Supplementary Material for Defending Against Repetitive Backdoor Attacks on Semi-supervised Learning through Lens of Rate-Distortion-Perception Trade-off

The content of Supplementary Material is summarized as follows: 1) In Sec. A, we discuss the details of SSL algorithms used in our work; 2) In Sec. B, we state the implementation and training details we used in the experiment in terms of datasets, hyper-parameters, and model architectures to ensure that our method can be reproduced; 3) In Sec. C, we first recall the rate-distortion theory. Then, we present the rate-distortion-perception (RDP) trade-offs from theoretical derivation. Last, we discuss the justification that repetitive trigger patterns are ineffective.

A. SSL Algorithms

We conducted the main experiments using five state-of-the-art SSL algorithms, briefly summarized as follows.

(a) MixMatch [2] creates various weakly augmented versions of each unlabeled sample. It then calculates the outputs of the current model for these versions and sharpens the average prediction by raising all its probabilities before normalization. This refined prediction acts as the label for the unlabeled sample. In addition, MixMatch employs mixup regularization on all training data and trains the model using cross-entropy loss.

(b) ReMixMatch [1] replaces weak data augmentation in MixMatch with AutoAugment and enhances consistency regularization through augmentation anchoring. This technique involves using predictions made on a weakly augmented version of an unlabeled sample as the target prediction for a strongly augmented version of the same sample. Additionally, it employs distribution alignment, which normalizes the new model predictions on unlabeled data using the running average of model predictions on unlabeled data, significantly enhancing the resulting model's performance.

(c) Unsupervised data augmentation (UDA) [14] exhibits superior performance on SSL with the benefit of strong data augmentations, such as RandAugment, instead of weak data augmentations used in MixMatch. Specifically, RandAugment randomly chooses a few powerful augmentations to improve the generalization and robustness of the model.

(d) FixMatch [11] combines consistency regularization and pseudo-labeling while simplifying the complex ReMixmatch algorithms. In FixMatch, weak augmentation follows a standard flip-and-shift strategy, randomly flipping images horizontally with a given probability. For strong augmentation, RandAugment and CTAugment are employed. Furthermore, Cutout is followed after these above operations.

(e) FlexMatch [18] presents a curriculum pseudo-labeling (CPL) approach, which flexibly sets the threshold of pseudo-labels in different categories in each training iteration. Then, according to the model's learning status, FlexMatch selects more informative unlabeled data and their pseudo-labels.

B. Experimental Details

B.1. Datasets and DNNs

We list the detailed dataset and model architecture used in our experiments, as summarized in Table 5, which shows the number of classes in each dataset alongside the number of training and test data. Compared to CIFAR10, SVHN is designed for recognizing street-view house numbers and is not class-balanced. STL10 is a 10-class classification task specially designed for semi-supervised learning research. Besides, we use WideResNet [16] as model architecture in our experiments.

B.2. Training size of Labeled data

Table 6 shows that different sizes of labeled data depend on the algorithm in our experiments. Even when there is little labeled training data (*i.e.*, 100 samples for CIFAR10), UPure still effectively alleviates backdoor effects and maintains model accuracy.

B.3. Training Settings

Following the training settings in [13], we adopted an SGD optimizer with a momentum of 0.9, a weight decay of 1×10^{-3}, layer decay of 1, a crop ratio of 0.875, and an initial learning rate of 3×10^{-2} in our experiments. With a batch size of 64, we trained the WideResNet-28-2 model for

Table 5. Statistics of datasets used in our experiments

Dataset	Input size	Classes	Unlabeled Training data	Test data	Model
CIFAR10	$3 \times 32 \times 32$	10	50000	10000	WideResNet-28-2
SVHN	$3 \times 32 \times 32$	10	73257	26032	WideResNet-28-2
STL10	$3 \times 96 \times 96$	10	100000	1000	WideResNet-28-2
CIFAR100	$3 \times 32 \times 32$	100	50000	10000	WideResNet-28-8

Table 6. Sizes of labeled data for training a model across different SSL algorithms.

Dataset	Algorithm				
	MixMatch	ReMixMatch	UDA	FixMatch	FlexMatch
CIFAR10	4000	100	100	100	100
SVHN	250	250	100	100	100
STL10	3000	1000	1000	1000	1000
CIFAR100	10000	2500	2500	2500	2500

$200,000$ iterations, as shown in Table 5. Note that for CIFAR100, we trained the WideResNet-28-8 model. All the other settings in SSL algorithms are the same as the original configurations in the USB package [13]. Furthermore, we executed the experiments with a single NVIDIA RTX3090 GPU. Nevertheless, SSL is still cost-expensive for computation. For instance, in our experiments, the FixMatch algorithm requires approximately 16 hours to complete $200,000$ iterations on CIFAR10, whereas, for MixMatch and ReMixMatch, each takes about 6 hours. Training for the same number of iterations on CIFAR100 using FixMatch extends to 3.5 days, leading us to exclude experiments with UDA and FlexMatch on CIFAR100.

B.4. UPure algorithms

Our training procedure is described in Algorithm 1. Specifically, UPure allows a defender to purify the unlabeled training data for SSL in the frequency domain using three strategies. This approach emphasizes preprocessing data before training, rather than identifying backdoor samples within the model.

B.5. Details of Backdoor Defense

We evaluate our experiments using four post-processing and one in-processing backdoor defenses for comparison, briefly summarized as follows.

(a) Fine-tuning is a common baseline to alleviate pernicious behavior on the backdoored model using additional clean labeled data. In our experiments, we utilize the labeled training data of SSL algorithm for finetune.

(b) Fine-pruning [8] first prunes the inactivated neurons of the last layer by benign data and then finetunes the model to prevent the activation of backdoors.

(c) Neural Attention Distillation (NAD) [7] fine-tunes a teacher model on a subset of benign data and distills the knowledge of the fine-tuned model into backdoored model

Algorithm 1: Training of UPure

```
Input: Training data $D_{train} = D_\ell \cup D_u$, an SSL learning algorithm $\mathcal{A}_{ss\ell}$, a loss function $\mathcal{L}_{ss\ell}$, a DCT function $\mathcal{T}_{dct}$ and an inverse DCT $\mathcal{T}_{idct}$, three strategies $\mathcal{S} = \{S_1, S_2, S_3\}$, where $S_1$ is "Turn to zero," $S_2$ is "Replace from other," and $S_3$ is "Add perturbation," and a predefined region $\tau \times \tau$ for perturbation.
Output: Clean Model $\mathcal{M}^*$.
/* Step 0: Pick a strategy */
$S^\star \leftarrow \mathcal{S}$
/* $D_u$ is a unlabeled dataset */
for $img \in D_u$ do
    $D_u = D_u \backslash \{img\}$   /* remove image from $D_u$ */
    /* Step 1: Transform to DCT spectrum */
    $spectrum = \mathcal{T}_{dct}(img)$
    /* Step 2: Apply UPure */
    if $S^* == S_1$ then
        $spectrum[-\tau :, -\tau :] = \mathbf{0}$
    else if $S^* == S_2$ then
        $img_\ell \sim D_\ell$   /* randomly select an image from $D_\ell$ */
        $spectrum_\ell = \mathcal{T}_{dct}(img_\ell)$
        $spectrum[-\tau :, -\tau :] = \mathbf{0}$
        $spectrum[-\tau :, -\tau :]$ += $spectrum_\ell$
    else if $S^* == S_3$ then
        $\eta \sim \mathcal{N}(\mathbf{0}, \sigma^2 I)$   /* sample a noise */
        $spectrum[-\tau :, -\tau :]$ += $\eta$
    /* Step 3: Inverse transform to pixel domain */
    $img = \mathcal{T}_{idct}(spectrum)$
    Apply cutout operation on $img$
    $D_u = D_u \cup \{img\}$   /* place image back into $D_u$ */
/* Adopt an SSL algorithm to train a model $\mathcal{M}^*$ */
Randomly initialize a model $\mathcal{M}$
$\mathcal{M}^*$=$\mathcal{A}_{ss\ell}(D_{train}, \mathcal{L}_{ss\ell}, \mathcal{M})$ return Model $\mathcal{M}^*$
```

for purification.

(d) Backdoor Adversarial Unlearning (I-BAU) [17] purifies the backdoored model by using an implicit hypergradient, which facilitates the model convergence and the generalizability of robustness given a small number of clean data.

(e) Detection-and-Purification (DePuD) [15] utilizes GradCam to detect suspicious region (*i.e.*, backdoor triggers) in images. According to the model's attention, the purification operation employs differential privacy to alleviate the effects of poisoned images.

B.6. Visualization results of UPure

We present visualization results of UPure in Fig. 7, comparing clean and backdoored samples with purified samples obtained from our three strategies. We utilize a repetitive backdoor attack in our work, as detailed in [10], with pixel intensity, width, and gap set to 30, 1, and 1, respectively. As can be seen from Fig. 7, the samples generated from "*Turn to zero*" and "*Replace from others*" are blurry while the "*Add perturbation*" strategy performs minimal disturbance on the backdoored samples, which is sharper than the other two. This indicates that "*Add perturbation*" is better than the other two to maintain the original fidelity of images. Note that the quantitative results of UPure are shown in Tab. 7.

Table 7. Quantitative results with our three strategies on CIFAR10.

Strategy	CIFAR10	
	PSNR	SSIM
Turn to zero	33.03	0.9618±.075
Replace from other	30.65	0.9461±.088
Add perturbation	45.43	0.9969±.012

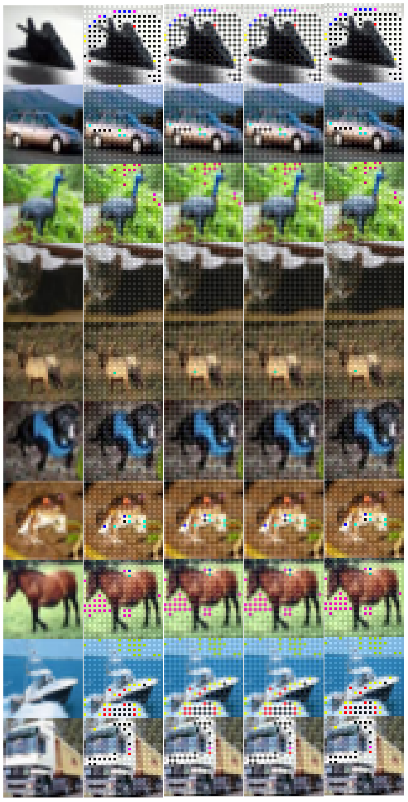


Figure 7. Visualization results on CIFAR10. (From Left to Right) The first and second columns display the clean and backdoored samples, respectively. The third, fourth, and fifth columns show purified samples obtained from using "*Turn to Zero*," "*Replace from others*," and "*Add Perturbation*" strategies of UPure, respectively.

B.7. More experiment results

B.7.1 Non-poisoned dataset applied UPure

Since UPure can be viewed as a form of data augmentation, we further train a model using UPure on a clean training dataset. For example, with CIFAR-10 as the training set, MixMatch and FlexMatch attain high BA, close to their original performance, as shown in Tab. 8. We find that ReMixMatch is more susceptible to high-frequency component perturbations, resulting in a significant decrease in BA. This observation implies that modern SSL algorithms can adapt UPure to protect unlabeled data and be robust to backdoor attacks in SSL scenarios.

Table 8. Evaluation on clean CIFAR10 dataset applied UPure.

	MixMatch	ReMixMatch	UDA	FixMatch	FlexMatch
BA	89.31%	77.82%	88.24%	87.96%	94.18%
ASR	0.21%	1.85%	0.34%	0.33%	0.26%

B.7.2 Non-targeted attacks

Non-targeted attacks refer to the accuracy of classifying clean and trigger inputs based on a trained model. As the clean and poisoned datasets tend to have different class distributions, we consider non-targeted attacks and observe the model's accuracy drop to measure the attack effectiveness if the trigger is input. Tab. 9 shows the purified model's accuracy in classifying clean and trigger inputs. More precisely, we find that even if the validation set does not contain the target class, the trigger inputs tend to be misclassified to certain classes (*e.g.*, "bird", "ship", and "airplane" in the test set). We conduct five SSL algorithms trained on CIFAR10 that are considered in Tab. 9. The purified model trained using FixMatch achieves 89.28% and 89.70% accuracy on clean and trigger inputs, respectively. However, the non-targeted BA of UDA algorithms drops more than the other four algorithms.

Table 9. Evaluation against non-targeted attacks on CIFAR10.

	MixMatch	RemMixMatch	UDA	FixMatch	FlexMatch
BA	87.85%	87.22%	93.59%	89.28%	94.27%
Non-target BA	84.47%	83.73%	77.47%	89.70%	92.14%

B.7.3 Impact of different perturbation area sizes.

We evaluate the impact of different perturbation area sizes, $\tau \times \tau$, in terms of BA and ASR on UPure in Fig. 9, where $\tau \in \{4, 8, 16, 24\}$. We find that a larger distortion region (*e.g.*, 24×24) in the DCT spectrum sacrifices a little BA (*i.e.*, a decrease of 2.33% in BA) but preserves a better ASR (*i.e.*, 0%). Fig. 9 also indicates that a small perturbation region in the high-frequency component is not enough to remove the backdoors. To better compromise between BA and ASR, we choose an appropriate size (*i.e.*, $\tau = 16$) in Sec. 4 for comparisons.

B.7.4 Impacts on poisoning rate of unlabeled data.

Fig. 8 displays the performance of UPure under different poisoning rates of unlabeled data. Specifically, we vary the poisoning rates, *i.e.*, 0.15%, 0.2%, 0.3%, 0.4%, and 0.5%. Regardless of the poisoning rates, UPure can suppress the occurrence of backdoors in terms of nearly zero ASRs and no significant drops in BA. This implies that UPure successfully breaks the association between the backdoor and target class with minimal changes to BA. Due to resource constraints, we perform these experiments only for a subset of combinations from Sec. 4.

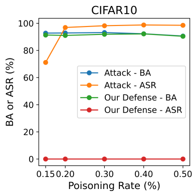


Figure 8. Poisoning rate vs. BA/ASR.

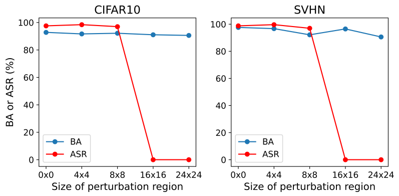


Figure 9. Different sizes of perturbation regions vs. BA/ASR.

B.8. Comparison with pre-processing methods

We compare UPure with three filter-based data pre-processing methods (*i.e.*, Gaussian Filter, Bilateral Filter [12], and Median Filter). We preprocess the unlabeled training data with these filters before feeding them into the model. From Tab. 10, we observe that Gaussian Filter can be effective in terms of lowering ASR across CIFAR10 and SVHN. However, it also significantly degenerates the BA performance in CIFAR10. Bilateral Filter does not eliminate the backdoor effects in both datasets. Median Filter reduces BA the most among other methods. UPure is more effective at resisting backdoor attacks than other methods, showing only a minor decrease in BA and a significantly low ASR.

Table 10. Results on filter-based data pre-processing methods.

Defense	CIFAR10			SVHN		
	BA	ASR	BA ↓	BA	ASR	BA ↓
No defense	92.80	97.54	-	97.61	98.76	-
Gaussian Filter	64.79	1.42	28.01	96.40	$\underline{2.37}$	1.21
Bilateral Filter	$\underline{88.16}$	99.15	4.64	**97.43**	99.84	0.18
Median Filter	42.32	$\underline{1.12}$	50.48	90.83	96.21	6.78
UPure	**91.05**	**0.00**	1.75	$\underline{96.49}$	**0.00**	1.12

C. Theory Details

C.1. Rate-Distortion Theory

Rate-distortion theory analyzes the fundamental tradeoff between the rate used for representing samples from a data source $X \sim p_X$, and the expected distortion incurred in decoding those samples from their compressed representations. Formally, the relation between the input X and output $\hat{X}$ of an encoder-decoder pair, is a (possibly stochastic) mapping defined by some conditional distribution $p_{\hat{X}|X}$. The expected distortion of the decoded signals is thus defined as

$$\mathbb{E}[\Delta(X, \hat{X})], \tag{6}$$

where the expectation is w.r.t. the joint distribution $p_{X,\hat{X}} = p_{\hat{X}|X} p_X$, and $\Delta : \mathcal{X} \times \hat{\mathcal{X}} \to \mathbb{R}^+$ is any full-reference distortion measure such that $\Delta(x, \hat{x}) = 0$ if and only if $x = \hat{x}$.

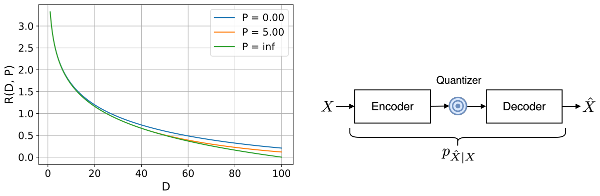
A key result in rate-distortion theory states that for an i.i.d. source X, if the expected distortion is bounded by D, then the lowest achievable rate R is characterized by the (information) rate-distortion function

$$R(D) = \min_{p_{\hat{X}|X}} I(X, \hat{X}) \quad \text{s.t.} \quad \mathbb{E}[\Delta(X, \hat{X})] \leq D, \tag{7}$$

where I denotes mutual information [5]. Closed-form expressions for the rate-distortion function $R(D)$ are known for only a few source distributions and under simple distortion measures (*e.g.*, squared error or Hamming distance). However several general properties of this function are known, including that it is always monotonic, non-increasing, and convex.

C.2. The Rate-Distortion-Perception Trade-offs

In this section, we introduce the RDP function and its solution for Gaussian sources. While there are various solutions under different source conditions, we focus on the Gaussian version due to its simplification of mathematical treatment, particularly under mean squared error (MSE) distortion. This allows us to conduct a theoretical analysis of the universal RDP representation.



(a) $R(D)$ under various P (b) Lossy compression

Figure 10. Rate-Distortion-Perception functions and lossy compression scheme. (a) The curve is computed from Eq. (8) in Theorem 2. Note that these curves represent the lower bound, with all points above them being considered feasible solutions.

Theorem 2. *[19] For a scalar Gaussian source* $X \sim \mathcal{N}(\mu_X, \sigma_X^2)$*, the information rate-distortion-perception function under squared error distortion and squared Wasserstein-2 distance is attained by some* $\hat{X}$ *jointly Gaussian with* X *and is given by*

$$R(D,P) = \begin{cases} \frac{1}{2}\log \frac{\sigma_X^2(\sigma_X - \sqrt{P})^2}{\sigma_X^2(\sigma_X-\sqrt{P})^2 - (\frac{\sigma_X^2+(\sigma_X-\sqrt{P})^2-D}{2})^2} & \textit{if } \sqrt{P} \leq \sigma_X - \sqrt{|\sigma_X^2 - D|}, \\ \max\{\frac{1}{2}\log\frac{\sigma_X^2}{D}, 0\} & \textit{if } \sqrt{P} > \sigma_X - \sqrt{|\sigma_X^2 - D|}. \end{cases} \quad (8)$$

As shown in Fig. 10a, we can see that under the same distortion, the smaller the P is, the larger the rate R is. A source signal $X \sim p_X$ is mapped into a coded sequence by an encoder and back into an estimated signal $\hat{X}$ by the decoder. Through this concept of RDP, we aim to satisfy three properties in our approach: (i) if the coded sequence has low rate, it implies that more repetitive trigger patterns are eliminated; (ii) the reconstruction $\hat{X}$ is similar to the source X on average (low distortion), which indicates that even the clean data is reconstructed well; (iii) the distribution $p_{\hat{X}}$ is similar to p_X so that decoded signals are perceived as genuine source signals (good perceptual quality), which means that the model can still learn the original distribution well (*e.g.*, high classification accuracy).

In addition, we also recall the concept of lossy compression, including an encoder, a decoder, and a quantizer, as depicted in Fig. 10b. The input and output signals follow a predefined distribution. In short, the lossy compression scheme can be viewed as a conditional probability distribution that can be analyzed in a mathematical treatment. For further details, please refer to the previous works [3,4,9,19].

C.3. Analysis of Theorem 1.

Since the common SSL algorithms in Sec. A adapt cutout operation as strong data augmentation on unlabeled training data, we devise Lemma 1 to explain the failure of a single trigger if any area α of the trigger intersects with the cutout region. We further present Lemma 2 to discuss how UPure can resist repetitive trigger patterns by perturbing high-frequency components. Their proofs are presented below.

C.3.1 Proof of Lemma 1.

Let p_f^{single} denote the failure probability of a single trigger when a randomly positioned cutout region intersects it. It can be observed that p_f^{single} reaches its minimum value when the trigger is placed at any corner of the image. This phenomenon can be attributed to the geometric constraints imposed by the image boundaries, which limit the spatial configurations available for the cutout region to intersect the triggers positioned at a corner.

Without loss of generality, we assume the trigger pattern is at the bottom-left corner of the image, where the coordinate is $O = (0, 0)$. Let the position of bottom-left corner of the cutout region be (w, h), as shown in Fig. 11. Under the constraint of minimal coverage area α that makes the trigger invalid, we can obtain the equation

$$(\mathsf{W}_t - w)(\mathsf{H}_t - h) = \alpha.$$

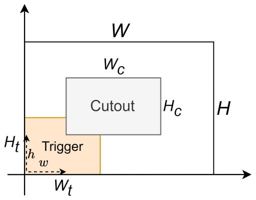


Figure 11. Example of cutout operation intersects with a trigger.

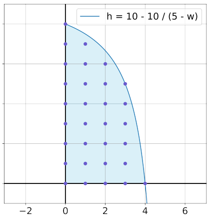


Figure 12. Example of lattice points under a hyperbolic.

With α being a constant, we can derive h and w as follows:

1. Solving for h, we obtain:

$$h = \mathsf{H}_t - \frac{\alpha}{(\mathsf{W}_t - w)}.$$

2. Solving for w, we find:

$$w = \mathsf{W}_t - \frac{\alpha}{(\mathsf{H}_t - h)}.$$

Let us define a function $\Phi(w)$ such that:

$$\Phi(w) = \mathsf{H}_t - \frac{\alpha}{(\mathsf{W}_t - w)}.$$

Analyzing the roots of $\Phi(w)$, specifically when $\Phi(w) = 0$, yields:

$$\mathsf{H}_t = \frac{\alpha}{(\mathsf{W}_t - w)} \Rightarrow w = \mathsf{W}_t - \frac{\alpha}{\mathsf{H}_t}.$$

Additionally, evaluating $\Phi(w)$ at $w = 0$ gives:

$$\Phi(0) = \mathsf{H}_t - \frac{\alpha}{\mathsf{W}_t}.$$

Given the function $\Phi(w) = \mathsf{H}_t - \frac{\alpha}{(\mathsf{W}_t - w)}$ (*e.g.*, the blue curve in Fig. 12), we aim to compute the number of integer solutions under the curve defined by $\Phi(w)$, which means every (w, h) makes overlapping region $\mathsf{Area}_{\mathsf{overlap}} \geq \alpha$. The number of feasible configurations of (w, h), in terms of the sum of integer solutions, can be calculated as follows:

$$\#\mathsf{config} = \sum_{w=0}^{\left\lfloor \mathsf{W}_t - \frac{\alpha}{\mathsf{H}_t} \right\rfloor} \left[\lfloor \Phi(w) \rfloor + 1 \right]. \tag{9}$$

Finally, the probability that the intersection $\mathsf{Area}_{\mathsf{overlap}}$ of the cutout region and trigger region is greater than α is:

$$p_f^{single} \geq \Pr\left(\mathsf{Area}_{\mathsf{overlap}} \geq \alpha\right) = \frac{\sum_{w=0}^{\left\lfloor W_t - \frac{\alpha}{H_t} \right\rfloor} [\lfloor \Phi(w) \rfloor + 1]}{(\mathsf{H} - \mathsf{H}_c + 1)(\mathsf{W} - \mathsf{W}_c + 1)},$$

where the denominator $(\mathsf{H} - \mathsf{H}_c + 1)(\mathsf{W} - \mathsf{W}_c + 1)$ represents the total number of distinct configurations in which the cutout region can manifest.

Remark. In Fig. 11, we illustrate the movement of the cutout region when the trigger region is positioned at the bottom-left corner. In addition, in Fig. 12, we provide an example of computing Eq. (9), where each point in the area represents a feasible integer solution of (w, h), the bottom-left corner of cutout region. Similarly, as discussed in Sec. 5 in our paper, we visualize the results of Lemma 1.

Given that the failure probability of a single trigger pattern is bounded in Lemma 1, we can extend our analysis to repetitive patterns and explore methods to bound their failure probability in Lemma 2.

C.3.2 Proof of Lemma 2.

As mentioned in [10], cutout [6] operations and other strong data augmentations also destroy low frequency in SSL training. Here, we only consider the high-frequency components from $\mathsf{M} + 1$ to $\mathsf{N} - 1$, the total number of changeable coefficients is $\mathsf{N} - \mathsf{M} - 1$. The first combination number should be $\binom{\mathsf{N}-\mathsf{M}-1}{\beta}$, and it's important to account for numbers greater than β as well. The proof is concluded.

References

[1] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019. 1

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *NIPS*, 32, 2019. 1

[3] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *CVPR*, pages 6228–6237, 2018. 5

[4] Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *ICML*, pages 675–685. PMLR, 2019. 5

[5] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley, 2012. 4

[6] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 6

[7] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *ICLR*, 2020. 2

[8] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International symposium on research in attacks, intrusions, and defenses*, pages 273–294. Springer, 2018. 2

[9] Claude E Shannon et al. Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec*, 4(142-163):1, 1959. 5

[10] Virat Shejwalkar, Lingjuan Lyu, and Amir Houmansadr. The perils of learning from unlabeled data: Backdoor attacks on semi-supervised learning. In *ICCV*, pages 4730–4740, 2023. 3, 6

[11] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NIPS*, 33:596–608, 2020. 1

[12] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846. IEEE, 1998. 4

[13] Yidong Wang, Hao Chen, Yue Fan, Wang Sun, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, et al. Usb: A unified semi-supervised learning benchmark for classification. *NIPS*, 35:3938–3961, 2022. 1, 2

[14] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *NIPS*, 33:6256–6268, 2020. 1

[15] Zhicong Yan, Jun Wu, Gaolei Li, Shenghong Li, and Mohsen Guizani. Deep neural backdoor in semi-supervised learning: Threats and countermeasures. *IEEE Transactions on Information Forensics and Security*, 16:4827–4842, 2021. 3

[16] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 1

[17] Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of backdoors via implicit hypergradient. In *ICLR*, 2021. 3

[18] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *NIPS*, 34:18408–18419, 2021. 1

[19] George Zhang, Jingjing Qian, Jun Chen, and Ashish Khisti. Universal rate-distortion-perception representations for lossy compression. *NIPS*, 34:11517–11529, 2021. 5