2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

AdvDetectGPT: Detecting Adversarial Examples Using Large Vision-Language Models

Ming Zhang[0000-0002-3441-7811], Huayang Cao, Cheng Qian(✉)

National Key Laboratory of Science and Technology on Information System Security, Beijing 100101, China
qiancheng@nudt.edu.cn

Abstract. Adversarial examples have been proven to be a substantial threat to the security applications of deep neural networks. Adversarial detection plays a pivotal role in defending against adversarial attacks. While the underlying concept is straightforward, the practical realization of adversarial detection is non-trivial, frequently encountering challenges of universality and effectiveness. In this study, we leverage the powerful capabilities of large vision-language models (LVLMs) and develop AdvDetectGPT, a novel adversarial detector based on LVLMs. AdvDetectGPT can learn to identify adversarial examples directly from clean and adversarial instances, independent of the victim model's outputs or internal responses. The extensive experiments show that AdvDetectGPT significantly outperforms the state-of-the-art baselines. AdvDetectGPT exhibits robust generalization, capable of detecting adversarial examples crafted by novel attacks on new models, as well as those with customized perturbations distinct from the training set. Code is available at https://github.com/mingcheung/AdvDetectGPT.

Keywords: Adversarial detection, Adversarial examples, Deep neural networks, LVLMs.

1 Introduction

Deep neural networks (DNNs), despite their remarkable success, have been demonstrated to be vulnerable to adversarial examples [14]. The adversarial examples, which incorporate carefully crafted perturbations but are virtually indistinguishable from clean examples, can successfully deceive DNNs into making incorrect decisions. This issue poses a serious challenge to the application of DNNs in safety- and security-critical domains. To enhance the ability of DNNs to resist adversarial examples, a spectrum of strategies has been rigorously investigated by researchers. This includes adversarial training that fortifies the learning process [1], structural enhancements that improve the model robustness [24], and the basic yet straightforward method of adversarial detection [2].

As one of the most straightforward defense methods, adversarial detection is simple in concept but not easy to implement [2]. Adversarial detection detects adversarial examples based on the intrinsic differences between adversarial and clean examples. Nonetheless, extracting discriminative features directly from adversarial and clean

examples is often difficult. Numerous studies [6,16,27] have harnessed the model's outputs or internal responses to amplify the differences between adversarial and clean examples, in order to achieve effective adversarial detection. However, the downside of this approach is that the trained detector is generally tailored to protect specific types of victim models.

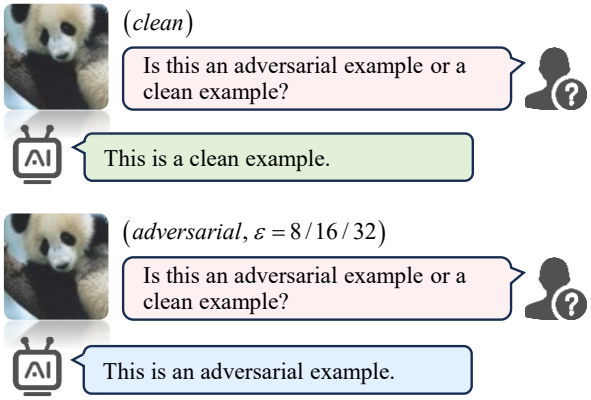

Fig. 1. AdvDetectGPT's adversarial detection via a question-and-answer mechanism.

Large language models (LLMs) have achieved remarkable success in various tasks such as translation, summarization, and question-answering. More recently, the advent of large vision-language models (LVLMs) has further expanded the horizons of artificial intelligence by integrating visual perception with linguistic understanding. Motivated by these advances, we explore the use of LVLMs to address the challenge of adversarial detection, and develop a novel detector, termed AdvDetectGPT, for identifying adversarial examples targeting image classification models. As depicted in Fig. 1, AdvDetectGPT can achieve adversarial detection in a question-and-answer style. In practice, AdvDetectGPT can be easily implemented by fine-tuning the LVLM. It learns to identify adversarial examples directly from clean and adversarial instances, independent of the victim model's outputs or internal responses. The extensive experiments on the ImageNet dataset demonstrate that AdvDetectGPT significantly outperforms the state-of-the-art baselines. AdvDetectGPT exhibits robust generalization, capable of detecting adversarial examples crafted by novel attacks on new models, as well as those with customized perturbations distinct from the training set.

This paper makes the following contributions:

- We propose to utilize large vision-language models to achieve adversarial detection in a question-and-answer style.
- We propose an adversarial detector, named AdvDetectGPT, which is based on LVLMs. It is designed to be easy to train and can achieve high accuracy in detecting adversarial examples targeting image classification models.

- We conduct extensive experiments to evaluate the performance of AdvDetectGPT, demonstrating its excellent generalization in adversarial detection and its superiority over the state-of-the-art baselines.

2 Related Work

In the realm of adversarial detection, prior research primarily utilizes statistical metrics such as kernel density [6], local intrinsic dimensionality (LID) [16] and Mahalanobis distance [15] to capture discriminative features from the model's outputs or internal responses. These methods are limited by their sensitivity to high-dimensional data and noise, which can significantly impact the performance and reliability. Meng and Chen proposed MagNet [17], which includes several detector networks and a reformer network. The detector networks are trained to distinguish adversarial examples from normal ones by modeling the manifold of normal examples. However, MagNet is only validated to be effective on small datasets like MNIST and CIFAR-10. Xu et al. [26] proposed feature squeezing, a technique to detect adversarial examples by reducing the input's feature space and comparing model predictions on original and squeezed inputs. They explored two feature squeezing methods: reducing the color bit depth of each pixel and spatial smoothing. These methods have also only been tested on small datasets like MNIST and CIFAR-10.

Recent advancements employ more sophisticated mechanisms to detect adversarial examples across diverse datasets and models. For example, Yang et al. [27] designed an innovative framework known as ML-LOO, which leverages multi-layer feature attributions to detect adversarial examples. ML-LOO utilizes the leave-one-out (LOO) method and the statistical measures such as the inter-quartile range (IQR) to quantify the dispersion of feature attribution maps. Argos [13] employs a generative model to create multiple views from the input image and the victim DNN's assigned label, and then exploits the inconsistencies between the generated images (i.e., views) and the input image to detect adversarial images. Deng et al. [4] proposed a practical approach called Lightweight Bayesian Refinement (LiBRe), which fortifies deep neural networks with uncertainty quantification by leveraging Bayesian neural networks, thereby achieving robust detection of adversarial examples across different scenarios. Wang and Gong [23] introduced a method for detecting adversarial examples by leveraging multi-layer saliency features. Their method focuses on the distinct evolution of saliency features between clean and adversarial examples across the hidden layers of a model. By analyzing these differences, their method provides an effective way to identify adversarial examples. Yang et al. [28] proposed ContraNet, a novel adversarial example detection framework that leverages the intrinsic contradiction between the semantic information of input samples and the discriminative features extracted by the target model. This framework comprises an encoder, a generator, and a similarity measurement model, which work together to identify adversarial examples by comparing synthetic outputs with original inputs. The authors demonstrated through comprehensive evaluations that ContraNet outperforms existing solutions, particularly under adaptive

attacks, by effectively capturing the semantic inconsistencies that are indicative of adversarial examples.

3 Methodology

3.1 Model Architecture

As shown in Fig. 2, AdvDetectGPT is composed of an image encoder, a text encoder, and an LLM. In this setup, both the image encoder and the text encoder are frozen, while the LLM is fine-tuned using the Low-Rank Adaptation (LoRA) [10] technique. The query image and the human's textual input (i.e., inquiring if the image is adversarial or clean) are processed by the image encoder and the text encoder, respectively, to generate corresponding image and text embeddings. The resulting image and text embeddings are concatenated and subsequently fed into the LLM to yield the final discrimination result. Notably, the image encoder and LLM constitute the core components of AdvDetectGPT, where the former generates preliminary discriminative encoding for adversarial and clean images, while the latter meticulously identifies nuanced discrepancies between their respective encodings.

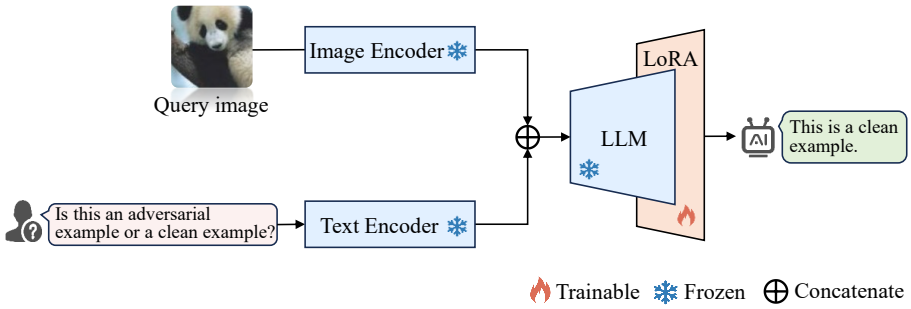

Fig. 2. The architecture of AdvDetectGPT.

3.2 Training Data

To fine-tune the LLM, we need to craft query images along with their respective questions. The query images include clean images and their corresponding adversarial variants, which can be generated using the iterative fast gradient sign method (I-FGSM) [14]. The accompanying question is straightforward and uniform, phrased as "*Is this an adversarial example or a clean example?*". The LLM's responses are programmed as follows: if the query image is clean, it answers with "*This is a clean example.*"; if it is adversarial, it responds with "*This is an adversarial example.*".

The aforementioned question-answer pairs constitute the essential data for fine-tuning the LLM, while the prompts fed into the LLM are slightly more complex and typically follow the format:

<BOS> ### Human: <Img> E_{img} </Img> Is this an adversarial example or a clean example? ### Assistant:

"<BOS>" indicates the start of the input sequence. "### Human:" denotes the portion of the human's input. "### Assistant:" indicates that the model should start responding. $E_{img} \in \mathbb{R}^{1\times d_{model}}$ represents the image embeddings, where d_{model} is the embedding dimension. In the prompt, all elements except for E_{img} are input to the text encoder to generate word embeddings, which are then concatenated with E_{img} to form the complete input for the LLM.

3.3 Image and Text Encoders

To achieve the adversarial detection var a question-and-answer mechanism, AdvDetectGPT integrates an image encoder and a text encoder to effectively process multi-modal inputs. The image encoder transforms the query image into image embeddings. In practice, this can be accomplished using multi-modal encoders like ImageBind [7], or by employing pre-trained image recognition models, such as the Vision Transformer (ViT) [5]. Concurrently, the text encoder processes the textual component of the query, translating the human's question into a sequence of word embeddings. In practice, the text encoder can be realized through the embedding layer of the LLM, or by utilizing pre-trained word embedding models, such as Word2Vec [18] or GloVe [19].

3.4 LLM and LoRA Fine-tuning

The LLM within AdvDetectGPT is tasked with interpreting the query image and the accompanying human's question, providing a definitive response to clarify whether the query image is adversarial or clean. The LLM is fine-tuned using the LoRA method. LoRA introduces low-rank matrices to the existing weight layers, allowing for efficient and effective parameter updates without the need for extensive retraining. In the fine-tuning stage, each training instance consists of an image I and a conversation data $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the human's question and the expected response of the system. The training objective of AdvDetectGPT is defined as minimizing the following loss function:

$$\mathrm{L} = -\sum_{t=1}^{T} \log p_\theta\left(\boldsymbol{y}_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, e_{\mathrm{I}}\right) \tag{1}$$

where e_{I} is the image embeddings; T is the length of the target sequence $\boldsymbol{y}$; $\boldsymbol{y}_{<t}$ denotes the sequence of tokens in $\boldsymbol{y}$ that precede the current timestep t; $\theta = \{\theta_f, \theta_l\}$, where θ_f and θ_l correspond to the frozen and the learnable parameters of the model; $p_\theta(\boldsymbol{y}_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}, e_{\mathrm{I}})$ represents the likelihood of predicting the token $\boldsymbol{y}_t$ at time step t given $\boldsymbol{x}$, e_{I} and $\boldsymbol{y}_{<t}$. Through LoRA-based fine-tuning of the LLM, AdvDetectGPT can efficiently adapt to the adversarial detection task while maintaining the generalization ability of the base language model. This lightweight parameter updating mechanism not only greatly reduces the training cost but also enables the model to accurately capture the semantic differences between adversarial and clean examples in the multi-modal embedding space.

4 Experiments

4.1 Experimental Settings

To verify the performance of AdvDetectGPT in detecting adversarial examples, we conduct extensive experiments in various scenarios. Table 1 provides an overview of the general experimental settings.

Table 1. The overview of the general experimental settings.

Training	Clean (10,000); Adversarial (10,000): ResNet-50 + I-FGSM (ϵ=16).
Testing	Scenario 1: novel attacks Adversarial (1,000 each): ResNet-50 + I-FGSM (ϵ=16)/DI-I-FGSM (ϵ=16)/ILA (ϵ=16)/LinBP (ϵ=16)/RAP (ϵ=16).
	Scenario 2: novel attacks + new models Adversarial (1,000 each): Inception-v3/DenseNet-121/VGG16 + DI-IFGSM (ϵ=16)/RAP (ϵ=16).
	Scenario 3: novel attacks + new models + customized perturbations Adversarial (1,000 each): Inception-v3/DenseNet-121/VGG16 + DI-IFGSM (ϵ=8, 32)/RAP (ϵ=8, 32).

Dataset. We randomly selected 10,000 images from the validation set of ImageNet*, along with their corresponding adversarial examples, as the training set for adversarial detectors. The ImageNet-compatible dataset† (containing 1,000 samples), along with the diverse adversarial examples crafted from it, constitute the testing set. The testing data used in each scenario are listed in Table 1. Here, "ResNet-50 + I-FGSM (ϵ=16)" denotes adversarial examples generated by applying I-FGSM on the ResNet-50 model with a maximum perturbation magnitude of ϵ=16.

Models. We selected four models with different architectures as the basis for generating adversarial examples, which are ResNet-50 [9], Inception-v3 [22], DenseNet-121 [11], and VGG16 [21].

Attacks. As for adversarial example generation methods, in addition to considering the basic I-FGSM [14] method, we also take into account methods such as DI-I-FGSM [25], ILA [12], LinBP [8] and RAP [20], which are capable of generating high transferable adversarial examples in black-box scenarios. All attacks utilize l_∞ as the metric for measuring the perturbation distance.

Baselines. We compare our AdvDetectGPT with three adversarial detection methods, i.e., ML-LOO [27], Argos [13], and LiBRe [4]. These methods have achieved state-of-the-art results in ImageNet-level adversarial detection, making them suitable as baselines for comparison.

Metrics. Adversarial detection is essentially a binary classification problem. We designate adversarial examples as positive instances and clean examples as negative

* https://image-net.org/download.php

† https://www.kaggle.com/c/nips-2017-non-targeted-adversarial-attack/data

instances. In experiments, we adopt the true positive rate (*TPR*, for adversarial examples) and the true negative rate (*TNR*, for clean examples) as the evaluation metrics.

Implementation details. We employ ImageBind [7] as the image encoder, Vicuna-7B [3] as the LLM, and the embedding layer of LLM as the text encoder. The size of the query image is set to 299×299×3. Training is conducted on two NVIDIA RTX A5000 GPUs over 50 epochs, with a learning rate of 5e-4 and a batch size of 16.

4.2 Detection Results: Novel Attacks

AdvDetectGPT and the comparative baseline methods are all trained with 10,000 clean examples and their corresponding adversarial examples (generated on ResNet-50 using I-FGSM with perturbations of ϵ=16/255). We first test their discrimination capabilities on clean examples and various types of adversarial examples (generated using I-FGSM and several novel attacks). The results, as shown in Table 2, indicate that on clean examples, AdvDetectGPT's *TNR* is 94.3%, slightly lower than LiBRe's 98.4%, but significantly higher than that of ML-LOO and Argos. On adversarial examples, AdvDetectGPT's detection ability is either superior to or on par with LiBRe, with *TPR* always approaching 100%, which is significantly higher than that of ML-LOO and Argos.

4.3 Detection Results: Novel Attacks + New Models

We next test the performance of various detection methods on adversarial examples generated by novel attacks on the new models. Specifically, the detectors, initially trained on adversarial examples generated by I-FGSM on ResNet-50, will now be tested against adversarial examples generated by DI-I-FGSM and RAP across three different models, i.e., Inception-v3, DenseNet-121, and VGG16. The results are shown in Table 3. It can be observed that AdvDetectGPT still maintains a *TPR* that is superior to or on par with LiBRe's on new types of adversarial examples, and significantly outperforms ML-LOO and Argos. This demonstrates AdvDetectGPT's excellent generalization ability in detecting new types of adversarial examples.

4.4 Detection Results: Novel Attacks + New Models + Customized Perturbations

We further evaluate the performance of AdvDetectGPT on detecting adversarial examples with customized perturbations. Specifically, we evaluate the performance of detectors using adversarial examples generated on new models with novel attacks and customized perturbations (ϵ =8/255, 32/255), deviating from the original training perturbations (ϵ =16/255). The results are shown in Table 4. It can be observed that when the maximum perturbation is increased to ϵ =32/255, AdvDetectGPT consistently maintains a TPR of 100%, outperforming all baseline methods. When the ϵ is reduced to 8/255, except for one special case (i.e., detecting adversarial examples generated using RAP on VGG16), AdvDetectGPT is still superior to all baselines. This further demonstrates the robust generalization of AdvDetectGPT.

Table 2. *TNR* (%) of clean examples and *TPR* (%) of adversarial examples.

Method	Clean	ResNet-50				
		I-FGSM	DI-I-FGSM	ILA	LinBP	RAP
ML-LOO [27]	77.1	80.4	81.2	80.1	79.4	77.5
Argos [13]	77.6	79.1	79.4	79.3	76.6	77.8
LiBRe [4]	**98.4**	99.5	**100.0**	99.9	99.6	98.5
AdvDetectGPT	94.3	**99.9**	**100.0**	**100.0**	**99.8**	**99.8**

Table 3. *TPR* (%) of adversarial examples generated by novel attacks on new models.

Method	Inception-v3		DenseNet-121		VGG16	
	DI-I-FGSM	RAP	DI-I-FGSM	RAP	DI-I-FGSM	RAP
ML-LOO [27]	78.9	77.4	79.2	78.7	79.9	79.1
Argos [13]	75.8	74.9	77.3	77.2	78.1	76.9
LiBRe [4]	**99.8**	98.1	**99.9**	98.9	**100.0**	98.7
AdvDetectGPT	99.7	**99.3**	**99.9**	**99.9**	**100.0**	**99.1**

*ILA and LinBP are not considered as their implementations [8] only target the ResNet-50 model.

Table 4. *TPR* (%) of adversarial examples with novel attacks, new models and customized perturbations.

Method	Inception-v3				DenseNet-121				VGG16			
	DI-I-FGSM		RAP		DI-I-FGSM		RAP		DI-I-FGSM		RAP	
	ϵ=8	ϵ=32	ϵ=8	ϵ=32	ϵ=8	ϵ=32	ϵ=8	ϵ=32	ϵ=8	ϵ=32	ϵ=8	ϵ=32
ML-LOO [27]	71.6	80.6	69.5	71.4	70.8	71.7	70.2	72.1	71.6	72.4	68.8	71.5
Argos [13]	71.7	76.4	70.7	75.9	76.7	77.9	74.1	77.5	76.2	78.8	**75.4**	77.2
LiBRe [4]	79.4	98.7	69.1	99.2	84.4	98.9	80.4	97.1	82.5	98.3	64.6	99.7
AdvDetectGPT	**82.0**	**100.0**	**74.9**	**100.0**	**94.1**	**100.0**	**89.1**	**100.0**	**91.9**	**100.0**	70.6	**100.0**

4.5 Benchmarking Computational Overhead

During experiments, AdvDetectGPT was deployed on a system with the following specifications: an Intel Xeon (R) Bronze 3206R CPU @ 1.90GHz (16 cores), 64GB RAM, and an NVIDIA RTX A5000 GPU with 24GB VRAM under CUDA 12.1. Our measurements indicate that during inference, the model consumes approximately 14.9GB of VRAM and 4.8GB of RAM, with an average detection time of 1.037 seconds per sample (for one 299×299 ×3 resolution image). The AdvDetectGPT is fine-tuned from the Vicuna-7B FP16 model. The actual GPU memory usage matched the theoretical value, while the RAM usage was mainly due to the image encoder of ImageBind. Notably, large vision-language models inherently demand substantial computational

resources, which is also a potential limitation of AdvDetectGPT in practical applications. Nevertheless, specialized optimization techniques (e.g., model pruning, quantization) could reduce the computational overhead. These techniques are beyond the scope of this study.

5 Conclusion

In this work, we delve into the application of LVLMs to tackle the challenge of adversarial detection. We design an adversarial detector based on LVLMs, named AdvDetectGPT. AdvDetectGPT is trained using the raw adversarial and clean examples, and can achieve effective adversarial detection via a question-and-answer mechanism. The extensive experiments demonstrate AdvDetectGPT's excellent performance and robust generalization. It is worth noting that in practical applications, AdvDetectGPT needs to guard against the impact of adaptive adversarial attacks (i.e., detection-aware attacks), which could serve as a future research direction. Additionally, exploring how to leverage more compact and low-energy large models for adversarial detection may also constitute a promising area for future investigation.

References

1. Addepalli, S., Jain, S., R., V.B.: Efficient and Effective Augmentation Strategy for Adversarial Training. In: Proceedings of the Annual Conference on Neural Information Processing Systems (2022)
2. Carlini, N., Wagner, D.: Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. pp. 3–14 (2017)
3. Chiang, W.L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J.E., Stoica, I., Xing, E.P.: Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality (2023), https://lmsys.org/blog/2023-03-30-vicuna/
4. Deng, Z., Yang, X., Xu, S., Su, H., Zhu, J.: LiBRe: A Practical Bayesian Approach to Adversarial Detection. In: Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 972–982 (2021)
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In: Proceedings of the 9th International Conference on Learning Representations (2021)
6. Feinman, R., Curtin, R.R., Shintre, S., Gardner, A.B.: Detecting Adversarial Samples from Artifacts. CoRR abs/1703.00410 (2017)
7. Girdhar, R., El-Nouby, A., Liu, Z., Singh, M., Alwala, K.V., Joulin, A., Misra, I.: ImageBind: One Embedding Space to Bind Them All. In: Proceedings of 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15180–15190 (2023)
8. Guo, Y., Li, Q., Chen, H.: Backpropagating Linearly Improves Transferability of Adversarial Examples. In: Proceedings of the 34th Conference on Neural Information Processing Systems (2020)

9. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
10. Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-Rank Adaptation of Large Language Models. In: Proceedings of the 10th International Conference on Learning Representations (2022)
11. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely Connected Convolutional Networks. In: Proceeding of 2017 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2261–2269 (2017)
12. Huang, Q., Katsman, I., Gu, Z., He, H., Belongie, S., Lim, S.N.: Enhancing Adversarial Example Transferability With an Intermediate Level Attack. In: Proceedings of 2019 IEEE/CVF International Conference on Computer Vision. pp. 4732–4741 (2019)
13. Kiani, S., Awan, S., Lan, C., Li, F., Luo, B.: Two Souls in an Adversarial Image: Towards Universal Adversarial Example Detection using Multi-view Inconsistency. In: Proceedings of the Annual Computer Security Applications Conference. pp. 31–44 (2021)
14. Kurakin, A., Goodfellow, I., Bengio, S.: Adversarial Examples in the Physical World. In: Workshop Track Proceedings of the 5th International Conference on Learning Representations (2017)
15. Lee, K., Lee, K., Lee, H., Shin, J.: A Simple Unified Framework for Detecting Outof-Distribution Samples and Adversarial Attacks. In: Proceedings of the Annual Conference on Neural Information Processing Systems. pp. 7167–7177 (2018)
16. Ma, X., Li, B., Wang, Y., Erfani, S.M., Wijewickrema, S.N.R., Schoenebeck, G., Song, D., Houle, M.E., Bailey, J.: Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. In: Proceedings of the 6th International Conference on Learning Representations (2018)
17. Meng, D., Chen, H.: MagNet: a Two-Pronged Defense against Adversarial Examples. In: Proceedings of the 2017 ACM Conference on Computer and Communications Security. pp. 135–147 (2017)
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. In: Workshop Track Proceedings of the 1st International Conference on Learning Representations (2013)
19. Pennington, J., Socher, R., Manning, C.D.: GloVe: Global Vectors for Word Representation. In: Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing. pp. 1532–1543 (2014)
20. Qin, Z., Fan, Y., Liu, Y., Shen, L., Zhang, Y., Wang, J., Wu, B.: Boosting the Transferability of Adversarial Attacks with Reverse Adversarial Perturbation. In: Proceedings of the 36th Conference on Neural Information Processing Systems (2022)
21. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. In: Proceedings of the 3rd International Conference on Learning Representations (2015)
22. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826 (2016)
23. Wang, S., Gong, Y.: Adversarial example detection based on saliency map features. Applied Intelligence 52(6), 6262–6275 (2022)
24. Xie, C., Wu, Y., Maaten, L.v.d., Yuille, A., He, K.: Feature Denoising for Improving Adversarial Robustness. In: Proceedings of 2019 IEEE Conference on Computer Vision and Pattern Recognition. pp. 501–509 (2019)

25. Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.L.: Improving Transferability of Adversarial Examples With Input Diversity. In: Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2725–2734 (2019)
26. Xu, W., Evans, D., Qi, Y.: Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In: Proceedings of 2018 Network and Distributed System Security Symposium (2018)
27. Yang, P., Chen, J., Hsieh, C.J., Wang, J.L., Jordan, M.I.: ML-LOO: Detecting Adversarial Examples with Feature Attribution. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence. pp. 6639–6647 (2020)
28. Yang, Y., Gao, R., Li, Y., Lai, Q., Xu, Q.: What You See is Not What the Network Infers: Detecting Adversarial Examples Based on Semantic Contradiction. In: Proceedings of the 2022 Network and Distributed System Security Symposium (2022)