Characteristic Examples: High-Robustness, Low-Transferability Fingerprinting of Neural Networks

Siyue Wang[1*], **Xiao Wang**[2*], **Pin-Yu Chen**[3], **Pu Zhao**[1] and **Xue Lin**[1]
[1]Northeastern University
[2]Boston University
[3]IBM Research
{wang.siy, zhao.pu, xue.lin}@northeastern.edu, kxw@bu.edu, pin-yu.chen@ibm.com

Abstract

This paper proposes *Characteristic Examples* for effectively fingerprinting deep neural networks, featuring high-robustness to the base model against model pruning as well as low-transferability to unassociated models. This is the first work taking both robustness and transferability into consideration for generating realistic fingerprints, whereas current methods lack practical assumptions and may incur large false positive rates. To achieve better trade-off between robustness and transferability, we propose three kinds of characteristic examples: *vanilla C-examples*, *RC-examples*, and *LTRC-example*, to derive fingerprints from the original base model. To fairly characterize the trade-off between robustness and transferability, we propose *Uniqueness Score*, a comprehensive metric that measures the difference between robustness and transferability, which also serves as an indicator to the false alarm problem. Extensive experiments demonstrate that the proposed characteristic examples can achieve superior performance when compared with existing fingerprinting methods. In particular, for VGG ImageNet models, using LTRC-examples gives $4\times$ higher uniqueness score than the baseline method and does not incur any false positives.

1 Introduction

With the rapid development of machine learning and artificial intelligence, the efforts and resources spent in developing state-of-the-art machine learning models such as deep neural networks (DNNs) can be tremendous, and therefore it is of utmost importance to be able to claim the ownership of a well-trained model and its derived versions (e.g. pruned models). For instance, the cost of training current state-of-the-art transformer based language model, GPT-3 [Brown and et al., 2020], is estimated to be at least 4.6 million US dollars[1]. Imagine an unethical model thief purposely pruned the pre-trained GPT-3 model and attempted to claim the ownership of the resulting compressed model. The solution to the challenge of "how to protect intellectual property for DNN models and reliably identify model ownership?" is literally worth million dollars.

Another motivating example is the surging trend of broad usage of neural network models across applications in cloud-based or embedded systems. For model owners deploying a model on the cloud, it is essential for them to verify the identity of the model to make sure that the model has not been tampered or replaced. Towards this direction, extensive research have been made to protect the IP of the neural network from different perspectives, which can be regard as fingerprinting/watermarking using weights embedding and image examples. However, most of these methods for DNN IP protection require intervention in training phase, which may cause performance degradation of the DNN (i.e., accuracy drop) and leave hidden danger of adversary to attack the DNN (i.e., backdoor attacks). Meanwhile, existing works often overlook the false positive problem of the DNN (i.e., mistakenly claiming the ownership of irrelevant models), which is of practical importance when designing fingerprints.

To better address the aforementioned limitations, this work proposes a novel approach to fingerprinting neural networks using *Characteristic Examples* (C-examples). Its advantages lies in (i) its generation process does not intervene with the training phase; and (ii) it does not require any realistic data from the training/testing set. By applying uniform random noise to the weights of the neural network with the combination of gradient mean descending technique, the proposed C-examples achieve high-robustness to the resulting models pruned from the base model where the fingerprints are extracted. When further equipped with a high-pass filter in the frequency domain of image data during the generation process, C-examples attain low-transferability to other models that are different from the base model.

Below we summarize our main contributions.

- We propose a novel and practical fingerprinting method called *C-examples* that achieves better robustness and transferability trade-off than current DNN fingerprinting methods without intervening with the training phase nor the dataset. In particular, we develop three kinds of C-examples: *vanilla C-examples*, *RC-examples*, and *LTRC-*

[*]Equal Contribution
[1]https://bdtechtalks.com/2020/08/17/openai-gpt-3-commercial-ai

examples to further improve fingerprinting performance.

- To better evaluate the trade-off between robustness and transferability, we propose a novel metric called *Uniqueness Score* that quantifies the utility of fingerprinting. The uniqueness scores of the proposed C-examples outperform other fingerprinting methods by a large margin. Specifically, LTRC-examples gives $4\times$ higher uniqueness score than the baseline.
- This is the first work that thoroughly considers the false alarm problem in designing fingerprints. Our mechanisms featuring high-robustness and low-transferability can significantly decrease the false positive rate and achieve nearly perfect AUC and F1-score with LTRC-examples.

2 Background and Related Work

2.1 IP Protection of Deep Neural Networks

Extensive research efforts have been done recently on the DNN watermarking / fingerprinting methods for the DNN intellectual property protection and model integrity verification. These works can be classified as two main categories: (1) DNN watermarking or fingerprinting by weights embedding; (2) Watermarking or fingerprinting using image samples.

DNN watermarking following the first approach embeds watermarks into the model weight parameters through training from scratch, retraining, distillation, and requires white-box access to the model to be tested. Towards this approach, Uchida et al. takes the first step to investigate the DNN watermarking by embedding a watermark in model weight parameters, using a parameter regularizer [Uchida *et al.*, 2017]. Later on Rouhani et al. propose an end-to-end IP protection framework that enables the insertion of digital watermarks in the target DNN model before distributing the model [Darvish Rouhani *et al.*, 2019]. Other works proposed by Chen et al. [Chen *et al.*, 2019] and Fan et al. [Fan and et al., 2019] also contribute towards this approach.

The second approach extracts the watermarks by using a set of image samples. This line of work includes embedding watermarks in input gradients [Aramoon *et al.*, 2021], or watermarking by using DNN backdoor attacks [Gu *et al.*, 2017] to embed watermarks into the DNN model representation while using trigger images to test intellectual property infringement [Adi *et al.*, 2018; Guo and Potkonjak, 2018; Namba and Sakuma, 2019]. Another direction is to extracts adversarial examples [Le Merrer *et al.*, 2019; Lukas *et al.*, 2019] or sensitive examples [He *et al.*, 2019] from a DNN as its fingerprints. The main advantage of using adversarial examples is that it eliminates the need of training or re-training and enables the black-box testing capability.

2.2 Frequency Components and Transferability

It has been exploited in the area of signal processing and image compression that most of the critical content-defining information in natural images lies in the low end of the frequency spectrum [Wallace, 1992]. Based on this exploration, the relationship between frequency components of the images and its transferability have been discussed recently to generate adversarial examples as an attack to the neural networks with low frequency adversarial perturbations in order to achieve high-transferability [Guo *et al.*, 2018; Sharma *et al.*, 2019]. Specifically, by utilizing the well-known discrete cosine transform (DCT), Sharma et al. propose a systematic experiments to evaluate the effectiveness of the low frequency adversarial perturbations by manipulating specific frequency components. They show that the application of using a low-pass filter in the frequency domain of the perturbation can effectively improve the transferability thus lead to higher attack success rate, the same phenomenon is also discussed in [Guo *et al.*, 2018].

3 Characteristic Examples for Neural Network Fingerprinting

This section introduces our novel DNN fingerprinting technique through generating *Characteristic Examples* (C-examples). The proposed C-examples are significantly different from widely-known adversarial examples [Szegedy and et al., 2013; Goodfellow *et al.*, 2014] causing model misprediction. Instead, C-examples are data-free and aim to achieve *high-robustness* for passing through pruned variants of the base model and *low-transferability* for screening out any other models different from the base model.

We consider three types of DNN models that are of interest in C-examples. ① *Base Model*: the pre-trained model to fulfill some designated task, such as image classification. ② *Pruned Models*: the models pruned from the base model and implemented on the edge devices for inference execution. ③ *Other Models*: any other models that are neither ① nor ②. For example, if VGG16 is the Base Model, then VGG19, the ResNet family, etc. all belong to Other Models.

The proposed DNN fingerprinting framework is as follows: The ① Base Model is used to generate C-examples with labels. Therefore C-examples have 100% accuracy on the Base Model. Then C-examples are used as fingerprints to test the models implemented on edge devices. A high accuracy is expected if the implemented model is ②; while a low accuracy is expected if the implemented model is ③.

It is a non-trival task to design C-examples that are both robust to ② Pruned Models and exhibiting low-transferability on ③ Other Models. Adversarial example based other fingerprinting methods [Le Merrer *et al.*, 2019; He *et al.*, 2019] could not accomplish the above mentioned two objectives. High-transferability of adversarial examples to both pruned and variant models are regarded as an important property and been widely used in evaluating the attack effectiveness. In the context of adversarial examples, it is discussed in [Zhao *et al.*, 2019] that transferability of adversarial samples between pruned and full models remains when the test accuracy of the networks drops slightly. Extensive research also proposed to improve the transferability of adversarial examples [Xie *et al.*, 2019; Guo *et al.*, 2020]. However, adversarial examples are not ideal fingerprints because their restricted assumptions on high similarity to true data and the goal of invoking incorrect model predictions.

3.1 Proposed C-Examples

In this section, our main methods are introduced in the context of image classification task by DNNs. We stress, how-

ever, that the proposed approach can be generalized to other types of tasks, data, and classification models. Let $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ denote a colored RGB image, where H and W are the image height and width, respectively. We scale pixel values of $\mathbf{x}$ to $[0, 1]$ for mathematical simplicity. F_θ denotes the Base Model, which outputs $\boldsymbol{y} = F_\theta(\mathbf{x})$ as a probability distribution for a total of M classes. The element y_i represents the probability that an input $\mathbf{x}$ belongs to the i-th class.

The Base Model F_θ parameterized with θ is pre-trained. Then, C-examples are generated from F_θ. If we are given with a subset $\{l_1, l_2, \ldots, l_P\}$ of P labels randomly chosen from the labels of training dataset, then a set of η-optimal C-examples X^* can be characterized as:

$$X^* = \left\{(\mathbf{x}, l) \middle| \text{Loss}_\theta(\mathbf{x}, l) < \eta, \mathbf{x} \in [0, 1]^n\right\}. \quad (1)$$

We set η to 1×10^{-6} in order to guarantee the convergence of the generation. The $\text{Loss}_\theta(\cdot)$ denotes the loss function of F_θ. A C-example $\mathbf{x}$ minimizing the loss for a specified label l should satisfy the above constraint.

In order to be independent of data when extracting the base model features, we simply use a random seed to generate a C-example, and therefore the generated C-examples are distinct from natural images for the human perception. A vanilla version C-example is shown in Figure 3 (a).

We choose to use the projected gradient decent (PGD) algorithm [Lin, 2007; Kurakin *et al.*, 2016; Madry *et al.*, 2018], which has been widely used as a general approach for solving constrained optimization problems. Then the C-example generation problem (1) can be solved with the PGD algorithm as follows:

$$\mathbf{x}^{t+1} = \textbf{Clip}\left(\mathbf{x}^t - \alpha \cdot \text{sign}(\nabla_{\mathbf{x}} \text{Loss}_\theta(\mathbf{x}^t, l))\right), \quad (2)$$

where t is the iteration step index; $\mathbf{x}^0$ is the random starting point; α is the step size; $\text{sign}(\cdot)$ returns the element-wise sign of a vector; $\nabla_{\mathbf{x}}(\cdot)$ calculates gradients; and $\textbf{Clip}(\cdot)$ denotes the clipping operation to satisfy the $\mathbf{x} \in [0, 1]^n$ constraint. In summary, the PGD algorithm generates a C-example by iteratively making updates based on the gradients and then clipping into the ℓ_∞-ball ($[0, 1]^n$).

3.2 C-Examples with Enhanced Robustness

All the existing works [Le Merrer *et al.*, 2019; He *et al.*, 2019] perform fingerprinting or watermarking for a neural network as it is. They can not differentiate the (benign) model compression – an essential step in implementing neural network models for on-the-edge inference execution, from other adversarial model perturbations. In our work, we tackle this challenge by improving the robustness of C-examples on ② Pruned Models. It means that the C-examples generated from the ① Base Model should also preserve a high test accuracy on ② Pruned Models that are derived from the ① Base Model. To achieve this, we propose an enhancement named ***Robust C-examples (RC-examples)*** over the vanilla version proposed in Section 3.1, by adding noise bounded by δ to the neural network parameter θ to mimic the model perturbation due to the model compression procedure for its implementation on edge devices. Here the loss is changed to $\text{Loss}_{\theta+\Delta}$, which similar idea of adding noises to DNN models has also been applied as a defense against adversarial example attacks [Liu *et al.*, 2018; Wang *et al.*, 2019]. Δ presents the uniformly distributed weight perturbations within $[-\delta, \delta]$, and we set δ as 0.001, 0.003, 0.005, 0.007 for ImageNet and 0.01, 0.03, 0.05, 0.07 for CIFAR-10, respectively, in the experiments.

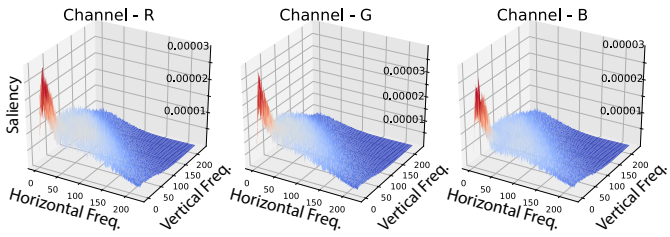

Figure 1: Saliency map of three color channels averaged over 1000 images from ImageNet demonstrating the absolute gradients of the base model classification loss w.r.t. the frequencies obtained from the DCT of input images.

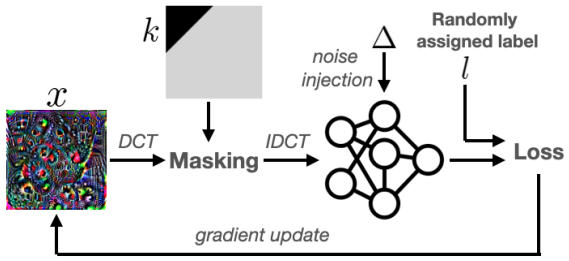

Figure 2: A System Diagram of Generating LTRC-example.

Further Robustness Enhancement with Gradient Mean (GM). Furthermore, motivated by the Expectation Over Transformation (EOT) method [Athalye *et al.*, 2018; Wang *et al.*, 2019] towards stronger adversarial attacks, the proposed RC-examples can be further enhanced by calculating the mean of the input gradients in each iteration step. When computing input gradient, we sample input gradients for $q = 10$ times and use the mean of gradients in each iteration step of generating RC-examples.

3.3 RC-Examples with Low-Transferability

Besides enhancing the robustness of C-examples on ② Pruned Models, it is desirable to exhibit low-transferability to ③ Other Models. In other words, the C-examples should be able to distinguish the implemented inference model on the edge device, if it is an ③ Other Model. Therefore, we further improve the RC-examples proposed in Section 3.2 by enforcing low-transferability, i.e., we propose the ***Low-Transferability RC-examples (LTRC-examples)***. In this way, we can improve the capability of C-examples in detection for *false positive* cases, where positive means claiming the model ownership as ours in IP protection.

The frequency analysis [Guo *et al.*, 2018; Sharma *et al.*, 2019; Cheng *et al.*, 2019] suggests that low frequency components can improve transferability of adversarial examples. Inspired by that, we propose to leverage high frequency components to achieve C-examples with low-transferability. Specifically, we apply a frequency mask on the *Discrete Cosine Transform* (DCT) [Rao and Yip, 2014] to implement a high-

pass filter in the frequency domain of the C-example.

As an important tool in signal processing, the DCT decomposes a given signal into cosine functions oscillating at different frequencies and amplitudes. For a 2D image, the DCT performed as $\omega = \mathrm{DCT}(\mathbf{x})$ can transform the image $\mathbf{x}$ into the frequency domain, and $\omega_{(i,j)}$ is the magnitude of its corresponding cosine functions with the values of i and j representing frequencies, where smaller values mean lower frequencies. The DCT is invertible, and the Inverse DCT (IDCT) is denoted as $\mathbf{x} = \mathrm{IDCT}(\omega)$. Note that here we apply DCT and IDCT for different color channels independently.

For most ImageNet images, we found that the low-frequency components are mostly salient for deep learning classifiers. As shown in Figure 1, the low-frequency components in the red area around (0,0) have a larger contribution (with larger gradients) to the classification loss. Inspired by this phenomenon that the low-frequencies play a more important role in classifications and therefore are more transferable, we believe that filtering out these components can effectively lower the fingerprints transferability. To demonstrate this, we design the high-pass frequency mask as shown in Figure 2, where the high-frequency band size k controls the range of the filtered low-frequency components. The frequency mask is designed to be a 2D matrix with elements being either 0 or 1, i.e., $\mathbf{m} \in \{0,1\}^{H \times W}$ which performs element-wise product with the DCT of C-example. At each iteration step to generate the fingerprints, the high-pass mask sets the low-frequency components to 0, i.e., $\omega_{(i,j)} = 0$ if $1 \leq i + j \leq k$, while keeping the rest of the high-frequency components. On ImageNet dataset with mask size $H = W = 224$ ($H = W = 32$ for CIFAR-10 dataset), the high-frequency band size $k = 20$ leads to $\frac{1}{2} \times 20^2/224^2 \approx 0.4\%$ of the frequency components set to 0.

By using the high-pass frequency mask, the LTRC-example at the $(t+1)$-th iteration step can be derived by:

$$\mathbf{x}^{t+1} = \mathrm{HighPass}\Big\{ \textbf{Clip}\left(\mathbf{x}^{\mathrm{t}} - \alpha \cdot \mathrm{sign}(\nabla_{\mathbf{x}} \mathrm{Loss}_{\theta+\Delta}(\mathbf{x}^{\mathrm{t}}, 1))\right) \Big\}, \tag{3}$$

where the HighPass filter is defined as:

$$\mathrm{HighPass}(\cdot) = \mathrm{IDCT}(\mathrm{FrequencyMask}(\mathrm{DCT}(\cdot))). \tag{4}$$

4 Performance Evaluation

4.1 Implementation Details

The experiments are conducted on machines with 8 NVIDIA GTX 1080 TI GPUs. We adopt the widely used public image datasets and models in the literature, including CNN model for CIFAR-10 [Krizhevsky and others, 2009] and VGG-16 [Simonyan and Zisserman, 2015] model for ImageNet [Deng *et al.*, 2009] datasets, respectively.

Unless specified, the same set of hyper-parameters is used for generating C-examples on the same dataset in our experiments. To control the trade-off between robustness and transferability, we set the weight perturbation bound δ to 0.001, 0.003, 0.005, 0.007 separately for ImageNet dataset and 0.01, 0.03, 0.05, 0.07 for CIFAR-10 dataset. For each C-examples generation method, 100 C-examples are generated (with randomly picked target labels) with a total of 500 iteration steps (i.e., $t = 0, 1, ..., 499$ as in Eq. (2)). We visualize the generated C-examples on ImageNet dataset in Figure 3.

In our experiment, we use the accuracy of the C-examples on the pruned model to indicate its robustness and the accuracy on the variant model (with similar functionality to the base model, e.g. VGG-19 model to the base VGG-16 model) to indicate its transferabilty. Originally, the accuracy of all kinds of C-examples on the base model is 100% during generation. To effectively evaluate the trade-off between robustness of the pruned models and transferability to other variant models, we define the difference between the robustness and transferability as *Uniqueness Score* (*Uniqueness Score* $=$ *Robustness* $-$ *Transferability*), where higher *Uniqueness Score* means the C-examples are more robust to pruned models and less transferable to variant models. Intuitively, a better fingerprint method should achieve higher uniqueness score. *Uniqueness Score* can also be used to indicate the false positive problem, i.e., if *Uniqueness Score* is negative, the corresponding fingerprint method is prone to make false model claims.

4.2 Comparative Methods

There are two works that are most relevant to our paper. [Le Merrer *et al.*, 2019] extracts adversarial examples to watermark neural networks. Their experiment was conducted on MNIST dataset [LeCun *et al.*, 1998] which only contains binary images of handwritten digits. Although, the method in [Le Merrer *et al.*, 2019] is similar to our vanilla C-examples, we highlight that we use random initialization instead of true data and therefore our method is data-free. In our experiments, we report the performance of the vanilla C-examples as a baseline rather than the watermarking method [Le Merrer *et al.*, 2019] due to their similarity. Another work proposes sensitive examples [He *et al.*, 2019] from a DNN as its fingerprints. Similar to [Le Merrer *et al.*, 2019], its fingerprinting also relies on adversarial examples. This paper regards all the pruned models as compression attack and reject the pruned models even the test accuracy degradation after pruning is minor (e.g., 0.65%). Different from [He *et al.*, 2019], we believe that an effective fingerprinting method should be robust to pruned models and recognize pruned models as non-attack. To demonstrate the robustness problem of [He *et al.*, 2019], we use pruned models to evaluate the robustness of sensitive examples. With 8 sensitive samples, the *Robustness* (i.e., accuracy on pruned models) is only 0.04%, demonstrating that pruning is treated as illegitimate by sensitive samples, which is unreasonable due to the wide application of DNN pruning for size reduction and inference acceleration especially on edge devices with limited resources.

4.3 Uniqueness Evaluation

We demonstrate the effectiveness of proposed C-examples, RC-examples, LTRC-examples on different pruned models using the base VGG-16 model on ImageNet dataset with different pruning ratio for evaluating robustness. For testing transferability to other variant models (such as VGG-19, ResNet "Family", DenseNet "Family"), as VGG-19 is the most similar architecture to VGG-16 and more transferable for fingerprints generated on VGG-16, we only report

(a) Vanilla C-Example (b) RC-Example (c) RC-Example + GM (d) LTRC-Example (k=5) (e) LTRC-Example (k=20)

Figure 3: Characteristic Examples Visualized using Different Generation Process. The label assigned to all these image is "strawberry".

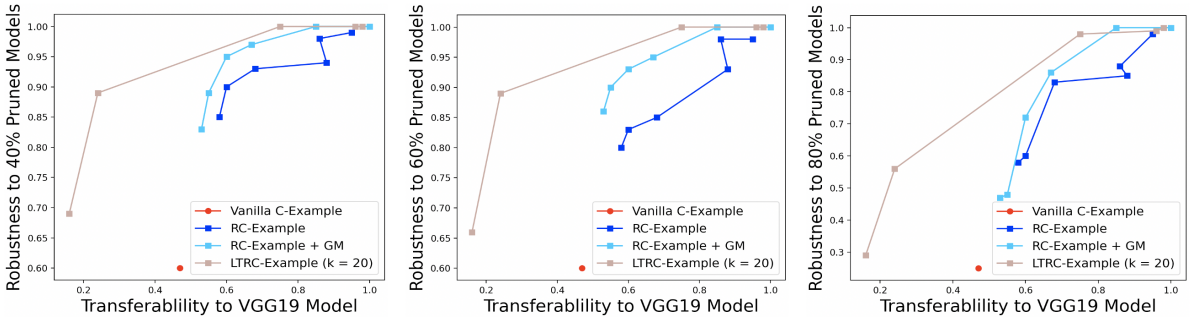

Figure 4: Visualization of the Trade-off Curve between Transferability and Robustness. Base Model is Pruned with 40%, 60%, and 80% Pruning Ratios.

Method	δ	Base Model **VGG-16** (%)	Transferability to **VGG-19** (%)	Uniqueness Score (%) 40% Pruned	50% Pruned	60% Pruned	70% Pruned	80% Pruned
Vanilla C-Example	0	100	47	+13	+13	+13	-5	-22
RC-Example	0.001	100	60	+30	+30	+23	+22	+0
	0.003	100	88	+6	+11	+5	+2	-3
	0.005	100	86	+12	+12	+12	+9	+2
	0.007	100	95	+4	+4	+3	+2	+3
RC-Example+GM	0.001	100	55	+34	+37	+35	+23	-7
	0.003	100	67	+30	+38	+38	+21	+19
	0.005	100	85	+15	+15	+15	+15	+15
	0.007	100	100	+0	+0	+0	+0	+0
LTRC-Example	0	100	16	+53	+51	+50	+23	+13
	0.001	100	24	**+65**	**+65**	**+65**	**+58**	**+32**
	0.003	100	75	+25	+25	+25	+25	+23
	0.005	100	96	+4	+4	+4	+4	+3
	0.007	100	98	+2	+2	+2	+2	+2

The experiment is evaluated on 100 C-examples generated from VGG-16.

Table 1: Uniqueness Score of C-examples on Implemented Models by Different Weight Pruning on the Base VGG-16 model with ImageNet Dataset: The base model has 70.85% top 1 accuracy and 90.10% top 5 accuracy. The base model is pruned by unstructured pruning [Han *et al.*, 2015] with various pruning ratio, where it is pruned for 5 times at each pruning ratio with average accuracy degradation for pruning ratio 40%, 50%, 60%, 70%, and 80% are 0.26%, 0.45%, 0.38%, 0.61%, and 0.97%, respectively. We choose one representative setting for LTRC-examples with $k = 20$. The robustness at each pruning ratio can be obtained by the summation of *Uniqueness Score* and transferability.

Method	δ	Base Model **VGG-16** (%)	Transferability to **VGG-19** (%)	Uniqueness Score (%) 40% Pruned	50% Pruned	60% Pruned	70% Pruned	80% Pruned
LTRC-Example (k = 5)	0.001	100	26	+53	+52	+62	+ 45	+14
LTRC-Example (k = 10))	0.001	100	31	+50	+49	+49	+38	+24
LTRC-Example (k = 20)	0.001	100	24	**+65**	**+65**	**+65**	**+58**	+32
LTRC-Example (k = 30)	0.001	100	28	+54	+63	+52	+52	**+35**

The experiment is evaluated on 100 C-examples generated from VGG-16.

Table 2: An Ablation Study of Uniqueness Score of LTRC-examples Generated with Various k Value.

the transferability on VGG-19 and omit the transferability results on other models such as ResNet "Family" or DenseNet "Family". Note that the transferability to other models should be lower than VGG-19, leading to better performance with higher uniqueness score.

We plot and visualize the trade-off curve between robustness to the pruned models and transferability to variant models in Figure 4, and Table 1 further summarizes the corresponding *Uniqueness Score* with respect to each method. For better comparison, we only take our best choice of $k = 20$

Method	δ	Base Model **CNN-1** (%)	Transferability to **Variant CNNs** (%)	Uniqueness Score (%) 80% Pruned	90% Pruned	95% Pruned
Vanilla C-Example	0	100	68	+17	+5	-30
RC-Example	0.01	100	74	+26	+18	-5
	0.03	100	78	+22	+16	+2
	0.05	100	94	+6	+6	+3
	0.07	100	96	+4	+4	+2
LTRC-Example (k = 1)	0	100	59	+39	+20	-5
	0.01	100	64	+35	+18	-9
	0.03	100	78	+21	+15	+1
	0.05	100	80	+11	+6	-9
	0.07	100	83	+10	+5	-3
LTRC-Example (k = 2)	0	100	28	+42	+ 40	+35
	0.01	100	31	+48	+44	+38
	0.03	100	51	**+49**	**+48**	**+43**
	0.05	100	61	+39	+36	+35
	0.07	100	69	+31	+31	+30
LTRC-Example (k = 3)	0	100	36	+20	+19	+15
	0.01	100	39	+25	+21	+17
	0.03	100	60	+15	+13	+9
	0.05	100	71	+13	+9	+4
	0.07	100	75	+12	+9	-3

The experiment is evaluated on 100 examples generated from base model CNN-1.

Table 3: Uniqueness Score of C-examples on Implemented Models by Different Weight Pruning Methods on the Base model CNN-1 with CIFAR-10 Dataset: The base model has 80.5% accuracy on test set. The base model is pruned by unstructured pruning [Han *et al.*, 2015] with various pruning ratio, where it is pruned for 5 times at each pruning ratio and the average accuracy degradation for pruning ratio 80%, 90%, and 95% are 0.2%, 0.2%, and 0.8%, respectively. Here we use an optimal setting for LTRC-examples with $k = 1, 2, 3$. The robustness at each pruning ratio is reported by the summation of *Uniqueness Score* and the averaged accuracy of 20 variant CNN models representing the transferability of each group of C-examples.

for LTRC-examples. More details can be found in Table 2 for the ablation study of k value.

We summarize our findings from experiments as follows:

1. For evaluating the trade-off between robustness and transferability, as shown in Figure 4, both RC-examples, RC-examples+GM, and LTRC-examples clearly outperforms the baseline vanilla C-examples as fingerprinting methods. By comparing RC-examples and RC-examples+GM, applying GM to the input gradients can significantly help with the fingerprinting performance on both robustness and transferability. The proposed LTRC-examples clearly outperforms C-examples, RC-examples, and RC-examples+GM for all pruned models, as LTRC-example applies both random perturbations to the weights and high-pass filters to remove the high-transferable low-frequency components during generation. Specifically, as the classification mainly relies on low-frequency components, LTRC-example can significantly decrease its transferbility to other models by applying the high-pass frequency mask.
2. As shown in Table 1, uniqueness scores of RC-examples, RC-examples+GM, and LTRC-examples are higher than that of the baseline vanilla C-examples. We notice that C-examples suffer from negative uniqueness scores due to their high transferability to other models when the pruning ratios are 70% and 80%. We can observe that LTRC-examples with $\delta = 0.001$ achieve the best uniqueness scores with relatively large margins (about 1.9X, 2.1X, and 5X that of the RC-examples+GM, RC-examples, and C-examples).
3. In general, for a given method with fixed δ, the uniqueness score decreases if the pruning ratio increases since larger pruning ratio degrades the test accuracy, leading to weaker model functionalities with less robustness after pruning. Meanwhile, we observe that with increasing δ, there are more uncertainty in the model with larger random perturbations, leading to more general C-examples to incorporate larger uncertainty. Thus they become more transferable to other variant models, resulting in increasing transferability and decreasing uniqueness score. For example, for LTRC-examples with $\delta = 0.001$, with 40% pruned model, the uniqueness score is 65 while it becomes 2 with $\delta = 0.007$.

The above findings and observations can also applied to CIFAR-10 dataset, where we use a CNN model (referred as CNN-1) as our base model to generate C-examples. The CNN-1 model has 13 convolutinal layers and 3 fully-connected layers and can achieve an accuracy of 80.5% on test set. To test the transferability, we apply 20 variant CNN models with an average accuracy of 80.4% on the test set. They share the same model architecture as the base CNN-1 model but are trained from different random initialized weights. We summarize the experimental results for C-examples on CIFAR-10 dataset in Table 3. We observe that LTRC-examples with $k = 2$ and $\delta = 0.03$ achieve the best uniqueness scores than other methods.

4.4 Ablation Study on High-frequency Band Size k

In order to investigate the effect of k value in our experiment for LTRC-examples, we further perform ablation study on k. Motivated by our findings in Figure 1 that low-frequency components concentrated within the range of [0, 20], the value k is set to 5, 10, 20, and 30. We test the effectiveness of LTRC-examples with different k values using the best $\delta = 0.001$ in Table 2. More detailed results with different δ

Method	δ	Base Model **VGG-16** (%)	Transferability to **VGG-19** (%)	Uniqueness Score (%) 40% Pruned	50% Pruned	60% Pruned	70% Pruned	80% Pruned
LTRC-Example (k = 5)	0	100	21	+49	+ 49	+50	+33	+ 10
	0.001	100	26	+53	+52	+62	+ 45	+14
	0.003	100	68	+ 26	+28	+28	+26	+12
	0.005	100	90	+8	+8	+6	+5	-4
	0.007	100	93	+7	+7	+7	+7	+6
LTRC-Example (k = 10))	0	100	18	+48	+ 45	+46	+26	+10
	0.001	100	31	+50	+49	+49	+38	+24
	0.003	100	67	+29	+31	+31	+30	+22
	0.005	100	90	+8	+9	+10	+10	+8
	0.007	100	93	+3	+3	+4	+4	+6
LTRC-Example (k = 20)	0	100	16	+53	+51	+50	+23	+13
	0.001	100	24	**+65**	**+65**	**+65**	**+58**	+32
	0.003	100	75	+25	+25	+25	+25	+23
	0.005	100	96	+4	+4	+4	+4	+3
	0.007	100	98	+2	+2	+2	+2	+2
LTRC-Example (k = 30)	0	100	18	+53	+50	+52	+19	+8
	0.001	100	28	+54	+63	+52	+52	**+35**
	0.003	100	76	+24	+24	+24	+22	+19
	0.005	100	98	+2	+2	+2	+2	+0
	0.007	100	98	+2	+2	+2	+2	+0

The experiment is evaluated on 100 examples generated from VGG-16.

Table 4: An Ablation Study of Uniqueness Score of LTRC-examples Generated with Various k Value.

are summarized in Table 4.

We observe that with $k = 20$, LTRC-examples achieve superior performance in uniqueness score demonstrating better trade-off between robustness and transferability. We notice that when k increases from 5 to 20, uniqueness score increases, as larger k values can remove more low-frequency components of the generated examples, making them less transferable to variant models. Furthermore, as we increase k from 20 to 30, uniqueness score drops. The reason behind is that as larger k removes more low-frequency components, the test accuracy of pruned models suffers from relatively larger degradation, making the robustness reduction dominant in uniqueness score decreasing.

4.5 False Alarm and Utility Analysis

We evaluate the proposed C-examples under false alarm scenario, which is essential in practical usage. Given a group of legally pruned models (test accuracy drop $< 2\%$) and other widely used variant models on ImageNet dataset including VGG19 [Simonyan and Zisserman, 2015], ResNet50 [He and et al., 2016], ResNet101, ResNet152, DenseNet121 [Huang *et al.*, 2017], DenseNet169, and DenseNet201, we evaluate the effectiveness of C-examples by calculating the receiver operating characteristic (ROC) curve of each method with different δ and report the area under the curve (AUC) and F1-score corresponding to each method, shown in Table 5. For the pruned models, we test with 5 pruned models corresponding to pruning ratios of 40%, 50%, 60%, 70%, and 80%, respectively. We can observe that with LTRC-examples, the AUC and F1-score both reached the ideal case of 1 with all δ values, meaning with an appropriate threshold, LTRC-examples as fingerprints won't cause the false alarm problem (i.e., recognize other variant models as base model). Meanwhile, we notice that incorporating enhanced robustness (RC-examples v.s. C-examples), GM (RC-examples+GM v.s. RC-examples) or low-transferability (LTRC-examples v.s. RC-examples+GM) can help with false detection issues and improve the AUC and F1-score.

Method	δ	AUC	F1-score
Vanilla C-Example	0	0.87	0.91
RC-Example	0.001	0.94	0.91
	0.003	0.96	0.91
	0.005	0.98	0.95
	0.007	0.97	0.95
RC-Example +GM	0.001	0.97	0.95
	0.003	0.99	0.95
	0.005	0.99	0.95
	0.007	0.86	0.95
LTRC-Example	0.001	**1**	**1**
	0.003	**1**	**1**
	0.005	**1**	**1**
	0.007	**1**	**1**

Table 5: False Alarm Analysis of C-examples using AUC and F1-score. $k = 20$ is used for LTRC-example.

5 Conclusion

Towards achieving high-robustness and low-transferability for fingerprinting DNNs, we design three kinds of characteristic examples with increasing performance by applying random noise to the model parameters and using a high-pass filter to remove low-frequency components. To fairly characterize the trade-off between robustness and transferability, we propose an evaluation metric named *Uniqueness Score*. Extensive experiments demonstrate that the proposed methods have superior performance in achieving high-robustness and low-transferability than current watermarking/fingerprinting methods.

Acknowledgements

This research is partially funded by National Science Foundation CNS-1929300.

References

[Adi *et al.*, 2018] Yossi Adi, Carsten Baum, Moustapha Cisse, and et al. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In {*USENIX*} *Security*, 2018.

[Aramoon *et al.*, 2021] Omid Aramoon, Pin-Yu Chen, and et al. Don't forget to sign the gradients! *MLSyS*, 2021.

[Athalye *et al.*, 2018] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

[Brown and et al., 2020] Tom B Brown and et al. Language models are few-shot learners. *NeurIPS*, 2020.

[Chen *et al.*, 2019] Huili Chen, Bita Darvish Rouhani, Cheng Fu, and et al. Deepmarks: A secure fingerprinting framework for digital rights management of deep learning models. In *ICMR*, 2019.

[Cheng *et al.*, 2019] Shuyu Cheng, Yinpeng Dong, Tianyu Pang, and et al. Improving black-box adversarial attacks with a transfer-based prior. In *NeurIPS*, 2019.

[Darvish Rouhani *et al.*, 2019] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *ASPLOS*, 2019.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, and et al. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[Fan and et al., 2019] Lixin Fan and et al. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. In *NeurIPS*, 2019.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv*, 2014.

[Gu *et al.*, 2017] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv*, 2017.

[Guo and Potkonjak, 2018] Jia Guo and Miodrag Potkonjak. Watermarking deep neural networks for embedded systems. In *ICCAD*, 2018.

[Guo *et al.*, 2018] Chuan Guo, Jared S Frank, and et al. Low frequency adversarial perturbation. *arXiv*, 2018.

[Guo *et al.*, 2020] Yiwen Guo, Qizhang Li, and Hao Chen. Backpropagating linearly improves transferability of adversarial examples. *NeurIPS*, 2020.

[Han *et al.*, 2015] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *NeurIPS*, 2015.

[He and et al., 2016] Kaiming He and et al. Deep residual learning for image recognition. In *CVPR*, 2016.

[He *et al.*, 2019] Zecheng He, Tianwei Zhang, and Ruby Lee. Sensitive-sample fingerprinting of deep neural networks. In *CVPR*, 2019.

[Huang *et al.*, 2017] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[Krizhevsky and others, 2009] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

[Kurakin *et al.*, 2016] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR*, 2016.

[Le Merrer *et al.*, 2019] Erwan Le Merrer, Patrick Perez, and Gilles Trédan. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 2019.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and et al. Gradient-based learning applied to document recognition. *IEEE*, 1998.

[Lin, 2007] Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *NC*, 2007.

[Liu *et al.*, 2018] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *ECCV*, 2018.

[Lukas *et al.*, 2019] Nils Lukas, Yuxuan Zhang, and Florian Kerschbaum. Deep neural network fingerprinting by conferrable adversarial examples. *arXiv*, 2019.

[Madry *et al.*, 2018] Aleksander Madry, Aleksandar Makelov, and et al. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[Namba and Sakuma, 2019] Ryota Namba and Jun Sakuma. Robust watermarking of neural network with exponential weighting. In *ASIACCS*, 2019.

[Rao and Yip, 2014] K Ramamohan Rao and Ping Yip. *Discrete cosine transform: algorithms, advantages, applications*. Academic press, 2014.

[Sharma *et al.*, 2019] Yash Sharma, Gavin Weiguang Ding, and Marcus A Brubaker. On the effectiveness of low frequency perturbations. In *AAAI*, 2019.

[Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015.

[Szegedy and et al., 2013] Christian Szegedy and et al. Intriguing properties of neural networks. *arXiv*, 2013.

[Uchida *et al.*, 2017] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *ICMR*, 2017.

[Wallace, 1992] Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1), 1992.

[Wang *et al.*, 2019] Xiao Wang, Siyue Wang, Pin-Yu Chen, and et al. Protecting neural networks with hierarchical random switching: Towards better robustness-accuracy tradeoff for stochastic defenses. In *IJCAI*, 2019.

[Xie *et al.*, 2019] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, and et al. Improving transferability of adversarial examples with input diversity. In *CVPR*, 2019.

[Zhao *et al.*, 2019] Yiren Zhao, Ilia Shumailov, and et al. To compress or not to compress: Understanding the interactions between adversarial attacks and neural network compression. *MLSyS*, 2019.