ENSEMBLE METHODS AS A DEFENSE TO ADVERSARIAL PERTURBATIONS AGAINST DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep learning has become the state of the art approach in many machine learning problems such as classification. It has recently been shown that deep learning is highly vulnerable to adversarial perturbations. Taking the camera systems of self-driving cars as an example, small adversarial perturbations can cause the system to make errors in important tasks, such as classifying traffic signs or detecting pedestrians. Hence, in order to use deep learning without safety concerns a proper defense strategy is required. We propose to use ensemble methods as a defense strategy against adversarial perturbations. We find that an attack leading one model to misclassify does not imply the same for other networks performing the same task. This makes ensemble methods an attractive defense strategy against adversarial attacks. We empirically show for the MNIST and the CIFAR-10 data sets that ensemble methods not only improve the accuracy of neural networks on test data but also increase their robustness against adversarial perturbations.

1 INTRODUCTION

In recent years, deep neural networks (DNNs) led to significant improvements in many areas ranging from computer vision (Krizhevsky et al., 2012; LeCun et al., 2015) to speech recognition (Hinton et al., 2012; Dahl et al., 2012). Some applications that can be solved with DNNs are sensitive from the security perspective, for example camera systems of self driving cars for detecting traffic signs or pedestrians (Papernot et al., 2016b; Sermanet & LeCun, 2011). Recently, it has been shown that DNNs can be highly vulnerable to adversaries (Szegedy et al., 2013; Goodfellow et al., 2014; Papernot et al., 2016a;b). The adversary produces some kind of noise on the input of the system to mislead its output behavior, producing undesirable outcomes or misclassification. Adversarial perturbations are carefully chosen in order to be hard, if not impossible, to be detected by the human eye (see figure 1). Attacks occur *after* the training of the DNN is completed. Furthermore, it has been shown that the exact structure of the DNN does not need to be known in order to mislead the system as one can send inputs to the unknown system in order to record its outputs to train a new DNN that imitates its behavior (Papernot et al., 2016b). Hence, in this manuscript it is assumed that the DNN and all its parameters are fully known to the adversary.

There are many methods on how to attack neural networks appearing in the literature. Some of the most well-known ones are the Fast Gradient Sign Method (Goodfellow et al., 2014) and its iterative extension (Kurakin et al., 2016), DeepFool (Moosavi-Dezfooli et al., 2016), Jacobian-Based Saliency Map Attack (Papernot et al., 2016c), and the L-BFGS Attack (Szegedy et al., 2013). This shows the need of building neural networks that are themselves robust against any kind of adversarial perturbations.

Novel methods on defending against adversarial attacks are appearing more and more frequently in the literature. Some of those defense methods are to train the network with different kinds of adversarially perturbed training data (Goodfellow et al., 2014; Papernot et al., 2016c), the use of distillation to reduce the effectiveness of the perturbation (Papernot et al., 2016d) or to apply denoising autoencoders to preprocess the data used by the DNN (Gu & Rigazio, 2014). It also has been noted that adversarial attacks can be detected (Metzen et al., 2017; Feinman et al., 2017), but

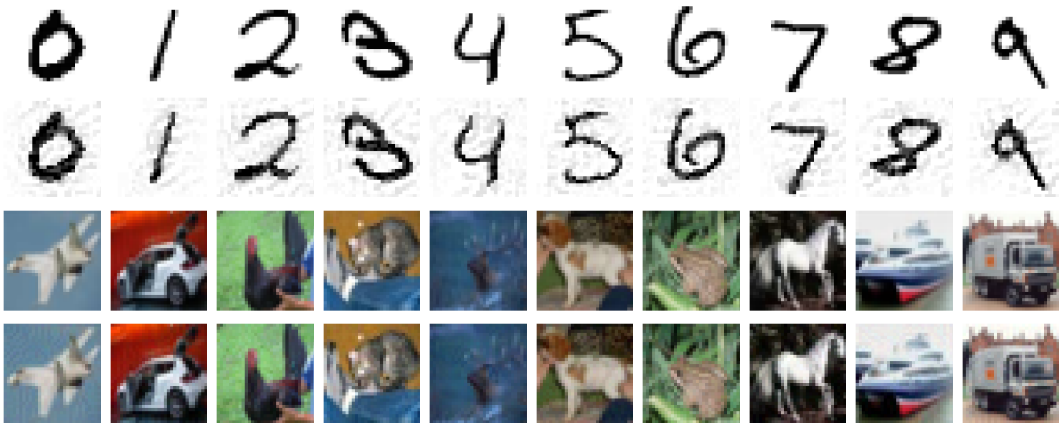

Figure 1: The first line shows original and correctly classified MNIST test data images. In the second line are the corresponding adversarial BIM attacks on a single classifier ($\epsilon = 0.2$, $\alpha = 0.025$, $n = 8$) which predicts (from left to right): 6, 8, 1, 5, 9, 3, 0, 2, 2, and 4. Analogously, the third line corresponds to correctly predicted examples of the CIFAR-10 test data set. In the bottom line are the corresponding adversarial BIM attacks on a single classifier ($\epsilon = 0.02$, $\alpha = 0.0025$, $n = 8$) which predicts (from left to right): deer, cat, deer, ship, bird, deer, deer, frog, automobile, and automobile.

these detection systems are again vulnerable to adversarial attacks. To our knowledge, there is no method that can reliably defend or detect *all* kinds of adversarial attacks.

In this manuscript, ensemble methods are used to obtain a classification system that is more robust against adversarial perturbations. The term ensemble method refers to constructing a set of classifiers used to classify new data points by the weighted or unweighted average of their predictions. Many ensemble methods have been introduced in the literature such as Bayesian averaging, Bagging (Breiman, 1996) and boosting (Dietterich et al., 2000). These methods frequently win machine learning competitions, for example the Netflix prize (Koren, 2009). Initial results on using ensembles of classifiers in adversarial context can be found in (Abbasi & Gagné, 2017; He et al., 2017).

This paper is organized as follows: In section 2, some methods for producing adversarial perturbations are briefly introduced. Section 3 describes the defense strategy proposed in this manuscript. In section 4, the previous methods are tested on the MNIST and CIFAR-10 data sets. Finally, in section 5 the conclusions are presented.

2 Adversarial Attack

In this section, two methods for producing adversarial attacks shall be briefly described. In the following, let θ be the parameters of a model, x the input of the model and y the output value associated with the input value x. Further, let $J(\theta, x, y)$ be the cost function used to train the DNN.

Fast Gradient Sign Method

The fast gradient sign method (FGSM) by Goodfellow et al. (2014) simply adds some small perturbations of size $\epsilon > 0$ to the input x,

$$x_{\text{FGSM}} = x + \epsilon \,\text{sign}[\nabla_x J(\theta, x, y)] \,,$$

where the gradient $\nabla_x J(\theta, x, y)$ can be computed using backpropagation. This relatively cheap and simple adversarial perturbation performs well on many DNNs. It is believed that this behavior is due to linear elements such as ReLUs or maxout networks in the DNNs (Goodfellow et al., 2014).

Basic Iterative Method

The basic iterative method (BIM) by Kurakin et al. (2016) is an iterative extension of FGSM. The idea is to choose $\epsilon \geq \alpha > 0$ and then apply some perturbations similar to FGSM to the input x and

repeat the process n times:

$$x_0 = x,$$
$$x_i = \text{clip}_{x,\epsilon}\left(x_{i-1} + \alpha\,\text{sign}[\nabla_{x_{i-1}} J(\theta, x_{i-1}, y)]\right),$$
$$x_{\text{BIM}} = x_n.$$

Here, $\text{clip}_{x,\epsilon}(\cdot)$ refers to clipping the values of the adversarial sample so that they remain within an ϵ-neighborhood of x.

3 Ensemble Methods

Ensemble methods are widely used to improve classifiers in supervised learning (Dietterich et al., 2000). The idea is to construct a set of classifiers that is used to classify a new data point by the weighted or unweighted average of their predictions. In order for an ensemble to outperform a single classifier it must be both accurate and diverse (Hansen & Salamon, 1990). A classifier is said to be accurate if it is better than random guessing, and a set of classifiers is said to be diverse if different classifiers make different errors on new data points.

As expected, when performing adversarial perturbations on new data points different classifiers perform quite differently on these points. Hence, we conclude that diversity on adversarial perturbations is given. Furthermore, for adversarial perturbations with small $\epsilon > 0$, the vast majority of classifiers was accurate. In other words, for any small $\epsilon > 0$, we could not find an adversarial attack that would turn the majority of classifiers into non-accurate classifiers.

In section 4, the following ensemble methods are used. Note that random initialization of the model parameters is used in all methods.

(i) The first method is to train multiple classifiers with the same network architecture but with random initial weights. This results in quite diverse classifiers with different final weights (Kolen & Pollack, 1991).

(ii) The second method is to train multiple classifiers with different but similar network architectures to ensure obtaining a set of even more diverse classifiers. That is, extra filters are used in one classifier or an extra convolution layer is added to another classifier.

(iii) Third, *Bagging* (Breiman, 1996) is used on the training data. The term Bagging is derived from bootstrap aggregation and it consists of drawing m samples with replacement from the training data set of m data points. Each of these new data sets is called a bootstrap replicate. At average each of them contains 63.2% of the training data, where many data points are repeated in the bootstrap replicates. A different bootstrap replicate is used as training data for each classifier in the ensemble.

(iv) The last method is to add some small Gaussian noise to the training data so that all classifiers are trained on similar but different training sets. Note that adding Gaussian noise to the training data also makes each classifier somewhat more robust against adversarial perturbations.

Once an ensemble of classifiers is trained, it predicts by letting each classifier vote for a label. More specifically, the predicted value is chosen to be the label that maximizes the average of the output probabilities from the classifiers in the ensemble.

Table 1: MNIST Network Architecture

Layer Type	Parameters
Relu Convolutional	32 filters (3×3)
Relu Convolutional	32 filters (3×3)
Max Pooling	2×2
Relu Fully Connected	128 units
Dropout	0.5
Relu Fully Connected	10 units
Softmax	10 units

Table 2: Experimental results on the MNIST and the CIFAR-10 data sets

MNIST Accuracy

Test Data		No Attack		Grad. 1		Grad. 2
Type	Method	Single	Ensemble	Single	Ensemble	Ensemble
FGSM	Rand. Ini.	0.9912	0.9942	0.3791	0.6100	0.4517
	Mix. Mod.	0.9918	0.9942	0.3522	0.5681	0.4609
	Bagging	0.9900	0.9927	0.4045	0.6738	0.5716
	Gauss Noise	0.9898	0.9920	0.5587	0.7816	0.7043
BIM	Rand. Ini.	0.9912	0.9942	0.0906	0.6518	0.8875
	Mix. Mod.	0.9918	0.9942	0.0582	0.6656	0.9076
	Bagging	0.9900	0.9927	0.1110	0.7068	0.9233
	Gauss Noise	0.9898	0.9920	0.5429	0.9152	0.9768

CIFAR-10 Accuracy

Test Data		No Attack		Grad. 1		Grad. 2
Type	Method	Single	Ensemble	Single	Ensemble	Ensemble
FGSM	Rand. Ini.	0.7984	0.8448	0.1778	0.4538	0.3302
	Mix. Mod.	0.7898	0.8400	0.1643	0.4339	0.3140
	Bagging	0.7815	0.8415	0.1822	0.4788	0.3571
	Gauss Noise	0.7160	0.7687	0.2966	0.6097	0.4707
BIM	Rand. Ini.	0.7984	0.8448	0.1192	0.5232	0.6826
	Mix. Mod.	0.7898	0.8400	0.1139	0.5259	0.6768
	Bagging	0.7815	0.8415	0.1280	0.5615	0.7166
	Gauss Noise	0.7160	0.7687	0.3076	0.6735	0.7277

Attacking an ensemble of neural networks is more challenging than attacking a single network. In order to attack a network with the methods from section 2 the gradient $\nabla_x J(\theta, x, y)$ must be computed. However, obtaining the gradient for an ensemble of classifiers is somewhat more complicated. Here, the following two methods are used to estimate the gradient of an ensemble, which are referred to as Grad. 1 and Grad. 2 for the rest of this manuscript:

Grad. 1 Use $\nabla_x J(\theta_i, x, y)$ of the i-th classifier. This is clearly not the correct gradient for an ensemble. But the question is whether an attack with this gradient can already mislead all classifiers in the ensemble in a similar manner.

Grad. 2 Compute the sum of the gradients $\sum_i \nabla_x J(\theta_i, x, y)$ from each classifier in the ensemble.

A comparison of the effects of these two gradients for attacking ensembles can be found in section 4.

4 Experiments

In this section the ensemble methods from section 3 are empirically evaluated on the MNIST (LeCun et al., 1998) and the CIFAR-10 (Krizhevsky & Hinton, 2009) data sets. All experiments have been performed on ensembles of 10 classifiers. Note that this choice has been done for comparability. That is, in some cases the best performance was already reached with ensembles of less classifiers while in others more classifiers might improve the results.

A summary of the experimental results can be found in Table 2 and the corresponding visualization in Figure 2. In the following all FGSM perturbations are done with $\epsilon = 0.3$ on MNIST and with $\epsilon = 0.03$ on CIFAR-10. Furthermore, all BIM perturbations are done with $\epsilon = 0.025$ and $n = 8$

iterations on MNIST and with $\epsilon = 0.0025$ and $n = 8$ on CIFAR-10. The abbreviations in Table 2 and in Figure 2 shall be interpreted in the following way: Rand. Ini. refers to random initialization of the weights of the neural network, Mix. Mod. means that the network architecture was slightly different for each classifier in an ensemble, Bagging refers to classifiers trained on bootstrap replicates of the training data, and Gauss noise implies that small Gaussian noise has been added to the training data. Each ensemble is attacked with FGSM and BIM based on the gradients from Grad. 1 and Grad. 2. In Table 2, the term Single refers to evaluating a single classifier.

MNIST

The MNIST data set consists of 60,000 training and 10,000 test data samples of black and white encoded handwritten digits. The objective is to classify these digits in the range from 0 to 9. A selection of images from the data set and some adversarial perturbations can be found in the two top rows of figure 1. In the experiments, the network architecture in Table 1 is used and it is trained with 10 epochs. All results from the experiments are summarized in Table 2.

On unperturbed test data the classification accuracy is roughly 99%. The difference between single classifiers and ensembles is below one percent throughout. The ensembles slightly outperform the single classifiers in all cases.

This picture changes dramatically if the networks are attacked by one of the methods described in section 2. Using the FGSM attack with gradients from Grad. 1 on a single classifier, the classification rate drops down to a range of roughly 35%–56%. The ensembles perform significantly better by producing an accuracy of 57%–78%. Evaluating the same with gradients from Grad. 2 it turns out that ensemble methods still obtain an accuracy of 45%–70%. The higher accuracy of Grad. 1 is expected since in contrast to Grad. 2 it computes the gradients with respect to just one classifier. Nevertheless, the ensembles outperform single classifiers in each case by approximately 7%-22%.

The decrease of the accuracy is even more extreme for single classifiers if the BIM method is used. Here, the accuracy can be as low as around 6% and only the classifiers trained with Gaussian noise significantly exceed the 10%. The accuracy of the ensemble methods against attacks using Grad. 1 is considerably higher with 65%–92%. Furthermore, ensembles are even more robust against BIM attacks based on Grad. 2 with a correct classification rate of 89%–98%. It is surprising that BIM attacks using Grad. 1 are more successful than those using Grad. 2, because Grad. 1 only attacks a single classifier in the ensemble. Concluding, the ensemble methods outperform single classifiers significantly by 37%-85% on BIM attacks.

Focusing on the different defense strategies, we observe that using random initialization of the network weights as well as using several networks of similar architectures for an ensemble generally improves the robustness against adversarial attacks considerably in comparison with single classifiers. Bagging outperforms both of the previous methods on adversarial perturbations, but performs slightly worse on unperturbed test data. Using ensembles with small Gaussian noise on the training data results in the best defense mechanism against adversarial attacks. This may be due to the fact that using additive noise on the training data already makes every single classifier in the ensemble more robust against adversarial perturbations. On the down-side, adding Gaussian noise to the training data performs worst from all considered ensemble methods on test data. However, such an ensemble still performs better than all single classifiers on MNIST.

CIFAR-10

The CIFAR-10 data set consists of 50,000 training and 10,000 test data samples of three-color component encoded images of ten mutually exclusive classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. A selection of images from the data set and some adversarial perturbations can be found in the two bottom rows of figure 1. In all experiments the network architecture described in Table 3 is used and the networks are trained with 25 epochs.

In general, the observations on the MNIST data set are confirmed by the experiments on CIFAR-10. Since the latter data set is more demanding to classify, the overall classification rate is already lower in the attack-free case, where single classifiers reach an accuracy of roughly 72%–80%, while ensembles show a higher accuracy of 77%–84%. Note that there are network architectures in the literature that outperform our classifiers considerably on test data (Graham, 2014).

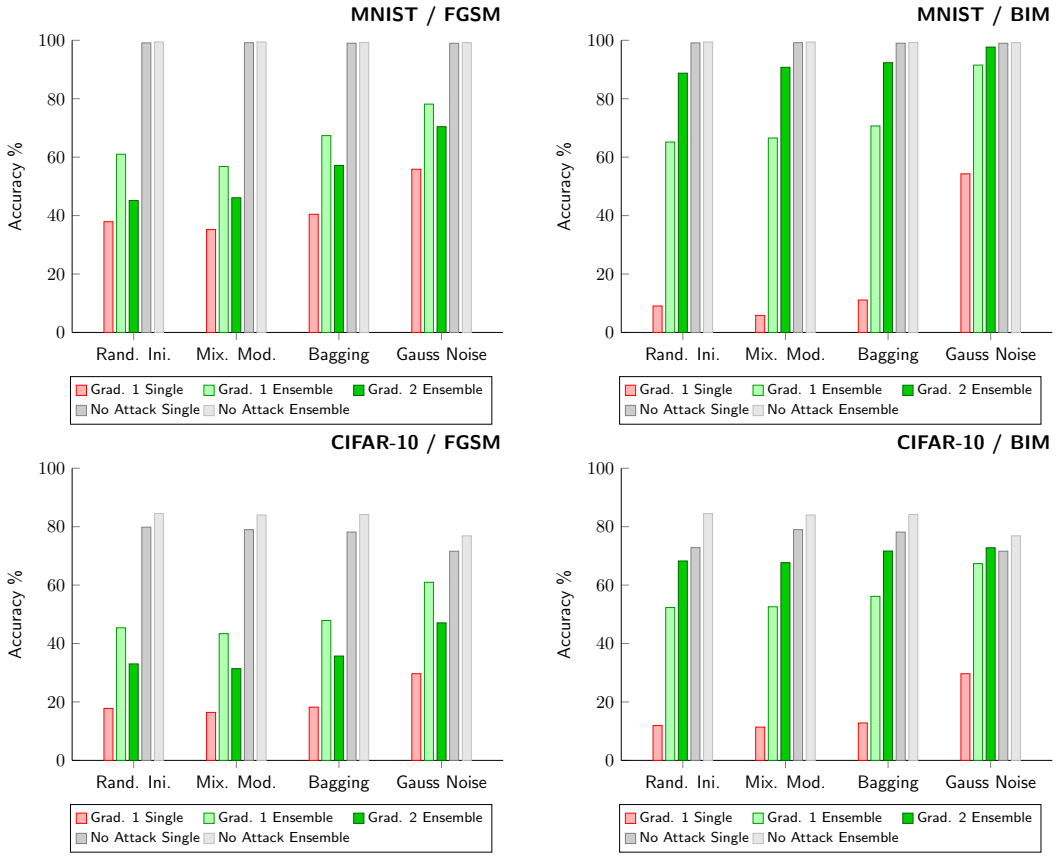

Figure 2: Visual comparisons of the accuracies presented in Table 2. Compared are the MNIST (top row) and CIFAR-10 (bottom row) data sets on the FGSM (left column) and the BIM (right column) attacks. Grad. 1 Single refers to attacks based on Grad. 1 on single classifiers, Grad. 1 Ensemble refers to attacks based on Grad. 1 on ensembles, Grad. 2 Ensemble refers to attacks based on Grad. 2 on ensemble classifiers, No Attack Single refers to single classifier on unperturbed data, and finally No Attack Ensemble refers to ensemble classifiers on unperturbed data.

The FGSM attacks on single classifiers using method Grad. 1 show a drop-down of the accuracy to 16%-30%. In contrast, ensembles are significantly better reaching accuracies of 43%-61% when attacked using Grad. 1 and 31%-47% when attacked with Grad. 2.

When using BIM attacks accuracies for single classifiers lie between 11% and 31%. Again, the ensemble methods outperform the single classifiers reaching accuracies of 52%-67% when attacked using Grad. 1 and 68%-73% when attacked with Grad. 2.

The same observations as on the MNIST data set can be made on the CIFAR-10 data set. All ensemble methods outperform single classifiers when comparing their robustness against adversarial perturbations. FGSM attacks on an ensemble using Grad. 1 outperform those using Grad. 2, as expected. Similar to the MNIST experiments, when using BIM attacks, ensembles are surprisingly more robust against gradient attacks from Grad. 2 than against gradient attacks from Grad. 1. The reason for this might be that the gradient portion from different classifiers using Grad. 2 in the ensemble try to reach a different local maximum and block each other in the following iterations.

As already observed on the MNIST data set, Bagging performs better than random initialization and than using similar but different network architectures. Again, adding small Gaussian noise on the training data performs best on adversarial perturbations but relatively poor on real test data on CIFAR-10.

Table 3: CIFAR-10 Network Architecture

Layer Type	Parameters
Relu Convolutional	32 filters (3×3)
Relu Convolutional	32 filters (3×3)
Max Pooling	2×2
Dropout	0.2
Relu Convolutional	64 filters (3×3)
Relu Convolutional	64 filters (3×3)
Max Pooling	2×2
Dropout	0.3
Relu Convolutional	128 filters (3×3)
Relu Convolutional	128 filters (3×3)
Max Pooling	2×2
Dropout	0.4
Relu Fully Connected	512 units
Dropout	0.5
Relu Fully Connected	10 units
Softmax	10 units

5 CONCLUSION

With the rise of deep learning as the state-of-the-art approach for many classification tasks, researchers noted that neural networks are highly vulnerable to adversarial perturbations. This is particularly problematic when neural networks are used in security sensitive applications such as autonomous driving. Hence, with the development of more efficient attack methods against neural networks it is desirable to obtain neural networks that are themselves robust against adversarial attacks.

In this manuscript, it is shown that several ensemble methods such as random initialization or Bagging do not only increase the accuracy on the test data, but also make the classifiers considerably more robust against certain adversarial attacks. We consider ensemble methods as sole defense methods, but more robust classifiers can be obtained by combining ensemble methods with other defense mechanisms such as adversarial training. Although only having tested simple attack scenarios, it can be expected that ensemble methods may improve the robustness against other adversarial attacks.

REFERENCES

Mahdieh Abbasi and Christian Gagné. Robustness to adversarial examples through an ensemble of specialists. *arXiv preprint arXiv:1702.06856*, 2017.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012.

Thomas G Dietterich et al. Ensemble methods in machine learning. *Multiple classifier systems*, 1857:1–15, 2000.

Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Benjamin Graham. Fractional max-pooling. *arXiv preprint arXiv:1412.6071*, 2014.

Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defenses: Ensembles of weak defenses are not strong. *arXiv preprint arXiv:1706.04701*, 2017.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

John F Kolen and Jordan B Pollack. Back propagation is sensitive to initial conditions. In *Advances in neural information processing systems*, pp. 860–867, 1991.

Yehuda Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 81:1–10, 2009.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal 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.

Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v1. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 2016a.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*, 2016b.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pp. 372–387. IEEE, 2016c.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pp. 582–597. IEEE, 2016d.

Pierre Sermanet and Yann LeCun. Traffic sign recognition with multi-scale convolutional networks. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pp. 2809–2813. IEEE, 2011.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.