Adversarial Error Correction

Mayank Mishra (2016EE30506), Tarun Kumar Yadav (2016CS10359), Chahat Chawla (2016MT10492)

***Abstract*—Deep neural networks have been enormously successful across a variety of classification tasks. However, recent research shows that DNNs are vulnerable to adversarial attacks, which pose a serious threat. In this paper, we propose a simple yet effective defense algorithm Adversarial Error Correction that uses variational autoencoder (VAE) to filter out adversarial noise from input images to a classifier. The proposed method is generic and can defend white-box attacks without the need of retraining the original CNN classifiers, and can further strengthen the defense.**

I. Introduction

Deep neural networks are highly expressive models that have recently achieved state of the art performance on speech and visual recognition tasks. Despite their ability to learn complicated functions, little is known about what they learn. Deep neural networks learn input-output mappings that are fairly discontinuous to a significant extent. We can cause the network to misclassify an image by applying a certain hardly perceptible perturbation, which is found by maximizing the network's prediction error. In addition, the specific nature of these perturbations is not a random artifact of learning: the same perturbation can cause a different network, that was trained on a different subset of the dataset, to misclassify the same input.

Machine learning models are vulnerable to these small perturbations. Such examples with these perturbations are calles adversarial examples. These adversarial examples look similar to their non-adversarial counterparts but lead to poor classification accuracy of deep learning systems on a particular task.

II. Adversarial attacks

A. Fast Gradient Sign Method (FGSM)

FGSM[1] is a single step attack. Let $L(\boldsymbol{x}, \boldsymbol{y})$ be the loss function of the classifier C given input $\boldsymbol{x}$ and target label $\boldsymbol{y}$. FGSM defines the perturbation $\boldsymbol{\rho}$ as

$$\boldsymbol{\rho} = \epsilon \times sign(\nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{y})) \tag{1}$$

where ϵ is a small factor. FGSM simply chooses the sign of change at each pixel to increase the loss $L(\boldsymbol{x}, \boldsymbol{y})$ and fool the classifier.

B. Projected Gradient Descent

PGD[2] is a more powerful multi-step attack with projected gradient descent. Projected gradient descent performs one step of standard gradient descent, and then clips all the coordinates to be within the box for box-constrained numerical optimization.

$$\begin{aligned} \boldsymbol{x}_0^{PGD} &= \boldsymbol{x} \\ \boldsymbol{x}_{t+1}^{PGD} &= \prod_S [\boldsymbol{x}_t^{PGD} + \alpha \times sign(\nabla_{\boldsymbol{x}} L(\boldsymbol{x}_t^{PGD}, \boldsymbol{y}))] \end{aligned} \tag{2}$$

where $\prod_S$ is the projection onto $S = \{\boldsymbol{x}' : ||\boldsymbol{x}' - \boldsymbol{x}||_\infty \leq \epsilon\}$.

III. Variational Autoencoders

Generative models have gained a lot of success since the advent of deep neural networks. In particular, GANs[3] and VAEs[4] have been used to model highly complex distributions whose implicit expressions might be intractable. We can obtain a lower bound on the log likelihood of the data as

$$\log p(\boldsymbol{x}) \geq -KL[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{z})] + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \tag{3}$$

To achieve the goal of maximizing the log-likelihood, one can simply maximize a lower bound, evidence lower bound (ELBO) on the log-likelihood. This can be done by using a VAE.

IV. Adversarial error correction

We leverage the expressive capability of VAEs for defense mechanism. Our proposed method is generic and can defend white-box and black-box attacks and no online iterative optimisation is involved.

We modify the VAE objective (that preserves the adversarial perturbations in reconstruction) to the one which constructs the corresponding clean image, enabling to correctly label the adversarial image.

An adversarial example is passed through a VAE whose encoder is K-Lipschitz. From a coding theory perspective, we claim that latent representation codes for the adversarial and clean image will be closed in latent space when the encoder is K-Lipschitz.

$$\begin{aligned} \boldsymbol{z} &\sim \text{Enc}(\hat{\boldsymbol{x}}) = q(\boldsymbol{z}|\hat{\boldsymbol{x}}) \\ \boldsymbol{x} &\sim \text{Dec}(\boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z}) \\ \mathcal{L} &= -\mathbb{E}_{q(\boldsymbol{z}|\hat{\boldsymbol{x}})}\left[\log \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{q(\boldsymbol{z}|\hat{\boldsymbol{x}})}\right] \\ &= -\mathbb{E}_{q(\boldsymbol{z}|\hat{\boldsymbol{x}})}[\log p(\boldsymbol{x}|\boldsymbol{z})] + KL(q(\boldsymbol{z}|\hat{\boldsymbol{x}})||p(\boldsymbol{z})) \end{aligned} \tag{4}$$

where $\hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\delta}$ is an adversarial image with δ-perturbation added on top of a clean image $\boldsymbol{x}$. This adversarial image is encoded to a latent variable $\boldsymbol{z}$, which is decoded to the underlying clean image $\boldsymbol{x}$.

V. Experiments and results

A. Datasets

For the purpose of this paper, we experimented on the Fashion MNIST dataset (black and white images of the dimension 28×28) and the SVHN dataset (coloured images of the dimension 32×32). Both datasets have 10 ground truth classes.

B. Adversarial attacks

We perform the two mentioned adversarial attacks in this paper i.e FGSM[1] and PGD[2]. The resulting images generated using FGSM for Fashion MNIST are shown in Fig. 1. The resulting images generated using PGD for Fashion MNIST are shown in Fig. 2 and for SVHN in Fig. 3.

C. Defense mechanisms

We try out two defense mechanisms to protect our classifiers as discussed below:

1) Adversarial retraining: We do adversarial retraining of the classifiers. Basically, in this approach the classifier are retrained on the adversarial examples.

2) Adversarial error correction: An adversarial example is passed through a VAE whose encoder is K-Lipschitz. From a coding theory perspective, we claim that latent representation codes for the adversarial and clean image will be closed in latent space when the encoder is K-Lipschitz. We experiment with both gradient norm penalty and gradient clipping for imposing Lipschitz constraint on encoder, and found the former to be giving slightly better results experimentally. Results in the table are reported with gradient norm penalty.

Our method can also be used as a simple detection mechanism to detect if an input image contains adversarial perturbations or not. This can be done by comparing the MSE between reconstructed image and the original image. If an image does contain adversarial perturbation, then this MSE measure will be relatively large as compared to when the image doesn't contain adversarial perturbations (refer Fig. 4). Assuming perfect VAE, the exact added δ-perturbation must be recovered after reconstruction. We set a criteria on this difference to detect the adversarial example, and results for the same has been reported in the table.

In this approach the adversarial examples are passed through a VAE whose encoder is K-Lipschitz. The claim is that if the encoder is K-Lipschitz the adversarial examples and the real examples would be close enough in the latent space. This Lipshitz contraint on the encoder is imposed by having a gradient penalty on the encoder.

D. Further studies

It is further seen that as the dimensionality of the latent space is increased the adversarial examples and the real data come closer in the latent space. (See Fig. 4)

VI. Conclusion

In this paper, we have proposed a simple yet effective defense algorithm Adversarial Error Correction that uses a variational autoencoder (VAE) to filter out adversarial noise from input images to a classifier. The proposed method is generic and can defend white-box and black-box attacks without the need of retraining the original classifiers, and can further strengthen the defense.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[4] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

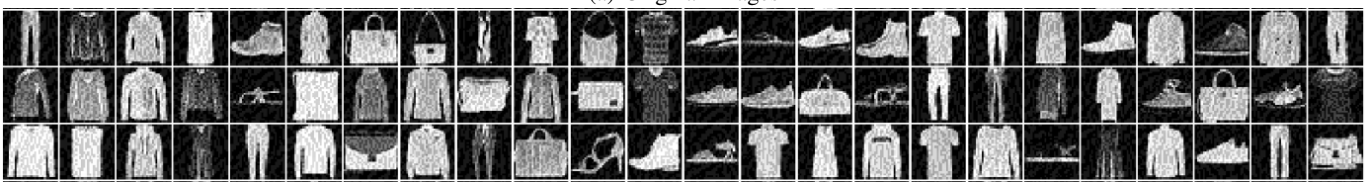
TABLE I: Accuracy on Fashion MNIST with and without defense mechanisms

		Adversarial retraining			Adversarial error correction
Attack	No defense	FGSM 0.3	PGD 0.1	PGD 0.3	VAE
No attack	91.2	91.4	89.9	91	82.2
FGSM ($\epsilon = 0.1$)	24.2	82.6	81	75.9	62.7
FGSM ($\epsilon = 0.3$)	9.1	89.4	42.4	74.4	49.2
PGD ($\epsilon = 0.1$)	5.9	12.1	71.7	61.8	50.5
PGD ($\epsilon = 0.3$)	5.7	5.6	7.1	68.1	44.2

TABLE II: Accuracy on SVHN with and without defense mechanisms

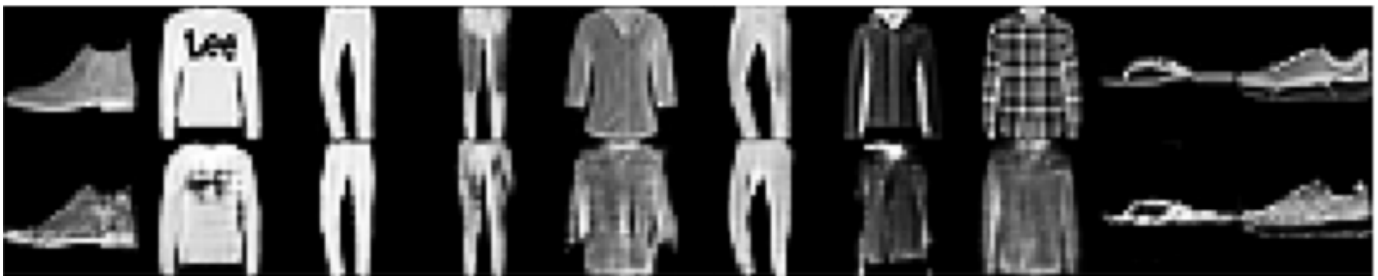
		Adversarial error correction
Attack	No defense	VAE
No attack	93.7	83.4
FGSM ($\epsilon = 0.05$)	11.4	66.4
FGSM ($\epsilon = 0.1$)	10.8	47.7
PGD ($\epsilon = 0.05$)	3.4	71.5
PGD ($\epsilon = 0.1$)	2.9	60.9

(a) Original images

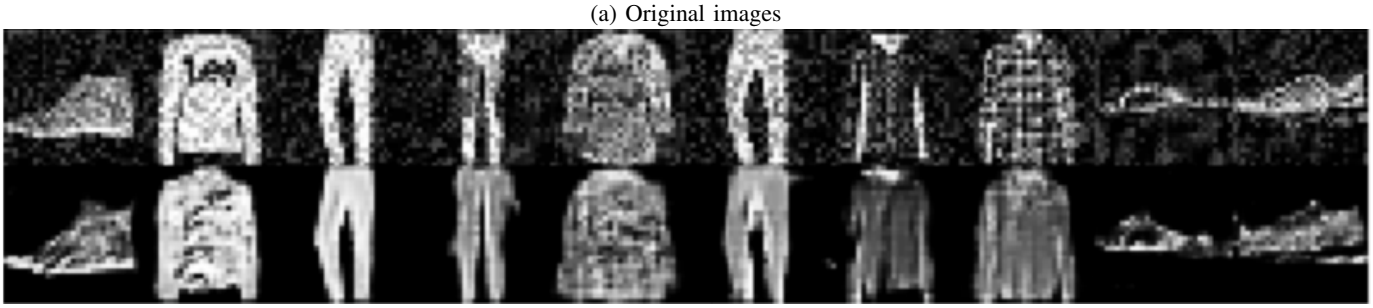


(b) FGSM with $\epsilon = 0.3$

Fig. 1: Fashion MNIST dataset comparing original and adversarial examples generated using FGSM[1] attack



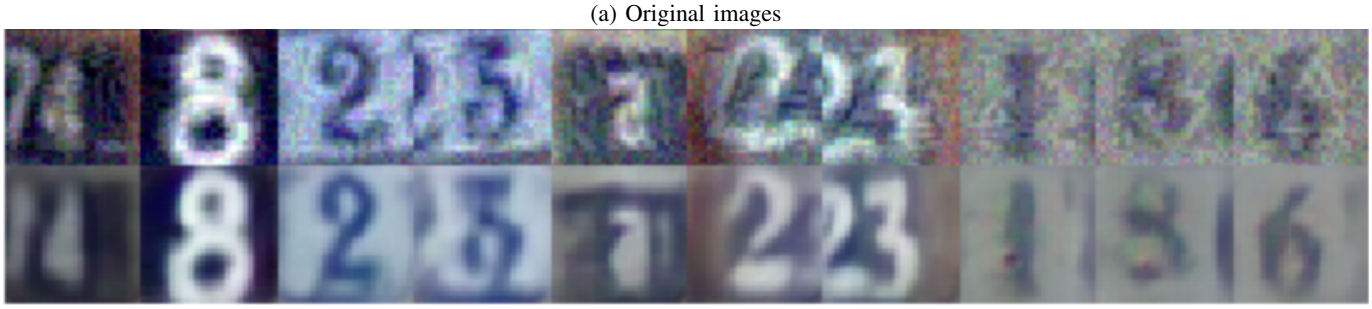
(a) Original images



(b) PGD with $\epsilon = 0.3$

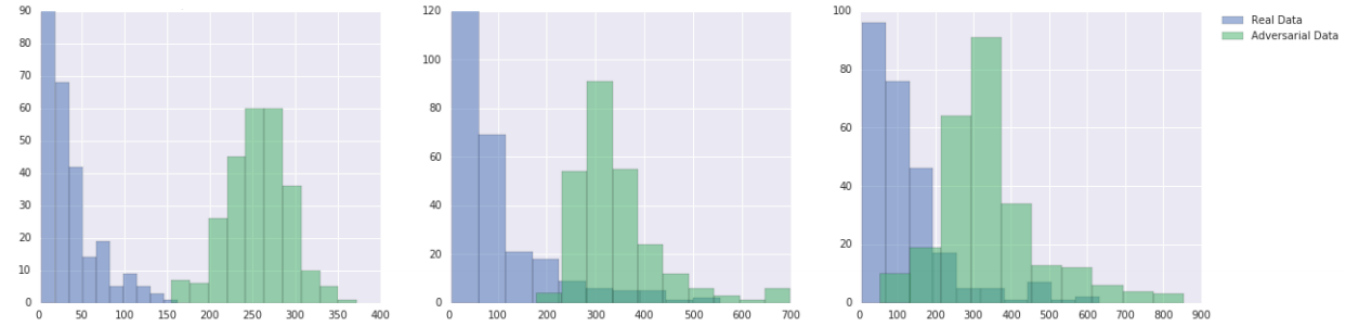
Fig. 2: Fashion MNIST dataset comparing original and adversarial examples generated using PGD[2] attack

(a) Original images

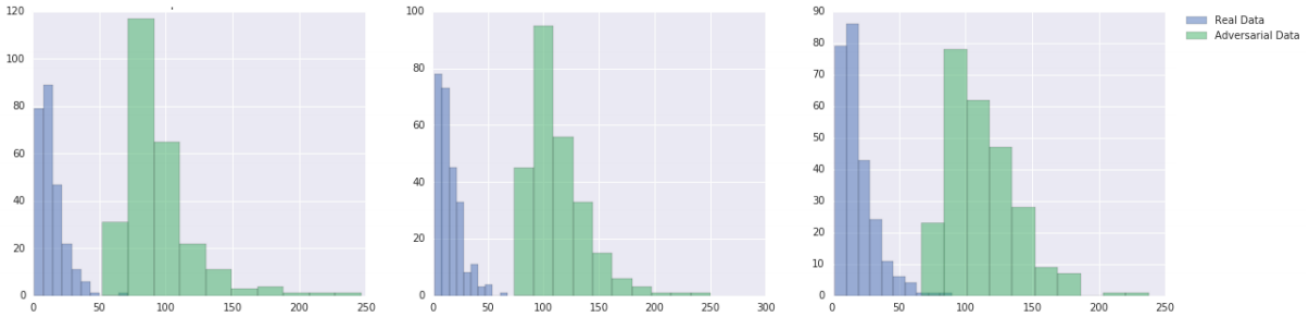


(b) PGD with $\epsilon = 0.3$

Fig. 3: SVHN dataset comparing original and adversarial examples generated using PGD[2] attack



(a) Fashion MNIST



(b) SVHN

Fig. 4: x-axis is the L_2 norm of the real and adversarial latent spaces and y-axis is the number of samples. Clearly, the peaks come closer as the latent dimesionality increased. Latent dimensionality increases from left to right (32, 64, 128).