature distillation loss and we use a dropout rate of 10%. Regarding

Table 3: The characteristics of the datasets for transfer learning we considered in this work. We includes the number of training samples per class, the number of testing samples per class, and the number of classes.

Dataset	Task Category	# Training Samples	# Testing Samples	# Classes	Abbreviation
Stanford Dogs [15]	Fine-grained classification	100	≈72	120	Dog
Caltech-UCSD Birds [37]	Fine-grained classification	≈30	≈ 29	200	Bird
Stanford 40 Actions [40]	Action classification	100	≈ 138	40	Action
MIT Indoor Scenes [24]	Indoor scene classification	80	20	67	Indoor
102 Category Flower [22]	Fine-grained classification	20	≈60	102	Flower

the parameters for crafting the adversarial examples, we set the perturbation budget B to 0.4, the number of iterations of PGD to be 40, m to be 1000 (following [27]), and the learning rate to be 0.01. We set the target t to be one-hot that always have one in the first neuron and zero for other neurons. We use AdverTorch [5] for generating adversarial examples using the above specified objective and parameters.

5.2. Ablating the proposed components

In this subsection, we are interested in understanding the importance of the different components in the proposed Renofeation. Specifically, we would like to understand the impact of random initialization, dropout, and stochastic weight averaging. To do so, we have three baselines: (1) DELTA, which is the best transfer learning method in clean performance as shown in Table 2, (2) Re-training without transfer, which is the most robust method in Table 2, and (3) DELTA-R, which is DELTA with random (as opposed to pre-trained weights) initialization. For each of these baselines, we add dropout (DO), stochastic weight average (SWA), and both of them to see how these techniques affect the clean data performance and attack success rate. We conduct all the experiments in this subsection using ResNet-18.

Importance of random initialization From Table 2, we can observe that DELTA has the best clean data performance while re-training has the best robustness. Since DELTA achieves transfer via pre-trained weights and feature distillation, an interesting question arises: *Do pre-trained weights help clean data performance and hurt robustness equally?* To answer this question, we compare DELTA and DELTA with random initialization (DELTA-R) in both clean data performance and ASR. As shown in Figure 3, we find that the pre-trained weights merely help clean data performance at the presence of feature distillation but hurts robustness significantly. This suggests that random initialization effectively makes the function less similar to the function induced by the pre-trained weights even in the presence of feature distillation. This is plausible because feature distillation is enforcing the two functions to be similar only at the training data points, which are scarce for transfer learning.

The effect of regularization In Renofeation, both dropout (DO) and stochastic weight averaging (SWA) are proposed to be incorporated into the transfer learning process. It would be of interest to understand their respective impact on both robustness and clean data performance for all three baselines. Let us first focus on DELTA, as shown in Fig. 3, we can observe that SWA does not work well in improving the robustness for DELTA, which might be due to the local optimality of the pre-trained weights that leads to less diverse model weights throughout the fine-tuning process. On the other hand, DO improves the robustness for DELTA, but the robustness still falls short when compared to re-training. Both techniques marginally improve the clean data performance for DELTA except for the Dog dataset, where we see a slight accuracy drop.

Considering SWA and DO for re-training, we find that both techniques significantly improve the clean data performance for re-training and this is expected based on the results from the DO and SWA papers. Nonetheless, the clean data performance of Re-train with both techniques still underperforms DELTA by a significant margin in most datasets.

Lastly, we consider SWA and DO for DELTA-R. We find that SWA works much better for improving robustness when compared to applying SWA to DELTA. This suggests that the weight initialization greatly affects the training trajectory. When considering the clean data performance, both techniques again marginally improve the clean data performance for DELTA-R except for the Dog dataset. Overall, Renofeation, which consists of DELTA-R, DO, and SWA, achieves the best of both worlds, with clean data performance comparable to DELTA and the robustness comparable to re-training.

5.3. More networks

So far, we have conducted our experiments and analyses based on ResNet-18. We are interested to see if Renofeation is still more preferable compared to re-training

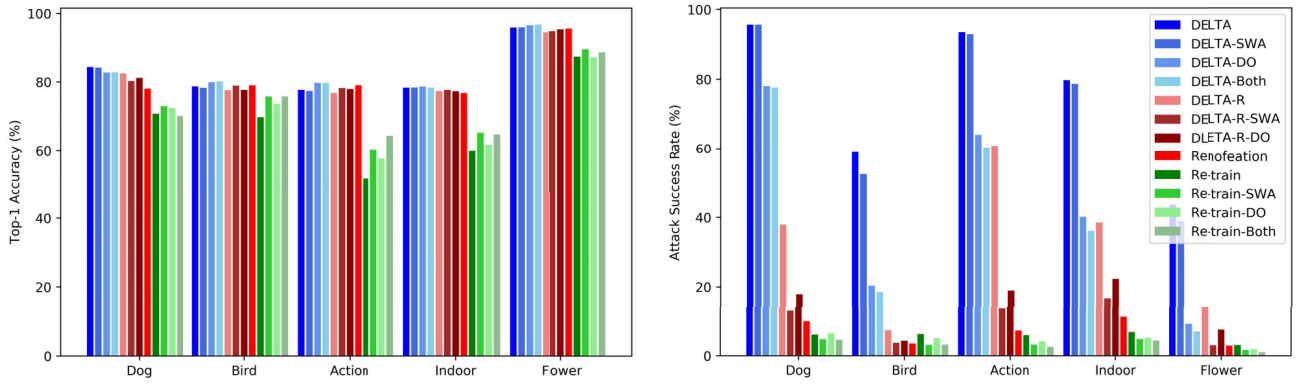

Figure 3: Ablation study of the effect of dropout (DO) and SWA on (Left) clean-data performance and (Right) attack success rate for the two baselines including DELTA and re-training.

Table 4: Comparing DELTA, Renofeation, and re-training for different ConvNets. Renofeation has the clean data performance comparable to DELTA and robustness comparable to re-training for different ConvNets we study.

			Dog	Bird	Action	Indoor	Flower	Average
ResNet-18	DELTA	CLEAN	**84.39**	78.75	77.69	**78.36**	**95.90**	-
		ASR	95.65	58.83	93.51	79.71	43.65	74.27
	Renofeation	CLEAN	78.11	**79.03**	**79.07**	76.79	95.59	-
		ASR	9.83	**3.41**	7.16	11.08	**2.86**	6.87
	Re-training	CLEAN	70.77	69.76	51.90	59.93	87.38	-
		ASR	**5.99**	6.14	**5.82**	**6.73**	3.00	5.54
ResNet-50	DELTA	CLEAN	**90.13**	**81.95**	81.87	79.93	96.63	-
		ASR	94.69	32.29	91.94	84.69	24.84	65.69
	Renofeation	CLEAN	83.57	79.27	**84.04**	**80.67**	**96.75**	-
		ASR	**5.08**	**3.96**	**3.33**	7.12	2.39	4.38
	Re-training	CLEAN	72.55	70.47	53.53	59.11	85.93	-
		ASR	6.30	7.45	6.11	**6.06**	**2.20**	5.62
ResNet-101	DELTA	CLEAN	**91.92**	**82.07**	82.61	80.00	**96.37**	-
		ASR	88.03	42.60	87.53	89.27	44.04	70.29
	Renofeation	CLEAN	83.88	80.98	**84.67**	**80.97**	96.33	-
		ASR	**4.38**	**3.54**	**3.69**	9.95	3.09	4.93
	Re-training	CLEAN	73.42	71.80	52.78	61.12	85.59	-
		ASR	6.64	7.21	6.64	**5.13**	**2.00**	5.52
MobileNetV2	DELTA	CLEAN	**84.86**	**78.51**	**78.94**	76.12	**96.68**	-
		ASR	82.89	40.30	57.00	52.45	21.08	50.75
	Renofeation	CLEAN	76.42	75.70	77.78	**76.49**	96.32	-
		ASR	11.62	**6.79**	**5.92**	**7.12**	2.84	6.86
	Re-training	CLEAN	67.95	69.50	52.86	61.49	88.73	-
		ASR	**8.56**	8.54	8.35	7.65	**2.71**	7.16

and DELTA for other networks. Specifically, we further consider deeper networks, *i.e.*, ResNet-50, and ResNet-101. Additionally, due to recent interests in reducing the computational overhead of ConvNets for deployment purpose [34, 31, 28, 3, 38], we also consider a compact network, *i.e.*, MobileNetV2 [28]. Due to computational considerations, for DELTA with other networks, we inherit the learning rate, weight decay, and momentum from ResNet-18 for each of the dataset.

As shown in Table 4, Renofeation achieves clean-data performance comparable to that of DELTA and has robustness similar to re-training across all ConvNets we have in-

vestigated. We note that while Renofeation in general has clean data performance comparable to DELTA, it is not the case for the Dog dataset, where DELTA consistently has higher accuracy compared to Renofeation. We hypothesize that the accuracy loss in this case may be due to the Dog dataset being a strict subset of the ImageNet dataset and therefore matching features alone on the scarce target dataset may not be sufficient to recover the features for good generalization. This can be inferred from the fact that the linear classifier alone has clean data performance matching DELTA for the Dog dataset as shown in Table 2. While it is less likely to conduct transfer learning to a target dataset that is a strict subset of the source dataset, this phenomenon also suggests that there is room for improvement for future research to tackle the studied threat model.

5.4. Data amount ablation

From previous results, we show that feature distillation using the target dataset is able to achieve competitive clean data performance compared to fine-tuning. Intuitively, if the amount of training data is large, feature distillation should be able recover the knowledge encoded in the pre-trained weights. However, in the transfer learning case, target datasets usually have much less training data compared to large-scale datasets such as ImageNet. In this section, we ablate the number of training samples to understand how it affects the effectiveness of Renofeation so as to further provide a guideline for when to use it. Specifically, we consider cases where the training data for each dataset is reduced to 33% and 66%. For each class in the dataset, we randomly sub-sample 33% and 66% of the training images. As a result, the overall training dataset is still balanced across classes.

As shown in Table 5, we find that as the training data size decreases, the clean data performance gap between Renofeation and DELTA increases. This is expected as feature distillation with fewer samples makes it an underdetermined problem to match the function of the pre-trained model. However, Renofeation still greatly improves over DELTA in robustness and greatly improves over re-training in clean data performance.

5.5. Adversarial training

While we showed that our proposed Renofeation approach, when compared to DELTA, achieves better robustness with comparable clean-data performance under our threat model, adversarial training can also be considered as a defense under our threat model. As a result, in this section, we compare our method with adversarial training to further demonstrate its effectiveness. To conduct adversarial training in our considered threat model, we train DELTA with $2\times$ more iterations and, we craft adversarial examples with three iterations of projected gradient descent. As shown in Table 6, adversarial training indeed achieves better robustness compared to the baselines but worse compared to Renofeation. This is because, as a defense method, Renofeation has used the prior that the attack is generated using the pre-trained model and defends accordingly using random initialization while adversarial training has not harnessed this prior.

Table 5: Ablating the number of training samples for each dataset to 33% and 66% and compare the performances among methods. As the training data gets smaller in size, Renofeation provides more improvement in clean data performance compared to re-training while having robustness much better than DELTA.

			Dog	Bird	Action	Indoor	Flower
33%	DELTA	CLEAN	**81.80**	**63.41**	**70.72**	**70.97**	**90.11**
		ASR	95.77	74.12	93.94	85.38	46.60
	RENOFEATION	CLEAN	74.13	61.75	69.22	70.22	88.32
		ASR	10.88	**5.56**	9.40	15.73	4.94
	RE-TRAINING	CLEAN	44.98	26.10	24.51	37.54	62.73
		ASR	**9.67**	14.68	**9.22**	**8.35**	**2.85**
66%	DELTA	CLEAN	**83.58**	73.04	75.52	**75.30**	**94.23**
		ASR	95.36	64.58	93.80	80.77	56.39
	RENOFEATION	CLEAN	77.25	**74.46**	**76.09**	74.48	93.56
		ASR	9.85	**3.55**	7.53	12.22	4.73
	RE-TRAINING	CLEAN	64.03	56.47	40.73	52.61	80.60
		ASR	**7.72**	10.79	**5.86**	**7.23**	**3.23**

Table 6: Comparison among DELTA, DELTA with PGD-3 adversarial training, and proposed Renofeation. Renofeation has the best robustness with comparable clean data performance with other DELTA variants.

		Dog	Bird	Action	Indoor	Flower
DELTA	CLEAN	**84.39**	78.75	77.69	**78.36**	95.90
	ASR	95.65	58.83	93.51	79.71	43.65
DELTA ADV. TRAINED	CLEAN	82.83	77.10	75.69	77.84	95.12
	ASR	85.86	16.77	85.19	61.84	23.85
DELTA ADV. TRAINED + SWA + DO	CLEAN	81.42	**80.20**	**79.12**	78.28	**96.81**
	ASR	53.03	8.93	63.72	36.60	4.92
RENOFEATION	CLEAN	78.11	79.03	79.07	76.79	95.59
	ASR	**9.83**	**3.41**	**7.16**	**11.08**	**2.86**

5.6. Regularization and feature distillation

We have shown in previous sections that both dropout (DO) and stochastic weight averaging (SWA) are helpful in reducing the attack success rate. It is not clear if this happens due to the reasons we expected: "reduce overfitting in terms of the feature distillation loss." As a result, we analyze the impact on feature distillation loss when DELTA-R is augmented with DO and SWA. As shown in Table 7, we observe that these regularization techniques indeed increase the feature distillation loss, which in turn improves the robustness of DELTA-R. Additionally, we find empirically that both dropout and SWA can work together to achieve

better regularization.

Table 7: The effect of dropout and SWA on feature distillation loss, clean-data performance, and robustness for DELTA-R and ResNet-18.

		Dog	Bird	Action	Indoor	Flower
DELTA-R	clean	**82.49**	77.58	76.79	**77.39**	94.49
	ASR	37.93	7.28	60.83	38.48	14.30
	Feature loss	0.70	1.48	0.72	0.68	0.56
DELTA-R + Dropout	clean	81.21	77.72	78.00	77.31	95.35
	ASR	17.91	4.24	18.98	22.30	7.44
	Feature loss	0.86	1.57	1.03	0.89	0.68
DELTA-R + SWA	clean	80.32	78.92	78.07	77.69	94.81
	ASR	12.87	3.65	23.62	16.72	2.95
	Feature loss	0.86	1.63	0.87	0.82	0.73
Renofeation	clean	78.11	**79.03**	**79.07**	76.79	**95.59**
	ASR	**9.83**	**3.41**	**7.16**	**11.08**	**2.86**
	Feature loss	**1.00**	**1.68**	**1.06**	**1.02**	**0.81**

5.7. Tuning hyperparameters for DELTA

A natural idea to reduce the impact of feature distillation is to tune its corresponding weight (λ_{feat}) on the training loss. As shown in Figure 4, even the best λ_{feat} still incurs high ASR for datasets such as Indoor, Dog, and Action. The performance gained obtained by Renofeation cannot be obtained by simply tuning the weight for the feature distillation loss.

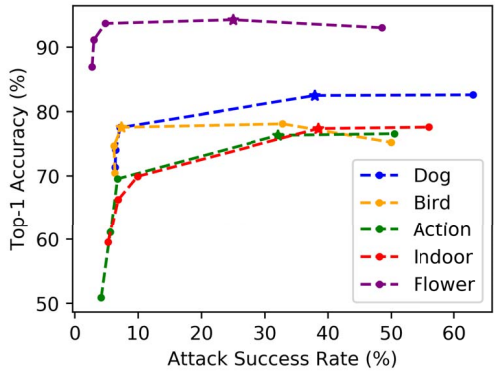

Figure 4: The effect of tuning λ_{feat} on the trade-off between clean-data performance and the attack success rate for ResNet-18. Star marks the λ_{feat} we use.

6. Conclusion

In this work, we first show that the attack proposed by Rezaei *et al.* [27] works not only for transfer learning by re-training the last linear layer, but also for end-to-end fine-tuning. This is concerning due to the widely adopted fine-tuning paradigm. We show that the attack success rate correlates well with the similarity between the pre-trained and the fine-tuned model. Based on this observation, we propose Renofeation, a transfer learning method that is significantly more robust to adversarial attacks crafted based on the pre-trained model when compared to state-of-the-art transfer learning methods based on fine-tuning. Renofeation has two key ingredients: (1) random initialization and (2) noisy feature distillation. We have extensively analyzed the proposed Renofeation empirically with ablation to demonstrate its effectiveness. While the threat model under consideration is relatively new [27], it is crucial to improve robustness under this threat model due to the practical popularity of fine-tuning. This work takes a first step towards improving the robustness under this threat model and sheds light on this topic for future study.

Acknowledgement

This research was supported in part by NSF CCF Grant No. 1815899, NSF CSR Grant No. 1815780, and NSF ACI Grant No. 1445606 at the Pittsburgh Supercomputing Center (PSC).

References

[1] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017. 2

[2] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26, 2017. 2

[3] Ting-Wu Chin, Ruizhou Ding, Cha Zhang, and Diana Marculescu. Legr: Filter pruning via learned global ranking. *arXiv preprint arXiv:1904.12368*, 2019. 6

[4] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[5] Gavin Weiguang Ding, Luyu Wang, and Xiaomeng Jin. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*, 2019. 5

[6] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 1, 2

[7] Weifeng Ge and Yizhou Yu. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1086–1095, 2017. 2

[8] Micah Goldblum, Liam Fowl, Soheil Feizi, and Tom Goldstein. Adversarially robust distillation. *arXiv preprint arXiv:1905.09747*, 2019. 2

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 2

[10] Google. Cloud automll. https://cloud.google.com/automl. 1

[11] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4918–4927, 2019. 1, 2

[12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 4

[13] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018. 4

[14] Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Learning what and where to transfer. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3030–3039, Long Beach, California, USA, 09–15 Jun 2019. PMLR. 2

[15] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, volume 2, 2011. 3, 5

[16] Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. *arXiv preprint arXiv:1909.11299*, 2019. 2

[17] Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Rethinking the hyperparameters for fine-tuning. In *International Conference on Learning Representations*, 2020. 2, 4

[18] Xingjian Li, Haoyi Xiong, Hanchao Wang, Yuxuan Rao, Liping Liu, and Jun Huan. DELTA: DEEP LEARNING TRANSFER USING FEATURE MAP WITH ATTENTION FOR CONVOLUTIONAL NETWORKS. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 4

[19] Olga Liakhovich and Claudius Mbemba. Food classification with custom vision service. https://www.microsoft.com/developerblog/2017/05/12/food-classification-custom-vision-service/, May 2017. 1

[20] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016. 2

[21] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 2

[22] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 3, 5

[23] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016. 2

[24] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420. IEEE, 2009. 3, 5

[25] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3342–3352. Curran Associates, Inc., 2019. 1

[26] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Advances in Neural Information Processing Systems*, pages 3342–3352, 2019. 2

[27] Shahbaz Rezaei and Xin Liu. A target-agnostic attack on deep models: Exploiting security vulnerabilities of transfer learning. In *International Conference on Learning Representations*, 2020. 1, 2, 3, 5, 8

[28] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 6

[29] Ali Shafahi, Parsa Saadatpanah, Chen Zhu, Amin Ghiasi, Christoph Studer, David Jacobs, and Tom Goldstein. Adversarially robust transfer learning. In *International Conference on Learning Representations*, 2020. 2

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 2

[31] Dimitrios Stamoulis, Ruizhou Ding, Di Wang, Dimitrios Lymberopoulos, Bodhi Priyantha, Jie Liu, and Diana Marculescu. Single-path nas: Designing hardware-efficient convnets in less than 4 hours. *arXiv preprint arXiv:1904.02877*, 2019. 6

[32] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 2

[33] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 4

[34] Kafeng Wang, Xitong Gao, Yiren Zhao, Xingjian Li, Dejing Dou, and Cheng-Zhong Xu. Pay attention to features, transfer learn faster {cnn}s. In *International Conference on Learning Representations*, 2020. 2, 6

[35] Siyue Wang, Xiao Wang, Pu Zhao, Wujie Wen, David Kaeli, Peter Chin, and Xue Lin. Defensive dropout for hardening deep neural networks under adversarial attacks. In *Proceedings of the International Conference on Computer-Aided Design*, pages 1–8, 2018. 4

[36] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Growing a brain: Fine-tuning by increasing model capacity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2471–2480, 2017. 2

[37] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 1, 3, 5

[38] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019. 6

[39] LI Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International Conference on Machine Learning*, pages 2825–2834, 2018. 1, 2, 3, 4

[40] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *2011 International Conference on Computer Vision*, pages 1331–1338. IEEE, 2011. 3, 5

[41] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014. 1, 2