Deep Learning - Final Project
Adversarial Examples in Traffic Sign Recognition

Tianyu Dai
Physics
Duke University
Durham, NC 27705
tianyu.dai@duke.edu

Spencer Hallyburton
Electrical and Computer Engineering
Duke University
Durham, NC 27705
spencer.hallyburton@duke.edu

Abstract

Traffic sign recognition is a crucial task to ensure the safety and reliability of autonomous vehicles. For most traffic sign recognition system, deep learning is applied to make instant decisions in response to the immediate surroundings without human intervention. However, recent developments in adversarial attacks have discovered that well-designed, imperceptible perturbations to the input data can drastically affect the performance of the deep perception models. In this work, we apply Deep Residual Network for traffic sign recognition on a German traffic sign dataset. We use two popular methods for adversarial examples against the neural network and evaluate them over sweeps of model parameters. We discuss the recently developed defenses against adversarial examples and show that current defenses are unsatisfactory. We then propose our own defenses: (1) validating deep learning results with human-engineered features; (2) leveraging perception data from multiple sensors.

1 Introduction

Autonomous vehicles (AVs) have already enjoyed millions of miles of partially automated road travel [1, 2, 3].

Perception is a fundamental element of autonomy and provides AVs an awareness of surroundings. Perception allows AVs to make well-informed safety-critical decisions, such as obeying traffic laws, that take into account the environment. Sensors such as cameras collect important data for perception. Inexpensive, high-quality cameras can provide high resolution, dense 2D outputs on limited fields of view, and cameras are used for many safety-critical perception tasks in AVs [1].

Due to AVs' safety-critical nature, misinformation or wrong decisions can quickly lead to severe adverse outcomes [4, 5]. The high-impact outcomes underscore the need for security research in the domain. In particular, the increasing reliance of AVs on deep neural networks (DNNs) for real-time perception has sparked security questions at the algorithm level. There is a growing body of AV security work: for perception algorithms, an attacker can perturb sensor data to change object classification (misclassification) [6], introduce fake objects (false positives) [7], and remove existing objects (false negative) [8], each with devastating consequences at the driving decision and control level.

Traffic sign recognition has been identified as a key safety-critical task of AVs [4, 5]. DNNs have been consistently proposed to be used to automate this task in self-driving. Thus, it is important to explore the robustness of these traffic sign detection and recognition algorithms to adversarial efforts.

We apply recently demonstrated *adversarial examples* to modern DNN algorithms capable of detecting hundreds of different traffic signs to test the robustness of perception models. We perform analysis of

the adversarial algorithms across multiple important attacker parameters. Damagingly for perception algorithms, we find that adversarial examples are very much a problem for traffic sign classification algorithms.

We survey efforts to secure perception models to adversarial examples. Unfortunately, we find that prior works have demonstrated each proposed defense falls short of guarding globally against adversarial examples. We find the only defense that has demonstrated widespread success is training the classification models against similar adversarial examples, which may not be possible if the attacker's capabilities are not known.

To advance the field and secure perception to adversarial examples, we propose two new defenses to guarding against state-of-the-art attacks. One defenses leverages human-derived features to validate the outcomes of the machine-learned model. The second defenses is a framework for multi-sensor fusion to secure the image classification problem.

2 Background

2.1 Adversarial Machine Learning

Machine learning is becoming an increasingly popular method for automation of tasks, even at the safety-critical level. In particular, perception system of the autonomous vehicle is usually heavily dependent on machine learning models.

Machine learning models are usually designed to solve the problem by assuming that the training and the test dataset are sampled as identically independently distributed random variables. However, in real world, adversaries in the data may invalidate this statistical assumption. These adversaries are imperceptible perturbations to the input data, but can drastically affect the performance of machine learning models. They can be exploited and designed as malicious attacks to the machine learning models.

2.2 Traffic sign recognition

2.2.1 Model

Traffic sign recognition is a vision-based technology to detect traffic signs on images, and classify them into different types. Several review papers survey the traffic sign recognition literature, and compare contributions in various stages on the same benchmark dataset [9, 10]. Traditional traffic sign recognition methods usually capture the manual features, like geometric shape, color and the oriented gradients of the traffic sign. Multiple works applying AdaBoost training [11], support vector machines [12], and other techniques [13, 14] to extract the manual features and improve the recognition accuracy. However, the manual features are difficult to be captured in difficult driving conditions, which limits the performance of the traditional traffic sign recognition approaches.

Modern traffic sign recognition technologies are usually based on deep neural network architectures. The recent developments in convolutional neural network (CNN) has made it desirable for implementing various computer vision tasks. CNN-based traffic sign recognition methods learn a hierarchy of features by building high-level features from low-level features, and perform better than traditional approaches.

In 2013, an approach using support vector machines for image processing and CNN for traffic sign detection and recognition is introduced [15]. In 2016, frameworks based on fully convolutional neural network are proposed for simultaneous detection and classification are proposed [16, 17]. To particularly detect the small-scale traffic sign, region-based CNN (R-CNN) are implemented to avoid selecting large number of regions [18, 19, 20]. Region proposals are produced to refine hierarchical feature extraction. In other works, color and geometric shape of traffic signs are also considered to improve the localization of the small traffic signs [21].

2.2.2 Datasets

Traffic signs include different types, e.g. stop sign, danger warning sign, speed limits, and are standardized by Vienna Convention on Road Signs and Signals. Most of the traffic sign recognition technologies are evaluated using one of the following datasets.

Figure 1: Sample images from the GTSRB dataset. Notice the variability in the image conditions, including the differences in lighting conditions, angles of image capture, scale of traffic sign, cropping of sign, colors of signs.

- Mapillary Traffic Sign Dataset [22]: $\sim 100k$ high-resolution street-level images with bounding box collected around the world, for both recognition and detection.
- Laboratory for Intelligent and Safe Automobiles (LISA) Dataset [9]: $\sim 7k$ annotated frames, obtained from different cameras on United States road, for both recognition and detection.
- German Traffic Sign Recognition Benchmark (GTSRB) [23]: $\sim 50k$ small-scale images obtained in German, for recognition only.
- Tsinghua-Tencent 100K dataset [24]: $\sim 100k$ street view images acquired in China, for both recognition and detection.

In this work, we use GTSRB [1] to evaluate our traffic sign recognition model. This dataset consists of 43 different classes with 39209 images in the training set and 12631 images in the test set. The images are distributed unevenly across different classes. All the images are small-scale with a single traffic sign in varying lighting conditions, and backgrounds. All classes are mutually exclusive and each image includes only a single traffic sign from a certain class.

Figure 1 shows several sample images from the GTSRB dataset. The data are collected across different environmental conditions, as evident from the differences in lighting, illumination, angle, color, and scale.

The benchmark evaluation on this traffic sign dataset shows that CNNs surpass the human-level performance [25].

2.3 Adversarial examples

Although CNN-based techniques have shown strong power on traffic sign detection and recognition, a well trained CNN can behave badly to adversarial attacks [26, 27]. Adversarial attacks are intentionally generated inputs, which are naturally indistinguishable but can cause incorrect classification of machine learning models [28].

[1] See https://benchmark.ini.rub.de/ to download the dataset.

Recently, many efforts are put to investigates the methods of generating adversarial examples [26, 29, 30, 31, 27]. As shown in [28, 32], adversarial threat models with different assumptions of the attacker's knowledge can be classified as:

- Zero-knowledge adversary (black-box attacks): adversarial examples are generated agnostic to the detector.
- Limited-knowledge adversary (grey-box attacks): adversary examples are generated being only aware of how the detector is trained.
- Perfect-knowledge adversary (white-box attacks): adversary examples are generated being aware of the model parameters of the detector.

These adversarial examples are usually robust and possible in physical world. Recent study shows that adversarial attacks can still fool the image recognition system after being printed out [33].

In this work, we only discuss the white-box attacks. Most of the white-box attacks are based on adjusting the input to maximize the gradient of the model loss. These adversarial threat model varies in the norm ball perturbation they consider, and the method they use for optimizing over the norm ball. We conclude three popular gradient-based adversarial threat models as follows [34].

Fast Gradient Sign Method (FGSM) [29] This attack calculates the gradient of the loss with respect to the input data, and then adjust the input data to maximize the loss. FGSM can be considered as a single projected gradient descent step under the l_∞ constraint. The optimization process can be written as

$$x' = \text{FGSM}(x) = \text{clip}_{[0,1]}(x + \epsilon(\nabla l_F(x, y))), \tag{1}$$

where l_F is the cross-entropy loss of the network F, ϵ constrain the step size, and the clip function keeps the adversarial example valid.

Projected Gradient Descent (PGD) / Basic Iterative Method (BIM) [33, 35] This attack attempts to maximize the model loss by performing projected gradient descent iteratively with a smaller step size. The optimization process can be written as

$$x'_{i+1} = \text{clip}_{[x-\alpha, x+\alpha]}\left(\text{FGSM}(x'_i)\right), \tag{2}$$

where the targeted adversarial example x'_i is controlled within a norm ball with size α.

Carlini and Wagner (CW) [31] This attack also iterates over the gradient descent to minimize the distance between the input and the adversarial example constraning by maximizing the loss function. They propose not only l_∞ attack, but also l_0 and l_2 attacks. To tackle the difficult constrained optimization, they solve the optimization process as

$$g(x') = \max(\max\{Z(x')_i : i \neq t\} - Z(x')_t, 0), \tag{3}$$

where $Z(\cdot)$ is the logits of the network and $g(\cdot)$ is a loss function.

2.4 Adversarial defense

Given that adversarial attacks is a potential vulnerability of deep learning system, significant amount of work focusing on increasing the system's robustness to adversarial attacks. Possible adversarial defenses are summarized as follows [36].

- Adversarial training [26, 29, 37]: a brute force approach, where the defender train the model with generated adversarial examples. This approach is only effective for adversarial attacks that the model was previously trained with, but not robust to black-box attacks.
- Gradient hiding [38]: an approach hiding the model's gradient information to defend gradient-based attacks. However, adversarial examples generated using a gradient-based surrogate model can still bypass this approach.
- Defensive distillation [31, 38]: a two-step strategy that train a smaller neural network with the probabilities of different classes trained from a larger neural network. However, the adversarial examples with strong transferibility across neural network models are able to bypass distillation defense.

- Model capacity enlargement [35]: an approach to increase the capacity of the neural network architecture. The classifier performance is sensitive to model capacity, since the adversarial examples change the decision boundary to a complex one. More capacity of the model can also decrease the transferability.
- Feature squeezing [39]: an approach to reduce the data representation. Encoding image colors with fewer values or applying smoothing filter on the image are feasible feature squeezing method. These feature squeezing techniques may work well against adversarial examples, but they will reduce the model accuracy on normal examples.

Although many adversarial defenses are proposed, most of them cannot provide valid defense for all types of attack scenarios. Defenses that works for small image dataset may defect on larger and more complex image dataset. The adversarial examples are usually generated by complex optimization process which is hard to be analyzed theoretically. Moreover, possible adversarial defenses may even decrease the prediction accuracy for normal inputs [36].

3 Attack Model

3.1 Goal and Capability

We consider an attacker who wishes to deteriorate the classification performance of a deep learning model. In particular, we consider the input space of images, defined as dense tensors ($C \times H \times W$ for channels, height, and width) with pixel intensity values in full generality clipped between 0 and 1, but more typically represented as integers between [0, 255].

The attacker is capable of perturbing the pixel intensity values arbitrarily. We assume the attacker has no way of modifying the network itself.

We consider that the attacker wishes to achieve the minimum norm perturbation. This could take the form of an explicit constraint on an allowable budget or could just be formulated as such without explicitly evaluating the constraint. Similarly, we consider both targeted and untargeted attack goals where, in the untargeted case, the attacker is indifferent to the model output of the adversarial example while in the targeted case, the attacker wishes to achieve some specific adversarial output class.

Thus, one possible formulation of the attack goal in the untargeted case is:

$$\begin{aligned} \text{find } x' = f(x) \text{ s.t. } C(x') \neq C(x) \\ \text{and } x' = \text{argmin} \|x' - x\| \end{aligned} \tag{4}$$

where a similar form in the targeted case is

$$\begin{aligned} \text{find } x' = f(x) \text{ s.t. } C(x') = y \neq C(x) \\ \text{and } x' = \text{argmin} \|x' - x\| \end{aligned} \tag{5}$$

where C is the classification function that takes in input image and returns the predicted class label, and the norm operator is some general $L - p$ norm that the attacker can specify.

3.2 Knowledge

Similar to [31, 29, 26], we consider that the attacker has full access to the neural network for classification. Thus, the adversary can perform operations leveraging the model weights and the gradient of the model. Thus, these methods are "white-box". Other methods exist for "black-box" attacks.

4 Baseline

In deciding our baseline image classification algorithm, we consult popular surveys on available algorithms [40, 41]. There are many suitable image classification algorithms, and we select the Deep Residual Network [42] (resnet) due to its wide adoption in the community and ease of availability in open-source packages. Resnets utilize skip-connections (also called "shortcuts") between layers and have been shown to dramatically increase model performance while also allowing for reduced model complexity [42].

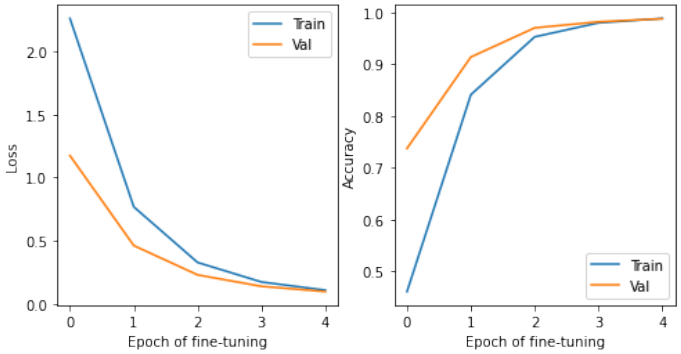


Figure 2: Training and validation loss and accuracy over the fine-tuning process. Model quickly converges to performance that exceeds that of a human, and validation early-stopping does not indicate over-fitting. Note that training performance appears "worse" than validation at each epoch - this is because training performance is aggregated across the epoch while validation is only evaluated at the end of the each epoch (with gradients off).

Resnets can be extended to arbitrary depth. Due to the simplicity of our application and the cleanliness of our data, we are able to select resnet-18, a commonly-used benchmark in small-scale image classification applications. We take some time to explore our dataset. Specifically, we investigate the class distribution and the mean pixel intensity distribution across the images and between images. For brevity, we choose not to include these here and refer the interested reader to [23].

We begin the baseline evaluation with a resnet-18 pre-trained [2] on the ImageNet dataset [43]. The purpose of using a pre-trained model to start is to reduce the computational burden required to train the model and also to allow the algorithm to learn how to extract general features without overfitting to an application-specific dataset.

After selecting a pre-trained model, we fine-tune resnet-18 on the GTSRB dataset. We split the data with available labels into training, validation, and testing sets using random samples of 80%, 10%, and 10% of the original training data, respectively. We use the validation set in the fine-tuning process to perform early stopping which has been shown to reduce overfitting on the test set in many algorithms and particularly in resnets [42]. Figure 2 shows the loss and accuracy of the fine-tuning process. Note that validation performance is only evaluated at the end of the epoch while training performance is evaluated within the epoch, thus average performance is worse for training at each epoch.

5 Attack

We choose to evaluate two popular attacks: the Fast Gradient Sign Method (FGSM) [29] and the Carlini-Wagner (CW) [31] attack, described in background. We move towards providing a comprehensive evaluation of the attack success against important model parameters. In table 1, we describe the attack success of FGSM depending on whether the attack was targeted and which norm was selected for the optimization.

Similarly, we sweep two important parameters of the models: the maximum allowable norm perturbation (ϵ) and the number of iterations used in the optimization for the CW attack. Figure 3 shows the intuitive outcome that increased allowable norm perturbation yields higher attack success in the FGSM attack and more iterations of the CW attack similarly lead to higher attack success.

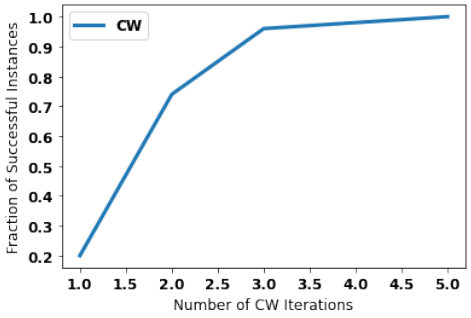
6 Defense

We first discuss several state-of-the-art defenses against adversarial examples. We also provide references showing how each of these defenses is not capable of defending against adversarial examples, globally.

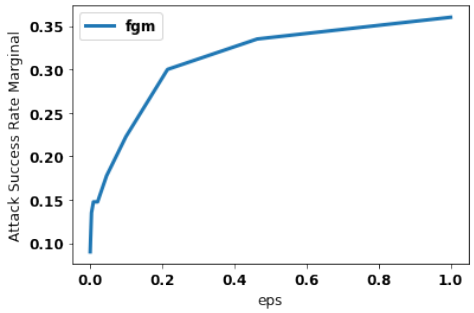
[2] see https://pytorch.org/vision/stable/models.html for available models

Attack Type	Norm	Targeted	**Accuracy**
FGSM	L2	NO	8.4%
FGSM	L2	YES	10.0%
FGSM	L∞	NO	32.0%
FGSM	L∞	YES	7.2%

Table 1: Attack accuracy of FGSM against several investigated parameters, including the choice of attack norm and targeted boolean. Runs were fixed at maximum norm perturbation value of 0.1.



(a) Success of the Carlini-Wagner attack against the number of optimization iterations.



(b) Success of the FGSM attack against the maximum norm perturbation.

Figure 3: We evaluated attack success rate of different methods over parameter sweeps to obtain an idea of performance distribution. (a) Carlini Wagner attack success increases sharply with number of iterations. (b) FGSM attack success increases as maximum allowable norm perturbation grows. Both results validate intuitive explanation.

We described the most common defenses in Section 2.4. Again, three of the most common defense techniques against adversarial examples are: **Defensive Distillation** that distills a larger model into a smaller model, **Gradient Hiding** which smooths gradients in an attempt to make the decision boundary more smooth, **Adversarial Training** which directly uses adversarial examples in the training process to fit true labels to adversarial examples.

However, [32] showed damaging results against defenses against adversarial examples. [32] evaluated 10 common defenses and showed how adaptive attacks can evade each of the defenses. The only promising method was *adversarial training*, which directly trains against the same types of adversarial examples that are used in the testing.

In the area of developing global defenses against adversarial examples in machine learning, there are two approaches that have been unexplored to date. First is the use of human-engineered features to validate the class outcomes, and second is the use of a multi-sensor fusion architectures.

First, adversarial machine learning is successful in deep learning applications due to unsmooth contours of the decision function which allow small perturbations to flip the classification. On the other hand, human-engineered features may be more robust due to enhanced smoothing of the decision function, however potentially at the cost of reduced accuracy of human-derived classifiers. Mixing the two strategies can have a positive impact on defending against adversarial examples. For example, training data distribution stats could be used to validate the class outcome of a testing example. Specifically, many classes of traffic signs have well-defined color distributions (e.g., stop signs are mostly red). Other interesting features would include the shape of the sign and the reflectance of the sign. Thus, any classification output (from a deep model) could be validated against the distribution from the training data (e.g., a speed limit sign could not be misclassified as a stop sign because it does not have enough red in it). A challenge with this approach is to engineer meaningful features from the data that can accurately discriminate classes, and future efforts in this area should make use of decision trees.

We also propose a defense using perception data from multiple sensors. This could the form of either redundant or complementary sensors. In a redundant sensor application, two cameras would have overlapping fields of view, for example. Perception models run on the independent streams

of data would need to provide consistent classification outputs and can be filtered out if the classes do not agree. This defense would severely limit the applicability of *untargeted* attacks, and attack coordination would need to occur between the different sensors. As an extension, it may be possible to use *complementary* sensors to validate the classification output. For example, since LiDAR sensors fully resolve objects in three dimensions, a LiDAR sensor could estimate the shape and reflectance of the traffic sign which may be useful in validating the true class.

7 Conclusion

Traffic sign detection and recognition is not secure when evaluated against adversarial examples. As a safety-critical application, efforts must be taken to protect the models against attack. Unfortunately, many efforts thus far in defending against adversarial examples fall woefully short, as demonstrated in the literature. To advance the field, we have proposed two defenses not yet seen in the literature. One merges human-engineered features with machine-learned features in a post-processing step and another uses sensor fusion to validate classification outcomes. Both are model-level changes, as opposed to input-level change (e.g., smoothing), and may be more promising to guard against adversarial examples.

8 Contributions

This project was a team effort. Tianyu performed a thorough literature review, identifying the 3 most promising adversarial methods, repositories where the code could be found, and candidate datasets for the traffic sign recognition task. She then wrote the background section as well as the abstract. Spencer integrated the Resnet-18 model in pytorch and performed fine-tuning of the model (pre-trained on imagenet) using the traffic sign dataset (GTSRB). He then evaluated the model against the adversarial attacks (FGSM and CW) over sweeps of parameters, and wrote up the results in sections 3, 4, and 5. The team discussed together possible defenses against adversarial examples and Spencer wrote section 6.

References

[1] A. Hawkins, "Waymo's autonomous cars have driven 8 million miles on public roads." `https://www.theverge.com/2018/7/20/17595968/waymo-self-driving-cars-8-million-miles-testing`, 2018.

[2] "The Evolution of Automated Safety Technologies." `https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety`, 2021.

[3] P. LeBeau, "Waymo starts commercial ride-share service." `https://www.cnbc.com/2018/12/05/waymo-starts-commercial-ride-share-service.html`, 2018.

[4] B. Schoettle and M. Sivak, "A preliminary analysis of real-world crashes involving self-driving vehicles," *University of Michigan Transportation Research Institute*, 2015.

[5] P. Kohli and A. Chadha, "Enabling pedestrian safety using computer vision techniques: A case study of the 2018 uber inc. self-driving car crash," in *Future of Information and Communication Conference*, pp. 261–279, Springer, 2019.

[6] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust Physical-World Attacks on Deep Learning Visual Classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.

[7] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on lidar-based perception in autonomous driving," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pp. 2267–2281, 2019.

[8] M. Abdelfattah, K. Yuan, Z. J. Wang, and R. Ward, "Adversarial Attacks on Camera-LiDAR Models for 3D Car Detection," *arXiv preprint arXiv:2103.09448*, 2021.

[9] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012.

[10] A. Arcos-Garcia, J. A. Alvarez-Garcia, and L. M. Soria-Morillo, "Evaluation of deep neural networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332–344, 2018.

[11] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pp. 255–260, IEEE, 2005.

[12] S. K. Berkaya, H. Gunduz, O. Ozsen, C. Akinlar, and S. Gunal, "On circular traffic sign detection and recognition," *Expert Systems with Applications*, vol. 48, pp. 67–75, 2016.

[13] S. Salti, A. Petrelli, F. Tombari, N. Fioraio, and L. Di Stefano, "Traffic sign detection via interest region extraction," *Pattern Recognition*, vol. 48, no. 4, pp. 1039–1049, 2015.

[14] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," *Machine vision and applications*, vol. 25, no. 3, pp. 633–647, 2014.

[15] Y. Wu, Y. Liu, J. Li, H. Liu, and X. Hu, "Traffic sign detection based on convolutional neural networks," in *The 2013 international joint conference on neural networks (IJCNN)*, pp. 1–7, IEEE, 2013.

[16] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2110–2118, 2016.

[17] Y. Zhu, C. Zhang, D. Zhou, X. Wang, X. Bai, and W. Liu, "Traffic sign detection and recognition using fully convolutional network guided proposals," *Neurocomputing*, vol. 214, pp. 758–766, 2016.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[19] D. Tabernik and D. Skočaj, "Deep learning for large-scale traffic-sign detection and recognition," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 4, pp. 1427–1440, 2019.

[20] J. Zhang, Z. Xie, J. Sun, X. Zou, and J. Wang, "A cascaded r-cnn with multiscale attention and imbalanced samples for traffic sign detection," *IEEE Access*, vol. 8, pp. 29742–29754, 2020.

[21] Y. Yang, H. Luo, H. Xu, and F. Wu, "Towards real-time traffic sign detection and classification," *IEEE Transactions on Intelligent transportation systems*, vol. 17, no. 7, pp. 2022–2031, 2015.

[22] C. Ertler, J. Mislej, T. Ollmann, L. Porzi, and Y. Kuang, "Traffic sign detection and classification around the world," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[23] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks*, no. 1288, 2013.

[24] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012. Selected Papers from IJCNN 2011.

[26] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[27] T. Gu, B. Dolan-Gavitt, and S. Garg, "Badnets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[28] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402, Springer, 2013.

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

[30] 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*, pp. 2574–2582, 2016.

[31] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*, pp. 39–57, IEEE, 2017.

[32] N. Carlini and D. Wagner, "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.

[33] A. Kurakin, I. Goodfellow, S. Bengio, *et al.*, "Adversarial examples in the physical world," 2016.

[34] N. Carlini, G. Katz, C. Barrett, and D. L. Dill, "Provably minimally-distorted adversarial examples," *arXiv preprint arXiv:1709.10207*, 2017.

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

[36] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," *arXiv preprint arXiv:1810.00069*, 2018.

[37] U. Shaham, Y. Yamada, and S. Negahban, "Understanding adversarial training: Increasing local stability of supervised models through robust optimization," *Neurocomputing*, vol. 307, pp. 195–204, 2018.

[38] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519, 2017.

[39] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017.

[40] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *International journal of Remote sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[41] P. Druzhkov and V. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognition and Image Analysis*, vol. 26, no. 1, pp. 9–15, 2016.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.