Adversarial Training with Maximal Coding Rate Reduction

Hsiang-Yu Chu*, Hongbo Zhao*†, Markus Flierl*

* KTH Royal Institute of Technology
Electrical Engineering and Computer Science
Stockholm, Sweden

† Harbin Institute of Technology
Department of Control Science and Engineering
Harbin, China

***Abstract*—Deep convolutional networks can solve various complex tasks in the field of image processing. However, adversarial attacks have been shown to have the ability of fooling deep learning models. Adversarial training is one commonly used strategy to improve the robustness of deep learning models against adversarial examples, which is performed by incorporating adversarial examples into the training process. Traditionally, during this process, cross-entropy loss is used as the loss function. In order to improve the robustness of deep learning models against adversarial examples, we propose in this paper two new methods of adversarial training by applying the principle of Maximal Coding Rate Reduction (MCR^2). We evaluate the performance of different adversarial training methods by comparing the clean accuracy and adversarial accuracy. It is shown that adversarial training with the MCR^2 loss function yields a more robust network than the traditional adversarial training method. In our experiments, adversarial accuracies are improved by up to 10%. The two loss functions are discussed by using a model.**

***Index Terms*—Machine learning, deep neural networks, adversarial example, adversarial attack, adversarial training, quadratic similarity queries on compressed data**

I. Introduction

Despite the great performance of deep neural networks on classification tasks, adversarial attacks have been shown to have the ability of fooling deep learning models. An adversarial attack is accomplished by applying specially designed perturbations on the input image of a deep learning model. The noise is hardly noticeable to the human eye, but can fool classifiers into making wrong predictions [1].

Adversarial training is one commonly used strategy to improve the robustness of deep learning models against adversarial samples. It is shown that adversarial training can provide an additional regularization benefit beyond that provided by using dropout [2]. Adversarial training is performed via generating and then incorporating adversarial examples into the training process. Traditionally, during this process, cross-entropy is used as the loss function. Although adversarial training can indeed increase the robustness of a deep neural network against adversarial attacks, it is a simple defense method that can only provide limited increase in classification accuracy [2]. We question if there is a method to make adversarial training a better defense method against adversarial attacks.

In order to address the question, we apply the principle of Maximal Coding Rate Reduction to the process of adversarial training, which maximizes the coding rate difference between the whole data set and the sum of each individual class [3]. We summarize our contributions as follows: (1) We propose new adversarial training methods which utilize the principle of Maximal Coding Rate Reduction. (2) Our purposed methods can lead to a deep neural network with higher robustness against adversarial examples than the traditional adversarial training method.

II. Proposed Approach

A. Adversarial Training

Adversarial training is performed by using adversarial examples as training set for an existing neural network to improve its performance against adversarial attacks. Along with the discovery of the existence of adversarial examples [4] for deep neural networks, Szegedy et al. notice that back-feeding adversarial examples for training may lead to an improvement in classification accuracy on adversarial examples. Goodfellow et al. [2] successfully reduce the error rate of their model on adversarial examples from 89.4% to 17.9% through adversarial training.

Intuitively, during the adversarial training process, the same loss function as the loss function used in the original training process is used. In most cases as well as in this work, the loss function used for training the deep neural network models is the cross-entropy [5] loss function, which can be expressed as

$$H(P||Q) = -\sum_{l\in\mathcal{L}} P(l)\log_2 Q(l), \tag{1}$$

where $\mathcal{L}$ is the set of all possible labels of an input, P is the true probability of each class label, and Q is the probability of each class label predicted by the neural network model.

B. Maximal Coding Rate Reduction

Yu et al. [3] point out that the cross-entropy loss has two major limitations. The first limitation of the cross-entropy loss is the lack of interpretability. The second limitation of the cross-entropy loss is that it can fit mislabeled labels [6]. In order to address these two limitations, the Maximal Coding Rate Reduction (MCR^2) loss function has been introduced.

Let $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_m] \in \mathbb{R}^{n\times m}$ be the input to a deep neural network, where each sample $\boldsymbol{x} \in \mathbb{R}^n$ is drawn from a mixture of k distributions $\mathcal{D} = \{\mathcal{D}_j\}_{j=1}^k$. For each sample $\boldsymbol{x}$, there is a label $\boldsymbol{a} \in \mathbb{R}^k$. A deep neural network

classifier could be viewed as a two-stage process. The first stage is to select some features $\boldsymbol{z}$ given the input $\boldsymbol{x} \in \mathbb{R}^n$ via a mapping $f(\boldsymbol{x}, \theta) : \mathbb{R}^n \to \mathbb{R}^s$, where θ are the parameters of the network. For samples from different classes, the features should be discriminative. The second stage is to predict the label $\boldsymbol{a}$ given the features via a classifier $g(\boldsymbol{z})$.

Yu et al. [3] demonstrated that given a set of input data $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_m]$, a good set of features $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{z}_3, ..., \boldsymbol{z}_m]$ should have a large difference between the number of bits needed to encode $\boldsymbol{Z}$ and the total number of bits needed to encode all subsets $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \boldsymbol{Z}_3, ..., \boldsymbol{Z}_k$ of $\boldsymbol{Z}$, where each $\boldsymbol{Z}_j$ corresponds to each class j.

The coding rate needed to encode $\boldsymbol{Z}$ up to a precision ϵ (the discriminative loss) is

$$R(\boldsymbol{Z}, \epsilon) = \frac{1}{2} \log(\det(\boldsymbol{I} + \frac{s}{m\epsilon^2} \boldsymbol{Z}\boldsymbol{Z}^T)). \tag{2}$$

The coding rate needed to encode all subsets $\boldsymbol{Z}_1, \boldsymbol{Z}_2, \ldots, \boldsymbol{Z}_k$ of $\boldsymbol{Z}$ (the compression loss) is

$$R^c(\boldsymbol{Z}, \epsilon|\boldsymbol{\Pi}) = \sum_{j=1}^{k} \frac{tr(\boldsymbol{\Pi}_j)}{2m} \log(\det(\boldsymbol{I} + \frac{s}{tr(\boldsymbol{\Pi}_j)\epsilon^2} \boldsymbol{Z}\boldsymbol{\Pi}_j\boldsymbol{Z}^T)), \tag{3}$$

where $\boldsymbol{\Pi} = \{\boldsymbol{\Pi}_j \in \mathbb{R}^{m \times m}\}_{j=1}^k$ is a set of matrices $\boldsymbol{\Pi}_j$ whose element $\boldsymbol{\Pi}_j(i, i)$ indicates the probability of sample i belonging to subset j. Learning via maximal coding rate reduction (MCR^2) maximizes the difference between the discriminative loss and the compression loss

$$\Delta R(\boldsymbol{Z}, \boldsymbol{\Pi}, \epsilon) = R(\boldsymbol{Z}, \epsilon) - R^c(\boldsymbol{Z}, \epsilon|\boldsymbol{\Pi}). \tag{4}$$

In practice, this is done by minimizing the difference between the compression loss and the discriminative loss

$$\Delta' R(\boldsymbol{Z}, \boldsymbol{\Pi}, \epsilon) = R^c(\boldsymbol{Z}, \epsilon|\boldsymbol{\Pi}) - R(\boldsymbol{Z}, \epsilon), \tag{5}$$

because a stochastic gradient descent (SGD) optimizer reduces a loss function.

C. Proposed Training Methods

We propose two adversarial training methods to improve the robustness of deep networks against adversarial examples.

The first method is to simply use the MCR^2 loss function in the adversarial training process instead of the cross-entropy loss function. The label of an input image is determined by applying a support vector machine on the features. After adversarial training, the output logits of the neural network model are interpreted as features. We call this method adversarial training with MCR^2.

The second method is to increase the number of outputs of the original network and retrain it on the clean data base with the MCR^2 loss function. We hope that increasing the number of features may lead to an increase in classification accuracy. After retraining the networks from scratch, we perform adversarial training using the MCR^2 loss function. We call this method the "fully MCR^2" method.

III. Experimental Results

A. Adversarial Attacks

We perform adversarial attacks on 2 different deep neural network models: the VGG-13 [7] model and the ResNet-18 [8] model, and 2 data sets: the CIFAR-10 [9] data set and the SVHN [10] data set. The classification accuracies on clean images are shown in Table I.

Model	Data Set	Training Acc	Testing Acc
ResNet-18	CIFAR-10	99.88%	91.12%
ResNet-18	SVHN	99.49%	95.41%
VGG-13	CIFAR-10	99.89%	90.00%
VGG-13	SVHN	99.49%	94.94%

TABLE I
TRAINING RESULTS FOR RESNET-18 AND VGG-13.

Four different adversarial attacks are used to generate adversarial examples: Iterative Fast Gradient Sign Method (I-FGSM) [11], DeepFool [12], Universal Adversarial Perturbation (UAP) [13], and Simple Black-box Attack (SIMBA) [14]. For the I-FGSM method, we implement a targeted version and a non-targeted version. The chosen adversarial attacks cover a large variety: targeted attacks and non-targeted attacks, white-box attacks and black-box attacks, as well as image-specific attacks and universal attacks.

We attack the testing sets of the CIFAR-10 and SVHN data sets with I-FGSM, I-FGSMt, DeepFool, UAP and SIMBA. The adversarial accuracies on the adversarial examples are shown in Table II.

Model	Data Set	Attack	Adv. Acc
ResNet-18	CIFAR-10	I-FGSM	0.95%
ResNet-18	SVHN	I-FGSM	0.46%
VGG-13	CIFAR-10	I-FGSM	1.30%
VGG-13	SVHN	I-FGSM	0.46%
ResNet-18	CIFAR-10	I-FGSMt	0.61%
ResNet-18	SVHN	I-FGSMt	0.28%
VGG-13	CIFAR-10	I-FGSMt	1.08%
VGG-13	SVHN	I-FGSMt	0.54%
ResNet-18	CIFAR-10	DeepFool	0.95%
ResNet-18	SVHN	DeepFool	2.48%
VGG-13	CIFAR-10	DeepFool	1.30%
VGG-13	SVHN	DeepFool	2.76%
ResNet-18	CIFAR-10	UAP	24.84%
ResNet-18	SVHN	UAP	33.93%
VGG-13	CIFAR-10	UAP	36.68%
VGG-13	SVHN	UAP	36.46%
ResNet-18	CIFAR-10	SIMBA	1.75%
ResNet-18	SVHN	SIMBA	3.65%
VGG-13	CIFAR-10	SIMBA	2.31%
VGG-13	SVHN	SIMBA	3.51%

TABLE II
RESULTS FOR DIFFERENT ADVERSARIAL ATTACKS.

B. Adversarial Training

We perform traditional adversarial training with cross-entropy loss function and the two new adversarial training methods as proposed in Sec. II-C.

For each attack algorithm, attacks on the CIFAR-10 testing data set generates 10000 adversarial samples. From the 10000

adversarial examples, we pick 7000 images as the training set for adversarial training. The remaining 3000 images are used as the testing set. For each attack algorithm, attacks on the SVHN testing data set generate 26032 adversarial samples. From the 26032 adversarial samples, we pick 18000 images as the training set for adversarial training. 8000 images are used as the testing set for adversarial training.

Data Set	Attack	Original	w/ CE	w/ MCR2	Fully MCR2
CIFAR-10	I-FGSM	1.06%	73.26%	85.84%	85.48%
CIFAR-10	I-FGSMt	0.43%	78.83%	90.90%	90.44%
CIFAR-10	DeepFool	4.93%	62.66%	85.96%	81.35%
CIFAR-10	UAP	24.93%	75.00%	89.15%	85.35%
CIFAR-10	SIMBA	2.00%	83.83%	92.37%	93.01%
SVHN	I-FGSM	0.41%	89.21%	91.06%	89.91%
SVHN	I-FGSMt	0.25%	91.36%	93.52%	91.01%
SVHN	DeepFool	2.45%	64.16%	81.61%	29.38%
SVHN	UAP	33.62%	85.05%	91.58%	80.36%
SVHN	SIMBA	3.40%	95.51%	96.26%	95.50%

TABLE III
ADVERSARIAL ACCURACY COMPARISON FOR RESNET-18.

Data Set	Attack	Original	w/ CE	w/ MCR2	Fully MCR2
CIFAR-10	I-FGSM	1.30%	69.80%	83.60%	84.86%
CIFAR-10	I-FGSMt	0.96%	77.20%	89.89%	90.32%
CIFAR-10	DeepFool	5.56%	59.40%	84.55%	82.00%
CIFAR-10	UAP	34.96%	78.23%	90.18%	86.88%
CIFAR-10	SIMBA	2.50%	83.06%	91.99%	94.73%
SVHN	I-FGSM	0.51%	88.26%	90.28%	88.35%
SVHN	I-FGSMt	0.53%	90.48%	92.30%	89.78%
SVHN	DeepFool	2.57%	60.40%	73.02%	36.05%
SVHN	UAP	35.77%	88.67%	93.06%	85.60%
SVHN	SIMBA	3.13%	95.67%	95.75%	95.31%

TABLE IV
ADVERSARIAL ACCURACY COMPARISON FOR VGG-13.

The adversarial accuracies of the ResNet-18 and VGG-13 models before and after adversarial training are presented in Tables III and IV, respectively. For ResNet-18 models, the average adversarial accuracy improves from 79.9% with cross-entropy to 89.8% with MCR^2 (82.2% with fully MCR^2). For VGG-13 models, the average adversarial accuracy improves from 79.1% with cross-entropy to 88.5% with MCR^2 (83.4% with fully MCR^2).

Since adversarial training might result in a drop in clean accuracy [15], we also compare the clean accuracies before and after adversarial training, which are presented in Tables V and VI. For ResNet-18, the average clean accuracies before and after adversarial training with cross-entropy, MCR^2 and fully MCR^2 are about 93%. For VGG-13, the average clean accuracies before and after adversarial training with cross-entropy and MCR^2 are about 92%. The average clean accuracy after adversarial training with fully MCR^2 increases slighly to 93%.

Adversarial training with the MCR^2 loss function results in significantly larger increases in adversarial accuracy than adversarial training with cross-entropy loss function. The clean accuracy of a deep neural network decreases after adversarial training in some cases, and increased in some other cases. In the cases where the clean accuracy decreases, the three adversarial training methods lead to a comparable decrease of clean accuracy.

Data Set	Attack	Original	w/ CE	w/ MCR2	Fully MCR2
CIFAR-10	I-FGSM	91.12%	89.54%	88.15%	90.11%
CIFAR-10	I-FGSMt	91.12%	93.28%	91.70%	92.39%
CIFAR-10	DeepFool	91.12%	93.28%	84.93%	86.29%
CIFAR-10	UAP	91.12%	74.92%	81.33	84.68%
CIFAR-10	SIMBA	91.12%	96.34%	93.02%	94.41%
SVHN	I-FGSMt	95.41%	97.36%	97.78%	96.68%
SVHN	DeepFool	95.41%	95.33%	97.21%	92.76%
SVHN	UAP	95.41%	95.37%	96.74%	95.27%
SVHN	SIMBA	95.41%	98.65%	98.53%	97.56%

TABLE V
CLEAN ACCURACY COMPARISON FOR RESNET-18.

Data Set	Attack	Original	w/ CE	w/ MCR2	Fully MCR2
CIFAR-10	I-FGSM	90.00%	88.33%	86.01%	90.36%
CIFAR-10	I-FGSMt	90.00%	91.56%	90.44%	92.61%
CIFAR-10	DeepFool	90.00%	88.06%	83.63%	89.05%
CIFAR-10	UAP	90.00%	73.21%	84.55%	84.49%
CIFAR-10	SIMBA	90.00%	95.97%	92.70%	95.40%
SVHN	I-FGSM	94.94%	96.78%	96.65%	96.55%
SVHN	I-FGSMt	94.94%	97.48%	97.78%	97.06%
SVHN	DeepFool	94.94%	95.90%	90.35%	94.74%
SVHN	UAP	94.94%	95.04%	97.46%	95.48%
SVHN	SIMBA	94.94%	98.83%	98.48%	98.11%

TABLE VI
CLEAN ACCURACY COMPARISON FOR VGG-13.

C. Cross Testing

In the previous section, we only test the adversarial accuracy against an attack algorithm after adversarial training with examples generated with that specific attack algorithm. In practice, an adversarial-trained deep neural networks might be attacked with an attack algorithm that was not used to generate adversarial examples in its adversarial training process. As a result, we would like to test the adversarial accuracy against attack algorithm B of a deep neural network that underwent adversarial training with adversarial examples generated with attack algorithm A.

For most combinations of attack algorithms A and B in this work, performing adversarial training with MCR^2 loss function against attack algorithm A leads to a higher adversarial accuracy against attack algorithm B than performing adversarial training with cross-entropy loss function against attack algorithm A. The "fully MCR^2" method generally provides similar performance as the adversarial training with MCR^2 loss function. But the performance variance of the "fully MCR^2" method is larger.

IV. RELEVANCE OF PRINCIPLED LOSS FOR TRAINING

It is known that using cross-entropy for the loss function may cause challenges when training neural networks. In particular, when the training loss reaches zero, a "neural collapse" can be observed [16], [17]. Sec. III shows that cross-entropy loss also poses a challenge for adversarial training. A so-called principled loss like MCR^2 can avoid this. Further, it may help us to better understand the training process.

In the following, we discuss the relevance of a principled loss for training. For a better discussion, we use a performance model for classification that considers both network complexity and classification accuracy.

A. Quadratic Similarity Queries on Compressed Data

Ingber et al. [18], [19] introduce a model for quadratic similarity queries on compressed data. This model defines a (quadratic) similarity between database sequences (i.e. training samples) and query sequences (i.e. test samples). Further, it defines an identification rate of "signatures" (i.e. features) that represent the vectors of the training samples.

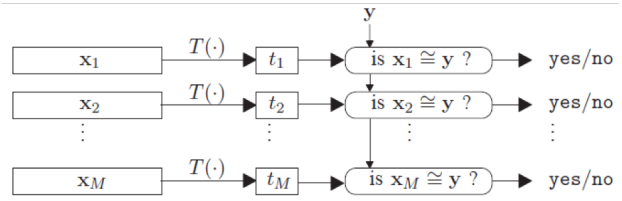

Fig. 1. Rate-R identification system (T, g) according to [19].

Fig. 1 shows the rate-R identification system (T, g) according to [19]. Let $\mathbf{x}_m = [x_1, \ldots, x_n]_m^T \in \mathbb{R}^n$ with $m = 1, \ldots, M$ be one of M database sequences of length n. Let $\mathbf{y} = [y_1, \ldots, y_n]^T \in \mathbb{R}^n$ be a query sequence of length n. The signature assignment $T : \mathbb{R}^n \to \{1, 2, \ldots, 2^{nR}\}$ maps a database sequence $\mathbf{x}_m$ to a signature t_m of rate R. The query function $g : \{1, 2, \ldots, 2^{nR}\} \times \mathbb{R}^n \to \{\text{no, yes}\}$ determines whether a query $\mathbf{y}$ is similar to a database sequence $\mathbf{x}_m$. In this work, a quadratic similarity is used such that

$$d(\mathbf{x}, \mathbf{y}) = \frac{1}{n}||\mathbf{x} - \mathbf{y}||_2^2. \tag{6}$$

Then $\mathbf{x}$ and $\mathbf{y}$ are called D-similar when

$$d(\mathbf{x}, \mathbf{y}) \leq D. \tag{7}$$

The model makes further assumptions to define a so-called identification rate R_{ID}: (a) The model assumes a vanishing probability of false negatives. (b) It defines D-admissible rates as rates of a compression scheme that also vanish the probability of false positives. Finally, the model defines the identification rate R_{ID} as the infimum of D-admissible rates. For Gaussian source data with variance σ^2 [19], the identification rate is:

$$R_{ID} = \begin{cases} \log_2\left(\frac{2\sigma^2}{2\sigma^2 - D}\right) & \text{for } 0 \leq D < 2\sigma^2 \\ \infty & \text{for } D \geq 2\sigma^2 \end{cases} \tag{8}$$

Note, if $D \geq 2\sigma^2$, almost all the sequences in the database will be similar to the query sequence. Then almost all the database needs to be retrieved, regardless of the compression. This makes the problem degenerate [19]. Fig. 2 shows the identification rate as a function of the normalized similarity.

B. Discussion of MCR^2 Loss

The loss function of MCR^2 uses a rate that reflects the compactness of the learned features as a whole and that is measured in terms of the average coding length per sample.

$$R_{\text{MCR}^2} \sim \frac{1}{2}\log_2\left(1 + \frac{\sigma_z^2}{\delta^2}\right) \tag{9}$$

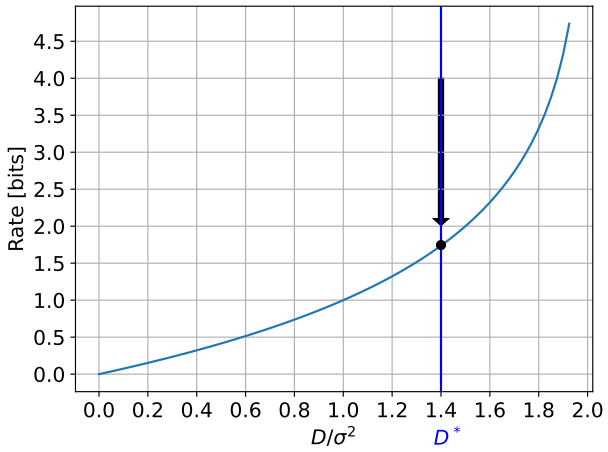

Fig. 2. Rate-similarity tradeoff and optimization for given similarity D^*.

The features with the variance σ_z^2 are subject to the distortion δ^2. And a limited distortion for the features

$$\mathbb{E}\left[\frac{1}{s}||\mathbf{z} - \tilde{\mathbf{z}}||_2^2\right] \leq \delta^2 \tag{10}$$

can constrain the similarity between data samples

$$\frac{1}{n}||\mathbf{x} - \mathbf{y}||_2^2 \leq D^*. \tag{11}$$

That is, if some distortion δ^2 is set before minimizing the MCR^2 loss function, some similarity threshold D^* can be found. Then, when minimizing the MCR^2 loss function during the training, the neural network is optimized, that is, its corresponding D-admissible rate is lowered. This can be interpreted as a constrained optimization of the D-admissible rate, given the target similarity D^*. Fig. 2 visualizes this constrained optimization process with the plotted arrow.

C. Discussion of Cross-Entropy Loss

Cross-entropy as a loss function for training uses only the probability of true and estimated labels. However, the probability of labels is not critical for rate-R identification systems. Further, the optimization does not constrain the similarity of data samples. The cross-entropy loss does not offer any control when dealing with the tradeoff between identification rate and D-similarity. Moreover, a neural collapse may occur that pushes the identification rate to zero.

V. Conclusions

We have proposed two new adversarial training methods which utilize the principle of Maximal Coding Rate Reduction (MCR^2). We conclude that adversarial training with the MCR^2 loss is a simple, yet effective method to defend against adversarial attacks. Moreover, this training leads to more robust deep neural networks. For adversarial attack methods like I-FGSM, I-FGSMt, DeepFool, UAP and SIMBA, adversarial training with the MCR^2 loss can achieve an adversarial accuracy of up to 90% when compared to the achieved adversarial accuracy of about 80% for adversarial training with cross-entropy.

Further, we have discussed cross-entropy loss and Maximal Coding Rate Reduction loss for adversarial training. For that we used a model for quadratic similarity queries on compressed data. It reveals a tradeoff between identification rate and similarity. In this context, a principled loss like the MCR^2 loss offers a clear advantage for training of deep networks. On the other hand, this model also underlines the deficiency of the across-entropy loss. Therefore, this model may help to develop new loss functions for training of deep neural networks.

REFERENCES

[1] Naveed Akhtar and Ajmal Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[2] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2015.

[3] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma, "Learning diverse and discriminative representations via the principle of maximal coding rate reduction," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, Vancouver, Canada, 2020.

[4] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2014.

[5] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958.

[6] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[7] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2015.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[9] Alex Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., University of Toronto, Toronto, ON, Apr. 2009, Available at https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[10] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[11] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," in *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

[12] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[13] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1765–1773.

[14] Chuan Guo, Jacob Gardner, Yurong You, Andrew Gordon Wilson, and Kilian Weinberger, "Simple black-box adversarial attacks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2484–2493.

[15] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang, "Adversarial training can hurt generalization," in *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019.

[16] V. Papyan, X. Y. Han, and D. L. Donoho, "Prevalence of neural collapse during the terminal phase of deep learning training," *Proceedings of the National Academy of Sciences*, vol. 117, no. 40, pp. 24652–24663, 2020.

[17] V. Kothapalli, "Neural collapse: A review on modelling principles and generalization," *Transactions on Machine Learning Research*, 2023.

[18] A. Ingber, T. Courtade, and T. Weissman, "Quadratic similarity queries on compressed data," in *Proceedings of the Data Compression Conference*, Snowbird, UT, Mar. 2013.

[19] A. Ingber, T. Courtade, and T. Weissman, "Compression for quadratic similarity queries," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2729 – 2747, May 2015.