A Hybrid Training-Time and Run-Time Defense Against Adversarial Attacks in Modulation Classification

Lu Zhang, *Graduate Student Member, IEEE*, Sangarapillai Lambotharan, *Senior Member, IEEE*, Gan Zheng, *Fellow, IEEE*, Guisheng Liao, Ambra Demontis, and Fabio Roli

***Abstract*—Motivated by the superior performance of deep learning in many applications including computer vision and natural language processing, several recent studies have focused on applying deep neural network for devising future generations of wireless networks. However, several recent works have pointed out that imperceptible and carefully designed adversarial examples (attacks) can significantly deteriorate the classification accuracy. In this letter, we investigate a defense mechanism based on both training-time and run-time defense techniques for protecting machine learning-based radio signal (modulation) classification against adversarial attacks. The training-time defense consists of adversarial training and label smoothing, while the run-time defense employs a support vector machine-based neural rejection (NR). Considering a white-box scenario and real datasets, we demonstrate that our proposed techniques outperform existing state-of-the-art technologies.**

***Index Terms*—DNNs, adversarial examples, projected gradient descent algorithm, adversarial training, label smoothing, neural rejection.**

I. Introduction

In RECENT years, deep learning (DL) has prompted significant interests in wireless communications. For example, various researchers [1], [2] have successfully applied DL into automatic modulation classification (AMC), which is critical in signal intelligence and surveillance applications including cognitive radio and dynamic spectrum access to monitor spectrum occupancy. Traditionally, the AMC has been achieved by higher-order statistical methods as well as by calculating the compact decision boundary using low-dimensional features that are carefully crafted by the developers. However, AMC can also be implemented by training a deep neural network (DNN) with raw signal samples which allows obtaining highly accurate classifications [3]. The goal of DL-based AMC is to recognize different types of modulations. The input to the DNN is the raw radio frequency (RF) signals that consist of both the in-phase and quadrature components (IQ samples), as in RML2016.10a [4]. Taking this RF signal as input, DNN is expected to output probability score for each possible modulation type BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB, and AM-DSB. The modulation type is determined by the DNN as the one that produces the largest probability score.

Manuscript received February 14, 2022; revised March 8, 2022; accepted March 9, 2022. Date of publication March 15, 2022; date of current version June 10, 2022. This work was supported by the Engineering and Physical Sciences Research Council under Grant EP/R006385/1 and Grant EP/N007840/1. The associate editor coordinating the review of this article and approving it for publication was S. Kafaie. *(Corresponding author: Lu Zhang.)*

Lu Zhang is with the School of Electronic Engineering, Xidian University, Xi'an 710071, China, and also with the Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough LE11 3TU, U.K. (e-mail: l.zhang6@lboro.ac.uk).

Sangarapillai Lambotharan and Gan Zheng are with the Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough LE11 3TU, U.K.

Guisheng Liao is with the School of Electronic Engineering, Xidian University, Xi'an 710071, China.

Ambra Demontis is with the Department of Electrical and Electronic Engineering, University of Cagliari, 09123 Cagliari, Italy.

Fabio Roli is with the Department of Informatics, Bioengineering, Robotics, and Systems Engineering, University of Genova, 16145 Genoa, Italy.

Digital Object Identifier 10.1109/LWC.2022.3159659

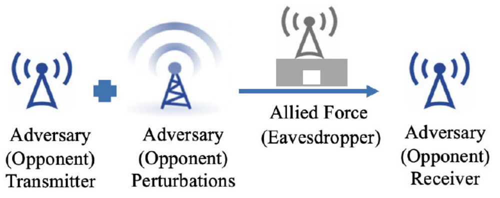

Adversary (Opponent) Transmitter
Adversary (Opponent) Perturbations
Allied Force (Eavesdropper)
Adversary (Opponent) Receiver

Fig. 1. A military scenario for the adversarial examples in modulation classification.

Despite their excellent performance, recent studies discovered that DNNs are vulnerable to adversarial examples, i.e., imperceptible and carefully designed modifications of the input that lead to misclassifications [5]. The earlier works about the vulnerability of DNNs to adversarial examples focus on the computer vision domain. However, Sadeghi and Larsson [6] showed that adversarial examples significantly decrease the classification accuracy also in AMC. Notably, AMC is also applied to critical applications such as military scenarios. During warfares, the units of each adversary (opponent transmitter and receiver as shown in Figure 1) exchange crucial information using radio signals. The allied forces (in this scenario taking the role of eavesdropper) can use AMC to discover the modulation used to intercept messages exchanged between the adversarial units (opponents). In order to deter the allied forces to eavesdrop messages, the adversary units (opponent) can make this modulation discovery more difficult by applying to the communication signals small perturbations, crafted ad-hoc to make the automatic discovery of the modulation eventually performed by the allied forces fail (adversarial examples). The allied forces can still discover the modulation, but in this case, it should have an AMC system that is robust to adversarial examples as considered in this letter.

There are only few works investigating the defense against adversarial examples in AMC. For example, Sahay *et al.* [7]

proposed a deep ensemble defense, an ensemble of different deep learning architectures trained on time and frequency domain representations of the received signals. However, this deep ensemble defense is designed for a black box scenario and investigates the transferability between signal domains which is different from a white box scenario as considered in this letter. In [8], a two-fold defense mechanism, adversarial re-training in conjunction with autoencoder (AE) based detection was proposed. In [9], an AE-based pre-training was proposed, which trains the encoder of the AE using the same architecture as the targeted CNN and then transfers the weights to the CNN classifier. In [8], a CNN is used for modulation classifications. The adversary is able to generate adversarial examples based on the knowledge of the CNN architecture used. However, in order to detect adversarial examples, an AE-based anomaly detector is used by the defense mechanism in [8] (to reject adversarial examples). As our proposed hybrid training-time and run-time defense (HTRD) also works based on the principle of anomaly detection which employs SVM to reject adversarial examples, we consider the work in [8] the most relevant one for performance comparison. Since the generation of adversarial examples in [8] does not consider the AE-based defense, it should be considered as a grey-box attack. Therefore, we have extended the attack scenario in [8] and developed a white-box scenario-based adversarial samples generation using the knowledge of both the CNN and the AE which is compared to our HTRD. More details are given in Section III. In terms of our previous work in [10], a neural rejection (NR) system and its enhancement, which uses label smoothing and Gaussian noise augmentation (LS-GNA), was proposed. However, as Gaussian noise augmentation resembles a white noise-like attack scenario and ignores the inherent features of the CNN architecture, we adopt a stronger form of attack generated by customized adversarial training (CAT) technique [11]. Briefly, CAT generates adversarial perturbations for each sample based on the parameters and architecture of CNN and customizes the perturbation level and the corresponding label during the adversarial training procedure. Our proposed CAT based technique outperforms the LS-GNA based DNN and the work in [8] as demonstrated using the real radio datasets RML2016.10a [4]. The motivation behind the proposed HTRD, which combines the CAT technique and NR system, is as follows. Adversarial training has the potential to maximize the input space margin, and the CAT could increase the lower bound of the margin in the input space. Obtaining a DNN with a larger margin in the input space increases the performance of the run-time defense that we consider in this letter which is an SVM-based NR detector. This detector can classify those samples that have low confidence as adversarial examples. The adversarial examples are pretty near in input space to their original counterpart. If the classifier has a higher margin, it is less likely to misclassify adversarial examples with high confidence. The attacker has to apply higher perturbations to have them misclassified with the same level of confidence, hence more transmission power is needed to succeed with the modulation classification attacks, which will hinder stealth operation of adversarial transmitter.

II. The Proposed Countermeasures Against Adversarial Attacks

A. *Customized Adversarial Training Based DNN*

We propose to employ a robust defense called CAT [11] for modulation classifications, which is a better defense against adversarial examples in AMC as compared to LS-GNA technique. Before delving into details, we first introduce the basics of adversarial training [12], which can be formulated as a min-max optimization. For a K-class classification problem, we denote $D = \{(x_i, y_i)\}_{i=1,\ldots,n}$ as the set of the training samples with $x_i \in \mathbb{R}^d$, $y_i \in \{1, \ldots, K\} =: [K]$. Let $f_\theta(x) : \mathbb{R}^d \rightarrow [K]$ denote a classification model parameterized by θ. The idea of adversarial training is to generate adversarial examples in each iteration of the training process and add them to the training dataset. Formally, adversarial training can be illustrated as: $\min_\theta \frac{1}{n} \sum_{i=1}^{n} \max_{x_i' \in (x_i, \epsilon)} l(f_\theta(x_i'), y_i)$, where (x_i, ϵ) is the l_p-norm ball centred at x_i with radius ϵ, and x_i' indicates the adversarial examples. The inner optimization problem aims to maximize the loss function between the output of the neural network $f_\theta(x_i')$ and the true label y_i so as to obtain an adversarial example for each data sample x_i. For a DNN, the inner maximization problem does not have a closed-form solution, hence, normally a gradient-based iterative solver is adopted. In this letter, we use the PGD attack [12] to solve the inner maximization problem.

However, it is discovered in [11] that a uniformly large ϵ is often harmful to the performance of the adversarial training because the classifier cannot correctly fit both the training data and the adversarial samples generated in this way. Therefore, the classifier will sacrifice its accuracy on some of the original training samples, which causes a distorted decision boundary. Two useful practices to mitigate this problem have been presented in [11]. First, identical and large ϵ for all data samples is avoided. Instead, for samples that are originally closer to the decision boundary, a smaller ϵ is used. This is to ensure that the original normal samples that are close to the decision boundaries do not influence significantly to alter the decision boundary, thereby, not adversely affecting accuracy of the normal data. Second, since a sample with a large perturbation introduces uncertainty on the class it belongs to, label smoothing should be applied instead of one-hot encoding. More label smoothing is applied to samples with higher perturbations.

To address these issues, CAT modifies adversarial training by adapting ϵ that results in adaptive label smoothing for each training sample. We denote ϵ_i as the level of perturbation allocated to each sample, which is calculated as, $\epsilon_i = \arg\min_\epsilon \{\max_{x_i' \in (x_i, \epsilon)} f_\theta(x_i') \neq y_i\}$. It means the ϵ will not be increased further if the adversarial sample can be misclassified within ϵ region. For each sample x_i, the perturbation power is calculated by ϵ_i^2. On the other hand, CAT uses an adaptive label smoothing method which assigns an adaptively smoothed label for each training sample to accommodate a different perturbation tolerance level to each sample. Specifically, given a one-hot encoded label y, the smoothed label is calculated as: $\tilde{y} = (1 - \alpha)y + \alpha u$, where u is a uniformly distributed random variable and $\alpha \in [0, 1]$ controls the smoothing level. For the adaptive setup, $\alpha = c\epsilon_i$ is used so that a larger perturbation tolerance would have a higher label uncertainty and c is a constant. Therefore, the adaptive label is obtained as:

$$\tilde{y_i} = (1 - c\epsilon_i) y_i + c\epsilon_i u. \tag{1}$$

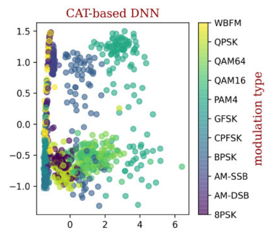
Overall, the objective function of the CAT algorithm is formulated as [11]:

$$\begin{aligned} &\min_{\theta} \; \frac{1}{n} \sum_{i=1}^{n} \max_{x_i' \in (x_i, \epsilon_i)} l(f_\theta(x_i'), \tilde{y_i}) \\ &s.t. \;\; \epsilon_i = \arg\min_{\epsilon} \left\{ \max_{x_i' \in (x_i, \epsilon)} f_\theta(x_i') \neq y_i \right\} \end{aligned} \tag{2}$$

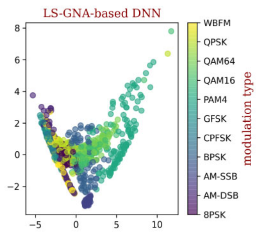
The algorithm for CAT used for modulation classification is adopted from [11]. The difference is that instead of using PGD attack with l_∞-norm constraint, we use PGD attack with the l_2-norm constraint as l_2-norm is a natural choice for AMC since it represents perturbation power. Specifically, during each iteration of the adversarial training, for each training sample, an adversarial example is generated attacking the updated model with the PGD algorithm [12]. After calculating PGD attack for each training sample, the adaptive smoothed label is calculated using (1). The DNN model is updated based on the calculated PGD attack and its corresponding adaptive smoothed label, during which ϵ_i is limited to make sure that the adversarial perturbation is not too large. Finally, if the generated adversarial example x_i' is successful under the current ϵ_i, i.e., $f_\theta(x_i') \neq y_i$, we keep the same ϵ_i for this sample in the next training, otherwise ϵ_i will be increased in the subsequent iteration.

B. Insights Into the Proposed Hybrid Training-Time and Run-Time Defense

From the input space margin maximization perspective, adversarial training with cross-entropy loss approximately maximizes a lower bound of the margin if the perturbation size ϵ is not greater than the soft logit margin [13]. Therefore, as suggested in [13], for adversarial training it is beneficial to start with a smaller ϵ and then to increase it gradually during the training as the lower bound of the margin is maximized at the beginning. The CAT technique considered for modulation classification complies with this finding, therefore, the margin of the last feature layer of the CAT-based DNN (i.e., the input space for the connected SVM) is expected to increase. To visualize this, the principal component analysis is used to reduce the dimension, and the visualization of the last feature layer for both the CAT-based DNN and the LS-GNA-based DNN is shown in Figure 2a and Figure 2b, respectively. It can be seen that the classes are more separated for the last feature layer of the CAT-based DNN. This higher separation will produce a larger rejection region for the NR system. A larger rejection region will force the adversary to use more transmission power to attack the modulation classification scheme, which will hinder stealth operation of the adversarial transmitter.



(a) Visualization of the last feature layer of the CAT-based DNN.



(b) Visualization of the last feature layer of the LS-GNA-based DNN.

Fig. 2. Visualization of the last feature layer of the DNN based on CAT and LS-GNA techniques.

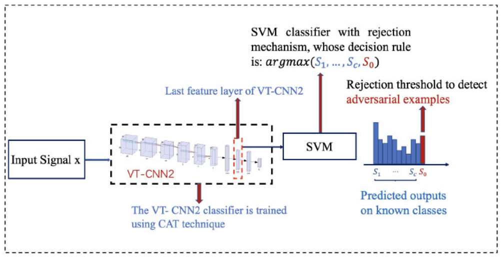

SVM classifier with rejection mechanism, whose decision rule is: $argmax(S_1, \dots, S_c, S_0)$
Last feature layer of VT-CNN2
Rejection threshold to detect adversarial examples
Input Signal x
VT-CNN2
SVM
S_1 ··· S_c S_0
The VT- CNN2 classifier is trained using CAT technique
Predicted outputs on known classes

Fig. 3. The architecture of the proposed HTRD.

C. Hybrid Training-Time and Run-Time Defense

Melis *et al.* [14] proposed a neural rejection (NR) system to detect adversarial examples. The NR system works by establishing a rejection region which could reject the samples that fall within a region of low confidence. The rationale behind the NR system is that there is an amplification effect for the difference between adversarial attacks and benign samples during the propagation of adversarial perturbations through layers of DNN, and this amplification phenomenon is significant at the last layer, hence the NR technique works using the last feature layer of DNN as the features of another classifier, i.e., SVM. The architecture of the proposed HTRD, which combines the benefits of CAT and NR system, is shown in Figure 3. The architecture is the same as that in [14], however, the difference is that DNN was trained using the CAT technique; hence, the DNN parameters would differ from [14]. To train the proposed HTRD, we first train the DNN using the CAT technique, then the NR system is employed on the CAT-based DNN. Specifically, the representations ζ extracted from the last feature layer of the CAT-based DNN are used to train the SVM classifier $S(\cdot)$. Our SVM classifier is a one-vs-all classifier with a radial basis function (RBF)-kernel, and its decision scores (i.e., the output of the SVM) can be presented as:

$$S(x) = \sum_{i=1}^{K} \alpha_i y_i \exp(-\gamma \|x - x_i\|^2) + b \tag{3}$$

where K is the total number of support vectors, α_i are dual variables, γ is the RBF kernel parameter x_i is the support vector and y_i is the class label for each sample x_i'. In order to train an RBF-SVM classifier, the hyper-parameter of SVM, γ, needs to be pre-set. In this letter, we use $\gamma = 0.01$. During the testing phase, given a new input signal x, we take the features

ζ from the pre-trained CAT-based DNN and feed them into the connected RBF-SVM. The decision scores of all the possible classes $S_1(\zeta), \ldots, S_c(\zeta)$ will be obtained from the output of the SVM. The decision function of the HTRD is:

$$c^* = \underset{k=1,\ldots,c}{\operatorname{argmax}} S_k(\zeta), \text{ only if } S_{c^*}(\zeta) > S_0, \tag{4}$$

which means the input signal x will only be correctly classified when the maximum of the decision score S_{c^*} is greater than a pre-defined threshold S_0, otherwise x will be classified as an adversarial example and rejected.

We use adaptive (white-box) attacks to evaluate the performance of the proposed HTRD. Consider that the objective function of HTRD is Ψ. Given a data sample X and the maximum size of the adversarial perturbation ε, an attacker with full knowledge of the defense system can perform white-box attacks by solving the following constrained optimization:

$$\begin{aligned} x^* &= \underset{x':\|x-x'\|_2 \le \varepsilon}{\operatorname{argmin}} \Psi(x') \\ \text{where} \quad \Psi(x') &= s_y(x') - \max_{j \notin \{0,y\}} s_j(x'). \end{aligned} \tag{5}$$

In (5), $\|x - x'\|_2 \le \varepsilon$ is the l_2-norm constraint. Besides, y means the true class of the input data sample, and 0 means the rejection class. Hence, given an input sample X, the adversary aims to minimize the confidence score of X that belongs to the true class while maximizing the confidence score of X that belongs to either rejection class or wrong class so as to achieve the untargeted evasion. To solve (5), the standard PGD algorithm [12] was used. Specifically, the gradient of the objective function $\nabla\Psi(x)$ is first calculated and a standard gradient descent procedure is applied. Then, a projector on the l_2-norm constraint $\|x-x'\|_2 \le \varepsilon$ is adopted to make the new data point within the feasible constraint domain, where $\varepsilon = \sqrt{PNR \cdot \|x\|_2^2/(SNR+1)}$. Here PNR is the perturbation to noise ratio and the SNR is the signal to noise ratio. The iteration will stop when either the objective function converges or the attack can escape the HTRD, i.e., $\|\Psi(x') - \Psi(x)\| \le t$ or $\max_{j\notin\{0,y\}} s_j(x') > \max_{i\in\{0,y\}} s_i(x')$. For our white-box scenario, the attacker has the full knowledge of HTRD achitecture, i.e., the attacker is assumed to have information of both the DNN and the SVM classifier. Specifically, to obtain the gradient $\nabla\Psi(x)$ for the adversarial examples, we first calculate the gradient of the feature vectors of DNN to the input data $\frac{\partial\zeta}{\partial x}$ using automatic differentiation package and calculate the gradient of RBF-SVM $\nabla S(\zeta) = \frac{\partial\Psi(x)}{\partial\zeta}$ as $\nabla S(\zeta) = \sum_i^N -2\gamma\alpha_i y_i \exp(-\gamma\|\zeta - \zeta_i\|^2) * (\zeta - \zeta_i)$. Finally the overall gradient is obtained using the chain rule as $\nabla\Psi(x) = \frac{\partial\Psi(x)}{\partial\zeta} * \frac{\partial\zeta}{\partial x}$.

III. Results and Discussion

We evaluate the performance of our proposed CAT-based DNN and HTRD against white-box untargeted adversarial attacks. To compare the effectiveness of the proposed system, we have implemented the state-of-the-art LS-GNA-based NR system [10] and the two-fold defense system proposed in [8], i.e., adversarial retraining in conjunction with AE detector. To make the comparison fair, we set the relevant thresholds in both the schemes such that the rejection rate of normal benign samples is equal to 10% for both schemes. Furthermore, as mentioned before, when generating adversarial examples, the authors of [8] calculate the gradient of the adversarial training considering only the CNN, which is equivalent to a grey-box attack, i.e., the knowledge of the AE-based anomaly detection was not considered. To consider the white-box scenario in [8], we modified adversarial example generation as follows: We denote the classifier of the adversarial training based CNN and AE detector as $g(\cdot)$ and $h(\cdot)$. For the k_{th} iteration of PGD generation process, we calculate the gradient considering two different conditions. First, if the sample $x'_{(k)}$ could not result in misclassification for the CNN $g(\cdot)$, i.e., $\underset{i}{\operatorname{argmax}}\, g_i(x'_{(k)}) = \mathit{true\ class\ y}$, the gradient of the PGD is generated with the intention of making misclassification, i.e., maximizing the loss function between $g(x'_{(k)})$ and y. The loss function could be the cross entropy loss $CE(g(x'_{(k)}), y)$ or the difference between the logits corresponding to the true class and the most competing wrong class, $g_y(x'_{(k)}) - \max_{j\notin\{y\}} g_j(x'_{(k)})$. Second, if $x'_{(k)}$ could result in misclassification for the CNN $g(\cdot)$ but can not escape from the detection of AE, i.e., $\underset{i}{\operatorname{argmax}}\, g_i(x'_{(k)}) \ne y$ and $\mathrm{MSE}(h(x'_{(k)}), x'_{(k)}) > \mathit{Threshold}$, the gradient of the PGD is generated with the intention of making anomaly detection at the AE fails. If the l_2-norm of adversarial perturbation is larger than the predefined bound ε, the sample $x'_{(k+1)}$ will be projected back to the region such that the adversarial perturbation is within the bound. The PGD iteration will continue until the occurrence of both CNN misclassification and detection of an anomaly at the AE (i.e., $\underset{i}{\operatorname{argmax}}\, g_i(x'_{(k)}) \ne y$ and $\mathrm{MSE}(h(x'_{(k)}), x'_{(k)}) < \mathit{Threshold}$), or until the maximum iteration is reached.

A. Experimental Setup

1) Dataset and Classifier: The GNU radio ML dataset RML2016.10a [4] contains 220,000 input samples, representing 11 different modulations BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB, and AM-DSB. These samples are crafted using 20 different SNR levels from -20dB to 18dB with a step of 2dB. We use half of the dataset as the training set and the rest as the testing set. The same VT-CNN2 classifier as in [6] was used.

2) Parameter Setting: For evaluating the security of the proposed defense system, we used 1000 data samples from the test set, which corresponds to SNR = 10dB. Accordingly, a total of 1000 white-box PGD-based attacks were generated for each algorithm.

3) Security Evaluation: To test the performance, we calculated the accuracy rate against various adversarial perturbations ε (or PNR). When there is no perturbation, i.e., $\varepsilon = 0$, the rejection rate is the misclassification rate of benign input data, i.e., unperturbed data that are wrongly rejected by the HTRD. For adversarial examples, i.e., $\varepsilon > 0$, the accuracy rate indicates the rate of the adversarial examples that are either rejected by the rejection mechanism or correctly classified.

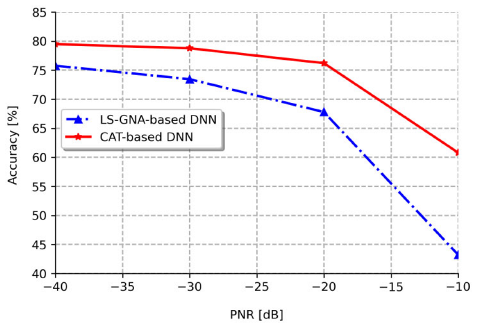


Fig. 4. Accuracy of the CAT-based DNN countermeasure as compared to the LS-GNA-based DNN.

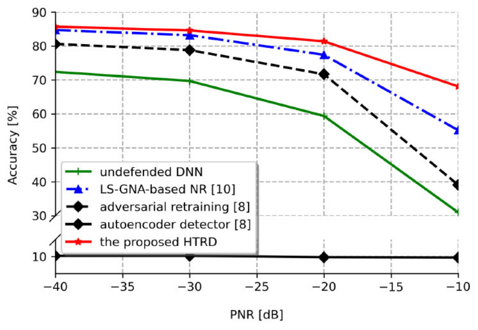


Fig. 5. Accuracy of the proposed HTRD countermeasure as compared to the LS-GNA-based NR countermeasure, the two-fold defense [8] and the undefended DNN.

B. Experimental Results

Figure 4 depicts the performance of DNN based on two different defense techniques: CAT, and LS-GNA [15]. As seen, the CAT-based DNN achieves higher accuracy than the LS-GNA-based DNN for a wide range of PNR. Especially, the improvement is more significant for higher PNR; for example, when PNR $= -10$dB, the CAT-based DNN achieves 17% higher accuracy than the LS-GNA-based DNN. Figure 5 depicts the performance of countermeasures based on the NR system. Specifically, the proposed HTRD outperforms the state-of-the-art LS-GNA-based NR system [10] and the two-fold defense system [8]. The gap becomes larger as PNR increases; for example, for PNR $= -10$dB, it improves about 12% as compared to the LS-GNA based NR system and 30% as compared to the adversarial re-training based DNN. For the white-box scenario, the accuracy of the auto-encoder-based detector is 10%. We have also observed that the normal accuracy for the undefended DNN is 74.5%, and for our proposed HTRD, the normal accuracy is 75.5% without NR and 78.7% with NR. Hence, the proposed HTRD is able to improve the robustness against adversarial examples without sacrificing the normal accuracy.

IV. Conclusion

We have proposed two defense schemes, namely a CAT-based DNN and a HTRD countermeasure for modulation classifications. Using real radio signals, we have shown that the proposed HTRD scheme, based on the adversarial training, label smoothing and neural rejection outperforms the state-of-the-art LS-GNA-based NR scheme and the two-fold defense mechanism. As a result, the adversary will be forced to use more transmission power to fool the HTRD used by the defender.

References

[1] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 168–179, Feb. 2018.
[2] S. Scholl, "Classification of radio signals and HF transmission modes with deep learning," 2019, *arXiv:1906.04459*.
[3] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," 2019, *arXiv:1901.05850*.
[4] T. J. O'Shea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proc. GNU Radio Conf.*, vol. 1, 2016, pp. 1–6.
[5] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–11.
[6] M. Sadeghi and E. G. Larsson, "Adversarial attacks on deep-learning based radio signal classification," *IEEE Wireless Commun. Lett.*, vol. 8, no. 1, pp. 213–216, Feb. 2019.
[7] R. Sahay, C. G. Brinton, and D. J. Love, "A deep ensemble-based wireless receiver architecture for mitigating adversarial attacks in automatic modulation classification," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 1, pp. 71–85, Mar. 2022.
[8] R. Sahay, D. J. Love, and C. G. Brinton, "Robust automatic modulation classification in the presence of adversarial attacks," in *Proc. 55th Annu. Conf. Inf. Sci. Syst. (CISS)*, 2021, pp. 1–6.
[9] S. Kokalj-Filipovic, R. Miller, N. Chang, and C. L. Lau, "Mitigation of adversarial examples in RF deep classifiers utilizing autoencoder pre-training," in *Proc. Int. Conf. Mil. Commun. Inf. Syst. (ICMCIS)*, 2019, pp. 1–6.
[10] L. Zhang, S. Lambotharan, G. Zheng, B. AsSadhan, and F. Roli, "Countermeasures against adversarial examples in radio signal classification," *IEEE Wireless Commun. Lett.*, vol. 10, no. 8, pp. 1830–1834, Aug. 2021.
[11] M. Cheng, Q. Lei, P.-Y. Chen, I. Dhillon, and C.-J. Hsieh, "CAT: Customized adversarial training for improved robustness," 2020, *arXiv:2002.06789*.
[12] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–23.
[13] G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang, "MMA training: Direct input space margin maximization through adversarial training," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–28.
[14] M. Melis, A. Demontis, B. Biggio, G. Brown, G. Fumera, and F. Roli, "Is deep learning safe for robot vision? Adversarial examples against the iCub humanoid," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Jan. 2017, pp. 751–759.
[15] A. Shafahi, A. Ghiasi, F. Huang, and T. Goldstein, "Label smoothing and logit squeezing: A replacement for adversarial training?" 2019, *arXiv:1910.11585*.