Memorization Weights for Instance Reweighting in Adversarial Training

Jianfu Zhang[1], Yan Hong[2], and Qibin Zhao[1] *

[1]RIKEN AIP
[2]Shanghai Jiao Tong University
jianfu.zhang@riken.jp, yanhong.sjtu@gmail.com, qibin.zhao@riken.jp

Abstract

Adversarial training is an effective way to defend deep neural networks (DNN) against adversarial examples. However, there are atypical samples that are rare and hard to learn, or even hurt DNNs' generalization performance on test data. In this paper, we propose a novel algorithm to reweight the training samples based on self-supervised techniques to mitigate the negative effects of the atypical samples. Specifically, a memory bank is built to record the popular samples as prototypes and calculate the memorization weight for each sample, evaluating the "typicalness" of a sample. All the training samples are reweigthed based on the proposed memorization weights to reduce the negative effects of atypical samples. Experimental results show the proposed method is flexible to boost state-of-the-art adversarial training methods, improving both robustness and standard accuracy of DNNs.

1 Introduction

DNNs have shown enormous success in the machine learning community. However, recent works indicate that DNNs are vulnerable to adversarial examples (Goodfellow, Shlens, and Szegedy 2015; Szegedy et al. 2014). Those adversarial examples are crafted by adding perturbations imperceptible to human eyes from the natural images, causing disastrous consequences. Since DNNs play critical roles in different artificial intelligence applications nowadays, affecting vast groups of people, developing robust DNNs against these adversarial attacks is an important topic.

To improve robustness of DNNs, existing works develop adversarial training (AT) (Madry et al. 2018) methods, which minimize DNN's error against adversarial perturbations by fitting DNNs on generated adversarial examples of all natural training data. Based on AT, many methods (Zhang et al. 2019; Wang et al. 2019; Zhang et al. 2020a; Wu, Xia, and Wang 2020; Chen et al. 2020a) are proposed to further enhance the robustness of DNNs. Although ATs can fit all training data as well as their adversarial counterparts to improve DNN's robustness, they still suffer from poor robustness performance (*i.e.*, accuracy on adversarial samples) on the test set (Tsipras et al. 2018; Schmidt et al. 2018). Adversarial examples generated by AT are stochastic and for worst-case scenario. DNNs need to be over-parameterized to fit these adversarial examples or compromise their performance.

Recent works (Xu et al. 2021; Feldman 2020; Sanyal et al. 2020; Dong, Liu, and Shang 2021) indicate that natural images and data distributions have a significant fraction of *atypical* examples which are *distinct from the subpopulations* and have *rare frequencies to appear* in the training set. Experimental results show that if DNN is trained with atypical and typical samples, atypical samples prevent DNNs from memorizing typical samples and hurt DNN's robust performance. These discoveries demonstrate that if no more training samples are provided, it is beneficial to remove or apply less weight on the atypical samples to reduce their negative effects for training robust DNNs.

In this paper, we propose a novel reweight method by leveraging the memorization effect in AT to distinguish atypical samples from typical samples adaptively with increasing model generalization ability. In detail, we adopt self-supervised learning techniques to learn a discrete codebook (Esser, Rombach, and Ommer 2021; van den Oord, Vinyals, and Kavukcuoglu 2017). The codebook receives embeddings extracted by DNN from input samples and records the most popular embeddings, which can be recognized as prototypes of typical samples. Typical samples are hard to perturb due to their large populations. With the help of codebook, each sample is assigned with an adaptive weight calculated based on how close each sample is to its nearest neighbor code in the codebook. Larger weights are assigned to the samples near the nearest neighbor code, which can be recognized as typical samples, otherwise, the samples which are far away from the nearest neighbor codes are atypical samples and assigned with lower weights. Our proposed reweight method is fast and flexible to boost existing adversarial training algorithms. Compare to existing reweight methods (Zhang et al. 2020b; Wang et al. 2020), our proposed method is more stable to defend strong and adaptive attacks. Experimental results show that the proposed method is effective in defending different backbones of DNNs on different datasets.

2 Preliminary and Related Works

We focus on image classification task in this paper. For a C-class classification problem, we consider a training dataset

*Corresponding author

$\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ independently drawn from a distribution $\mathcal{D}$ and a DNN $f_\theta(\mathbf{x})$ parameterized by θ. $f_\theta(\mathbf{x})$ predicts the label of an input data via $f_\theta(\mathbf{x}) = \arg\max_k \mathbf{p}_k(\mathbf{x}; \theta)$, with $\mathbf{p}_k(\mathbf{x}; \theta)$ being the predicted probability (softmax on logits) of the k-th class.

2.1 Adversarial Training (AT)

AT's objective functions imply the optimization of adversarially robust networks, with one *inner step* generating adversarial data and one *outer step* minimizing loss on the generated adversarial data. Let $(\mathcal{X}, d_\infty)$ denote the input feature space $\mathcal{X}$ with the infinity distance metric $d_\infty(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}^{'}\|_\infty$, $\mathcal{B}_\epsilon(\mathbf{x}) = \{\mathbf{x}' \in \mathcal{X} \mid d_\infty(\mathbf{x}, \mathbf{x}') \leq \epsilon\}$ with $\epsilon > 0$ be the closed ball of radius ϵ centered at x in $\mathcal{X}$, and dataset $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{0, 1, ..., C-1\}$. The objective function of standard adversarial training (AT) (Madry et al. 2018) is:

$$\underbrace{\arg\min_{f_\theta \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^n \ell_{out}\left(f_\theta\left(\hat{\mathbf{x}}_i\right), y_i\right),}_{\text{outer step}}$$
$$\text{s.t. } \hat{\mathbf{x}}_i = \underbrace{\arg\max_{\hat{\mathbf{x}} \in \mathcal{B}_\epsilon(\mathbf{x}_i)} \ell_{in}\left(f_\theta(\hat{\mathbf{x}}), y_i\right),}_{\text{inner step}} \tag{1}$$

and $\hat{\mathbf{x}}$ is the most "adversarial" data within the ϵ-ball centered as $\mathbf{x}$, and the loss functions $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}$ are specific loss functions (*e.g.*, cross-entropy loss). AT employs the most adversarial data generated according to the inner step for updating the current model by outer step. The PGD method (Madry et al. 2018) is commonly adopted for the inner step: for a natural example $\mathbf{x}_i$, it starts with random noise and repeatedly computes:

$$\delta_i^{(t)} \leftarrow$$
$$\text{Proj}\left[\delta_i^{(t-1)} + \alpha \,\text{sign}\left(\nabla_\theta \ell\left(\mathbf{x}_i + \delta_i^{(t-1)}, y_i; \theta\right)\right)\right] \tag{2}$$

with the clipping operation Proj such that $\delta_i^{(t)}$ is always in the ϵ bound, and sign the signum function. Due to non-convexity, we approximate the optimal solution iteratively by $\delta_i^{(T)}$ with T being the maximally allowed steps. Accordingly, $\delta_i^{(T)}$ is viewed as the perturbation for the most adversarial example, *i.e.*, $\hat{\mathbf{x}} = \mathbf{x}_i + \delta_i^{(T)}$.

Intuitively, AT corresponds to *the worst-case robust optimization*, continuously augmenting the training dataset with adversarial variants that highly confuse the current model. Therefore, it is a practical learning framework to alleviate the impact of adversarial attacks. However, it still results in unsatisfactory model performance regarding adversarial robustness, due to:

Noises: Adversarial training generates adversarial examples stochastically because of random initialization. The model is not apt and not necessary to memorize such very noisy information.

Training-friendly: AT has an overwhelming smoothing effect in fitting highly adversarial examples (Zhang et al. 2019), and thus consumes a large model capacity to learn from individual data points.

To alleviate these problems, SWA (Chen et al. 2020b) and AWP (Wu, Xia, and Wang 2020) are proposed which smooth the weights of DNNs in AT. In (Zhang et al. 2020a), Friendly Adversarial Training (FAT) is proposed which stops PGD early to avoid generating adversarial data which are too strong to train DNNs for enhanced robustness.

2.2 Instance Reweighting for Adversarial Training

Instance reweighting methods are used to improve adversarial training. In the framework of reweighting, weights are assigned to the loss associated with individual samples. The goal of reweighting is to minimize the empirical weighted training loss, where objective function Eqn. 1 is modified as,

$$\min_{f_\theta \in \mathcal{F}} \sum_i \omega_i \ell_{out}\left(f_\theta\left(\hat{\mathbf{x}}_i\right), y_i\right)$$
$$\text{s.t. } \omega_i \geq 0 \text{ , and } \sum_i \omega_i = 1.$$

The constraints are required since the risk after weighting is consistent with the original one without weighting.

Previous works in instance reweighting for adversarial robustness (Zhang et al. 2020b; Wang et al. 2020; Zeng et al. 2021) largely focus on designing heuristic functions of various notions of margins to use for the sample weight w_i to evaluate the "difficulty" of the sample.

A reweighting method for adversarial training typically (I) should focus on misclassified samples and (II) should favour samples to the decision boundary.

(Wang et al. 2019) shows that we can apply more weights to the incorrect samples for better performance. (Zhang et al. 2020b) claimed that training examples should have unequal significance in AT, and proposed the Geometry-Aware Instance-Reweighted Adversarial Training (GAIRAT). They revealed that data near decision boundary are much vulnerable to be attacked and require large weights. Margin-Aware Instance Reweighting Learning (MAIL) (Wang et al. 2020) relies on the local linearity of ReLU networks and the fact that for samples near the margin, the relative scale of predicted class-likelihoods directly corresponds to the distance to the decision boundary.

Although these techniques improve adversarial training over regular AT, they still have problems: *Vulnerability:* Existing reweighting methods mostly depend on *supervised metrics*. They may be biasing the model towards certain samples by re-scaling the loss and lead the model to be susceptible to attacks that scale the logits. In (Hitaj et al. 2021), a simple adaptive attack: logit scaling attack is proposed, which can easily decrease GAIRAT's robust accuracy.

2.3 Memorization Effects of Adversarial Training

There are several existing methods discuss the relationship between memorization effects and adversarial training. As well known, overparameterized DNNs have tremendous capacity to make them easy to perfectly fit the training dataset (Zhang et al. 2021; Neyshabur et al. 2017; Belkin et al.

2019; Bubeck and Sellke 2021; Zhang et al. 2017). While, this property of DNNs in practice cannot be well explained by standard theories about model generalization (Evgeniou, Pontil, and Poggio 2000) from the regularization perspective. At a high level, the standard theories underline the importance of regularization on the model complexity, to make DNNs to avoid overfitting or memorizing the outliers and nonuseful samples in the training data. In (Sanyal et al. 2020; Xu et al. 2021), the authors investigated memorization behavior in adversarial training and have pointed out that overfitting some rare examples in training can just improve the clean accuracy, and ignoring these rare examples helps in adversarial robustness. In (Dong, Liu, and Shang 2021), the authors show that using high-quality (not so rare or hard) examples can indeed improve both natural performance and adversarial robustness. In (Dong et al. 2021), a new mitigation algorithm was proposed to impede overconfident predictions by a regularization term for avoiding the excessive memorization of adversarial examples with possibly noisy labels. Unfortunately, distinguishing atypical samples from useful typical samples is not a free lunch. These methods need to build a validation set to evaluate the quality of the samples, which is cumbersome to calculate.

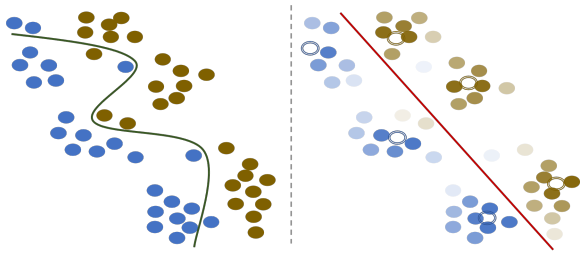
3 Memorization Weights (MeoW) for Adversarial Training

We define those *samples from main populations and with high frequencies to appear* as *typical samples*, otherwise are *atypical samples*. Data distributions of training samples and test samples may have shifts, but are commonly similar. Therefore, typical samples in training set are easy to be generalized on test set. When samples are easily manipulated by small perturbations to the input, they are referred as to be non-robust; likewise when they are not easily manipulated, they are said to be robust. By definition, typical samples have major populations which are easy to be memorized by DNNs and hard to be manipulated due to the large population. Hence, features of typical samples are more robust than those of atypical samples.

Based on the above assumptions, we develop Memorization Weights (MeoW) to explicitly evaluate the "typicalness" of natural samples. Considering that deep model has sufficient capacity to memorize the entire training samples either typical or atypical samples. We design a capacity-restricted memory bank to guide deep model to memorize robust features by relying more overly on typical adversarial samples than atypical adversarial samples. Specifically, we firstly construct a memory bank to remember and store robust features, and update the memory bank continuously with improvement of model generalization ability on incorrect samples. Then, based on memory bank, we assign adaptive weights for each adversarial samples for robust training without overfitting on atypical samples.

3.1 Memory Bank Construction

We remove the last layer of classifier $f_\theta(\cdot)$ as feature extractor $g_\theta(\cdot)$, which is used to extract embeddings of natural samples $\mathbf{x}_i$ (*resp.,* adversarial sample $\hat{\mathbf{x}}_i$) as $\mathbf{F}_i$ (*resp.,* $\hat{\mathbf{F}}_i$)



(a) Complex decision boundary before reweighting

(b) Robust decision boundary after reweighting

Figure 1: Motivation of our approach. *Left*: training samples from two classes are assigned with same weights without difference among those samples, leading to complex decision boundary and enforce the model to memorize all training samples. *Right*: training samples (denoted as colored •) are assigned with different weights (samples with larger weights are colored with less transparent dot), the learned clusters are denoted as colored circle ○. The samples closer to clusters are regarded as more typical samples which provide robust features for learning essential distribution of training data.

$\in \mathbb{R}^d$, where d is the feature dimension. We build a memory bank to adaptively learn the typical sample prototypes. There are many works study how to build memory banks for DNNs (Sun et al. 2021; Wu et al. 2018; Song et al. 2019), and we apply the method in (van den Oord, Vinyals, and Kavukcuoglu 2017; Esser, Rombach, and Ommer 2021). We learn a codebook $\mathcal{B} = \{\mathbf{e}_k|_{k=1}^N\} \in \mathbb{R}^{N\times d}$ where N is the number of items in the codebook. Each code $\mathbf{e}_k \in \mathbb{R}^d$ in codebook $\mathcal{B}$ can be regarded as a set of stable modes from distribution for training samples. In each batch, we use embeddings $\{\mathbf{F}\}$ to update the codebook. Similar to (van den Oord, Vinyals, and Kavukcuoglu 2017), we quantize embeddings $\mathbf{F}_i$ by looking up its nearest neighbor in the codebook $\mathcal{B}$. Specifically, each $\mathbf{F}_i$ can be replaced with its nearest item $\mathbf{e}_k$ in the codebook $\mathcal{B}$ by comparing the distances between $\mathbf{F}_i$ and all items in $\mathcal{B}$. This process can be represented by optimizing:

$$\ell_m = \sum_i \| \mathrm{sg}[\mathbf{F}_i] - \tilde{\mathbf{F}}_i \|_2^2, \tag{3}$$

where s.t. $\tilde{\mathbf{F}}_i = \mathbf{e}_k, k = \arg\min_j \|\mathbf{F}_i - \mathbf{e}_j\|_2$, and $sg[\cdot]$ represents the stop-gradient operation and $\|\cdot\|_2^2$ denotes L_2 loss. The codebook is trained concurrently with the classifier, which reveals the prototype representations of the classifier adaptively during the training phase. Please note, the memory loss does not return gradients to DNNs. The memory loss will enhance the codes which appear more frequently in the feature space, which is closely related to the "typical" sample concept.

3.2 Weight Assignment for Adversarial Training

With updated codebook $\mathcal{B}$, we can measure the weights of embeddings $\{\mathbf{F}_i\}$ for different samples. Items $\{\mathbf{e}_k\}$ in codebook $\mathcal{B}$ represents the stable major population of training

features, *i.e.*, typical samples. Given embeddings $\{\mathbf{F}\}$, we can calculate the Euclidean distance $d(\mathbf{F}, \mathcal{B})$ between $\mathbf{F}$ and its nearest code $\mathbf{e}_k$ in codebook $\mathcal{B}$. The distance $d(\mathbf{F}, \mathcal{B})$ indicates the "typicalness" of embedding $\mathbf{F}$. The smaller the distance $d(\mathbf{F}, \mathcal{B})$, the more typical the embedding $\mathbf{F}$ is. In this way, we define two opposite weights of samples:

$$\begin{aligned} w^-(\mathbf{F}) &= \exp(-d(\mathbf{F}, \mathcal{B})/T), \\ w^+(\mathbf{F}) &= \exp(d(\mathbf{F}, \mathcal{B})/T) \end{aligned} \tag{4}$$

where $\exp(\cdot)$ is the natural exponential function and $T = 10$ is a constant hyperparameter as temperature. Intuitively, w^- will suppress the effect of atypical samples while w^+ will amplify the effect of atypical samples.

3.3 Objective Functions

In (Feldman and Zhang 2020), the "long tail theory" shows that atypical samples in natural image distribution help generalization even with significant memorization. However, the long-tail distributions of adversarial examples may be harmful to DNNs, which is opposite to the phenomenon for natural examples. Different from natural samples, atypical adversarial samples contain more noisy information and may be more training-unfriendly, as we have mentioned in Section 2.1. We choose to apply w^- on adversarial samples and w^+ on natural samples for MeoW. Here, we demonstrate how to apply the proposed reweight method on AT (Madry et al. 2018) and TRADES (Zhang et al. 2019).

For AT, the objective function (outer step in Eqn. (1)) changes to:

$$\begin{aligned} &\ell_{\mathrm{AT}}^{\mathrm{MeoW}}\left(f_\theta\left(\hat{\mathbf{x}}_i\right), y_i\right) \\ &= \alpha \cdot w^+(\mathbf{F}) \cdot \mathrm{CE}\left(\mathbf{p}\left(\mathbf{x}; \boldsymbol{\theta}\right), y\right) \\ &+ (1-\alpha) \cdot w^-(\hat{\mathbf{F}}) \cdot \mathrm{CE}\left(\mathbf{p}\left(\hat{\mathbf{x}}; \boldsymbol{\theta}\right), y\right). \end{aligned} \tag{5}$$

For TRADES, the objective function changes to:

$$\begin{aligned} &\ell_{\mathrm{TRADES}}^{\mathrm{MeoW}}\left(f_\theta\left(\hat{\mathbf{x}}_i\right), y_i\right) \\ &= w^+(\mathbf{F}) \cdot \mathrm{CE}\left(\mathbf{p}\left(\mathbf{x}; \boldsymbol{\theta}\right), y\right) \\ &+ \lambda \cdot w^-(\hat{\mathbf{F}}) \cdot \mathrm{KL}\left(\mathbf{p}\left(\mathbf{x}; \boldsymbol{\theta}\right) \| \mathbf{p}\left(\hat{\mathbf{x}}; \boldsymbol{\theta}\right)\right), \end{aligned} \tag{6}$$

where α and λ are trade-off hyperparameters.

3.4 Implementation Details and Discussions

Latency of MeoW: In Table 1, we report the model complexity and average time cost for each training batch of MeoW and baseline backbones. $N = 2048$ in Table 1. *We can see the additional latency of MeoW is minor (about 0.01s per batch) for training. It is also worth noting that MeoW does not introduce any latency during the test phase.*

Embedding Selection: As mentioned in Section 2.2, reweighting methods focus on samples which are misclassified or near decision boundary. MeoW can also achieve this goal by selecting a subset of misclassified natural/adversarial embeddings from each batch for calculating the memory loss in Eqn. 3. Given natural sample $\mathbf{x}_i$ (*resp.,* adversarial sample$\hat{\mathbf{x}}_i$) with predicted label $\tilde{y}_i$ (*resp.,* $\hat{y}_i$). If predicted $\tilde{y}_i = y_i$ (*resp.,* $\hat{y}_i = y_i$), the corresponding input x_i (*resp.,* $\hat{\mathbf{x}}_i$) is referred to as correct natural (*resp.,* adversarial) sample, otherwise the input is referred to as wrong natural (*resp.,* adversarial) sample. We divide natural features $\{\mathbf{F}_i\}$ (*resp.,* adversarial features $\{\hat{\mathbf{F}}_i\}$) into two subsets consisting of correct natural features $\{\mathbf{F}_i^C\}$ (*resp.,* correct adversarial features $\{\hat{\mathbf{F}}_i^C\}$) and wrong natural features $\{\mathbf{F}_i^W\}$ (*resp.,* incorrect adversarial features $\{\hat{\mathbf{F}}_i^W\}$). We select wrong natural features $\{\mathbf{F}_i^W\}$ and wrong adversarial features $\{\hat{\mathbf{F}}_i^W\}$ to update our codebook $\mathcal{B}$. In Appendix, we will discuss the effect of embedding selection for MeoW in detail.

Network	Time	Parameters(M)
ResNet18	0.597	11.2
+MeoW	0.608	12.2
WRN-34-10	1.121	46.2
+MeoW	1.129	47.5

Table 1: Batch train time and number of model parameters.

Training Process: For each batch of training, we first calculate the adversarial examples based on Eqn. 2. Then we calculate the embeddings $\mathbf{F}$ from the codebook and calculate weights in Eqn. 4. After that we will update the codebook and the network respectively. Please refer to the Appendix for the pseudocodes of the whole algorithm.

Remarks: Generally, MeoW will force the model to pay more attention to the adversarial samples which appear more frequently in the memory bank. This will make the adversarial training process stable and smooth. The capacity of the codebook is limited, hence, DNN is regularized to ignore the noise of adversarial samples caused by random initialization or optimization. Intuitively, extreme training-unfriendly adversarial samples are out-of-distribution samples. Hence, they are suppressed by MeoW. After adopting reweighted scheme, the model would be focused on memorizing stable features from typical samples instead of relying overly on atypical samples, in this way, the model is able to improve the robustness of predictions. It is also worth noting that MeoW also amplifies the weights of atypical natural samples, which avoids ignoring minor populations of the training dataset.

4 Experiments

To evaluate the effectiveness of MeoW, we conducted extensive experiments on three datasets, including CIFAR-10 (Krizhevsky et al. 2009), CIFAR-100 (Krizhevsky et al. 2009), and SVHN (Netzer et al. 2011). We choose popular PreActResNet (He et al. 2016) and Wide ResNet (Zagoruyko and Komodakis 2016) as backbones in our experiments.

4.1 Experimental Setup

Training Parameters We train the networks batch stochastic gradient descent with momentum 0.9, weight decay 5×10^{-4}. The training process takes 200 epochs. Batch size is 128, and the initial learning rate is 0.1. Learning rate

Datasets	Methods	Robustness		Natural	
		Best	Last	Best	Last
CIFAR-10	AT	52.81 ± 0.23	45.30 ± 0.40	85.30 ± 0.18	84.61 ± 0.17
	AT+MeoW	**55.07 ± 0.27**	**54.61 ± 0.18**	**85.33 ± 0.19**	**85.08 ± 0.22**
CIFAR-100	AT	27.09 ± 0.18	21.97 ± 0.31	56.13 ± 0.20	55.39 ± 0.23
	AT+MeoW	**30.47 ± 0.25**	**30.09 ± 0.29**	**56.39 ± 0.22**	**55.87 ± 0.25**
SVHN	AT	54.51 ± 0.13	45.39 ± 0.33	93.01 ± 0.22	90.64 ± 0.30
	AT+MeoW	**59.49 ± 0.20**	**57.23 ± 0.26**	**93.33 ± 0.13**	**92.68 ± 0.32**

Table 2: Average accuracy (%) of AT and AT+MeoW on different datasets with PreActResNet-18.

Defense	Natural	PGD-100	AutoAttack
AT (Madry et al. 2018)	86.07 ± 0.16	55.72 ± 0.28	52.51 ± 0.22
AT+MeoW	**87.03 ± 0.18**	**57.73 ± 0.30**	**53.92 ± 0.24**
TRADES (Zhang et al. 2019)	84.37 ± 0.20	55.93 ± 0.25	53.00 ± 0.27
TRADES+MeoW	**85.07 ± 0.19**	**58.72 ± 0.23**	**55.60 ± 0.29**
FAT (Zhang et al. 2020a)	87.93 ± 0.23	55.30 ± 0.31	52.09 ± 0.23
FAT+MeoW	**88.10 ± 0.19**	**57.82 ± 0.25**	**53.80 ± 0.21**
AWP (Wu, Xia, and Wang 2020)	85.22 ± 0.19	58.78 ± 0.30	56.03 ± 0.29
AWP+MeoW	**85.93 ± 0.19**	**59.42 ± 0.28**	**56.61 ± 0.22**
RST (Carmon et al. 2019)	89.67 ± 0.16	62.03 ± 0.18	59.41 ± 0.20
RST+MeoW	**90.01 ± 0.14**	**63.79 ± 0.21**	**60.70 ± 0.19**

Table 3: Average accuracy (%) of different methods on CIFAR10 with WideResNet of 5 runs.

is divided by 10 after the 100-th and 150-th epoch. To some extent, this setup can alleviate the impact of adversarial overfitting (Pang et al. 2020; Wang et al. 2019). All training process is conducted on nVidia A100 GPU.

Hyperparameters For the proposed MeoW, we set the size of the codebook $N = 2048$ and the temperature $T = 10$ in Eqn. (4). The trade-off parameter for AT α in Eqn. (5) was set to 0.2 and for TRADES λ in Eqn. (6) it was 6. For the detailed ablative studies of these hyperparameters, please refer to Appendix.

Robustness Evaluation We evaluated our methods and baselines using the standard accuracy on natural test data (NAT) and the adversarial robustness based on several representative attack methods, including the PGD method with 100 iterations (Madry et al. 2018), and AutoAttack (AA) (Croce and Hein 2020). We evaluate all the methods on white-box setting that all these methods have full access to the model parameters and all the attacks are constrained by the same perturbation limit as above. Black-box attack methods (Bai et al. 2020; Chen et al. 2020a; Li et al. 2020) are relatively easy to defense (Chakraborty et al. 2018), so here we do not focus on them.

4.2 Comparing Vanilla AT and AT+MeoW

In this part, we conduct a case study on vanilla AT and AT+ MeoW across three benchmark datasets (CIFAR-10 (Krizhevsky et al. 2009), CIFAR-100 (Krizhevsky et al. 2009), and SVHN (Netzer et al. 2011)) and L_∞ threat model using PreActResNet-18 for 200 epochs. We follow the same settings in (Rice, Wong, and Kolter 2020): for L_∞ threat model, $\epsilon = 8/255$, step size is $1/255$ for SVHN, and $2/255$ for CIFAR-10 and CIFAR-100. The training/test attacks are PGD10/20. The test robustness is reported in Table 2, where "Best" means the highest robustness that ever achieved at different checkpoints for each dataset and threat model while "Last" means the robustness at the last epoch checkpoint.

We can see that AT+ MeoW consistently improves the test robustness for all cases. It indicates that MeoW is generic and is applicable for different datasets to reduce the negative effects of "atypical" samples based on learned memory bank.

4.3 Applying MeoW to Other Defense

In this part, we evaluate the robustness of our proposed MeoW on CIFAR-10 (Krizhevsky et al. 2009) to benchmark the state-of-the-art robustness against white-box attacks. Two types of adversarial training methods are considered here: One is only based on original data: 1) AT (Madry et al. 2018); 2) TRADES (Zhang et al. 2019), 3) FAT (Zhang et al. 2020a), and 4) AWP (Wu, Xia, and Wang 2020). The other type is RST (Carmon et al. 2019) using additional data.

For CIFAR-10 under L_∞ attack with ϵ = 8/255, we train WideResNet 28-10 for RST (Carmon et al. 2019), while WideResNet 34-10 for remaining adversarial training methods, following their original papers. All defenses are trained

Baseline	Reweight	Natural	PGD-100	AutoAttack
AT	Vanilla	86.07 ± 0.16	55.72 ± 0.28	52.51 ± 0.22
	GAIRAT (Zhang et al. 2020a)	86.20 ± 0.46	57.46 ± 0.48	41.39 ± 0.17
	MAIL (Wang et al. 2020)	84.98 ± 0.36	57.51 ± 0.36	47.29 ± 0.21
	MeoW	**87.03 ± 0.18**	**57.73 ± 0.30**	**53.92 ± 0.24**
TRADES	Vanilla	84.37 ± 0.20	55.93 ± 0.25	53.00 ± 0.27
	GAIRAT (Zhang et al. 2020a)	85.01 ± 0.30	58.49 ± 0.51	48.71 ± 0.20
	MAIL (Wang et al. 2020)	84.05 ± 0.37	57.74 ± 0.59	52.81 ± 0.22
	MeoW	**85.07 ± 0.19**	**58.72 ± 0.23**	**55.60 ± 0.29**

Table 4: Average accuracy (%) of reweight methods on CIFAR10 with WRN-34-10.

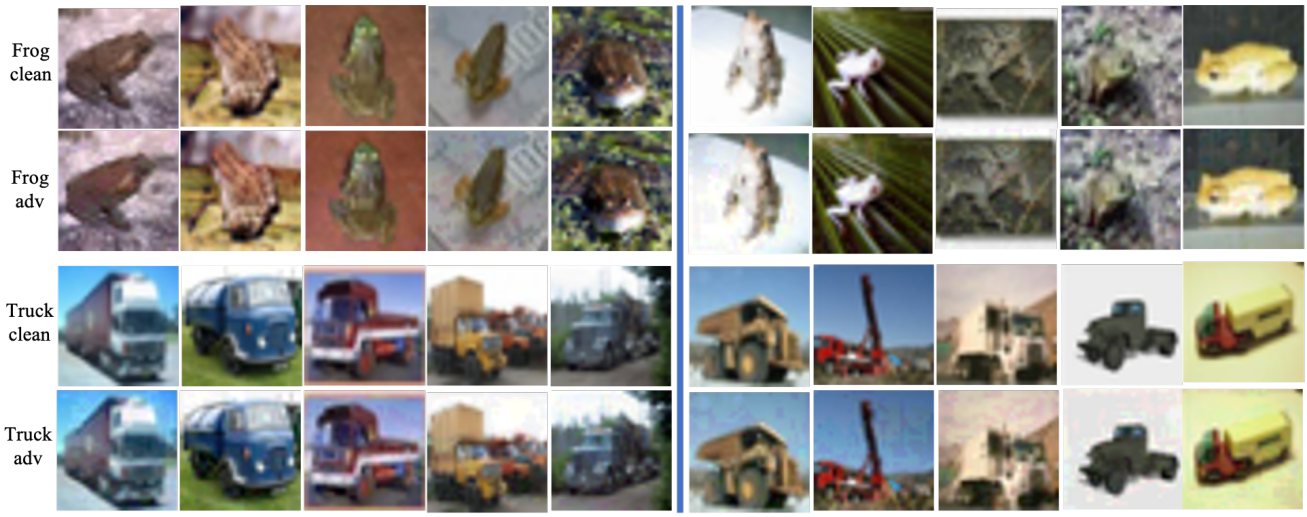


Figure 2: Visualization of natural and adversarial sample with learned weights on CIFAR-10. In the left column (*resp.*, right) column, we show five training sample with the highest (*resp.*, lowest) weights obtained by our method.

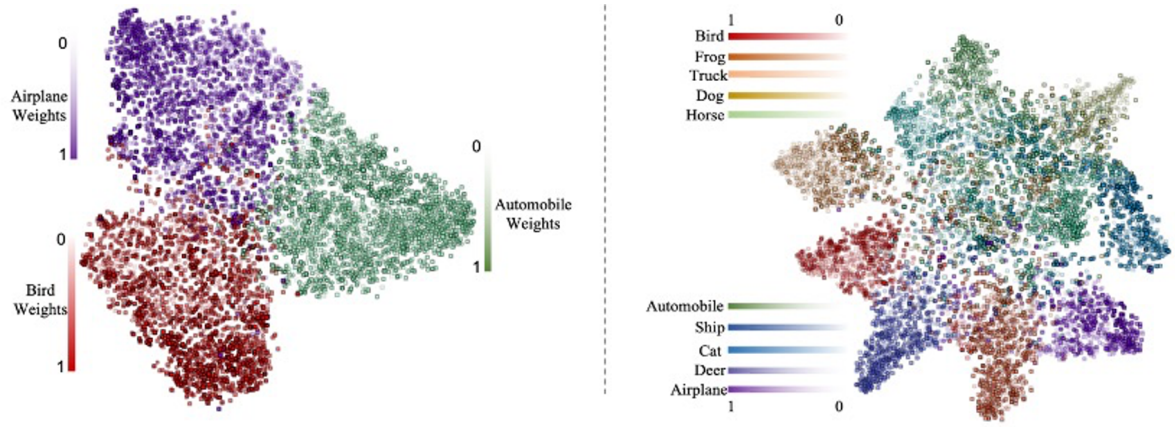
for 200 epochs using SGD with momentum 0.9, weight decay 5×10^{-4}, and an initial learning rate of 0.1 that is divided by 10 at the 100-th and 150-th epoch. Simple data augmentations such as 32×32 random crop with 4-pixel padding and random horizontal flip are applied. The training attack is PGD-10 with step size $2/255$. For AWP (Wu, Xia, and Wang 2020), we set $\gamma = 5 \times 10^{-3}$. Other hyper-parameters of the baselines are configured as per their original papers.

Table 3 reports the "best" test robustness (the highest robustness ever achieved at different checkpoints for each defense against each attack) against white-box and black-box attacks. "Natural" denotes the accuracy on natural test examples. First, we test PGD-100 (Madry et al. 2018). MeoW almost improves the robustness of state-of-the-art methods against all types of attacks. Second, we test MeoW against AutoAttack (AA) (Croce and Hein 2020), which is a strong and reliable attack to verify the robustness via an ensemble of diverse parameter-free attacks. Compared with original results of those state-of-the-art methods, MeoW can further boost their robustness, ranking the first on both with and without additional data. This verifies that MeoW is flexible to be integrated with existing defense methods (*resp.,* AT, TRADES, FAT, AWP, and RST) to further improve adversarial robustness.

4.4 Comparison with Other Reweighting Methods

In this part, we compare MeoW with other reweighting methods using the same setting as Section 4.3. For comparison, we firstly report the results without any reweighting strategy referred to as "Vanilla" in table 4. Then, we report the results of competitive reweighting baselines including GAIRAT (Zhang et al. 2020b), and MAIL (Wang et al. 2020) in table 4. As we can see, the superiority of our method is apparent. Comparing the results between "MeoW+AT" (*resp.,* "MeoW + TRADES") and "AT" (*resp.,* "TRADES", "MART", "MAIL"), MeoW can lead to promising robustness, especially for the strong ensemble attack AA. The reason is that "MeoW" can measure the robustness of training samples by calculating the distance between its feature and its nearest robust cluster in learned codebook. As a result, it can help accurately assign high weights for those robust samples during training.

In Appendix, we also show MeoW is more robust to state-of-the-art reweighting methods against logit-scale attack (Hitaj et al. 2021), which is specially designed to attack



(a) t-SNE of embeddings with different weights from three classes on CIFAR-10 dataset.

(b) t-SNE of embeddings with different weights from 10 classes on CIFAR-10 dataset.

Figure 3: Illustration of our motivation of designing reweighting scheme. In subfigure (a) (*resp.,* subfigure (b)), we visualize the results of our reweighted scheme on natural and adversarial samples from selected three classes (*resp.,* all ten classes) on CIFAR-10 dataset as example. Natural training samples (*resp.,* adversarial training samples) are denoted with colored circles (○) (*resp.,* boxes (□)). The larger the weights, the more transparent the colored circles (○) (*resp.,* boxes (□)). In each class, samples with larger weights are gathered into several clusters, while sample with small weights are scattered in sparse space far from those clustered samples.

reweighting methods.

4.5 Analysis of the Learned Weight

Examples with Small/Large Weights: We show some examples training natural samples (*resp.,* adversarial samples) from CIFAR10 dataset in Figure 2, those example samples are selected based on the weights learned from our proposed method. In Figure 2, natural (*resp.,* adversarial) samples with the highest weights are shown in the left subfigure, we can see that the objects in training samples with higher weights are clear and the background information is simple, (*i.e.,* the whole body of frogs and trucks in the left subfigure can be captured by camera and its bodies account for a large proportion in the image). In contrast, samples with small weights in the right subfigure are far different from large-weighted samples. Some of them have unusually colored skin (*i.e.,* frog in 7-th and 10-th column), which are quite different from usual samples. For trucks with small weights (*i.e.,* 8-th and 9-th column) seem like toys, which is challenging for model to classify them into real trucks.

t-SNE Visualization of the Learned Representation: To further demonstrate our motivation, on CIFAR-10 dataset with our proposed reweight scheme, we visualize the t-SNE of reweighted embeddings of natural samples $\mathbf{F}$ and embeddings of adversarial samples $\hat{\mathbf{F}}$ from three/ten class samples in Figure 3. From the figure, we can find that the natural samples (*resp.,* adversarial samples) with larger weights (colored with less transparent ○ (*resp.,* □)) are gathered into several clusters, while natural samples (*resp.,* adversarial samples) with small weights (colored with more transparent ○ (*resp.,* □)) are scattered in sparse space far from those clusters. Figure 3 can indicate that the learned larger weights are assigned to typical samples, whose features are gathered into stable clusters in feature space, while those samples with smaller weights are scattered in more parse feature space.

5 Conclusions

In this paper, we leverage memorization effect in DNNs to construct a memory bank using vector quantization techniques. Learned memory bank can guide model to distinguish typical samples from atypical samples for robust training by assigning larger weight to typical samples while alleviating overly relying on atypical samples. Extensive experiments on various datasets with different backbones demonstrate the effectiveness of our proposed instance reweighting method. It is interesting to consider designing adaptive attacks to MeoW based on the codebook it learned. However, the codebook is learned adaptively during the training process and the effect is complex. Also, it is hard to design attack methods against reweighting methods. We take designing attacks against MeoW as future work.

Acknowledgments

The work is partially supported by JSPS KAKENHI Grant Number 20H04249, 20H04208.

References

Bai, Y.; Zeng, Y.; Jiang, Y.; Wang, Y.; Xia, S.-T.; and Guo, W. 2020. Improving query efficiency of black-box adversarial attack. In *ECCV*.

Belkin, M.; Hsu, D.; Ma, S.; and Mandal, S. 2019. Reconciling modern machine-learning practice and the classi-

cal bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32): 15849–15854.

Bubeck, S.; and Sellke, M. 2021. A Universal Law of Robustness via Isoperimetry. *arXiv preprint arXiv:2105.12806*.

Carmon, Y.; Raghunathan, A.; Schmidt, L.; Duchi, J. C.; and Liang, P. S. 2019. Unlabeled data improves adversarial robustness. *NeurIPS*.

Chakraborty, A.; Alam, M.; Dey, V.; Chattopadhyay, A.; and Mukhopadhyay, D. 2018. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

Chen, T.; Liu, S.; Chang, S.; Cheng, Y.; Amini, L.; and Wang, Z. 2020a. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *ICCV*.

Chen, T.; Zhang, Z.; Liu, S.; Chang, S.; and Wang, Z. 2020b. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*.

Croce, F.; and Hein, M. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*.

Dong, C.; Liu, L.; and Shang, J. 2021. Data Profiling for Adversarial Training: On the Ruin of Problematic Data. *arXiv preprint arXiv:2102.07437*.

Dong, Y.; Xu, K.; Yang, X.; Pang, T.; Deng, Z.; Su, H.; and Zhu, J. 2021. Exploring Memorization in Adversarial Training. *arXiv preprint arXiv:2106.01606*.

Esser, P.; Rombach, R.; and Ommer, B. 2021. Taming transformers for high-resolution image synthesis. In *CVPR*.

Evgeniou, T.; Pontil, M.; and Poggio, T. 2000. Regularization networks and support vector machines. *Advances in computational mathematics*, 13(1): 1–50.

Feldman, V. 2020. Does learning require memorization? a short tale about a long tail. In *ACM SIGACT Symposium on Theory of Computing*.

Feldman, V.; and Zhang, C. 2020. What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation. In *NeurIPS*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. In Bengio, Y.; and LeCun, Y., eds., *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hitaj, D.; Pagnotta, G.; Masi, I.; and Mancini, L. V. 2021. Evaluating the robustness of geometry-aware instance-reweighted adversarial training. *arXiv preprint arXiv:2103.01914*.

Krizhevsky, A.; et al. 2009. Learning multiple layers of features from tiny images. *Citeseer*.

Li, J.; Ji, R.; Liu, H.; Liu, J.; Zhong, B.; Deng, C.; and Tian, Q. 2020. Projection & probability-driven black-box attack. In *ICCV*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning. *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; and Srebro, N. 2017. Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*.

Pang, T.; Yang, X.; Dong, Y.; Su, H.; and Zhu, J. 2020. Bag of Tricks for Adversarial Training. In *ICLR*.

Rice, L.; Wong, E.; and Kolter, Z. 2020. Overfitting in adversarially robust deep learning. In *ICML*.

Sanyal, A.; Dokania, P. K.; Kanade, V.; and Torr, P. 2020. How Benign is Benign Overfitting? In *ICLR*.

Schmidt, L.; Santurkar, S.; Tsipras, D.; Talwar, K.; and Madry, A. 2018. Adversarially Robust Generalization Requires More Data. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *NeurIPS*.

Song, J.; Yang, Y.; Song, Y.-Z.; Xiang, T.; and Hospedales, T. M. 2019. Generalizable person re-identification by domain-invariant mapping network. In *CVPR*.

Sun, G.; Hua, Y.; Hu, G.; and Robertson, N. 2021. MAMBA: Multi-level Aggregation via Memory Bank for Video Object Detection. In *AAAI*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In Bengio, Y.; and LeCun, Y., eds., *ICLR*.

Tsipras, D.; Santurkar, S.; Engstrom, L.; Turner, A.; and Madry, A. 2018. Robustness May Be at Odds with Accuracy. In *ICLR*.

van den Oord, A.; Vinyals, O.; and Kavukcuoglu, K. 2017. Neural Discrete Representation Learning. In *NeurIPS*.

Wang, Q.; Liu, F.; Han, B.; Liu, T.; Gong, C.; Niu, G.; Zhou, M.; and Sugiyama, M. 2020. Probabilistic Margins for Instance Reweighting in Adversarial Training. *NeurIPS*.

Wang, Y.; Zou, D.; Yi, J.; Bailey, J.; Ma, X.; and Gu, Q. 2019. Improving adversarial robustness requires revisiting misclassified examples. In *ICLR*.

Wu, D.; Xia, S.-T.; and Wang, Y. 2020. Adversarial Weight Perturbation Helps Robust Generalization. *NeurIPS*.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*.

Xu, H.; Liu, X.; Wang, W.; Ding, W.; Wu, Z.; Liu, Z.; Jain, A.; and Tang, J. 2021. Towards the Memorization Effect of Neural Networks in Adversarial Training. *arXiv preprint arXiv:2106.04794*.

Zagoruyko, S.; and Komodakis, N. 2016. Wide Residual Networks. In *British Machine Vision Conference 2016*. BMVC.

Zeng, H.; Zhu, C.; Goldstein, T.; and Huang, F. 2021. Are Adversarial Examples Created Equal? A Learnable Weighted Minimax Risk for Robustness under Non-uniform Attacks. In *AAAI*.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2017. Understanding deep learning requires rethinking generalization. In *ICLR*.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; El Ghaoui, L.; and Jordan, M. 2019. Theoretically principled trade-off between robustness and accuracy. In *ICML*.

Zhang, J.; Xu, X.; Han, B.; Niu, G.; Cui, L.; Sugiyama, M.; and Kankanhalli, M. 2020a. Attacks which do not kill training make adversarial learning stronger. In *ICML*.

Zhang, J.; Zhu, J.; Niu, G.; Han, B.; Sugiyama, M.; and Kankanhalli, M. 2020b. Geometry-aware Instance-reweighted Adversarial Training. In *ICLR*.