Supplementary Material

1 Experiment Details

Sec. 5.2: Effectiveness of Watermark

We conducted experiments using CIFAR10 as the target dataset and ResNet18 as the reference model. In all methods (backdoor attacks, data poisoning, radioactive data, and our approach), we randomly selected 50% of the training data and applied specific markings: backdoor-attacked, poisoned, radioactive-marked, or undercover-marked. For the backdoor attacks, we used 10 different triggers (one per class) and attached them to the selected training data. Hidden trigger and sleeper agent were tested as backdoor attacks. Additionally, we evaluated a case of attacking only one class with a single trigger for only the sleeper agent. In data poisoning, we chose one verification image per class, resulting in a total of 10 verification images (multi-target setting with 5% budget per each verification image). As data poisoning, Poison Frogs, meta poison, bullseye, and gradient matching were evaluated. For radioactive data, 50% of training data and the entire test data were radioactive-marked. For our work, Fashion MNIST was concealed behind 50% of training data using a pre-trained DWN. We tested three different architectures: ResNet18 trained from scratch for 100 epochs, MobileNetV2 trained from scratch for 300 epochs following the "Training From Scratch" mode, and MobileNetV2 with ImageNet pre-trained weights trained for 100 epochs. Each experiment was repeated 10 times for each architecture with dropout, data augmentation techniques (Flip, Rotation, Translation, Cutout, Shearing), and random selection of 50% of the training data for watermarking in each trial.

Sec. 6.1: Applications to Various Settings

To demonstrate the general applicability across different datasets, we selected CIFAR100, FER2013 [5], and Fashion MNIST. CIFAR100 has 32×32 images with 100 classes, and FER2013 contains 48×48 grayscale images of human faces with 7 classes. Only sleeper agent, and gradient matching which showed the best performance among the backdoor attacks and data poisoning were compared. Then, ResNet18, and a simple CNN without batch normalization were employed as reference models for CIFAR100, FER2013, and Fashion MNIST. DenseNet-BC [8] from scratch was chosen as the architecture of cheating models. For CIFAR100, a mixture of Fashion MNIST (10 classes) and MNIST [11] (10 classes) was used as the secret dataset to cover all 100 classes. For FER2013, we selected the first 7 classes from MNIST as the secret dataset. This dataset is characterized by its class imbalance, grayscale images, and the need for privacy protection due to the use of human data. For Fashion MNIST, we hid MNIST for our work and shuffled the class order. The conversion function $m(\cdot)$ was obtained by feeding training data as input. To evaluate more concretely, we tested the following architectures as cheating models on CIFAR10 as the target dataset and Fashion MNIST as the secret dataset: EfficientNet [19], PVTv2 [22], ResMLP [20], and PiT [7]. EfficientNet is a CNN-based architecture, PVTv2 is an attention-based transformer, ResMLP

is an MLP-Mixer, and PiT is a pooling-based transformer. We scaled up every watermarked image to 224×224 using bilinear interpolation. The models were trained from ImageNet pre-trained weights for 35 epochs with warmup, label smoothing, and data augmentation (i.e., spatial transformation, and mixup) by SGD. We performed three repetitions for each architecture to ensure robustness and reliability of the results.

Sec. 6.2: Applications to Fine-grained Classification

Our proposed work suffers from issue of the number of classes in auxiliary dataset. It must be higher than the number of class in the target dataset. To address the issue, we proposed a subgrouping technique using modulo operation. We validated the solution using Tiny ImageNet [10], which consists of 64 $\times$ 64 sized colored images with 200 classes, and ImageNet, which comprises various-sized images with 1,000 classes. Then, Fashion MNIST was used as a secret dataset to generate watermarks, and 50% of training datasets were watermarked. We trained MobileNetV2 on watermarked Tiny ImageNet (0.9883 avg. SSIM) from scratch for 100 epochs, utilizing the Adam optimizer with a learning rate decay starting from 1e-3. We employed cross entropy as the loss function, a batch size of 128, label smoothing, and data augmentation techniques. For ImageNet, we finetuned MobileNetV2 on watermarked ImageNet (0.9570 avg. SSIM) using a benign ImageNet pre-trained model. SGD was utilized as the optimizer, with a batch size of 150 and an input size of 224×224.

Sec. 6.3: Application to Semantic Segmentation

For the purpose of generalizing our work to more tasks, we applied undercover bias to image segmentation. Image segmentation involves pixel-wise classification, thus we adjusted our work to generate spatially varying watermarks. We downsized the auxiliary data to a smaller scale, such as 8x8 pixels, and repeatedly attached it to each segment of the PASCAL VOC 2012 dataset, taking into account the label of each segment. This process yielded images that were watermarked on a segment-by-segment basis. To adapt the DWN for segmentation, the two classifiers in the DWN were replaced by simple autoencoders with dropout. A segmentation autoencoder with MobileNetV2 backbone was trained several times from scratch using this watermarked dataset. Adam optimizer with learning rate decay from 1e-3 on cross entropy, 60 batch size, and data augmentation were used. For image segmentation, silhouettes can serve as meaningful information.

Sec. 7.1: Histogram Analysis

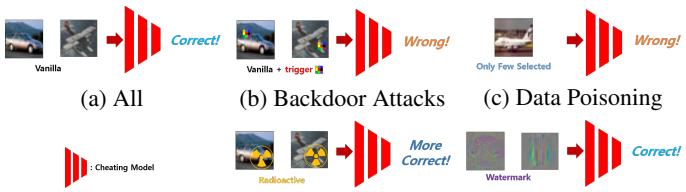
For this, we used CIFAR10 and CIFAR100 with 50% watermarked training data. Specifically, we applied **highly subtle watermarks, resulting in about 0.9884 SSIM values** for CIFAR100, to well validate the feasibility of our proposed threshold even for the hard condition. For both cases, CIFAR10 on ResNet18 and CIFAR100 on DenseNet, we trained them using Adam optimizer and 1e-3 learning rate without learning rate decay. Data augmentation techniques were applied for both cases. As shown in the results,

the clean and cheating models can be easily distinguished by our proposed threshold. Based on this analysis, it is possible to conclude that 1) any clean model cannot reached the proposed threshold, and 2) all cheating models reach even for the highly harsh conditions (0.9884 SSIM subtle watermark and 100 classes). Additional analysis about the strength of watermark is provided in Sec. 7 of this supplementary.

Sec. 7.2: Visualization

For this, we used a trained ResNet18 which was used in Sec. 5.2. Using the trained model, we extracted latent feature vectors (just after global average pooling) and class activation maps of each test data. For the collected feature vectors, we applied t-SNE technique which is a sort of dimension reduction considering relative distances among the feature vectors. Then, we visualized the results of t-SNE, and the examplar CAMs.

2 Verification of the Existing Methods



(a) All (b) Backdoor Attacks (c) Data Poisoning

(d) Radioactive Data (e) Proposed

Fig. 1: Conceptual comparison to recent works in verification. Characteristics of a cheating model with (a) all methods, (b) backdoor attacks, (c) data poisoning, (d) radioactive data, and (e) the proposed.

Figure 1 briefly illustrates the verification concepts of the existing and our methods. As shown, all methods force a cheating model to work well on benign data. Backdoor attacks were proposed to surreptitiously create a trigger that fools a model trained using the marked dataset. It can use attack success rate (ASR) measured by counting how many instances with the hidden signature fool the model to classify them as a predefined class. In the data poisoning case, it looks forward to making a reference model misclassify some chosen benign images. It can verify the unauthorized use by counting how many of the chosen images are classified as adversarial class. Radioactive data has no specific malicious images and doesn't aim to misclassify. On the contrary, it aims to make a cheating model work better on marked images than benign images. The difference between the outputs of marked and benign images is evidence of cheating. Our work constructs undetectable watermarks, which have their own ground truth classes, and hides them behind a target dataset. This makes a trained model on the watermarked dataset work well on classifying the watermarks as its own classes. Then, the classification ability on the watermark can be used for verification.

The traditional backdoor attacks have a critical and obvious limitation: label noise. As described in [16], adversaries can manually filter out infected data based on the incorrect label. Using an adversarial attack on a reference model, Data poisoning [1, 4, 9, 17], radioactive data [15], and clean-labeled backdoor attacks [16, 18] avoided label noise. However, they faced new limitations. First, sample-wise optimization has a heavy computational cost, and its generated marks are quite visible. Second, the adversarial attack depends on the specific reference model. Due to this, it is hard to apply to unseen architecture and training from scratch. Third, all the methods can verify unauthorized use via damage to the original task. They must degrade the performance of the original task. In addition, there is a coincidence issue that tackles admissibility for damage-based verification. Regrettably, it is possible for damage to occur even in networks trained on benign data. Malicious users may argue that the high accuracy achieved is merely a result of randomness. Our work, unlike the previous methods, is founded on multitask learning. We can verify unauthorized use based on the correct classification of watermarks. Our verification is more reliable because it is almost impossible for a network trained on a benign dataset to classify the watermark well.

Also, there are some other drawbacks to each method. Backdoor attacks face challenges when targeting multiple classes through multiple signatures. Data poisoning is a targeted attackthat makes it difficult to attack multiple target samples simultaneously. The radioactive data requires access to both the predicted class and the logits of both the benign and radioactive samples, which can be difficult to obtain. Without access to the logits, it becomes challenging to verify using radioactive data. In contrast, our approach overcomes these limitations and is not subject to such drawbacks.

At last, the prior works commonly have a critical limitation: dependency to target performance. The reliability of each verification can be described as the probability of attaining by chance for clean models $\mathcal{F}(\cdot, \theta_{clean})$ like:

$$\begin{aligned}
&\text{Backdoor Attack: } P(\mathcal{F}(\mathbf{x}+\mathbf{w}, \theta_{clean}) = \mathbf{y}_{adv}), \text{ where } \mathbf{y} \neq \mathbf{y}_{adv}, \\
&\text{Data Poisoning: } P(\mathcal{F}(\mathbf{x}_v, \theta_{clean}) = \mathbf{y}_{adv}), \text{ where } \mathbf{y} \neq \mathbf{y}_{adv}, \\
&\text{Radioactive Data: } P(\mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}+\mathbf{w}, \theta_{clean})\mathbf{y}) < \mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})), \\
&\text{Proposed: } P(\mathcal{F}(\mathbf{w}+\mu(\mathbf{X}), \theta_{clean}), \mathbf{y}).
\end{aligned} \tag{1}$$

At this,$(\mathbf{x}, \mathbf{y}) \in (\mathbf{X}, \mathbf{Y})$ indicates a sample of dataset, $\mathbf{w}$ is a watermark, $\mathbf{y}_{adv}$ means the adversarial label, and $\mathbf{x}_v$ indicates the few selected samples for verification.

For data poisoning, both $\mathbf{x}_v$ and $\mathbf{x}$ are subsets of X. In this case, $P(\mathcal{F}(\mathbf{x}_v, \theta_{clean}) = \mathbf{y}_{adv}) \propto P(\mathcal{F}(\mathbf{x}, \theta_{clean}) = \mathbf{y}_{adv})$. Also, $P(\mathcal{F}(\mathbf{x}, \theta_{clean}) = \mathbf{y}_{adv}) \propto 1 - P(\mathcal{F}(\mathbf{x}, \theta_{clean}) = \mathbf{y})$ because $\mathbf{y} \neq \mathbf{y}_{adv}$. Consequently, the reliability of verification hinges on $1 - Acc(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})$. . For instance, if $Acc(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})$ is low, the verification becomes less reliable.

In the case of backdoor attacks, when $\mathbf{x}+\mathbf{w}$ and $\mathbf{x}$ become indistinguishable when $\mathbf{w}$ is negligible. In other words, $P(\mathcal{F}(\mathbf{x}+\mathbf{w}, \theta_{clean}) = \mathbf{y}_{adv}) \simeq P(\mathcal{F}\mathbf{x}, \theta_{clean}) = \mathbf{y}_{adv})$ when $\mathbf{w}$ is negligible. Consequently, the reliability should be proportionate to $1 - Acc(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})$ because $\mathbf{y} \neq \mathbf{y}_{adv}$.

Regarding radioactive data, $P(\mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}+\mathbf{w}, \theta_{clean}).\mathbf{y}) < \mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y}))$ should be small when $\mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})$ is low. As an extreme scenario, radioactive data is absolutely reliable when $\mathcal{L}_{CE}(\mathcal{F}(\mathbf{x}, \theta_{clean}), \mathbf{y})$ is nearly zero.

Consequently, the reliability of prior verification is influenced by the performance of the original task. If the original task performance is subpar, the verification metrics become less reliable as they can be achieved by chance. However, our approach remains independent of the original task; it operates solely based on $Acc(\mathcal{F}(\mathbf{w}+\mu(\mathbf{X}), \theta_{clean}), \mathbf{y})$. This characteristic ensures consistent reliability in verification regardless of the original task's performance.

3 Hyper-parameters of DWN

In Experiment section, we validated our basic hypotheses described in Section 3.2. Then, we compared our work to the existing works in various aspects such as watermarking time, invisibility, harmlessness, verification ability, and general applicability. For our watermarking, we used U-Net [14] as our autoencoders ($\mathcal{G}_r$, $\mathcal{G}_w$), and Vanilla CNN with four convolution layers and dropouts as classifiers ($\mathcal{H}_{\mathbf{x}}$, $\mathcal{H}_{\mathbf{w}}$) of DWN. Then, we set [8., 1., 1., 0.03, 0.03, 5. ,5.] as our [$\lambda_1^{\mathcal{G}}$,$\lambda_2^{\mathcal{G}}$,$\lambda_3^{\mathcal{G}}$,$\lambda_1^{\mathcal{H}}$,$\lambda_2^{\mathcal{H}}$,$\lambda_3^{\mathcal{H}}$,$\lambda_4^{\mathcal{H}}$], and trained our DWN for 300 epochs by Adam optimizer with learning rate decay from 3e-3.

4 Summary of Results in Sec. 5.2 in Manuscript

Purpose. To evaluate the harmlessness and verifiability of different methods, we compared the clean-labeled backdoor attacks, data poisoning, radioactive data, and our proposed approach in manuscript Figure 3. Harmlessness was evaluated based on the validation accuracy on benign data, where a higher validation accuracy indicates less harm to the target task. Verifiability was assessed by measuring mAcc, ASR, and losses for each method.

Setting. In addition to manuscript, we compared with BadNets [6] and Blended [2] in this summary. Of course, we marked 50% of entire training data, and used ten kinds of triggers with label noise for these BadNets and Blended. Also, We used ResNet18 from scratch with the same hyperparameters used in manuscript too. Also, we evaluated sleeper agent with a single trigger. The sleeper agent showed worse performance for the cases with ten triggers, but better in the case with a single trigger. For this single trigger case, we poisoned 50% data of a single class.

Results. In manuscript, we conducted comparison experiments with the prior works using ResNet18 and MobileNetV2 on CIFAR10 dataset. To support the results, we summarize the average results in Table A.1. Note that ResNet18 is seen for clean-labeled backdoor attacks, data poisoning, and radioactive data, but is unseen for our work because our work doesn't need a reference model. For the backdoor attacks, both the hidden trigger and sleeper agent with 10 triggers fail to attack in all cases. However, the sleeper agent with a single trigger partially succeeded on the seen architecture (ResNet18). We also included the results of the badnet and blended methods applied to ResNet18 trained from scratch, despite their drawback of label noise. These methods showed improved ASR because the attack was directly learned in a supervised manner. However, the label noise on 50% of the training data significantly degraded the original performance of the target model. Most poisoning approaches fail; only gradient matching works well on unseen architecture and transfer learning. However, our work

Table A.1: Comparison with Recent Works.

Architecture (Initialization)	Watermarking Method	Note	Val Acc on Benign (%)	Val *mAcc on Watermark (%)	Val **mASR (%)	Val Loss on Benign	Val Loss on Watermarked
ResNet18 from Scratch (Seen for prior works, but unseen for the proposed method)	Benign	N/A	93.83	10.72	N/A	0.2679	0.3143
	BadNets [6]	**with Label Noise**	89.13	***N/A	96.78	0.4763	N/A
	Blended [2]	**with Label Noise**	82.35	N/A	87.79	0.6851	N/A
	Hidden Trigger [16]	10 Triggers	93.47	N/A	1.82	0.2806	N/A
	Sleeper Agent [18]	1 Trigger	93.56	N/A	45.17	0.2834	N/A
	Sleeper Agent [18]	10 Triggers	90.89	N/A	16.14	0.4036	N/A
	Poison Frogs [17]	10 Verification Images	91.31	N/A	8.00	0.4113	N/A
	MetaPoison [9]	10 Verification Images	91.01	N/A	23.00	0.4286	N/A
	Bullseye [1]	10 Verification Images	91.21	N/A	13.00	0.4191	N/A
	Gradient Matching [4]	10 Verification Images	91.54	N/A	**70.00**	0.3950	N/A
	Radioactive Data [15]	10 Radioactive Classes	91.87	N/A	N/A	0.3816	0.3289
	Proposed	10 Hidden Classes	92.54	63.67	N/A	0.3339	**0.1045**
MobileNetV2 from Scratch (Unseen for all methods)	Benign	N/A	94.67	11.77	N/A	0.3782	0.5168
	Hidden Trigger [16]	10 Triggers	94.13	N/A	1.85	0.4276	N/A
	Sleeper Agent [18]	1 Trigger	94.26	N/A	16.72	0.4019	N/A
	Sleeper Agent [18]	10 Triggers	92.20	N/A	5.95	0.4972	N/A
	Poison Frogs [17]	10 Verification Images	92.22	N/A	1.00	0.4885	N/A
	MetaPoison [9]	10 Verification Images	92.04	N/A	1.00	0.4940	N/A
	Bullseye [1]	10 Verification Images	92.26	N/A	0.00	0.4995	N/A
	Gradient Matching [4]	10 Verification Images	92.09	N/A	42.00	0.4846	N/A
	Radioactive Data [15]	10 Radioactive Classes	92.92	N/A	N/A	0.4834	0.3723
	Proposed	10 Hidden Classes	93.07	**69.42**	N/A	0.4445	**0.1847**
MobileNetV2 from ImageNet-pretrained (Unseen for all methods)	Benign	N/A	95.53	10.62	N/A	0.6439	0.7493
	Hidden Trigger [16]	10 Triggers	95.07	N/A	1.22	0.6823	N/A
	Sleeper Agent [18]	1 Trigger	95.22	N/A	8.47	0.6603	N/A
	Sleeper Agent [18]	10 Triggers	93.44	N/A	4.17	0.6822	N/A
	Poison Frogs [17]	10 Verification Images	93.41	N/A	0.00	0.6818	N/A
	MetaPoison [9]	10 Verification Images	93.27	N/A	1.00	0.6829	N/A
	Bullseye [1]	10 Verification Images	93.41	N/A	0.00	0.6771	N/A
	Gradient Matching [4]	10 Verification Images	93.58	N/A	40.00	0.6657	N/A
	Radioactive Data [15]	10 Radioactive Classes	94.31	N/A	N/A	0.7349	0.6502
	Proposed	10 Hidden Classes	94.32	**69.50**	****23.03	0.6771	**0.4712**

*mAcc: Mean Class Accuracy
**mASR: Mean Class Attack Success Rate
*** We report only their own metric for each approach, and others are denoted as "N/A."
**** Attack success rate of the proposed method is explained in Section 11.

achieved higher validation accuracy for both the benign CIFAR10 and the watermark, demonstrating its harmlessness and better verifiability compared to poisoning methods. It's worth noting that the data poisoning can impact on only specific verification images, while our method has no limitation on the number of watermarks. Furthermore, our work consistently achieved meaningful accuracy on the watermarks in every trial, indicating its reliability compared to other methods. Comparing to radioactive data, our work shows a lower validation loss on the benign CIFAR10 and a larger difference between the validation losses of the benign and watermarked datasets. Based on these findings, we concluded that the proposed method is superior to the existing methods in terms of harmlessness, verification ability, and stability.

5 Comparison of Adversarial Attack and Autoencoder

Purpose. It is possible to use adversarial attack instead of autoencoders. Many of the existing works adopted adversarial attack for marking. However, we anticipated that it would be inferior to the autoencoder for the following reasons: 1) adversarial attack tends to fool the reference model rather than hide auxiliary data, 2) adversarial attack must be highly dependent on the reference model, making it difficult to be applied to other architectures, and 3) instance-wise optimization results in inconsistent watermarks.

Setting. To compare attack-based and autoencoder-based watermarking, we trained two MobileNetV2 trained on their watermarked CIFAR10 datasets, respectively. CIFAR10 and Fashion MNIST were adopted as the target and auxiliary datasets for the autoencoder-based approach, along with a pre-trained DWN on the datasets. For adversarial attack, a pre-trained ResNet18 trained on benign Fashion MNIST was used as reference model $F_{ref}(\cdot, \theta_{ref})$. Following the previous works, we applied optimization as follows:

$$\mathbf{w} = \underset{\Delta}{\operatorname{argmin}} ||\mathbf{\Delta}|| \quad \text{s.t.} \quad \mathcal{F}_{ref}(\mathbf{x} + \mathbf{\Delta}, \theta_{ref}) = \mathbf{y}^{\mathbf{z}}. \tag{2}$$

The average accuracy of 10 trials is presented for both cases, and each trial was trained from scratch. Then, half of the training samples were randomly watermarked for each trial. Then, we evaluated watermarking time, invisibility, harmlessness, and verifiability of them.

Table A.2: Comparison of autoencoder and adversarial attack for watermarking using MobileNetV2.

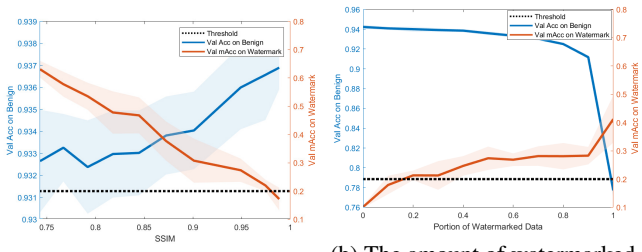
	Time Cost (sec/image)	Invisibility (SSIM)	Val Acc on Benign (%)	Val mAcc on Watermark (%)
Adversarial Attack	2.87e-1	0.8882	92.15	24.87
DWN (Autoencoder)	**1.07e-3**	**0.9785**	93.07	**69.42**

Results. As shown in Table A.2, autoencoder is about $300\times$ faster than adversarial attack for watermarking. The watermark generated by adversarial attack is not only dependent on reference model, but also more visible than the one by autoencoder. In addition, the autoencoder-based watermark is less harmful on benign CIFAR10 and more effective than the adversarial-attack-based watermark. Based on these analyses, we found that the autoencoder-based strategy is more efficient than the adversarial-attack-based approach.

6 Sensitivity to Strength of Watermark and the Amount of Watermarked Data

Purpose and Setting. In addition, we assessed the mAcc for watermarked images with varying SSIM and varying amount of watermarked data in entire training dataset. For this, we employed DenseNet on the Fashion MNIST dataset, maintaining the same setup as outlined in Section 4.2 because its mAcc was close to $\frac{2}{N_{cls}^s}$. We generated watermarked images with varying SSIM values by adjusting the hyperparameters of training DWN. Then, we trained DenseNet multiple times using the various watermarked datasets. Note that we mainly used 0.9491 SSIM 50% watermarked data in our experiments.

Results. Figure 2 (a) illustrates the impact of watermark strength. Notably, it's feasible to regulate verifiability by adjusting the SSIM of watermarked images. Stronger watermarks, indicated by lower SSIM values, lead to higher mAcc, while weaker watermarks with higher SSIM values result in lower mAcc. The results reveal that the challenge



(a) Strength

(b) The amount of watermarked data

Fig. 2: Impact of watermark strength (SSIM) and the amount of watermarked data on mAcc. The shading represents the variations of all trials.

isn't in determining the best threshold that distinguishes cheating models from clean models. Instead, the focus should be on 1) establishing a threshold that clean models cannot attain, and 2) generating the least visible watermarks to compel cheating models to achieve an mAcc surpassing the threshold.

As shown in Figure 2 (b), the more portion of watermarked data makes the better verifiability. For validation accuracy on the benign CIFAR10, there are only negligible damages until 60% of ratio, and very small damages from 70% to 90%. Only the case whose all training data is watermarked shows a remarkably negative impact on the benign accuracy.

Based on these results, we could conclude that it is possible to adjust the expected mAcc on watermark by watermark strength and the amount of watermarked data in entire training dataset. Therefore, dataset owners can control their dataset to achieve higher mAcc than threshold.

7 Additional Results for Histogram Analysis of mAcc on Watermark

Purpose and Setting. In the manuscript, we conducted histogram analysis to underscore the difficulty in achieving an mAcc higher than a specified threshold for clean models. To further underscore this difficulty, we imposed more stringent conditions using CIFAR100, highly subtle watermarks with 100 classes. In this section, we compared the histograms using two watermarked CIFAR100 datasets—one with an SSIM of 0.9710 and the other with an SSIM of 0.9884—trained on DenseNet. Note that we watermarked 50% training data for both cases. To do this, we applied the same training methodology as outlined in one of the histogram analyses presented in the main manuscript.

Results. In the manuscript, we have already presented histograms depicting the performance of clean models and models with subtle watermarks, demonstrating successful differentiation based on our pre-defined threshold. Furthermore, Fig. 3 illustrates the outcomes of comparing subtle and strong watermark variants. Notably, the strong watermark exhibits a more substantial deviation from the chance level, averaging around

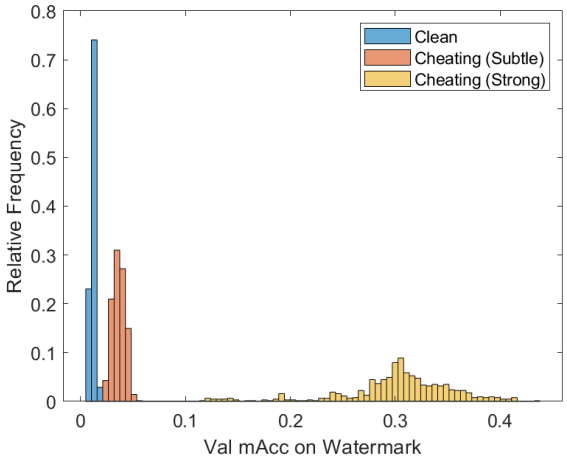


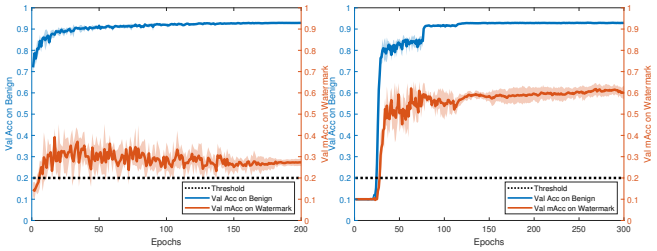
Fig. 3: Additional result of histogram analysis.

30% mAcc. This observation suggests that the proposed watermarking method can effectively accommodate both subtle and strong watermark variants. Consequently, the anticipated deviation from the pre-defined threshold can be managed by adjusting the watermark's strength.

8 Simultaneous Training of Target and Hidden Watermark

Purpose and Setting. Target context of dataset and our watermarks can be jointly trained. Then, our previous experiments validate that its feasibility and applicability to verification of unauthorized use of dataset. In this section, we visualize that the two tasks are simultaneously learned by representing epoch-wise history of target performance and mAcc on watermark. For this, we employed two settings: DenseNet on the Fashion MNIST dataset and MobileNetV2 on CIFAR10 dataset with Adam optimizer and learning rate decay. We employed cosine decay for DenseNet with Fashion MNIST, and step decay for MobileNetV2 with CIFAR10.

Results. Figure 4 presents the historical data for the two cases. As illustrated, both tasks are trained concurrently. For the case of the MobileNetV2 and CIFAR10 scenario, the simultaneous training is clearly demonstrated. In the case of DenseNet with Fashion MNIST, the convergence of the mAcc is comparatively lower. However, it's observable that there is a simultaneous increase in performance for both the hidden and target tasks. These results validate our hypothesis in a practical setting, confirming that both the hidden watermarks and the target task can be effectively learned through dataset overlaying.



(a) DenseNet + Fashion MNIST (b) MobileNetV2 + CIFAR10

Fig. 4: Histories of validation accuracy of target task and validation mAcc on watermark.

9 Additional Experiments about Noise Placements

Purpose and Setting. In manuscript, we were concerned that the noise placement is weak to spatial transformation (i.e., flip or rotation), which is commonly used as data augmentation. To concretely validate this, we evaluated the noise placement on CIFAR10, utilizing another architecture (MobileNetV2) from random initialization and employing random flip as a data augmentation. We intentionally designed the noise patches, as displayed in Figure 5, to consist of mirrored pairs. Also, we adjusted the average SSIM of the watermarked images to be similar to our approach's.

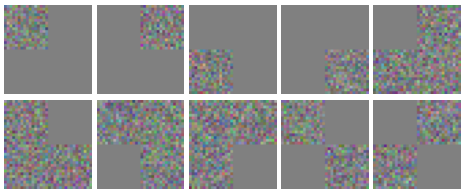


Fig. 5: Examples of noise patch placement.

Results. As shown in Table A.3, noise placement is significantly influenced by spatial transformations. When solely employing horizontal flip, mAcc is close to 50% due to the presence of five pairs of mirrored patterns in the utilized noise patches. When using both horizontal and vertical flips, the first four patches and the subsequent four patches are confused due to combined flips, resulting in a 25% mAcc. The last two patches confused each other, leading to a 50% mAcc. Thus, the average value is about 30%, closely matching the achieved mAcc. If we used more precise positions, the verification would be additionally affected by rotation and translation. These indicate that the watermarks must be robust to data augmentation. Leveraging a auxiliary dataset stands out as an evident approach to achieving structural watermarks.

Table A.3: Results of watermarks based on noise placement

Random Flip H*	Random Flip V*	Val Acc on Benign (%)	Val mAcc on Watermark (%)
✓	✗	93.02	48.99
✓	✓	92.01	28.55

*"H" and "V" indicate the horizontal, and vertical flip.

10 Additional Results for Overcoming Limitation in the Number of Classes

Purpose and Setting. We propose a solution to overcome the limitation where the auxiliary dataset needs to contain more classes than the target dataset, and validated using ImageNet dataset. Additionally, we validated the solution using Tiny ImageNet [10], which consists of 64 $\times$ 64 sized colored images with 200 classes. Then, Fashion MNIST was used as a auxiliary dataset to generate watermarks, and 50% of training datasets were watermarked. We trained MobileNetV2 on watermarked Tiny ImageNet (0.9883 avg. SSIM) from scratch for 100 epochs, utilizing the Adam optimizer with a learning rate decay starting from 1e-3. We employed cross entropy as the loss function, a batch size of 128, label smoothing, and data augmentation techniques.

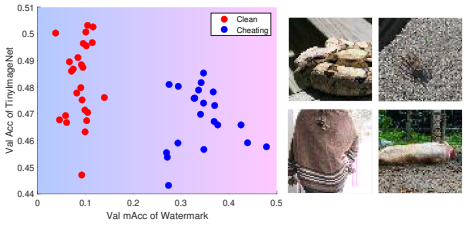


Fig. 6: Results and examplar images of Tiny ImageNet.

Results. Figure 6 presents the results obtained on Tiny ImageNet. As shown, the clean trials show almost a chance-level, 10% for the watermark, while the cheating trials show significantly higher accuracies. In terms of the validation accuracy on benign data, the cheating trials show slightly lower validation accuracy within 1%p on average. In this experiment, the cheating models can be perfectly verified by our work. This finding also support the verifiability of our approach, even when the target dataset contains a greater number of classes compared to the auxiliary dataset. Consequently, it is not necessary to prepare a auxiliary dataset that matches the target dataset in terms of the number of classes.

11 Applicability to Backdoor Attack

Purpose. Prior experiments employed watermarked data adhering to the condition $\mathbf{y}^{\mathbf{x}} = \mathbf{y}^{\mathbf{w}}$, where $\mathbf{y}^{\mathbf{x}}$ corresponds to both the target sample $\mathbf{x}$ and the watermark $\mathbf{w}$. We investigated the network's behavior when operating on watermarked samples with $\mathbf{y}^{\mathbf{x}} \neq \mathbf{y}^{\mathbf{w}}$.

In this case, the network would be confused because $\mathbf{x}$ leads to $\mathbf{y^x}$, whereas $\mathbf{w}$ attracts to $\mathbf{y^w}$. In short, we applied our work as backdoor attack.

Setting. For evaluating our approach as a backdoor attack, we employed a MobileNetV2 model that was trained on a watermarked dataset, referred to as the "cooperative set," wherein $\mathbf{y^x} = \mathbf{y^w}$ was satisfied. In the context of backdoor attacks, the term "victim model" is used interchangeably with "cheating model." After training, we tested the trained model on an "adversarial set," which adhered to $\mathbf{y^x} \neq \mathbf{y^w}$. We conducted 10 trials and reported the average results.

Our expectations were as follows: the victim models should perform well on both benign target data and watermarked data from the cooperative set, but exhibit poor performance on watermarked data from the adversarial set. Additionally, we anticipated that the victim models would classify both the cooperative and adversarial watermarks as $\mathbf{y^w}$ accurately.

Table A.4: Results of cooperative and adversarial watermarks.

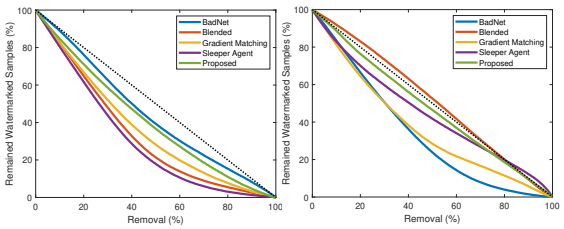
		Val mAcc on Target Label (%)	Val mAcc on Adversarial Label (%)
Cooperative ($\mathbf{y^x} = \mathbf{y^w}$)	Watermarked ($\hat{\mathbf{x}}$)	98.42	N/A
	Benign Target ($\mathbf{x}$)	94.32	N/A
	Watermark ($\mathbf{w}$)	69.50	N/A
Adversarial ($\mathbf{y^x} \neq \mathbf{y^w}$)	Watermarked ($\hat{\mathbf{x}}$)	64.98	**23.03**
	Benign Target ($\mathbf{x}$)	94.32	N/A
	Watermark ($\mathbf{w}$)	N/A	69.42

Results. The average results of 10 trials on the cooperative and adversarial validation datasets are presented in Table A.4, aligning with our expectations. The trained weights were applied to the adversarial dataset, resulting in no discernible difference in performance on the benign target dataset. Additionally, there was no significant distinction in performance on the watermark for both cases, with mAcc of 69.5% for the cooperative watermark and 69.42% for the adversarial watermark, respectively. Notably, the models achieved high accuracy on cooperatively-watermarked images, while facing challenges in classifying adversarially-watermarked images. The accuracy on adversarially-watermarked data with the adversarial label serves as the ASR, a well-established metric for assessing backdoor attacks. Specifically, our method achieved 23.03% ASR for a 10-trigger case on MobileNetV2, surpassing that of a sleeper agent. The following observations can be made based on the results: 1) The discriminability of the watermark is comparable to a 34% drop in accuracy for the watermarked dataset, reflecting the decline in validation mAcc on the target label due to adversarial watermarking. 2) The network's performance is significantly influenced by the learned watermark, lending itself as admissible evidence. 3) Our approach demonstrates competitive performance with a 23.03% ASR for a backdoor attack. Based on this property, our approach can also be used for misclassification-based verification.

12 Robustness to Defense

Purpose and Setting. There have been numerous studies focusing on defense mechanisms to proactively filter out watermarked or attacked samples. In this evaluation, we assessed the robustness of four methods: blended, gradient matching, sleeper agent, and our proposed approach. To compare their effectiveness, we selected two popular defense methods, namely STRIP [3] and Spectral signature [21]. Both of these defense methods require a model trained on a watermarked dataset, so we reused a ResNet18 model trained on attacked CIFAR10. Our evaluation involved filtering a dataset consisting of 50,000 clean samples and 50,000 attacked samples.

STRIP assumes that the trigger or hidden watermark still has effect even for the superimposed images. To detect the presence of such watermarks, STRIP employs a technique where it combines a target image with multiple clean images and measures the average entropy. When the target image is attacked, the average entropy is expected to decrease significantly. Conversely, if the target image is clean and devoid of any hidden watermark, the average entropy should remain relatively high. By comparing the average entropy values, STRIP aims to identify whether the target image has been attacked or is in its original clean state. Spectral signature employs singular value decomposition (SVD) on the per-sample features extracted by the trained model. Based on the observations made by the spectral signature, attacked samples tend to exhibit higher singular values compared to clean samples. Leveraging this insight, it becomes feasible to identify and filter out attacked samples by focusing on samples with the highest singular values. By removing samples associated with these top singular values, the majority of attacked samples can be effectively eliminated from the dataset.



(a) STRIP (b) Spectral Signature

Fig. 7: Results of two defense methods. The proposed watermark is hard to be detected by the two defense methods.

Results. The results are depicted in Figure 7. The x-axis represents the gradual removal of samples, while the y-axis indicates the ratio of remaining watermarked samples. A straight line connecting the points (0,100) and (100,0) indicates that the defense method fails to differentiate between clean and attacked data. For instance, the sleeper agent method can eliminate 50% of attacked samples by removing about 30% of all samples, resulting in the filtering out of 25,000 attacked samples out of 30,000 removed samples. It is noteworthy that the badnet and blended approaches incorporate label noise, making it easier to identify and filter out attacked samples through human inspection. In

contrast, our proposed method exhibits the highest level of robustness to defense methods. Importantly, our proposed method doesn't introduce label noise and maintains the robustness to various defense methods.

13 Robustness to Debias

Purpose and Setting. Our study is founded upon deliberately incorporating hidden biases. This raises questions about the efficacy of debiasing methods in removing the proposed watermark. To assess its robustness to debiasing, we employed the "Learning from Failure (LfF)" approach [12], which operates effectively without prior knowledge of bias. According to Nam et al., bias is not always harmful; it becomes malignant when it is easier to learn than the target knowledge. Therefore, LfF endeavors to differentiate between benign and malignant biases based on their ease to learn. In this method, two identical networks, $\mathcal{F}_B$ (biased network) and $\mathcal{F}_D$ (debiased network), are trained by following process for every iteration. First, $\mathcal{F}_B$ is trained using generalized cross entropy (GCE) [23], which facilitates faster learning of easier knowledge. Subsequently, $\mathcal{F}D$ is trained using naive cross entropy along with a weighting factor defined as:

$$\frac{\mathcal{L}_{CE}(\mathcal{F}_B)}{\mathcal{L}_{CE}(\mathcal{F}_B) + \mathcal{L}_{CE}(\mathcal{F}_D)}, \tag{3}$$

where $\mathcal{L}CE(\mathcal{F})$ represents the cross entropy loss of network $\mathcal{F}$. Through repeated iterations of these two processes, $\mathcal{F}_B$ learns biased knowledge while $\mathcal{F}_D$ learns debiased knowledge. We applied this LfF approach to our watermarked CIFAR10 dataset using ResNet18, varying the slope hyperparameter of GCE (q). For the training regimen, we conducted 100 epochs using the Adam optimizer with cosine decay, commencing with a learning rate of 1e-3, and implemented spatial transformation as a form of data augmentation.

Table A.5: Results of debias

q	Model	Val Acc on Benign (%)	Val mAcc on Watermark (%)
$q = 0.3$	$\mathcal{F}_B$	92.27±0.26	62.09±4.09
	$\mathcal{F}_D$	30.86±1.88	9.92±0.31
$q = 0.5$	$\mathcal{F}_B$	91.70±0.32	55.90±2.87
	$\mathcal{F}_D$	28.45±1.56	10.47±1.22
$q = 0.7$	$\mathcal{F}_B$	90.33±0.29	43.35±3.29
	$\mathcal{F}_D$	35.33±2.31	28.33±16.46

Results. If LfF successfully removed our watermark, $\mathcal{F}_D$ would be expected to fail in classifying it while maintaining accuracy on benign data. The average results of ten

trials are presented in Table A.5. As depicted, approximately 62% accuracy on benign data needs to be compromised to achieve a 10% mAcc (chance level) on the watermark. These findings suggest that the proposed watermark presents a similar learning difficulty compared to the benign data, indicating high robustness to debiasing method.

14 Robustness to Denoising

Purpose and Setting. We assessed the robustness of our work and Gradient Matching to denoising using official blind image denoising [13]. We denoised the watermarked train set of CIFAR10, and finetuned ImageNet-pretrained MobileNetV2 using the recipe of Section 5 in main manuscript.

Table A.6: Results of Training on Denoised Data

Method	Val mAcc on Benign (%)	Val mAcc on Watermark (%)
Gradient Matching	87.34±1.57	36.00±8.94
Proposed	86.39±0.31	33.08±1.11

Results. Because our watermark is subtle (SSIM$\simeq$0.98), it can be removed by denoising. However, it maintained verifiable watermark accuracy above the threshold, indicating **robustness to denoise**.

15 Qualitative Examples

This section presents further examples of our watermarked CIFAR10 data alongside their corresponding Class Activation Maps (CAMs) generated by clean and cheating ResNet18 models. As depicted in Fig. 8, both clean and cheating models effectively concentrate on the regions containing discriminative features pertinent to CIFAR10 context, regardless of whether the data is benign or watermarked. However, notably, only the cheating model adequately directs its attention to the regions corresponding to the embedded watermarks. The clean model ignores or focuses on non-watermark regions. This observation underscores the capability of cheating models to discern and focus on the watermark regions, highlighting the effectiveness of our verification.

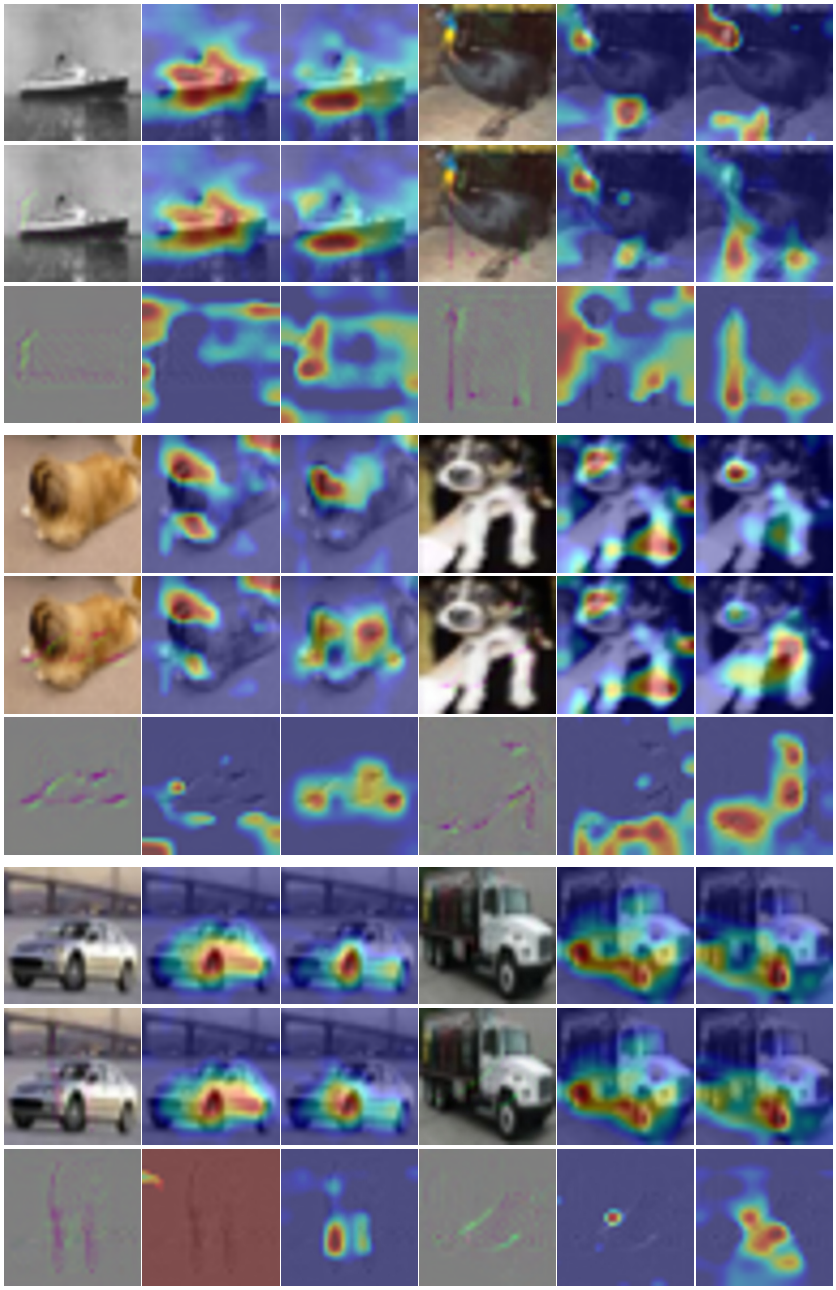


Fig. 8: Additional examples and their corresponding CAMs achieved by ResNet18. Each row represent **Top**: benign image, **Middle**: CAM for watermarked image, and **Bottom**: CAM for watermark. Then, each column indicates **1st&4th**: input image, **2nd&5th**: CAM of clean model, and **3rd&6th**: CAM of cheating model.

16 Human Evaluation

Purpose and Setting. We conducted a Google Form using a randomly selected 50 images (23 benign & 27 watermarked) from ImageNet. Totally 63 volunteers (26 experts in AI and CV area & 37 non-experts) participated to identify the watermarked images. The participants were informed about the presence of watermarked images but were not given details (i.e., the amount of watermarked images).

Table A.7: Results of Human Evaluation

	Precision (%)	Recall (%)
Experts	69.87±24.61	47.26±28.65
Non-Experts	65.22±20.48	46.45±31.62

Results. As shown in Table A.7, the results well represent the difficulty to distinguish our watermarks by visual inspection. As shown in Supp. 6, it is verifiable if only 20% of training data is watermarked. Also, please note that the participants were aware of the existence of watermarks as prior knowledge, making the task easier.

References

1. Aghakhani, H., Meng, D., Wang, Y.X., Kruegel, C., Vigna, G.: Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In: EuroS&P. pp. 159–178 (2021) 4, 6
2. Chen, X., Liu, C., Li, B., Lu, K., Song, D.: Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526 (2017) 5, 6
3. Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D.C., Nepal, S.: Strip: A defence against trojan attacks on deep neural networks. In: Proceedings of the 35th Annual Computer Security Applications Conference. pp. 113–125 (2019) 13
4. Geiping, J., Fowl, L.H., Huang, W.R., Czaja, W., Taylor, G., Moeller, M., Goldstein, T.: Witches' brew: Industrial scale data poisoning via gradient matching. In: ICLR (2021) 4, 6
5. Goodfellow, I.J., Erhan, D., Carrier, P.L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.H., et al.: Challenges in representation learning: A report on three machine learning contests. In: NeurIPS (2013) 1
6. Gu, T., Liu, K., Dolan-Gavitt, B., Garg, S.: Badnets: Evaluating backdooring attacks on deep neural networks. IEEE Access **7**, 47230–47244 (2019) 5, 6
7. Heo, B., Yun, S., Han, D., Chun, S., Choe, J., Oh, S.J.: Rethinking spatial dimensions of vision transformers. In: ICCV (2021) 1
8. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017) 1
9. Huang, W.R., Geiping, J., Fowl, L., Taylor, G., Goldstein, T.: Metapoison: Practical general-purpose clean-label data poisoning. NeurIPS **33**, 12080–12091 (2020) 4, 6
10. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N **7**, 7 (2015) 2, 11
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998) 1
12. Nam, J., Cha, H., Ahn, S., Lee, J., Shin, J.: Learning from failure: De-biasing classifier from biased classifier. NeurIPS **33**, 20673–20684 (2020) 14
13. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR. pp. 10684–10695 (2020) 15
14. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241. Springer (2015) 5
15. Sablayrolles, A., Douze, M., Schmid, C., Jégou, H.: Radioactive data: tracing through training. In: ICML. pp. 8326–8335 (2020) 4, 6
16. Saha, A., Subramanya, A., Pirsiavash, H.: Hidden trigger backdoor attacks. In: AAAI. vol. 34, pp. 11957–11965 (2020) 4, 6
17. Shafahi, A., Huang, W.R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., Goldstein, T.: Poison frogs! targeted clean-label poisoning attacks on neural networks. NeurIPS **31** (2018) 4, 6
18. Souri, H., Fowl, L., Chellappa, R., Goldblum, M., Goldstein, T.: Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. NeurIPS **35**, 19165–19178 (2022) 4, 6
19. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML. pp. 6105–6114 (2019) 1
20. Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J., et al.: Resmlp: Feedforward networks for image classification with data-efficient training. ICLR (2021) 1
21. Tran, B., Li, J., Madry, A.: Spectral signatures in backdoor attacks. NeurIPS **31** (2018) 13
22. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvt v2: Improved baselines with pyramid vision transformer. Computational Visual Media pp. 1–10 (2022) 1

23. Zhang, Z., Sabuncu, M.: Generalized cross entropy loss for training deep neural networks with noisy labels. NeurIPS **31** (2018) 14