SELF-SUPERVISED ADVERSARIAL TRAINING

Kejiang Chen[⋆†] *Yuefeng Chen*[†] *Hang Zhou*[⋆] *Xiaofeng Mao*[†] *Yuhong Li*[†] *Yuan He* [†] *Hui Xue* [†] *Weiming Zhang*[⋆] *Nenghai Yu*[⋆]

[⋆] University of Science and Technology of China
[†] Alibaba Group

ABSTRACT

Recent work has demonstrated that neural networks are vulnerable to adversarial examples. To escape from the predicament, many works try to harden the model in various ways, in which adversarial training is an effective way which learns robust feature representation so as to resist adversarial attacks. Meanwhile, the self-supervised learning aims to learn robust and semantic embedding from data itself. With these views, we introduce self-supervised learning to against adversarial examples in this paper. Specifically, the self-supervised representation coupled with k-Nearest Neighbour is proposed for classification. To further strengthen the defense ability, self-supervised adversarial training is proposed, which maximizes the mutual information between the representations of original examples and the corresponding adversarial examples. Experimental results show that the self-supervised representation outperforms its supervised version in respect of robustness and self-supervised adversarial training can further improve the defense ability efficiently.

***Index Terms*—** Adversarial training, self-supervised, defense, kNN

1. INTRODUCTION

Deep Learning has made a significant progress in computer vision, natural language processing and etc. Various kinds of techniques based on deep learning have been applied in practical engineering, such as autonomous vehicles [1], disease diagnosis [2]. These empowered applications are life crucial, raising great concerns in the field of safety and security. However, recently, many studies have shown that the classifiers using neural network are not robust when encountering attacks, especially adversarial examples.

Szegedy *et al.* proposed the concept of adversarial example for the first time [3], which means that a subtle perturbation is added to the input of the neural network to produce a wrong output with high confidence. After that, plenty of methods for generating adversarial examples have been developed, including gradient-based [4, 5, 6, 7], optimization-based [3, 8, 9] and etc. These methods show the fragility of deep learning models.

On the opposite side, many defenses against adversarial examples have been proposed along two directions: model hardening [4, 10, 11, 12, 13] ,input preprocessing [14, 15, 16, 17, 18]. As for model hardening, adversarial training has been proven to be an effective defense method. One convinced reason is that adversarial training forces the neural network to learn the robust feature [19], which is rarely affected by adversarial examples. Inspired by this view, we are eager to find neural networks that naturally learn the robust feature

This work was supported in part by the Natural Science Foundation of China under Grant U1636201, and by Anhui Initiative in Quantum Information Technologies under Grant AHY150400. Contact mail: zhangwm@ustc.edu.cn

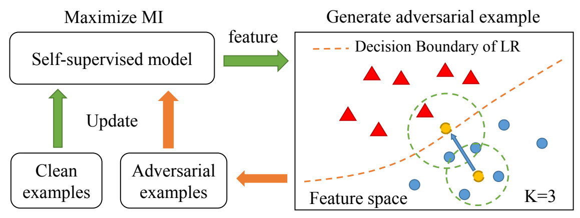

Fig. 1. The diagram of self-supervised adversarial training.

of images. Fortunately, self-supervised learning pursues the similar destination and has been developed quickly in recent years. Self-supervised learning aims to learn robust and semantic embedding from data itself and formulates predictive tasks to train a model, which can be seen as learning the robust representation.

Generally, given the self-supervised feature, the classification can be done with linear regression (LR) or k-Nearest Neighbors (kNN). In this paper, we choose self-supervised feature coupled with kNN as the final classifier. The reason can be intuitively observed from the right part of Fig. 1 that even the modified sample has crossed the decision boundary of LR, but it is still correctly classified by kNN, meaning that kNN owns stronger robustness than LR.

To further enhance the robustness, self-supervised adversarial training (SAT) is proposed. The object of SAT is to maximize the mutual information (MI) between the representations of clean images and their corresponding adversarial examples, so the learned feature can mitigate the effect of adversarial perturbation. The method can be divided into two parts: generating adversarial examples, maximizing the MI. The adversarial examples are generated using gradient-based method, due to its high efficiency. Subsequently, MI between the feature representations of clean and adversarial examples is maximized. In implementation, noise contrast estimator is utilized to estimate MI. Then the model is updated by minimizing the opposite value of estimated MI.

Our experimental results demonstrate that using the state-of-the-art self-supervised feature representation coupled with kNN shows stronger robustness against adversarial examples produced by both gradient-based and optimization-based methods with respect to supervised feature representation by a clear margin on CIFAR-10 and STL-10. Besides, the robustness of self-supervised models can be largely improved with SAT efficiently. Implementation-related file will be available at https://github.com/everange-ustc/SAT.git.

2. RELATED WORKS

2.1. Adversarial Examples

Adversarial examples are designed by an adversary to make machine learning system producing erroneous outputs. Most adversarial examples on deep neural networks are generated by adding small perturbation to clean samples. For kNN classification methods, the attack operates by adding a perturbation δ to the input such that its representation, $f(x)$, moves closer to representations of x_g, a nearest group of training instances from a different class (x_g^i for $i \in \{1, 2, ..., m\}$). Intuitively, adversarial examples can be generated by solving the optimization problem[20]:

$$\hat{\delta} = \arg\min_{\delta} \sum_{i=1}^{m} \left\| f\left(x_g^i\right) - f(x+\delta) \right\|_2^2 \quad \text{such that } \|\delta\|_p \leq \epsilon \text{ and } x+\delta \in [0,1]^d \tag{1}$$

The optimization can be formulated as a Lagrangian, and we can binary search the Lagrangian constant that yields the minimal perturbation. For example, the optimization can be solved with Adam optimizer.

2.2. Defense

Many defenses against adversarial examples have been proposed along two directions: model hardening, input preprocessing. For model hardening, adversarial training shows satisfying performance against adversarial examples. The **standard adversarial training (AT)** in Madry's work[5] can be formulated as:

$$\arg\min_{\theta} \mathbb{E}_{(x,y)\in \hat{p}_{\text{data}}} (\max_{\delta \in S} L(\theta, x+\delta, y)) \tag{2}$$

where $\hat{p}_{\text{data}}$ is the underlying distribution of training data, $L(\theta, x, y)$ is the loss function at data point x with the true label y for the neural network with parameters θ. δ is the permutation introduces by PGD[5]. The accuracy drops fast using AT, there is an alternate version[21], **Mix-minibatch adversarial training (MAT)**:

$$\arg\min_{\theta} [\mathbb{E}_{(x,y)\in \hat{p}_{\text{data}}} (\max_{\delta \in S} L(\theta, x+\delta, y)) + \mathbb{E}_{(x,y)\in \hat{p}_{\text{data}}} (L(\theta, x, y))] \tag{3}$$

which helps to pursue the trade-off between accuracy on the clean examples and robustness on the adversarial examples. **Adversarial logit pairing (ALP)**[11] matches the logits from a clean example x and its corresponding adversarial example $\tilde{x}$ during training, which exhibits better performance:

$$J(\mathbb{B}, \theta) + \lambda \frac{1}{m} \sum_{i=1}^{m} L\left(f\left(x; \theta\right), f\left(\tilde{x}; \theta\right)\right) \tag{4}$$

where $\mathbb{B}$ is a minibatch including clean examples x and the corresponding adversarial examples $\tilde{x}$. $f(x;\theta)$ is function mapping from inputs to logits of the model and $J(\mathbb{B}, \theta)$ is the cost function used for adversarial training. One potential reason of adversarial training is that it forces the neural network to learn robust feature, which can mitigate the affect of adversarial examples[19].

2.3. Self-supervised Learning

Self-supervised learning exploits internal structures of data and formulates predictive tasks to train a model, which can be seen as learning the robust feature. Here are some representative works in this aspect: Contrastive Predictive Coding (CPC) [22] uses a probabilistic contrastive loss which induces the latent space to capture information that is maximally useful to predict future samples. Deep Infomax (DIM) [23] maximizes mutual information between global features and local features. Augmented Multiscale DIM (AMDIM) [24] maximizes mutual information between features extracted from multiples views of a shared context.

Actually, the self-supervised representation has been used for defense in previous works. [20] utilized the feature representation coupled with kNN for classification. Both supervised and self-supervised features are adopted. However, as mentioned in [20], their method does not perform well on datasets bigger than MNIST. [25] combined the self-supervised loss into the loss of traditional adversarial training, but this training process is still time-consuming.

3. METHOD DESCRIPTION

As illustrated before, forcing the neural network to learn the robust feature of the instance can help improve the robustness of the model. Meanwhile, the self-supervised learning focuses on the robust feature, for example. they can predict the missing part of images using itself. Inspired by these point-views, we propose using self-supervised representation cooperated by k-Nearest Neighbour for defending against adversarial examples. Besides, we can maximize the mutual information representation between clean and adversarial examples by adjusting the existing model, so that the model can further mitigate adversarial perturbation.

3.1. Self-supervised Representation for Defense

The self-supervised representation is coupled with kNN for classification. After self-supervised training, the neural network is frozen and adopted as a feature extractor. All instances in the training set are fed into the network to obtain their representations on a specified layer, and then these representations serve as the feature library. Given an image, extract its feature representation, search the k-nearest representations from the feature library, and then predict the label of the image.

3.2. Self-supervised Adversarial Training

To further improve the robustness of self-supervised representations cooperated with kNN, we propose a method called self-supervised adversarial training (SAT), which maximizes the mutual information between the representations of clean images and the corresponding adversarial examples. As shown in Fig. 1, given the pretrained self-supervised model, the framework of SAT is divided into two parts: generating adversarial examples and maximizing the mutual information.

3.2.1. Generating Adversarial Examples

Due to the introduced attack method in Section 2.1 is time-consuming, we modified the generating method inspired by PGD[5]. In detail, the gradient of the image is obtained firstly:

$$g = \nabla_{x_{\text{adv}}^{t-1}} \sum_{i=1}^{m} \|f\left(x_{\text{g}}^i\right) - f\left(x_{\text{adv}}^{t-1}\right)\|_2^2 \tag{5}$$

where ∇ is the gradient operator, and $m = 300$ is the default setting. Then update the image:

$$x_{\text{adv}}^t = x_{\text{adv}}^{t-1} - \epsilon_s \cdot \text{sign}(g) \tag{6}$$

where ϵ_s is the update step size. To restrict the generated adversarial examples within the ϵ-ball of x_{adv}, we can clip x_{adv} after each update. For better distinction, we address the former method in Section 2.1 as optimization-based method and this as gradient-based method.

3.2.2. *Maximizing Mutual Information*

After obtaining the adversarial examples, we are going to maximize the MI on the feature representation space. Formally, the MI between X and X_{adv}, with joint density $p(x, x_{adv})$ and marginal densities $p(x)$ and $p(x_{adv})$, is defined as the Kullback–Leibler (KL) divergence between the joint and the product of the marginals:

$$\begin{aligned} I(X; X_{\text{adv}}) &= D_{\text{KL}}(p(x, x_{adv}) \| p(x)p(x_{adv})) \\ &= \mathbb{E}_{p(x,x_{adv})}\left[\log \frac{p(x,x_{adv})}{p(x)p(x_{adv})}\right] \end{aligned} \tag{7}$$

As for the feature representation, the MI can be defined as:

$$I\left(z_i; \hat{z}_i\right) = \underset{z_i,\hat{z}_i}{\mathbb{E}}\left[\log \frac{p\left(z_i, \hat{z}_i\right)}{p\left(z_i\right) p\left(\hat{z}_i\right)}\right] \tag{8}$$

where $z_i, \hat{z}_i$ are the feature representations of clean images and the corresponding adversarial version, respectively. It is hard to obtain the explicit distribution of representations, meaning that the MI cannot be calculated. Instead, several methods have been proposed to estimate MI, and here noise contrast estimator (NCE) is adopted, whose estimated MI has been proved to be a low bound of MI[26], defined by:

$$\begin{aligned} I(Z; \hat{Z}) &\geq \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} \log \frac{\Phi(\{z_i, \hat{z}_i\})}{\frac{1}{N}\sum_{j=1}^{N} \Phi(\{z_i, \hat{z}_j\})}\right] \\ &\triangleq I_{\text{NCE}}(Z; \hat{Z}) \end{aligned} \tag{9}$$

where $\hat{z}_j$ is the representation of other adversarial example different from $\hat{z}_i$. Here, we refer to representations from joint distribution as positives, i.e. $pos \sim p\left(z_i, \hat{z}_i\right)$, $pos = \{z_i, \hat{z}_i\}$, and representations from the product of marginal distributions as negatives, i.e. $neg \sim p\left(z_i\right) p\left(\hat{z}_j\right)$, $neg = \{z_i, \hat{z}_j\}$. N in in Equation (9) is the number of negative pairs, and $\Phi(\cdot)$ is the score function that is higher for positive pairs but lower for negative pairs. $\Phi(\cdot)$ can be any continuous and differentiable parametric functions, such as cosine similarity function. Here, the matching score function is defined as a simple dot product:

$$\Phi\left(z_i, \hat{z}_i\right) \triangleq \phi_1\left(z_i\right)^{\top} \phi_2\left(\hat{z}_i\right) \tag{10}$$

where $\phi_1(\cdot)$ and $\phi_2(\cdot)$ are small neural networks, for they can approximate any superb score functions. In implementation, the estimated MI is maximized by minimizing its opposite value, named the contrast loss:

$$\mathcal{L}_{\text{contrast}} = -\underset{\{z_i,\hat{z}_i\}}{\mathbb{E}}\left[\frac{1}{N}\sum_{j=1}^{N} \log \frac{\Phi\left(\{z_i, \hat{z}_i\}\right)}{\sum_{j=1}^{N} \Phi\left(\{z_i, \hat{z}_j\}\right)}\right] \tag{11}$$

The self-supervised neural network can be fine-tuned using back-propagation through minimizing $\mathcal{L}_{\text{contrast}}$. The process will be kept iterating until the performance meeting the requirement. To point out, the whole process does not require the true label of data, similar to the self-supervise learning. The pseudo-code of the framework is given in **Algorithm 1**.

4. EXPERIMENTS

4.1. Setting

Dataset CIFAR-10 and STL-10 are selected as the dataset. The CIFAR-10 dataset consists of 60000 32 × 32 labeled color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. STL-10 is composed of 10 classes 5000 96 × 96 labeled color images, 100000 unlabeled images for training and 8000 labeled images for testing. For speedy training, we resize the images in STL-10 to 64 × 64.

Attack Method The attack is implemented under white-box setting: the attacker has full information about the model (i.e. knows the architecture, parameters, etc.). Both gradient-based and optimization-based attack methods are utilized to evaluate the robustness of models. All the adversarial examples are generated on the 1000 correctly predicted images on the testing set. For the gradient-based attack methods, there are two kinds of setting: $\epsilon = 0.03$, $\epsilon_s = 0.005$, and 10 iterations; $\epsilon = 0.06$, $\epsilon_s = 0.005$, and 20 iterations, which is denoted as **small** perturbation and **large** perturbation, respectively.

Evaluation Metric For kNN classification, k is 75 and faiss[27] is adopted for speed consideration. The penultimate layer of neural network is adopted as the representation. The defense successful rate (**DSR**), defined as the correct prediction rate on the adversarial examples, is utilized to evaluate the robustness of the model against gradient-based attack. ℓ_2 distance, the average ℓ_2-norm of perturbation required to mislead the classifier, is used to measure the robustness against optimization-based attack. Larger ℓ_2 distance leads to better robustness. The accuracy (**ACC**) of clean examples is also presented to show the precision of the model.

Algorithm 1 Self-supervised Adversarial Training (SAT)

```
Require: Training samples $X$, perturbation bound $\epsilon$, step size $\epsilon_s$,
    maximization iterations per minimization step $K$, and minimiza-
    tion learning rate $\tau$.
 1: Initialize $\theta$ with a pretrained self-supervised model $f$.
 2: for epoch $= 1 \ldots N_{ep}$ do
 3:     for minibatch $B \subset X$ do
 4:         Build $x_{adv}$ for $x \in B$:
 5:             Assign a random perturbation
 6:                 $r \leftarrow U(-\epsilon, \epsilon)$
 7:                 $x_{adv} \leftarrow x + r$
 8:             for $k = 1 \ldots K$ do
 9:                 $L \leftarrow \sum_{i=1}^{m} \left\| f\left(x_g^i\right) - f(x_{adv}) \right\|_2^2$
10:                 $g_{adv} \leftarrow \nabla_{x_{adv}} L$
11:                 $x_{adv} \leftarrow x_{adv} - \epsilon_s \cdot \text{sign}(g_{adv})$
12:                 $x_{adv} \leftarrow \text{clip}(x_{adv}, x - \epsilon, x + \epsilon)$
13:             end for
14:         Calculate the representation of samples:
15:             $\hat{z} \leftarrow f(x_{adv}), z \leftarrow f(x)$
16:         Update $\theta$ with stochastic gradient descent:
17:             $g_\theta \leftarrow \mathbb{E}_{x \in B}[\nabla_\theta \mathcal{L}_{\text{contrast}}(z, \hat{z})]$
18:             $\theta \leftarrow \theta - \tau g_\theta$
19:     end for
20: end for
```

Table 1. The defense results of self-supervised representation and supervised representation of AMDIM with kNN on CIFAR-10 and STL-10.

Dataset	Method	ACC	DSR		ℓ_2 distance
			Small	Large	
CIFAR-10	SUP	**92.02%**	18.7%	15.4%	0.378
	SSL	84.64%	**51.9%**	**27.7%**	**0.667**
STL-10	SUP	75.41%	24.2%	16.3%	0.970
	SSL	**86.13%**	**54.9%**	**44.7%**	**1.591**

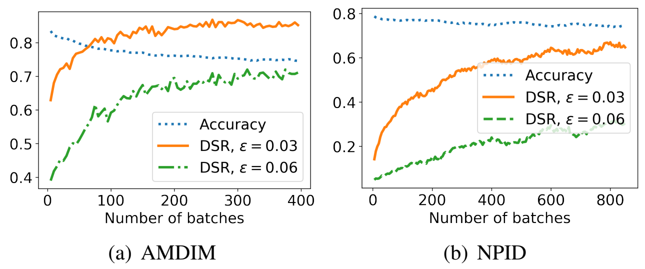

Fig. 2. The defense results of AMDIM and NPID using SAT on CIFAR-10.

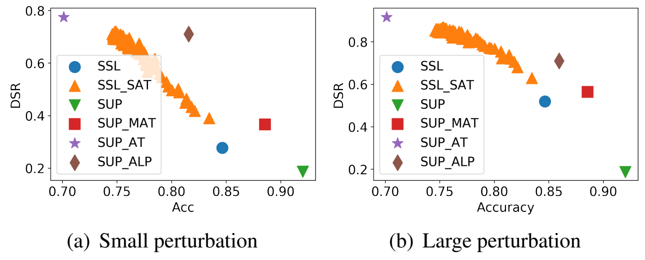

Fig. 3. The defense results of among self adversarial training and supervised adversarial training on CIFAR-10. AMDIM is selected as the seed model, and SUP and SSL mean the supervised and self-supervised version.

4.2. Superior of Self-supervised Representation

In this experiment, the state-of-the-art self-supervised learning method, AMDIM [24], and its supervised versions are compared to present the superior of self-supervised feature representation in the respect of robustness. These two methods are denoted by SSL and SUP, respectively. The backbone of AMDIM is an encoder based on the standard ResNet[28], with changes to make it suitable for DIM. More details about the encoder, the readers can refer to [23]. The parameters of the encoder for CIFAR-10 and STL-10 are set as (ndf=128, nrkhs=128, ndepth=3), (ndf=128, nrkhs=1024, ndepth=8), respectively. For supervised learning, Adam optimizer with learning rate 0.001 is adopted, and we trained the model for 400 epochs. For self-supervised learning, the learning rate is 0.0002 and the number of epoch is 300.

As shown in Table 1, the DSR of the self-supervised learning model (SSL) of AMDIM outperforms its supervised learning model (SUP) with a clear margin against gradient-based attack and the required ℓ_2-distance of successfully attacking SSL is larger than that of the SUP on two datasets. On CIFAR-10, the ACC on the clean images is lower than its supervised version, but the gain on the DSR is large, 33.2% for small perturbation attack. On STL-10, the performance is considerable, whose ACC outperforms that of supervised version, benefiting from unlabeled images in the training phase. In conclusion, the self-supervised representation owns stronger robustness.

4.3. Effectiveness of SAT

To verify the effectiveness of SAT, unsupervised model NPID[29] and self-supervised model AMDIM[24] are selected as the seed models. The adversarial examples are generated by gradient attack method with the small perturbation. Batch size 100, learning rate 0.0001

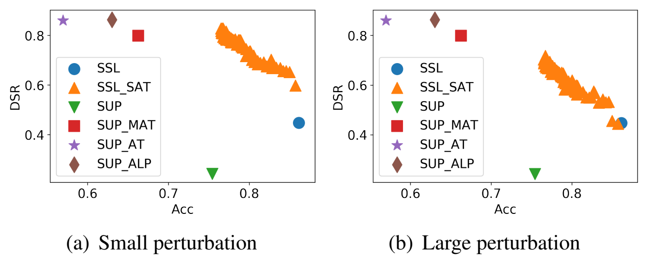

Fig. 4. The defense results of among self adversarial training and supervised adversarial training on STL-10.

and Adam optimizer are the other setting for SAT. The results are presented in Fig. 2. It can be seen that the DSR improves a lot after SAT with slight drop of accuracy for both AMDIM and NPID against small and large perturbation attacks, verifying the effectiveness of SAT.

We have also compared with the supervised adversarial training methods, and the adversarial examples are generated using PGD with the same setting as the gradient-based method does. AMDIM is selected as the seed model, and the results of these methods against small and large perturbation attack are shown in Fig. 3, Fig. 4, and the suffix represents which adversarial training method is adopted. The closer to the top right corner in figures, the better the performance. For sufficient label dataset CIFAR-10, the SAT is worse than the MAT, ALP, due to the original classification performance of self-supervised learning is worse than supervised models. Analyzing the development of self-supervised learning, we can see the gap between self-supervised and supervised model is closer. With stronger self-supervised model, the performance of SAT will become considerable. Furthermore, the time cost of SAT is much cheaper than that of MAT and ALP. For dataset with a few labels, like STL-10, the performance of SAT is significant. The trade-off between the robustness and accuracy is better achieved by SAT than supervised versions, especially for small perturbation attack, shown in Fig. 4. Since the dataset with a little supervised information is common in many downstream tasks, the proposed method SAT has a good prospect.

5. CONCLUSION

In this paper, we utilize self-supervised representation coupled with kNN for classification, where the underlying reason is that self-supervised model learns the robust feature of data. To further strengthen the defense ability of self-supervised representation, a general framework called self-supervised adversarial training is proposed, which maximizes the mutual information between the representations of original examples and adversarial examples. The experiments show that the self-supervised representation of AMDIM outperforms its supervised representation in the aspect of robustness on CIFAR-10 and STL-10. Furthermore, self-supervised adversarial training has been verified that it can be efficiently applied to AMDIM and NPID, and significantly improve the robustness against adversarial examples with slight drop of accuracy.

It is interesting to design self-supervised learning which considers adversarial attack in the training phase, so that the self-supervised representation naturally owns strong robustness, which is one direction of our future work.

6. REFERENCES

[1] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.

[2] Rongjian Li, Wenlu Zhang, Heung-Il Suk, Li Wang, Jiang Li, Dinggang Shen, and Shuiwang Ji, "Deep learning based imaging data completion for improved brain disease diagnosis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2014, pp. 305–312.

[3] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[5] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[6] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.

[7] Felix Kreuk, Yossi Adi, Moustapha Cisse, and Joseph Keshet, "Fooling end-to-end speaker verification with adversarial examples," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1962–1966.

[8] 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.

[9] Nicholas Carlini and David Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.

[10] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 582–597.

[11] Harini Kannan, Alexey Kurakin, and Ian Goodfellow, "Adversarial logit pairing," *arXiv preprint arXiv:1803.06373*, 2018.

[12] Saeid Asgari Taghanaki, Kumar Abhishek, Shekoofeh Azizi, and Ghassan Hamarneh, "A kernelized manifold mapping to diminish the effect of adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11340–11349.

[13] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He, "Feature denoising for improving adversarial robustness," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 501–509.

[14] Weilin Xu, David Evans, and Yanjun Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[15] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766*, 2017.

[16] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow, "Thermometer encoding: One hot way to resist adversarial examples," 2018.

[17] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Hassan Foroosh, "Comdefend: An efficient image compression model to defend adversarial examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6084–6092.

[18] Guoqing Jin, Shiwei Shen, Dongming Zhang, Feng Dai, and Yongdong Zhang, "Ape-gan: Adversarial perturbation elimination with gan," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3842–3846.

[19] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry, "Adversarial examples are not bugs, they are features," *arXiv preprint arXiv:1905.02175*, 2019.

[20] Chawin Sitawarin and David Wagner, "Defending against adversarial examples with k-nearest neighbor," *arXiv preprint arXiv:1906.09525*, 2019.

[21] Eric Wong and J Zico Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," *arXiv preprint arXiv:1711.00851*, 2017.

[22] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[23] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio, "Learning deep representations by mutual information estimation and maximization," *arXiv preprint arXiv:1808.06670*, 2018.

[24] Philip Bachman, R Devon Hjelm, and William Buchwalter, "Learning representations by maximizing mutual information across views," *arXiv preprint arXiv:1906.00910*, 2019.

[25] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song, "Using self-supervised learning can improve model robustness and uncertainty," *arXiv preprint arXiv:1906.12340*, 2019.

[26] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic, "On mutual information maximization for representation learning," *arXiv preprint arXiv:1907.13625*, 2019.

[27] Jeff Johnson, Matthijs Douze, and Hervé Jégou, "Billion-scale similarity search with gpus," *arXiv preprint arXiv:1702.08734*, 2017.

[28] 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.

[29] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," *arXiv preprint arXiv:1805.01978*, 2018.