GAN-Wars: Adversarial example generation and defense

Anver Khusainov Mike Boss Orhan Saeedi
Department of Computer Science, ETH Zurich, Switzerland

Abstract—**Deep Neural Networks can oftentimes be fooled by adversarial examples to incorrectly classify an example while keeping structural or visual consistency with the original data.**

Defending against such attacks is usually done by either training the network using adversarial examples or removing the adversarial perturbation before inference. In recent times, multiple GAN architectures have been proposed to generate adversarial examples as well as remove adversarial perturbations. We will analyse two such GAN architectures AdvGAN [1] and APE-GAN [2], proposed for adversarial example generation and defense respectively.

We re-produce the results of AdvGAN on the MNIST-challenge [3]. Using the attacks from the trained AdvGAN as well as FGSM [4], and PGD [5], we train APE-GAN models. We investigate the performance of the attacks under defense from their respective APE-GAN models as well as models trained on the other attacks.

I. Introduction

Motivation. Deep Neural Networks have been shown to be able to be fooled by bounded perturbations in the original data. These attacks can be harmful and represent a weakness in the model's architecture and training as the model may be misguided to misclassify data. Adversarial attacks that are visually indistinguishable from the original data are especially dangerous as the model may misclassify without a visually discernible reason. Finding visually indistinguishable attacks while keeping a high attack success rate is a non-trivial task. For this reason, AdvGAN was proposed using a GAN architecture to generate perturbations leading to adversarial examples on original data for a target model. The discriminator forces the model to generate attacks that are similar to the original data.

As adversarial attacks can be dangerous to certain model's performances, defenses have been proposed to mitigate such attacks. These defenses mostly either improve the model training by using attacks, combine multiple models to improve robustness, or try to remove the adversarial perturbations beforehand. Removing the perturbation is especially interesting as no change to the target model has to be applied therefore such defenses can be added later. A general approach to remove perturbation has been proposed in APE-GAN, utilizing a GAN architecture to map the space of attack images to the original data distribution.

Contribution. We investigate the performance of both GAN models in their respective area. We attack APE-GAN trained on AdvGAN, noting both the attack success rate of AdvGAN and defense error rate of APE-GAN. Furthermore, we train APE-GAN models on FGSM and PGD attacks and investigate how the defense performance transfers if an APE-GAN model trained on one attack is attacked by another attack. We perform these attacks on all combinations of AdvGAN, distilled AdvGAN, FGSM, and PGD and their respective APE-GAN models. Last but not least, we propose AdvGAN-Reverse. Another defense technique, which is based both on AdvGAN and APE-GAN. It uses the reverse of the adversarial loss proposed in AdvGAN and the content loss from APE-GAN to defend against attacks.

Related Work. The field of adversarial examples and adversarial training got a lot of recognition over the years. Szegedy et al. [6] discovered that deep learning models were vulnerable to adversarial examples. Furthermore they found that adversarial examples transfer between models, thus enabling black-box attacks on deployed models. Several attack strategies to generate adversarial examples have been proposed since then. Goodfellow et al. [7] argued that linear behavior in high-dimensional spaces is sufficient to cause adversarial examples. With this insight they proposed FGSM. Kurakin et al. [8] extended this attack by applying it several times with a smaller step size. There are also several attack in other distance classes. DeepFool proposed by Moosavi-Dezfooli et al. [9] is a l_2-bounded attack.

Also various defensive techniques against adversarial examples in deep neural networks have been proposed. Gu & Rigazio [10] propose Deep Contractive Networks (DCN) as a robust Deep Neural Network. This model follows the idea of adding noise to the image and then removing adversarial examples with a denoising autoencoder. Another idea from Papernot et al. [11] is to distill the target model with Knowledge Distillation [12], [13] so that it gets more robust. This approach is also very relevant for the black-box attack of AdvGAN. Parseval networks proposed by Cisse et al. [14] make their networks robust by constraining their Lipschitz constant of linear, convolutional and aggregation layers to be smaller than 1. Moreover, recent works by Sinha et al. [15], Raghunathan et al. [16], and Kolter & Wong [17] even succeed in providing robustness for small perturbations on MNIST.

II. Models and Methods

A. Problem Definition

We consider a standard classification task with an underlying data distribution $\mathcal{D}$ with data $\mathbf{x} \in \mathbb{R}^d$ and corresponding labels $y_{true} \in \mathbb{Z}_k$. We identify a classification model f

which maps the input to class scores $f(\mathbf{x}) \in \mathbb{R}_k$ We further assume a given suitable loss function (i.e. cross entropy) $L(f(\mathbf{x}), y)$.

The adversary's objective is to find adversarial examples $\mathbf{x}^{adv} \in \mathbb{R}^d$ for some target model f and inputs $(\mathbf{x}, y_{true})$ such that:

1) the model f misclassifies $\mathbf{x}^{adv}$: $f(\mathbf{x}^{adv}) \in \mathbb{Z}_k \backslash \{y_{true}\}$
2) the difference between $\mathbf{x}$ and $\mathbf{x}^{adv}$ is *perceptually small.*

How the visual quality of adversarial attacks should be measured is not clearly defined. Most proposed threat models are l_∞-bounded meaning that for a distance ε the generated examples $\mathbf{x}^{adv}$ have to satisfy $\|\mathbf{x}^{adv} - \mathbf{x}\|_\infty < \varepsilon$. It is to note, that such a bound does not correspond directly to perceptually small as the bound holds even if all pixels are changed by at most ε clearly visibly altering the image. Threat models can be subdivided into white-box attacks and black-box attacks. White-box attacks have access to the models loss while black-box attacks only have access to the models outputs.

B. Generative Adversarial Networks

Goodfellow et al. [4] introduced Generative Adversarial Networks, a framework for training deep generative models using a minimax game between a generator G and a discriminator D. The goal is to learn a generator distribution $P_G(\mathbf{x})$ that matches the real data distribution $P_{\text{data}}(x)$. The GAN does so by teaching its generator G to generate samples from the generator distribution P_G by transforming a noise variable $z \sim P_{\text{noise}}(z)$ into a sample $G(z)$. The discriminator learns to differentiate generated images from real images. The following loss is minimized

$$\begin{aligned}\mathcal{L}_{GAN} :=& \mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}[\log D(x)] \\ &+ \mathbb{E}_{z\sim P_{\text{noise}}}[\log(1 - D(G(z)))]\end{aligned} \tag{1}$$

C. Attacks

Fast Gradient (Sign) Method (FGSM). Goodfellow et al. [7] introduced the fast gradient method attack (FGSM). A method that applies a first order approximation of the loss function to construct adversarial samples,

$$\mathbf{x}^{adv}_{\text{FGSM}} := \mathbf{x} + \varepsilon \cdot \text{sign}(\nabla_{\mathbf{x}} L(f(\mathbf{x}), y_{true})). \tag{2}$$

This method is simpler and faster than other methods, however it has a lower attack success rate since it was designed to be fast rather than optimal.

Projected Gradient Descent. Madry et al. [5] introduced a more powerful multi-step variant, which is essentially projected gradient descent (PGD) with step size $\alpha \geq \varepsilon/k$ on the negative loss function. **AdvGAN.** Xiao et al. [1] proposed AdvGAN. The idea was to generate adversarial examples using a GAN architecture. AdvGAN's objective combines three seperate losses

$$\mathcal{L} = \mathcal{L}_{adv} + \alpha\mathcal{L}_{GAN} + \beta\mathcal{L}_{hinge}$$

where $\mathcal{L}_{GAN}$ stands for the standard GAN loss, $\mathcal{L}_{hinge}$ for the hinge loss to minimize the perturbation, and $\mathcal{L}_{adv}$ for the adversarial loss on the target model. $\mathcal{L}_{adv}$ depends on if the attack is targeted or un-targeted, either maximizing towards the target or away from ground-truth. Both a black-box and white-box version of the attack are described. The black-box attack learns to distill the target model to perform the white-box attack on the distilled model. The distillation and attack are alternating resulting in dynamic distillation of the target model during the attack. The white-box attack may be classified as semi-whitebox as the target model is only needed during training.

D. Defenses

APE-GAN. The GAN architecture is used to learn the mapping from adversarial images to the original image distribution. To achieve this APE-GAN uses the standard adversarial GAN loss in combination with a content loss, the pixel-wise MSE-loss, to learn to generate images close to the original image distribution. APE-GAN uses DC-GAN [18] as it's GAN architecture owing to the stability of DC-GAN.
AdvGAN-Reverse. We introduce another defense based on a combination of the AdvGAN attack and content loss of APE-GAN. Instead of maximizing distance from the ground truth to generate adversarial attacks we instead minimize the distance to generate images that are easier to classify for the target model. As this reverses the function of the adversarial loss we call this model AdvGAN-Reverse. To mitigate overfitting to the target model by the generator we introduce the same pixel-wise MSE-loss as in APE-GAN, comparing generated images to the original data. It is to note, that such a model actually allows to improve the performance of a fixed target model by adapting the original dataset to the target models distribution. However, in the context of this work we are interested in the defense performance under adversarial attacks instead.

E. MNIST-Challenge

The aim of the MNIST challenge [3] is to provide a robust and equal performance measure to different adversarial attacks. The black-box version has concluded in 2017 but the white-box version is still open to contributions. The challenge provides three models natural, adversarial trained, and secret. For the black-box challenge the natural and adversarial trained models were available with corresponding weights to train attacks on. The secret model was used to evaluate the attacks as a test model. After concluding the black-box version the secret model was released and is open to attack in the white-box version. The challenge restricts attacks to a maximum L_∞ norm perturbation of the original

MNIST images of less than $\varepsilon = 0.3$. The images that are evaluated are the standard test split images from MNIST.

III. Implementation

We re-implement both models in pytorch-lightning [19] to try to re-produce the original results as well as to easily train them with each other.

Dataset. We use the MNIST pytorch-lightning data-module provided by lightning-bolts [20]. Notably, we use the standard train/validation split in MNIST to train our models. As the MNIST challenge uses the test data from MNIST our trained models have not seen these images before. Therefor, testing on the MNIST challenge is the same as testing on an unseen test set.

FGSM and PGD. We use the FGSM and PGD implementations provided in CleverHans [21] wrapped in a pytorch module as our attacks. PGD performs 40 steps instead of 100 like in the MNIST challenge pgd attacks.

AdvGAN. Our AdvGAN implementation follows the paper and builds on top of existing implementations [22], [23]. We use $\alpha = 0.0004$ and $\beta = 0.002$ in the loss function, giving most weight to the adversarial loss. This corresponds with other implementations as well as the nature of the challenge. As the L_∞ criterion is used perceptual similarity, normally a feature of AdvGAN, is not needed only absolute maximal per-pixel perturbation matters. To stay in the challenge bounds the perturbation is clipped to $\varepsilon = 0.3$ and the resulting adversarial image to box $= [0, 1]$. Different to the original paper but in accordance with implementations, we do not just maximize the distance to ground truth in the adversarial loss. Instead the difference of the ground truth prediction to the maximal predicted incorrect label is minimized. This is similar to a targeted attack, however the target is not fixed. We implement distillation for the black-box attack based on the distillation described in [11]. As far as we know, this is the first available implementation of AdvGAN with distillation. We allow for the target model to be loaded as a tensorflow model. However, for performance reasons we converted the original tensorflow model used in the challenge to pytorch.

APE-GAN. Our APE-GAN implementation follows the paper and builds on top of existing implementations [24], [25]. Instead of directly generating restored images from the images, as done in other implementations, we generate the perturbation. The perturbation is clipped to $\varepsilon = 0.3$ and the restored images to box $= [0, 1]$. The original learning rate of $2e - 4$ used in the implementation lead to unstable training in our implementation. We use a decreased learning rate of $5e - 5$.

AdvGAN-Reverse. We invert the adversarial loss from the AdvGAN model and add the MSE-loss to the generator loss from AdvGAN.

Training. We use a batch size of 256 and 128 for AdvGan and APE-GAN respectively. We trained all models for 50 epochs at 16 bit mixed precision. We monitor the accuracy of the target model on the adversarial images for AdvGAN and restored images for APE-GAN. For evaluation we load the best model based on this monitoring.

IV. Results

AdvGAN. In the MNIST challenge AdvGAN ranks first place with an accuracy of 92.76% in the black-box setting. We trained our AdvGAN implementation on the test data directly achieving accuracies of about 94% and 91% in the black-box and white-box setting respectively. These results lead us to believe that with more challenge specific fine-tuning the accuracy of 92.76% is achievable. We chose not to further investigate this instead opting to use the test dataset as unseen test data. The results presented in Table I are accuracies achieved on the MNIST test data not seen before by all models. As such, learned models are at an disadvantage as the data has not been seen before during training. However, we believe using unseen test data is more realistic to for model performance measurements.

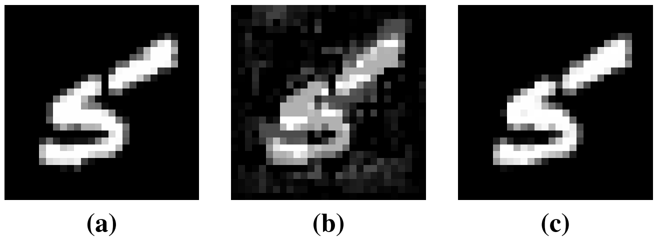

Figure 1. Example of perturbations after applying AdvGAN and APE-GAN. (a) shows the correctly classified original image. (b) shows the corresponding adversarial attack by AdvGAN with $L_\infty \leq 0.3$. The target model misclassifies this image as 6. (c) is the correctly classified restored image generated by APE-GAN given (b) as input.

Threat Model	Black-Box Setting	White-Box Setting
AdvGAN	97.51 %	95.70 %
AdvGAN-dist	95.88 %	95.77 %
FGSM	97.82 %	97.07 %
PGD	98.01 %	95.44 %

Table I
Accuracies for attacks without defender

As seen in Table I attack performances are comparable between these attacks. AdvGAN with a distilled target model achieves better performance than AdvGAN without distillation in the black-box setting. In the white-box setting both AdvGAN models achieve about the same performance. AdvGAN over all show strong attack performance, achieving best performance in the black-box setting and only slightly losing out to PGD in the white-box setting. It is to note however, that AdvGAN does not achieve attack performance

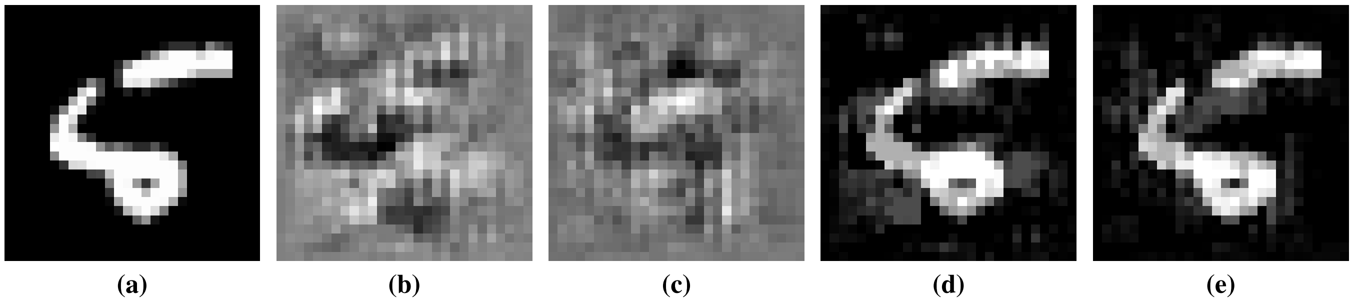
Figure 2. (a) is the original example. (b) and (d) are perturbation and corresponding adversarial image which is still classified correctly as 5. (c) and (e) are perturbation and adversarial example which is misclassified as 6.

similar to the MNIST challenge attack. This indicates that it's performance on unseen data is quite different from targeted attacks on specific data.

APE-GAN. We present attack performances of all attacks on APE-GAN models trained on these different attacks in Table II and Table III for the black-box and white-box setting respectively. In both tables, the rows indicate the respective attack while the columns indicate with which attack APE-GAN was trained with. The $+/-$ indicates the change in performance compared to Table I over one row or column. Therefor, in the right $+/-$ column a positive value indicates where APE-GAN defended successfully against a certain attack over all trained APE-GANs. In the bottom $+/-$ row a positive value indicates a certain APE-GAN successfully defending over all attacks. Bold values indicate improvements by APE-GAN over the attacks in Table I.

Black-Box Setting	AdvGAN	AdvGAN-dist	FGSM	PGD	+/-
AdvGAN	**98.29 %**	**98.25 %**	97.39 %	**97.61 %**	1.50 %
AdvGAN-dist	95.44 %	**98.02 %**	93.49 %	95.50 %	−1.07 %
FGSM	**97.83 %**	**97.83 %**	**97.91 %**	**98.10 %**	0.39 %
PGD	97.90 %	98.00 %	97.85 %	**98.31 %**	0.02 %
+/-	0.24 %	2.88 %	−2.58 %	0.30 %	

Table II
ATTACK (ROW) AGAINST APE-GAN USING ATTACK (COLUMN) IN BLACKBOX SETTING

In the black-box setting, the results show that APE-GAN successfully improves the accuracy in the majority of attack scenarios. However, as seen in the $+/-$ column over all trained APE-GAN models improvements are smaller and variy depending on the specific attack. AdvGAN with distilled target model performs best against all APE-GAN models. It successfully attacks three models only losing out to APE-GAN on the model trained with the same attack.

In the white-box setting we observe worse results for APE-GAN over most attacks. This is unsurprising in itself as attack performances increased in the white-box setting. However, we observe a greater variance in model performances resulting in higher improvements over all APE-GAN models for certain attacks but higher decreases in accuracy for others as well. Notably, APE-GAN successfully defends against the PGD attack with all models in the white-box setting compared to only one model in the black-box setting. With FGSM as an attack the corresponding APE-GAN defends well but all other APE-GANs reduce accuracy quite significantly. In the $+/-$ row we clearly see that over all attacks each APE-GAN model actually reduces the accuracy.

White-Box Setting	AdvGAN	AdvGAN-dist	FGSM	PGD	+/-
AdvGAN	**98.38 %**	**97.53 %**	94.12 %	**96.01 %**	3.24 %
AdvGAN-dist	**96.26 %**	**97.88 %**	91.39 %	95.43 %	−2.12 %
FGSM	74.45 %	84.98 %	**97.47 %**	83.01 %	−48.37 %
PGD	**95.48 %**	**97.50 %**	**96.89 %**	**97.75 %**	5.86 %
+/-	−19.41 %	−6.09 %	−4.11 %	−11.78 %	

Table III
ATTACKS (ROW) AGAINST APE-GAN USING ATTACK (COLUMN) IN WHITEBOX SETTING

AdvGAN-Reverse. Using the adversarial loss AdvGAN improves on the performance achieved by APE-GAN in all models that are trained on the same attack except for the AdvGAN with a distilled target model in the black-box setting slightly missing out. However, the model shows very poor generalisability to other attacks as it performs significantly worse in all attacks that the defense model was not trained on. APE-GAN also struggled with generalisability of it's model to other attacks however accuracy staid mostly comparable to the original attack. For the AdvGAN-Reverse model this is clearly not true as accuracy reduces up to 10% in a lot of cases.

In the black-box setting all models except the AdvGAN-dist* are using a distilled target model to train the defense. This improved the performance in the black-box setting significantly. AdvGAN-dist* already uses a distilled target model in the attack it is trained against and saw reduced performance if the defense also used a distilled model. No models with distilled target models were used in the white-box settings as they achieved superior performance.

Black-Box Setting	AdvGAN	AdvGAN-dist*	FGSM	PGD
AdvGAN	**98.89 %**	95.05 %	96.53 %	96.79 %
AdvGAN-dist	78.26 %	98.00 %	82.87 %	87.75 %
FGSM	95.66 %	84.87 %	**98.26 %**	**98.66 %**
PGD	96.15 %	88.39 %	**98.35 %**	**98.67 %**

Table IV
ATTACKS (ROW) AGAINST ADVGAN-REVERSE USING ATTACK (COLUMN) IN BLACKBOX SETTING

White-Box Setting	AdvGAN	AdvGAN-dist	FGSM	PGD
AdvGAN	**98.74 %**	**98.19 %**	76.36 %	82.56 %
AdvGAN-dist	83.65 %	**98.66 %**	73.94 %	88.91 %
FGSM	74.45 %	83.00 %	**97.47 %**	**98.55 %**
PGD	70.34 %	85.20 %	96.58 %	**98.61 %**

Table V
ATTACK (ROW) AGAINST ADVGAN-REVERSE USING ATTACK (COLUMN) IN WHITEBOX SETTING

Examples. Figure 1 demonstrates work of both AdvGAN and APE-GAN. It shows that AdvGAN can successfully fool even the robust target model that was suggested in the MNIST Challenge. Visually the shape of the digit didn't change significantly, but the model misclassifies it. Nevertheless, APE-GAN eliminates this perturbation and helps to classify the image correctly.

Not all of the examples are easy to classify even for humans. Figure 2 gives an example with digit which is hard to classify. It may seem as an easy target to attack for any adversarial model. However, even after 10 out of 50 epochs the image is still classified correctly, i.e. AdvGAN didn't find perturbation which will fool the target model. However, in the end of training, AdvGAN found a suitable perturbation. It worth noting that the images visually don't seem very different, but perturbations vary noticeably.

V. DISCUSSION

The results of the AdvGAN model show a strong attack that misses out on some performance if used on unseen data. It performs very well in a black-box setting compared to other attacks, however lost out in the white-box setting to the PGD attack. If training to on specific data is possible we think this attack can achieve very high performance while keeping high visual consistency with the original data.

APE-GAN manged to defend quite well against attacks it was trained against in both the black-box and white-box setting. However, APE-GAN fails to generalize to other attacks as seen in our results. In the best case over all attacks it leads to some accuracy improvements however in the case of FGSM in the white-box setting to reduction in accuracy averaged over all APE-GAN models of about 10%. In the white-box setting the all APE-GAN models reduce the accuracy averaged over all attacks. Further study is needed in how to make models such as APE-GAN robust against other attacks. Training with multiple attack is a future possibility to explore as well as some regularization.

Our proposed AdvGAN-Reverse model improves over APE-GAN in all attacks it was trained against. However, improvements are modest relative to APE-GAN. AdvGAN-Reverse shows an increased susceptibility to performance losses on attacks it was not trained on. These are in certain cases quite significant only seen in APE-GAN in one case in this magnitude. Therefor, the model generalized much worse than APE-GAN.

We conclude that deep neural networks such as in this case GAN architecture show both great prospects for adversarial example generation and defense against such attacks. However, the defensive models we investigate show little to no ability to generalize to attacks they were not trained against. Further research in making robust defenses against adversarial attacks using generative models is needed as well as more thorough evaluation of existing models.

VI. SUMMARY

We investigated the APE-GAN model for defense against adversarial attacks. We attacked the model with AdvGAN, an attack based on the GAN architecture, AdvGAN using a distilled target model, FGSM, and the PGD attack. We evaluated the performance of APE-GANs trained on these attacks against the respective attack as well as APE-GANs trained on the other attacks. We evaluated the performance on the MNIST challenge in a black-box and white-box setting. We found the AdvGAN attack to be quite strong in the black-box setting. APE-GAN improved the robustness against all attacks if the APE-GAN model was trained against the specific attack. However, APE-GAN failed to generalize to the other attacks it was not trained on.

References

[1] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[2] S. Shen, G. Jin, K. Gao, and Y. Zhang, "Ape-gan: Adversarial perturbation elimination with gan," *arXiv preprint arXiv:1707.05474*, 2017.

[3] Mnist challenge github. [Online]. Available: https://github.com/MadryLab/mnist_challenge

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2019.

[6] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2014.

[7] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[8] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," 2017.

[9] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," 2016.

[10] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," 2015.

[11] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," 2016.

[12] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?" 2014.

[13] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[14] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples," 2017.

[15] A. Sinha, H. Namkoong, R. Volpi, and J. Duchi, "Certifying some distributional robustness with principled adversarial training," 2020.

[16] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," 2020.

[17] E. Wong and J. Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," 2018.

[18] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016.

[19] W. Falcon and T. P. L. team, "Pytorch lightning: The lightweight pytorch wrapper for high-performance ai research. scale your models, not the boilerplate." 3 2019. [Online]. Available: https://www.pytorchlightning.ai

[20] W. Falcon and K. Cho, "A framework for contrastive self-supervised learning and designing a new approach," *arXiv preprint arXiv:2009.00104*, 2020.

[21] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, "Technical report on the cleverhans v2.1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2018.

[22] Advgan pytorch implemenation. [Online]. Available: https://github.com/mathcbc/advGAN_pytorch

[23] Advgan pytorch implemenation. [Online]. Available: https://github.com/GiorgosKarantonis/Adversarial-Attacks-with-Relativistic-AdvGAN

[24] Apegan pytorch implemenation. [Online]. Available: https://github.com/owruby/APE-GAN

[25] Apegan pytorch implemenation. [Online]. Available: https://github.com/shenqixiaojiang/APE-GAN