

Defense-PointNet: Protecting PointNet Against Adversarial Attacks

Yu Zhang^{1*}, Gongbo Liang¹, Tawfiq Salem², Nathan Jacobs¹

1 Department of Computer Science, University of Kentucky, Lexington, KY, USA

2 Department of Computer Science and Software Engineering, Miami University, Oxford, Ohio, USA

Email: y.zhang@uky.edu*

Abstract—Despite remarkable performance across a broad range of tasks, neural networks have been shown to be vulnerable to adversarial attacks. Many works focus on adversarial attacks and defenses on 2D images, but few focus on 3D point clouds. In this paper, our goal is to enhance the adversarial robustness of PointNet, which is one of the most widely used models for 3D point clouds. We apply the fast gradient sign attack method (FGSM) on 3D point clouds and find that FGSM can be used to generate not only adversarial images but also adversarial point clouds. To minimize the vulnerability of PointNet to adversarial attacks, we propose Defense-PointNet. We compare our model with two baseline approaches and show that Defense-PointNet significantly improves the robustness of the network against adversarial samples.

Index Terms—point cloud, adversarial attack, pointnet, defensive network

I. INTRODUCTION

Convolutional neural networks (CNNs) achieve remarkable performance across a broad range of image-related tasks [1] [2] [3], but CNNs have been shown to be vulnerable to adversarial attacks [4] [5] [6] [7]. Goodfellow *et al.* [8] propose the fast gradient sign attack method (FGSM) for generating adversarial samples and claim that CNNs can be easily misled with high confidence by adding imperceptible perturbations to real input images. Recently, researchers propose multiple ways to generate adversarial samples for 2D images and explain this phenomenon theoretically [9] [10]. Besides generating adversarial samples, some recent works focus on how to protect CNNs against adversarial attacks. ShieldNets [11] proposes a probabilistic method to defend against adversarial attacks on 2D images. Defense-GAN [12] models the distribution of unperturbed images, and Erraqabi *et al.* [13] propose an adversarially augmented adversarial training (A3T) approach to improve the adversarial robustness of CNNs by using a discriminator to filter the adversarial noise and achieve good performance on MNIST [14].

Many works focus on adversarial attacks and defenses on 2D images, so one question naturally arises: How can we transfer these attacking and defending approaches from 2D images to 3D real world data? There are many ways to represent 3D objects in the real world. In this work, we focus on 3D point clouds, which can be obtained from LiDARs and depth cameras. Some recent works [15] [16] [17] use neural networks to process point cloud data and achieve great

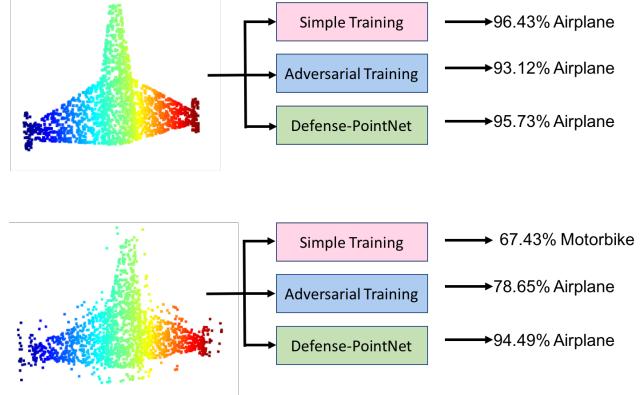


Fig. 1. The upper sample is a clean airplane point cloud and the lower sample is its adversarial counterpart generated by FGSM. Both samples are fed into three models. For the real sample, all three approaches classify correctly but for the adversarial sample, it fools the simple training model.

success. Among these works, PointNet [15] is the earliest and also the most commonly used architecture for 3D point cloud data, and we show that it can be attacked by adversarial samples. Figure 1 illustrates that when the PointNet is misled by the adversarial sample, our model can still make the correct prediction.

Previous works [15] [18] show that PointNet is more difficult to attack than ordinary CNNs. It has been proven that PointNet is robust against multiple kinds of perturbations. We validate this by showing that PointNet can achieve high accuracy as 86.57% when attacked by FGSM with $\epsilon = 0.1$ and 64.78% with $\epsilon = 0.2$. One major difficulty for generating adversarial point clouds is that we cannot simply use most of the attacking approaches designed for 2D images because it is impossible for us to modify pixel values for point clouds unless we have point-level features. Xiang *et al.* [18] propose multiple approaches to generate adversarial point clouds by shifting points or adding extra points to the original point cloud. In this work, we focus more on how to improve the adversarial robustness than how to generate adversarial samples.

To protect PointNet against adversarial attacks, we propose Defense-PointNet which uses a discriminator to learn to filter the adversarial noise in the latent representation space. We

split the PointNet into two parts, the feature extractor and the classifier. We attach the discriminator to the last layer of the feature extractor and train these three modules jointly. Our discriminator is trained to distinguish latent representations of real point clouds from the ones of adversarial point clouds. The feature extractor is trained not only to extract features for the classifier to correctly classify the training samples but also to fool the discriminator.

We evaluate our Defense-PointNet on ShapeNet [19], a well-known 3D point cloud dataset, and find that it outperforms two baseline approaches and improves the adversarial robustness of PointNet. We also find that Defense-PointNet can give higher probability on correctly classified samples comparing with the traditional adversarial training approach, which means Defense-PointNet can predict not only more accurately but also more confidently. We apply t-Distributed Stochastic Neighbor Embedding (t-SNE) [20] for dimensionality reduction and use it to visualize the latent vectors. Our experiments show that Defense-PointNet can enhance the intra-class compactness of feature clusters, thereby reducing the overlap of different classes, leading to the improvement of robustness against bounded input perturbations. The key contributions of this work are:

- Adversarial Point Clouds Generation: We apply fast gradient sign attack method to point clouds and find that even slightly shifting the points can mislead PointNet to give incorrect predictions with high confidence.
- Adversarial Training: By feeding real batches and adversarial batches alternatively in the network during training, the adversarial robustness of PointNet is improved significantly. We use this adversarial training approach as one baseline for evaluation.
- Defense-PointNet: We propose the Defense-PointNet which can protect PointNet against adversarial attacks. By adding a discriminator, we enforce resistance to adversarial attack of latent representations and improve the adversarial robustness of PointNet.
- Visualization: We provide t-SNE plots to explain how our approach affects the feature space representations.

II. RELATED WORK

A. Fast Gradient Sign Attack Method

Fast gradient sign attack method (FGSM) [8] is the earliest and most fundamental technique for generating adversarial samples. Many following attacking methods, for instance, Basic Iterative Method (BIM) [21], Momentum Iterative Method (MIM) [22], Carlini & Wagner Attack (C&W) [6], and Projected Gradient Descent (PGD) [23] are all based on FGSM or using FGSM as one of their steps. In FGSM, we calculate the gradient of the cost with respect to the input pixels and generate a perturbation matrix with same dimension of the input images. By adding the perturbation matrix to the input images, we can generate adversarial images with limited perturbations. The adversarial samples it generates can mislead the CNNs to predict incorrect labels with high confidence.

Our main focus is to provide a defending strategy against the fundamental but powerful FGSM. In our paper, we directly apply FGSM on 3D point clouds, which means we shift points instead of modifying pixel values.

B. Defense-GAN

Defense-GAN [12] is a new framework leveraging the expressive capability of generative models to defend deep neural networks against adversarial attacks. Defense-GAN is trained to model the distribution of unperturbed images. At inference time, it finds a close output to a given image which does not contain the adversarial changes, then feed this image to the classifier. It has been proven that Defense-GAN is effective against different attack approaches and improves on existing defending strategies.

C. A3T

The original idea of augmenting a network with a discriminator to make hidden representation filter the noise is the essence of the work Ganin *et al.* [24] on domain adaptation, where the network learns features that adapt to different domains for the same task. That idea is exploited by Adversarially Augmented Adversarial Training (A3T) [13], which divides a CNN into an encoder and a residual classifier. A discriminator is used to enforce resistance to adversarial attack of latent representations. The A3T approach has been evaluated on MNIST and achieved higher accuracy than using the adversarial training approach. The success of these two works [13], [24] on 2D images inspire us to augment deep neural networks (DNNs) for 3D point cloud data and design the Defense-PointNet.

D. PointNet

Unordered point sets in 3D are usually considered to be difficult to model by using DNNs. PointNet [15] is the first paper using DNNs to model 3D point cloud data. PointNet uses max pooling [25] to reduce the unordered and varying length input to a fixed-length global feature vector make it possible for end-to-end training. In our paper, we show that PointNet can still be attacked using adversarial samples generated by the fundamental approach FGSM, and we propose the Defense-PointNet to protect PointNet against adversarial attacks.

E. t-SNE

The t-Distributed Stochastic Neighbor Embedding (t-SNE) [20] that we used is a technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. The technique can be implemented via Barnes-Hut approximations, allowing it to be applied on large real-world datasets. t-SNE has been approved to be able to apply on data sets with up to 30 million examples [26] [27] [28]. In this work, we use t-SNE to map our high dimensional latent vectors to 2D space for visualization. That helps us explain how our model affects the feature space representations.

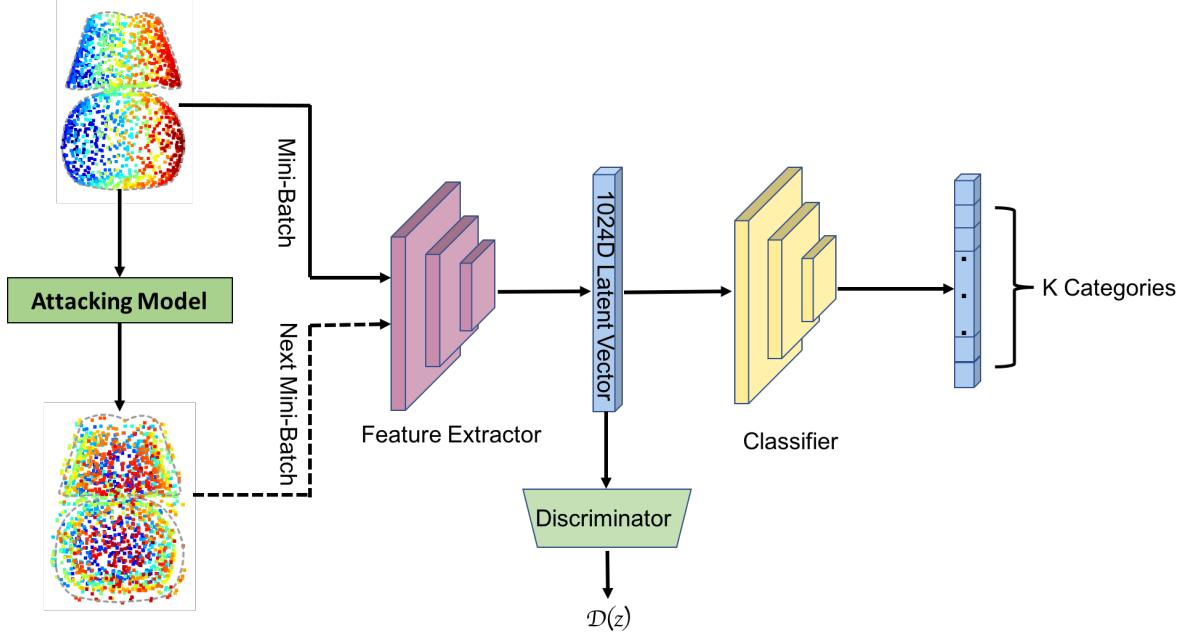


Fig. 2. The Architecture of Defense-PointNet

III. APPROACH

We first use FGSM to attack the PointNet. Then we propose the architecture of our Defense-PointNet. We optimize the parameters of our model by minimizing three loss functions: the classifier loss, the discriminator loss, and the feature extractor loss simultaneously.

A. Adversarial Point Clouds Generation

We generate adversarial point clouds by shifting points using FGSM.

Our PointNet takes as input a mini-batch of real point clouds x and its associated targets y . We then calculate the gradients of that batch of point clouds and add the gradients to x to get perturbed point clouds as adversarial samples x_p as following:

$$x_p = x + \epsilon \nabla_x(\theta, x, y).$$

By adding a small perturbation to the input point clouds x , we get a new mini-batch of point clouds x_p , which is shifted slightly from x and we use x_p as our adversarial point clouds.

B. Adversarial Training

In this part, we train our model on a mixture of clean and adversarial samples. Specifically, for each iteration, we first feed a mini-batch of real point clouds to the network, then generate and feed the corresponding mini-batch of adversarial point clouds alternatively.

C. Defense-PointNet

We extend the adversarial training procedure by proposing the Defense-PointNet. We split the PointNet into two parts. The first part is the feature extractor and the second part is the classifier. A discriminator is attached to the last layer of

the feature extractor. Figure 2 illustrates the architecture of the Defense-PointNet.

The classifier is trained to classify each input correctly and the feature extractor is trained to not only extract features for the classifier but also fool the discriminator. The output of the feature extractor is a 1024D latent vector. That latent vector is the input of the discriminator, which is a two-layer network to enforce an invariance across real samples and their adversarial counterparts at the level of the latent representations. If a latent vector is extracted from a real point cloud, it is labeled as $t = 0$, and if it is extracted from an adversarial sample, then it is labeled as $t = 1$. The discriminator is trained as a binary classifier and it learns to output the probability of the input latent vector's tag is $t = 0$.

We then use the response of the discriminator to train the feature extractor. The feature extractor is trained to fool the discriminator and try to mislead the discriminator to label every real/adversarial vector as real ($t = 0$).

D. Loss Function

To optimize the parameters of Defense-PointNet, we design three different loss functions respectively for the classifier, the discriminator, and the feature extractor. We optimize these three loss functions simultaneously during training.

Classifier Loss: Our first loss, L_{cls} , is the loss of the classifier. The purpose of the classifier is for multi-class classification. We use negative log likelihood as our loss function here as the following equation:

$$L_{cls}(x_{pred}, y) = -\log P(y|x_{pred}).$$

The inputs are the prediction vector x_{pred} and the target y . Notice that L_{cls} is used to update parameters in the entire PointNet, including both the classifier and the feature extractor.

Discriminator Loss: Our second loss, L_{dis} , is the loss of the discriminator. For every latent vector z , the discriminator predicts the probability $D(z) = P(t = 0|z)$ that z is from real ($t = 0$). We use binary cross entropy for the discriminator loss as following:

$$L_{dis}(D(z), t) = -\frac{1}{n} \sum_{i=1}^n [t^i \log(D(z)^i) + (1 - t^i) \log(1 - D(z)^i)].$$

For the real mini-batch, we use $t = 0$; for the adversarial mini-batch, we use $t = 1$.

Feature Extractor Loss: The goal of our feature extractor is not only to extract features for the classifier but also to fool the discriminator. We use the response from the discriminator to train the feature extractor. We use binary cross entropy loss here for the feature extractor as same as what we use for L_{dis} , the only difference is for L_{feat} , we always use $t = 0$ for both real and adversarial mini-batches. The loss function is shown in the following equation:

$$L_{feat}(D(z), t = 0) = -\frac{1}{n} \sum_{i=1}^n [t^i \log(D(z)^i) + (1 - t^i) \log(1 - D(z)^i)].$$

L_{feat} is only used to update parameters in the feature extractor.

IV. EVALUATION

We evaluate our proposed approach quantitatively and qualitatively. We begin by introducing the dataset we use, then we show the implementation details. Finally we show the results of our model and compare our results with different baseline approaches.

A. Dataset

To evaluate our approaches, we use ShapeNet dataset [19], which is one of the most widely used benchmark datasets of 3D point clouds. It is a richly-annotated, large-scale repository of shapes represented by 3D CAD models of objects. In our evaluation, we use a subset, which contains 15,011 3D point clouds belonging to 16 categories. We split the dataset to 80% training and 20% testing. This results in 12,137 samples for training and 2,874 samples for evaluating. The training set distribution is showed in Figure 3.

B. Implementation Details

Our approach was implemented in Pytorch [29] and trained and tested on a Linux computer server with two Nvidia GTX 1080 GPUs. For the FGSM, we use $\epsilon = 0.1$. We split the PointNet into a feature extractor and a classifier. The classifier is a combination of the last three fully connected layers of the

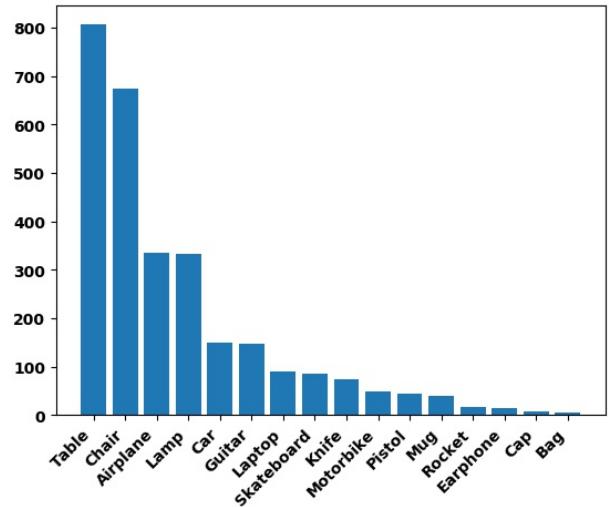


Fig. 3. Training Set Distribution

PointNet and we set the output dimension of FC3 to 16, which is the number of the classes. Anything before the classifier in the PointNet architecture is considered as the feature extractor. The output of the feature extractor is a 1024D latent vector. The discriminator is a combination of 2 linear layers which is attached to the end of the feature extractor. The dimension of the discriminator's input is 1024 and the output dimension is 2. We use softmax as the activation function to predict the probability of the latent vector is real ($t = 0$). We use Adam [30] as our optimizer and decay the learning rate of each parameter group by $\gamma = 0.5$ every 20 epochs.

C. Classification Accuracy

We use simple training and adversarial training approaches as our two baselines. Simple training means we train the PointNet model only on real data (the original ShapeNet data). For the adversarial training, we train the model on the mixture of the original ShapeNet data and the adversarial data generated by us used FGSM. We compare the evaluation results of our approach with these two baselines, and our experiments show the proposed Defense-PointNet outperforms both of these two baselines on testing accuracy.

Table I shows the quantitative testing results on the three compared models. The table reveals when evaluating these approaches on the real samples, all of them achieve reasonably good performance (the accuracy between 93.04% and 96.62%). However, when testing on the adversarial samples, the accuracy of the simple training approach decreases significantly (from 96.96% to 86.57%). The adversarial training

TABLE I
ACCURACY ON TEST SET

Methods	Acc. on real	Acc. on adversarial
Simple Training	96.62%	86.57%
Adversarial Training	93.04%	94.92%
Defense-PointNet	94.08%	96.35%

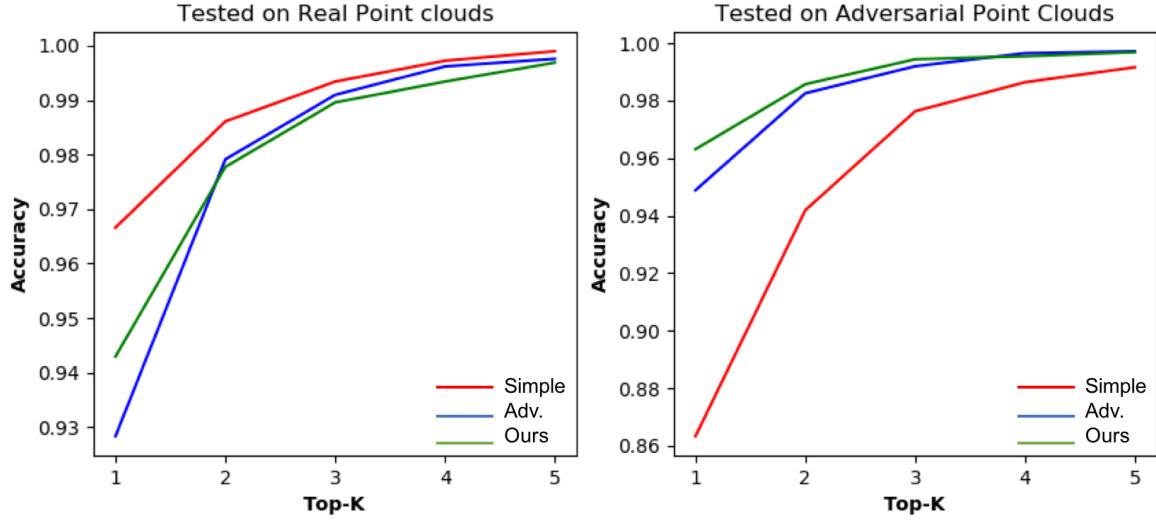


Fig. 4. Top-K accuracy: In each subfigure, the horizontal axis stands for top-K (K in the range $[1, 5]$) and the vertical axis stands for top-K accuracy, which means that the correct prediction gets to be in the top-K probabilities for it to count as correct. (left) Shows all three approaches have similar performance when testing on real samples. (right) Shows our approach outperforms the other two baseline approaches when testing on adversarial samples.

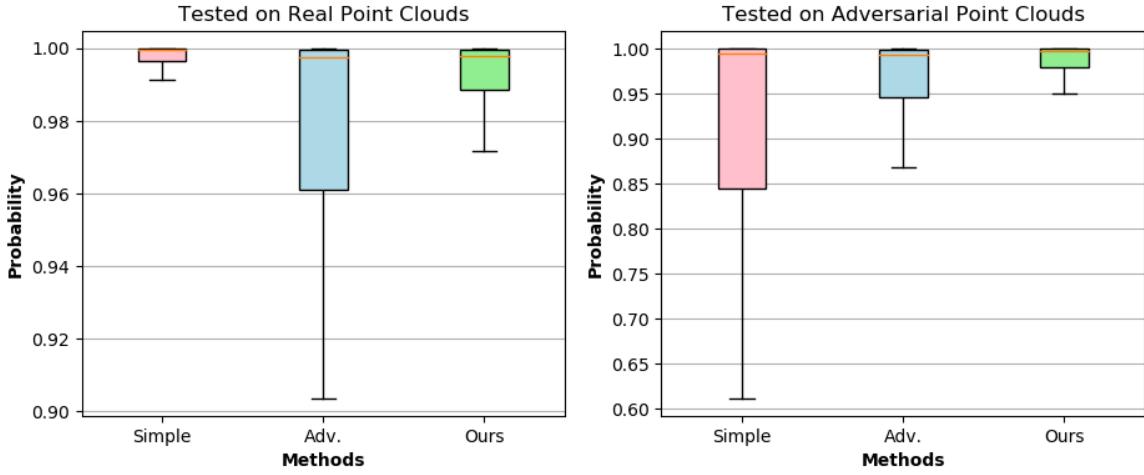


Fig. 5. Two box plots, which show the prediction confidence of the proposed approach and the two baseline approach on the real (left) and adversarial (right) samples.

slightly increases its accuracy to 94.92%, and the performance of our approach is significantly improved to 96.35%. Figure 4 shows the top-1 to top-5 accuracy for all the three approaches on both real and adversarial testing samples. The figure also indicates our approach outperforms the other two baseline approach when testing on adversarial samples.

Figure 4 (left) shows that our approach does not outperform the simple training approach, but this results are based only on real samples, not on adversarial samples. We expect our model to achieve similar performance as the simple training approach when evaluating on real samples only.

D. Prediction Confidence

One important thing we noticed that the proposed approach is not only able to maintain the accuracy level during

an adversarial attack but also able to keep the prediction confident. Figure 5 shows the prediction confidence of the proposed approach and the two baseline approaches on the real and adversarial samples. The figure reveals that on the real samples, the simple training approach generates the most confident prediction results with the predicted probability between 0.99 to 1.0. The proposed approach produces the second most confident prediction results (the predicted probability between 0.97 to 1.0). The adversarial training approach has the worst prediction confidence, which ranges between 0.9 to 1.0. However, when testing on the adversarial samples, the prediction confidence of the simple training decreased dramatically. The lowest prediction confidence dropped to 0.62, which original was 0.99 with the real samples. Similarly,

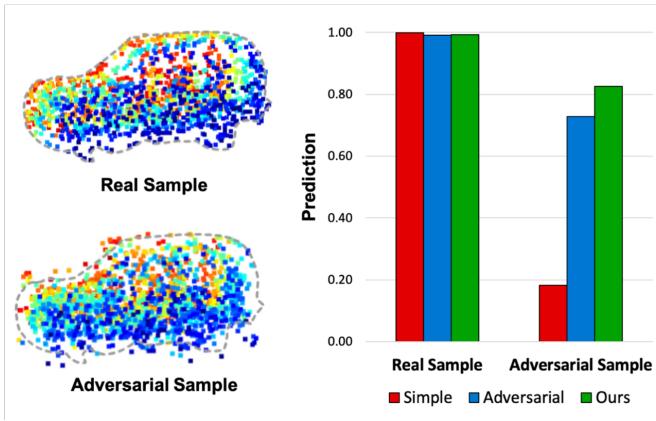


Fig. 6. Classification result of **real car** sample (left-top) and **adversarial car** sample (left-bottom) of different methods.

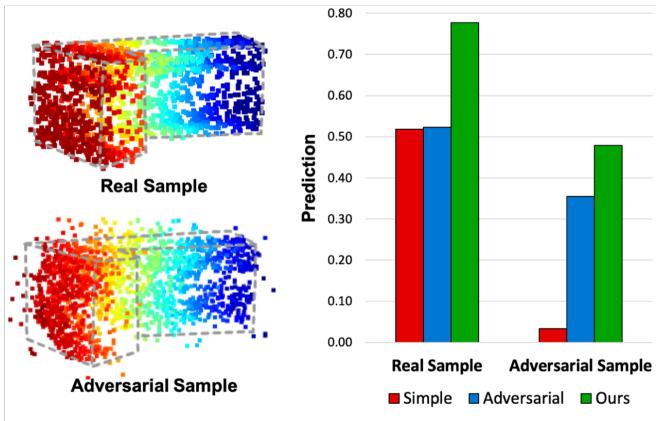


Fig. 7. Classification result of **real table** sample (left-top) and **adversarial table** sample (left-bottom) of different methods.

the prediction confidence of the adversarial training is also decreased significantly. Unlike the baseline models, the proposed approach is able to maintain the prediction confidence at the similar level.

We give two examples to show the prediction confidence. Figure 6 reveals that when testing with the real car sample, all the three models perform an almost perfect prediction. However, when testing on the adversarial car sample, only the proposed approach is able to maintain the prediction confidence. Figure 7 indicates a similar result of the real table sample and the adversarial sample.

E. Feature Space Visualizations

We visualize the 2-D t-SNE plots of 1024D latent vectors extracted by simple training model in Figure 8 and Defense-PointNet model in Figure 9. We can see that, for simple training model, feature clusters of adversarial samples (Figure 8 right) are less separate and the overlap of different classes causes the success of adversarial attack. Defense-PointNet enhances intra-class compactness (Figure 9 right), thereby reducing the feature cluster overlap, leading to a lower adversary success for a bounded perturbation $\epsilon \leq 0.1$.

V. CONCLUSION

We introduced a novel approach for protecting PointNet against adversarial attacks. In several critical experiments, we demonstrated our proposed approach can significantly improve the robustness against adversarial samples of PointNet, as well as maintain the high prediction confidence. We also provided a interpretation of how our model affects the feature space representations by visualizing latent vectors. We hope this work can be able to provide a baseline as well as a guideline for future 3D adversarial attacks, defending strategies and interpretability research.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support of NSF CAREER (IIS-1553116).

REFERENCES

- [1] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 6105–6114.
- [2] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 734–750.
- [3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [4] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, 2019.
- [5] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. 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.
- [6] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 39–57.
- [7] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. 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.
- [8] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.
- [9] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.
- [10] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.
- [11] R. Theagarajan, M. Chen, B. Bhanu, and J. Zhang, "Shieldnets: Defending against adversarial attacks using probabilistic adversarial robustness," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6988–6996.
- [12] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," in *International Conference on Learning Representations*, 2018.
- [13] A. Erraqabi, A. Baratin, Y. Bengio, and S. Lacoste-Julien, "A3t: Adversarially augmented adversarial training," *NeurIPS Machine Deception Workshop*, 2017.
- [14] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [15] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.

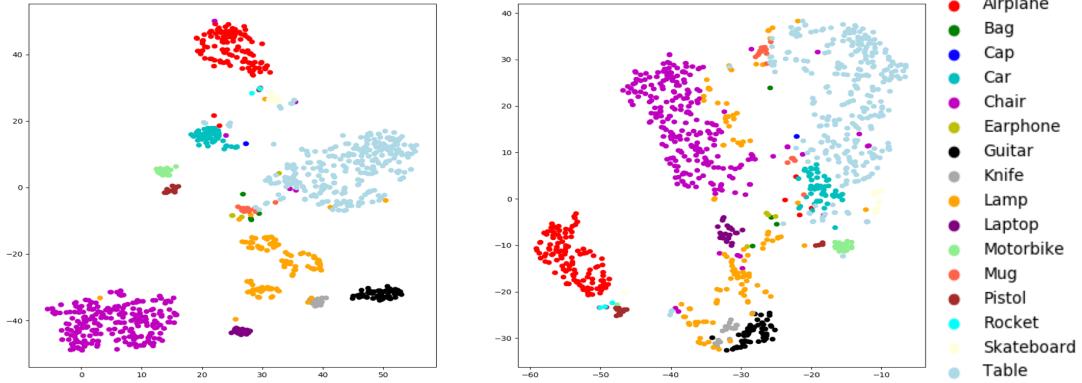


Fig. 8. Two t-SNE plots, which visualize the feature cluster compactness of the simple training baseline approach. The real sample features (left) are more separate while the adversarial samples (right) have more inter-class overlap.

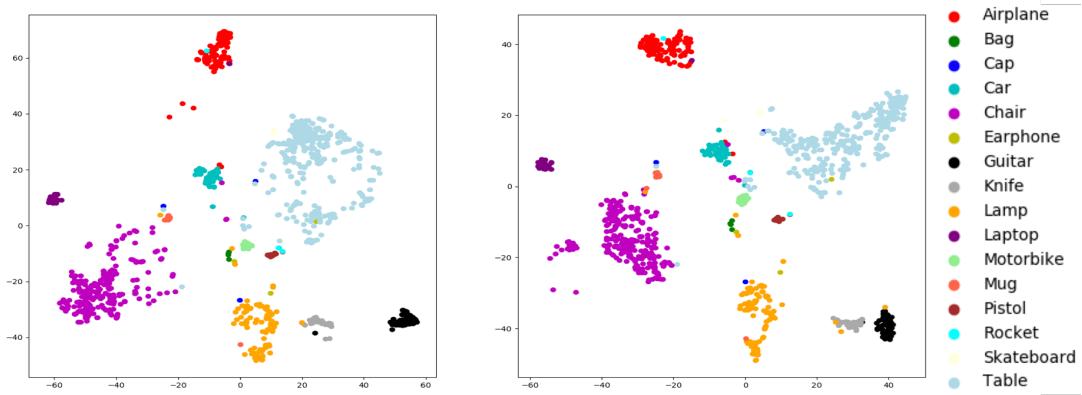


Fig. 9. Two t-SNE plots, which visualize the feature cluster compactness of the Defense-PointNet approach. Our approach enhances intra-class compactness which makes adversarial sample features (right) as separate as real sample features (left).

- [16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.
- [17] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph cnn for learning on point clouds,” *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, p. 146, 2019.
- [18] C. Xiang, C. R. Qi, and B. Li, “Generating 3d adversarial point clouds,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9136–9144.
- [19] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, “Shapenet: An information-rich 3d model repository,” *arXiv preprint arXiv:1512.03012*, 2015.
- [20] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [21] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” *ICLR Workshop*, 2017.
- [22] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, “Boosting adversarial attacks with momentum,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9185–9193.
- [23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” *International Conference on Learning Representations*, 2018.
- [24] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [25] D. Ciresan, U. Meier, and J. Schmidhuber, “Multi-column deep neural networks for image classification,” in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642–3649.
- [26] L. Van Der Maaten, “Accelerating t-sne using tree-based algorithms,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3221–3245, 2014.
- [27] L. Van der Maaten and G. Hinton, “Visualizing non-metric similarities in multiple maps,” *Machine learning*, vol. 87, no. 1, pp. 33–55, 2012.
- [28] L. Van Der Maaten, “Learning a parametric embedding by preserving local structure,” in *Artificial Intelligence and Statistics*, 2009, pp. 384–391.
- [29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.
- [30] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference on Learning Representations*, 2015.