Test-time Adversarial Detection and Robustness for Localizing Humans Using Ultra Wide Band Channel Impulse Responses

Abhiram Kolli, Muhammad Jehanzeb Mirza, Horst Possegger, Horst Bischof
Institute of Computer Graphics and Vision, Graz University of Technology, Austria.

***Abstract*—Keyless entry systems in cars are adopting neural networks for localizing its operators. Using test-time adversarial defences equip such systems with the ability to defend against adversarial attacks without prior training on adversarial samples. We propose a test-time adversarial example detector which detects the input adversarial example through quantifying the localized intermediate responses of a pre-trained neural network and confidence scores of an auxiliary softmax layer. Furthermore, in order to make the network robust, we extenuate the non-relevant features by non-iterative input sample clipping. Using our approach, mean performance over 15 levels of adversarial perturbations is increased by 53.3% for the fast gradient sign method and 60.9% for both the basic iterative method and the projected gradient method when compared to adversarial training.**

***Index Terms*—Test time adversarial robustness, ultra wideband (UWB) sensors, channel impulse response, human localization**

I. Introduction

Digital keys or keyfobs are widely used to unlock cars using UWB sensors. Typically, the channel impulse responses (CIRs) of multiple UWB sensors in and around the car are analysed by a neural network to localize these keys and verify the access permissions. However, such systems are very vulnerable to a wide range of attacks on protocol level [1]–[3] and for neural network level [4]. The vulnerabilities of such systems that use neural network can be mitigated by making the networks robust towards the attacks. Traditionally, these approaches rely on adversarial training [5]. Although these methods provide a certain level of robustness, they have plenty of limitations. For example, they require a maximum bound of the attack [6] and need to go through an extensive adversarial training phase [6], which might reduce the test accuracy of the neural networks [6], [7]. However, even after this tedious training process, if the specifications of the attack changes, the network again becomes vulnerable to attacks.

In contrast to adversarial training, an effective alternative is test-time robustness to adversarial attacks which either adapts the model to the incoming inputs or modify the inputs [8], [9]. All these defensive mechanisms are iterative methods which reduce the throughput and also increase the computational resources, thus prohibiting their application in edge computing devices, which are required for the keyfob access verification in vehicles.

This work was funded by the Austrian Research Promotion Agency (FFG) under the research project SEAMAL-Front (No.: 880598).

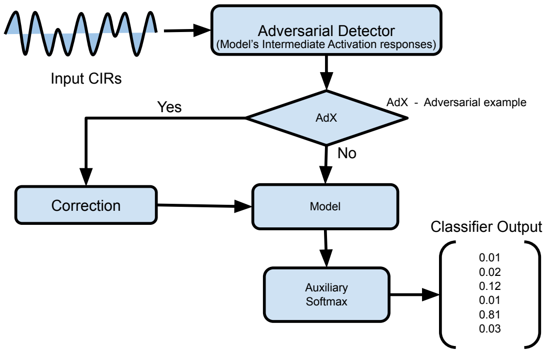

Fig. 1: An overview of the adversarial example detector and robustness architecture.

To address these issues, we propose an adversarial example detector which improves robustness using the model's intermediate responses. We quantify the discrepancy between these intermediate responses and use it as temperature parameter to an auxiliary softmax. This determines the classifier's confidence towards the input signal and whether an input is adversarially perturbed or not. We also propose a novel non-iterative input clipping method for test-time robustness. Since it is a non-iterative method, it improves throughput over iterative methods and is more energy efficient. Our method achieves adversarial robustness by detecting and extenuating a probable adversarial signal in the input sample through intermediate activation responses. We demonstrate the adversarial detection and robustness on fast gradient sign method (FGSM), basic iterative method (BIM) and projected gradient descend (PGD) attacks. An overview of our approach is shown in Fig.1. Several approaches [8]–[18] have studied test-time iterative defences for images but to the best of our knowledge we are the first to propose a test-time non-iterative adversarial detection and defence mechanism for localizing humans within and surrounding a car using UWB CIRs for evasion attacks.

II. Related Work

Our work is related to methods which study conventional adversarial robustness, detection, test-time defences and more specifically to approaches which manipulate the input signal to achieve adversarial robustness at test-time. Since, there are no known direct studies on adversarial attacks and robustness on CIRs for UWB sensors based keyfob location classification, we compare our work with image classification based adversarial attacks and robustness.

A. Adversarial Robustness

Adversarial Robustness is traditionally achieved by adding an adversarial regularization term during training [19], which is broadly termed adversarial training. Madry et al. [20] use a natural saddle point (min-max) formulation during optimization to achieve robustness. Kannan et al. [21] proposed adversarial logit pairing during training for robustness. Gowal et al. [22] study different design choices for training deep neural networks to achieve robustness. Similarly, other works, e.g. [23] achieve robustness by designing a sophisticated method for updating model parameters and the sample through self-supervision [24] and by generative inversion [25].

B. Test-Time Defences

Test-Time Defences for adversarial robustness counter attacks from a different perspective – adapt to adversarial perturbations 'as and when' they are encountered. The goal of these methods is to adapt to an attack which a neural network might become exposed to at test-time.These methods can be divided into two categories. One category updates the model weights at test-time to achieve robustness. For example, DENT [10] minimizes prediction entropy as adversarial examples are encountered during test-time. Chen et al. [11] modify the activation responses for robustness against attackers. Kang et al. [12] design a neural ordinary differential equation and show that they are more robust than ordinary DNNs.

The second category which is more related to our work is achieving test-time defence by *purifying inputs*. For example, Wu et al. [13] achieve input sample purification by attacking all classes simultaneously in such a manner that the misplaced predictions can be flipped into correct ones. Some other approaches, e.g. [14], [15] use generative diffusion models to remove adversarial perturbations. Mao et al. [16] propose to achieve robustness by reversing the adversarial examples by finding the minimum reverse vector through gradient back propagation. Hwang et al. [17] add an auxiliary network for purifying inputs. Alfarra et al. [18] combat adversaries by adding an additional layer which reverses the direction of the attack. Zhu et al. [9] proposed to add saliency maps to the inputs that extenuate the insignificant features.

Test-time robustness approaches [9], [16]–[18] are closely related to our work. However, we differ to them in the following fundamental ways: First, in contrast to [16], [18] we do not search for the bound of attack and then try to mitigate it. Second, contrary to [9], our method does not rely on approaches such as generating saliency maps iteratively to enlarge the distance between original data and the decision boundary. Third, they update pre-trained model's batch norm statistics by forward passing the test samples and use gradients to determine the direction to the original class. Our method obtains the direction by the sign of intermediate responses from the network. Fourth, Zhu et al. [9] proposed a method which takes several steps to predict the adversarial distortion and the direction of the gradients. This might lead to either vanishing or exploding gradients. Since our method is non-iterative, this scenario does not arise. Fifth, multi-step approaches are not desirable in resource constrained embedded devices where UWB pulses arrive with the repetitive frequency between 1 to 50 MHz [26]. Sixth, unlike in [17], though we also purify the inputs we do not use any auxiliary neural network. We only use an auxiliary softmax layer which does not require any training.

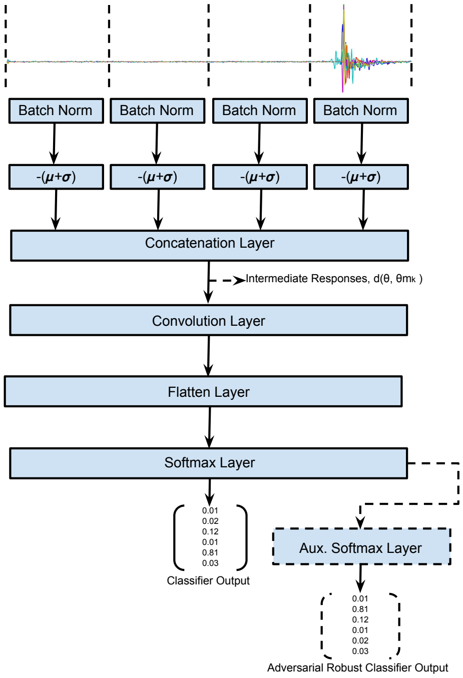

Fig. 2: An overview of our network architecture for adversarial example detector and robustness.

III. Methodology

In this section, we detail the architecture of our pre-trained network (Fig. 2), the mechanism of adversarial detection and adversarial robustness.

A. Architecture

Our signal classification model (Fig. 2) consists of an input layer which receives the input UWB CIRs of dimension B × N × S, where B is the batch size, N is the number of sensors, S is the CIR for each sensor, 1024 in this case. This is followed by 4 parallel blocks (k = 0, 1, 2, 3) where the input to each block is a B × N × 256 patch of non-overlapping input tensor. Each block consists of a batch normalization layer followed by the mean (μ) and variance (σ) subtraction layers along the S dimension of the input (Fig. 2). The output of these modules are concatenated and fed to a single convolution layer consisting of a single filter of size 3 × 3 with L2 regularization. This is followed by a flattened layer and a softmax layer. The intermediate responses

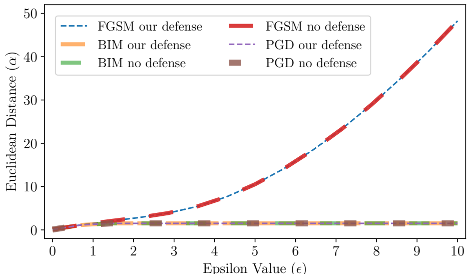

Fig. 3: Euclidean distance between the intermediate responses of the network with respect to adversarial perturbations on the training set. With the increase in perturbation higher is better.

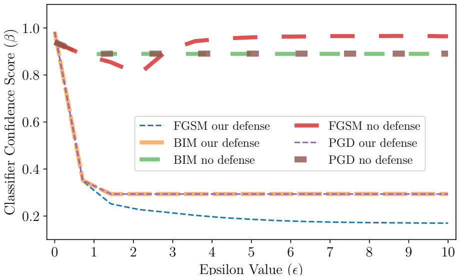

Fig. 4: Confidence scores of the classifier with respect to adversarial perturbations on the training set. With the increase in perturbation lower is better.

of the network (dashed arrows in Fig. 2) are used for both adversarial detection and robustness. An auxiliary softmax layer (dashed box in Fig. 2) uses the intermediate responses to quantify our classifier's confidence and outputs the robust classification results.

B. Adversarial detection

During test-time our classifier detects if an incoming sample is an adversarial example or not. We do this by taking the Euclidean distance between the pre-trained model's intermediate responses after the concatenation layer (Fig. 2).

$$d(\theta, \theta_{m_k}) = \|f(x, \theta) - f(x, \theta_{m_k=0})\|, \tag{1}$$

where, $f(\cdot)$ is the output of the pre-trained model's intermediate response after the concatenation layer, x is the single input sample, θ represents the weights of the intermediate response, m_k is the momentum of the batch normalization layer, $k \in [0, \ldots, 3]$ is the block number of batch normalization layer as illustrated in Fig 2. The distance between clean samples is expected to be near 0 as the network is pre-trained with clean samples and increases for adversarial samples (see Fig.3). This distance is also the temperature parameter for the auxiliary softmax layer which computes

$$\hat{y}(x_i) = \frac{\exp(\frac{y_i}{d(\theta, \theta_{m_k}) + \zeta})}{\sum_j^N \exp(\frac{y_i}{d(\theta, \theta_{m_k}) + \zeta})} \tag{2}$$

where, $\hat{y}(x_i)$ is the output of the auxiliary softmax for class i, y_i is output of networks softmax layer for class i, N is the total number of classes, ζ is for numerical stability (not used in these experiments). We take the Euclidean distance as the temperature parameter because for an incoming clean test sample the distance is smaller so the confidence of prediction remains mostly unchanged (see Fig. 4). When the input is an adversarial example, this distance increases which decreases the confidence of the model's prediction.

An input signal is considered as adversarial if

$$\hat{x} = \begin{cases} \text{clean} & \text{if} \, (\alpha < 0.2) \wedge (\beta > 0.98) \\ \text{adversarial} & \text{otherwise} \end{cases} \tag{3}$$

where, α is Euclidean distance from (1) and β is the confidence score of the auxiliary softmax from (2). The threshold values of 0.2 and 0.98 are y-axis values for clean samples as illustrated in the Fig. 3 and Fig. 4 respectively.

C. Adversarial robustness

After passing through the adversarial detection block, we clip those input samples which are flagged as adversarial sample by the detector.

$$\hat{x} = \begin{cases} x * (\sigma(\gamma) - \text{sgn}(\gamma)) & if \, (\alpha > 0.2) \vee (\beta < 0.98) \\ x & \text{otherwise} \end{cases} \tag{4}$$

where, x is the incoming sample, σ is the sigmoid activation function. $\gamma = f(x, \theta) - f(x, \theta_{m_k=0})$ is the difference of intermediate responses of the module stated in Section III-B.

IV. RESULTS

We evaluate 3 adversarial defensive approaches, namely standard adversarial training, our noniterative adversarial clipping method without adversarial detector and our noniterative adversarial clipping method with adversarial detector on a custom dataset. The dataset consists of 461 samples where each sample consists of N=6 CIRs representing 6 sensor outputs. Among these 6 sensors, 4 sensors are deployed surrounding the car and 2 are placed within the car. These 6 CIRs together determine the presence of a keyfob into namely *back*, *back seat*, *driver seat*, *front*, *left* and *right*. For all the experiments in this paper, we used randomly shuffled 231 samples to pre-train the network and the remaining 230 samples are subjected to adversarial perturbations. We modeled three types of adversarial attacks namely fast gradient sign method, basic iterative methods and projected gradient descent. These attacks are modeled as mentioned in [27]. A total of 15 levels per each type of attack are used where level1 (ϵ = 0) is clean sample and level15 is highest adversarial perturbation (ϵ = 10). The intermediate values of ϵ used are linearly spaced between 0 through 10. 15 levels of adversarial attacks are used to study the effect of adversarial attacks on our defence mechanism in fine detail.

We trained the network with raw input samples as opposed to normalised inputs. This is due to the observation that

TABLE I: Comparison of adversarial example detection of a pre-trained neural network without and with our defensive mechanism. Here, L1 is without any distortion and L15 is the highest distortion level. 'Source' is the model without our defensive mechanism.

Distortion level	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15
Source (FGSM)	0.65	0.65	0.66	0.66	0.66	0.65	0.64	0.63	0.62	0.62	0.62	0.63	0.63	0.64	0.65
Source (BIM)	0.78	0.77	0.78	0.78	0.78	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79
Source (PGD)	0.78	0.77	0.78	0.78	0.78	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79	0.79
Our (FGSM)	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Our (BIM)	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Our (PGD)	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

TABLE II: Comparison of adversarial robustness of a pretrained neural network without and with our defensive mechanism. Here, L1 is without any distortion and L15 is the highest distortion level. 'Source' is the model without our defensive mechanism. 'Source (attack method)$_{AdX}$' represents adversarial trained network's performance. Bold values and underlined values are best and second best among adversarial defensive mechanisms.

Distortion level	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15
Source (FGSM)	0.93	0.80	0.67	0.28	0.09	0.06	0.06	0.06	0.06	0.06	0.06	0.06	0.07	0.07	0.07
Source (BIM)	0.93	0.80	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Source (PGD)	0.93	0.80	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Source (FGSM)$_{AdX}$	0.57	0.38	0.23	0.22	0.22	0.22	0.22	0.22	0.21	0.21	0.21	0.21	0.20	0.20	0.19
Source (BIM)$_{AdX}$	0.59	0.36	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18
Source (PGD)$_{AdX}$	0.72	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18	0.18
Our (FGSM)$_{WoD}$	$\underline{0.89}$	0.83	0.77	0.65	0.56	0.57	0.60	0.70	0.77	0.84	0.88	0.89	0.90	0.90	0.90
Our (BIM)$_{WoD}$	$\underline{0.89}$	0.84	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82
Our (PGD)$_{WoD}$	$\underline{0.89}$	0.84	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82
Our (FGSM)$_{WD}$	**0.92**	0.83	0.77	0.66	0.56	0.57	0.61	0.70	0.77	0.83	0.88	0.90	0.90	0.90	0.90
Our (BIM)$_{WD}$	**0.92**	0.84	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82
Our (PGD)$_{WD}$	**0.92**	0.83	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82	0.82

normalised inputs performed similar to clean sample scenarios but are very vulnerable to adversarial perturbations. Since the input sample's min max values range between -3000 to 3000 (linear scale), we selected the adversarial perturbation magnitude between 0 to 10. Thus ϵ range is usually very high when compared to adversarial attacks on images which remains between $\frac{8}{255}$ to $\frac{64}{255}$. Another reason for having very high ϵ is to test whether the hyperparameters (threshold values) α and β in (3) hold good over large varying incoming input samples which they have never seen.

A. Adversarial Detection

Table I shows the performance of our detector for all classes various adversarial distortions. Performance of source model (without defence mechanism) is around 65% for FGSM attack. With our approach (see (3)) the network is able to detect with an accuracy of 98% or above for FGSM attack. The performance of source model is around 79% for both BIM and PGD attacks. Our mechanism is consistently 98% and above with all types of attacks as shown in Table I.

B. Adversarial Robustness

The classifier's adversarial robustness performance is shown in Table II (output of (4)). Performance of the source model (without defence mechanism) drops immediately even with small adversarial perturbations for FGSM attack. Adversarial training (see Table II, row, Source (attack method)$_{AdX}$), a standard defensive mechanism where the network is trained with $\epsilon \in [0, 5, 10]$, the accuracy of clean samples dropped due to label leaking. However, accuracy over other distortion levels improved by over 10 to 15%.

On the contrary, our defensive mechanism (see Table II, row, Our (attack method)$_{WD}$) performed better by more than 30%. Though the accuracy of our model drops to 56% at level5, it increases and stabilizes at 90% for higher levels of FGSM adversarial perturbations. This phenomenon can be attributed to the usage of the sigmoid activation function in our input clipping function (see (4)). FGSM attack has a mean performance of 78% over 15 levels with our defence mechanism whereas it is just 22.6% for source model. This is a 55% improvement over the source model. The performance of the network remains consistently higher for BIM and PGD attacks at 6% higher than the source model. The drop in accuracy was not similar for BIM and PGD attacks. This could be attributed to not much change in (1) (see Fig.3) for different levels of adversarial perturbations.

The performance of another method, Our (attack method)$_{WoD}$ has 89% accuracy on clean samples and for other distortion levels remains the same as with our previous defensive mechanism. This demonstrates that even if the adversarial detector (see (3)) fails to detect the incoming sample correctly the network's performance does not get adversely impacted.

V. Conclusion

We show that adversarial examples can be detected with high accuracy even at low and high magnitudes of adversarial perturbations. With this performance level of adversarial detection, our system can also be used to discard the samples without classifying them. We also achieve high adversarial robustness with noniterative clipping function even for higher ϵ values. Since our approach is a noniterative method, it provides better throughput over iterative methods and better energy efficiency. Due to its very low memory foot print (162.1 KB, tflite model without quantization), our model with defensive mechanism can be deployed on an edge computing device that can be deployed on a vehicle.

References

[1] M. Singh *et al.*, "Security analysis of ieee 802.15. 4z/hrp uwb time-of-flight distance measurement," in *Proceedings of the ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2021, pp. 227–237.

[2] P. Leu *et al.*, "Ghost peak: Practical distance reduction attacks against hrp uwb ranging," in *USENIX Security Symposium*, 2022, pp. 1343–1359.

[3] J. Classen *et al.*, "Evil never sleeps: When wireless malware stays on after turning off iphones," in *Proceedings of the ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2022, pp. 146–156.

[4] C. Szegedy *et al.*, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[5] T. Bai *et al.*, "Recent advances in adversarial training for adversarial robustness," *arXiv preprint arXiv:2102.01356*, 2021.

[6] S.-M. Moosavi-Dezfooli *et al.*, "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.

[7] Y.-Y. Yang *et al.*, "A closer look at accuracy vs. robustness," *Advances in neural information processing systems*, vol. 33, pp. 8588–8601, 2020.

[8] F. Croce *et al.*, "Evaluating the adversarial robustness of adaptive test-time defenses," *arXiv preprint arXiv:2202.13711*, 2022.

[9] Y. Zhu *et al.*, "Efficient adversarial defense without adversarial training: A batch normalization approach," in *International Joint Conference on Neural Networks*. IEEE, 2021, pp. 1–8.

[10] D. Wang *et al.*, "Fighting gradients with gradients: Dynamic defenses against adversarial attacks," *arXiv preprint arXiv:2105.08714*, 2021.

[11] Z. Chen *et al.*, "Towards robust neural networks via close-loop control," *arXiv preprint arXiv:2102.01862*, 2021.

[12] Q. Kang *et al.*, "Stable neural ode with lyapunov-stable equilibrium points for defending against adversarial attacks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 925–14 937, 2021.

[13] B. Wu *et al.*, "Attacking adversarial attacks as a defense," *arXiv preprint arXiv:2106.04938*, 2021.

[14] Y. Song *et al.*, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766*, 2017.

[15] W. Nie *et al.*, "Diffusion models for adversarial purification," *arXiv preprint arXiv:2205.07460*, 2022.

[16] C. Mao *et al.*, "Adversarial attacks are reversible with natural supervision," in *Proceedings of the International Conference on Computer Vision*, 2021, pp. 661–671.

[17] D. Hwang *et al.*, "Aid-purifier: A light auxiliary network for boosting adversarial defense," *arXiv preprint arXiv:2107.06456*, 2021.

[18] M. Alfarra *et al.*, "Combating adversaries with anti-adversaries," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 5992–6000.

[19] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *Computing Research Repository*, vol. abs/1412.6572, 2014.

[20] A. Madry *et al.*, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[21] H. Kannan *et al.*, "Adversarial logit pairing," *arXiv preprint arXiv:1803.06373*, 2018.

[22] S. Gowal *et al.*, "Uncovering the limits of adversarial training against norm-bounded adversarial examples," *arXiv preprint arXiv:2010.03593*, 2020.

[23] A. Shafahi *et al.*, "Adversarial training for free!" *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[24] M. Naseer *et al.*, "A self-supervised approach for adversarial robustness," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2020, pp. 262–271.

[25] Y. Zhang *et al.*, "The secret revealer: Generative model-inversion attacks against deep neural networks," in *Proceedings of the conference on computer vision and pattern recognition*, 2020, pp. 253–261.

[26] E. M. Staderini, "Everything you always wanted to know about uwb radar...: a practical introduction to the ultra wideband technology," in *Online Symposium for Electronics Engineers*, 2001.

[27] K. Ren *et al.*, "Adversarial attacks and defenses in deep learning," *Engineering*, vol. 6, no. 3, pp. 346–360, 2020.