BUILDING DEEP NEURAL NETWORKS ROBUST AGAINST ADVERSARIAL ATTACKS

Anthony Dickson
Department of Computer Science
University of Otago
dican732@student.otago.ac.nz

May 31, 2019

ABSTRACT

Deep neural networks, and in particular convolutional neural networks, have seen great success over recent years and have achieved state-of-the-art performance in a wide range of tasks. However, recent work has shown that these networks are vulnerable to adversarial attacks which cause these networks to fail spectacularly. In the case of image data perturbations as small as a single pixel have been shown to be enough to fool these networks into assigning incorrect labels with a high level of confidence, and even just small random translations and rotations can fool convolutional neural networks in a similar way. A growing body of work has started on addressing these issues and offer ways of mitigating the effects of adversarial attacks. In this literature review I will give a brief overview of adversarial attacks and the methods proposed that aim to build neural networks robust against these attacks.

(a) (b) (c) (d)

Figure 1: Examples of successful impersonation attacks where a deep neural network (DNN) is forced to identify somebody as a specific person and dodging attacks where the DNN is forced to identify someone as anybody else. Figure (a) shows S_A (subject A, top) and S_B (subject B, bottom) dodging against a DNN. Figure (b)–(d) show impersonations. Impersonators carrying out the attack are shown in the top row and corresponding impersonation targets in the bottom row. Figure (b) shows S_A impersonating Milla Jovovich (by Georges Biard; source: https://goo.gl/GlsWlC); (c) S_B impersonating subject C S_C; and (d) S_C impersonating Carson Daly (by Anthony Quintano; source: https://goo.gl/VfnDct).[1]

[1]Figure and caption adapted from [1].

(a) (b)

Figure 2: Adversarial examples generated for AlexNet [3]. (Left) is a correctly predicted sample, (center) difference between correct image, and image predicted incorrectly magnified by 10x (values shifted by 128 and clamped), (right) adversarial example. All images in the right column are predicted to be an *"ostrich, Struthio camelus"*. Average distortion based on 64 examples is 0.006508. The examples are strictly randomly chosen. There is not any postselection involved.[2]

1 Introduction

Convolutional neural networks have seen great success over recent years over a vast range of computer vision tasks [2, 3, 4]. However, recent work has identified adversarial attacks that can cause state of the art deep neural networks (DNN) to fail spectacularly [5, 6, 7]. While most of the work on adversarial attacks has been focused on convolutional neural networks that deal with image data, there has been work showing that adversarial attacks are possible for other domains such as natural language processing and speech recognition [8, 9].

In the context of computer vision tasks, adversarial attacks perturb an image from the training dataset such that the target network assigns the incorrect label with a high level of confidence (see Figure 2). The perturbations applied to the original images can be arbitrarily large or even as small as a single pixel [10]. Sometimes it is enough even to just randomly rotate and translate the image [11]. These attacks are typically engineered such that the original image and adversarial example are indistinguishable to the human eye.

There have been several theories proposed about how and why adversarial attacks work. Quoting the authors of [12]:

> Different works have argued that this vulnerability stems from their discontinuous nature [5], their linear nature [6], or is a result of high-dimensional geometry and independent of the model class [13].

An intriguing property of these adversarial attacks is that an adversarial example generated for a given network can successfully fool other networks, even ones that have a different architecture or were trained on different data [5].

These vulnerabilities also raise questions about our understanding of the inner workings of convolutional neural networks. Previous work on visualising the features that convolutional neural networks (CNN) learn would suggest that these networks learn recognisable features [14, 15, 16], and the fact that these networks create and use recognisable high-level features might suggest that these networks have a high-level understanding of visual data that is in some capacity comparable to that of humans. However a recent study would suggest that CNNs are more sensitive to texture than to shape [17], which is contrary to popular belief that CNNs recognise objects primarily through shape in a way similar to that of humans.

[2]Figure and caption adapted from [5].

Although machine learning agents built upon neural networks have been approaching human-level, or even super-human levels, of performance in various tasks [4, 18], it is concerning that they can be so easily fooled. The authors of [1] highlight a particular concern regarding systems that use neural networks for facial recognition (see Figure 1), and how their vulnerability to adversarial attacks could possibly lead to security issues in such systems.

This all goes to show that while CNN have reached impressive levels of performance, they are still far from being on par with humans in terms of robustness. So this leads me to the question:

How can we can build neural networks that are robust against adversarial attacks?

In this literature review I will provide a brief survey of some of the literature that addresses this question.

2 Adversarial Training

Szegedy et al. in their 2013 paper [5] were among the first to show the vulnerability of DNNs to adversarial examples. They formulate a method of constructing adversarial examples. They cast this problem as a box-constrained optimisation problem where they try to find a perturbation that results in a image that is as close as possible to the original image and causes the network to assign the wrong label. More formally they try solve the following optimisation problem:

$$\text{Minimise } ||r||_2 \text{ subject to } f(x+r) = l \text{ and } x + r \in [0,1]^m$$

where r is the perturbation to be applied to the input x, $f(.)$ is a function that maps an input to a label, l is a label that differs from the correct label of x, and m is the number of dimensions in the input. They use this method to generate adversarial examples for AlexNet and discover that these examples fool the network even though the examples are indistinguishable from the original images (see Figure 2).

The authors then run a series of experiments to explore the effects of adversarial examples on generalisation. In their preliminary experiments they train a neural network with two hidden layers of 100 units each on the MNIST dataset and a pool of adversarial examples. The pool of adversarial examples are updated as training progresses. Their main result is that they were able to decrease the test error rate from 1.6% to 1.2% by training on adversarial examples. In other words, training the network on adversarial examples improved generalisation.

In further experiments they explored how adversarial examples generated on one model *generalise* to other models. They train five neural networks with either zero or two hidden layers and either 100 or 200 units in the hidden layers in a way similar to what was described above. Then they generated adversarial examples from each of these networks. They then fed these examples to the all of the other models other than the one that they were generated on. The interesting thing about their results is that sometimes the other networks were robust to the adversarial examples from the other networks only reporting an error rate of about 2.2%. However, sometimes these networks would misclassify up to 87.1% of these adversarial examples.

They further show that it is possible to train identical networks on subsets of the MNIST dataset, generate adversarial examples from the resulting networks, and fool the other networks with these examples. This method is slightly less effective, but nonetheless it demonstrates that adversarial examples are not particular to the network that generated them and generalise to some extent to other networks that either have a different architecture, were trained on a different set of data, or both.

They also run some informal experiments using convolutional neural networks. While they do not present the full results as with the fully connected neural networks trained on MNIST, the few examples they give are convincing (see Figure 2) and serve as a motivating example for future work.

Overall Szegedy et al. bring forth the problem of adversarial attacks and propose a method for producing such examples. With their method they are able to consistently generate examples that fool a given network and are visually identical with the original image. They further show that these examples can fool networks that have a different architecture or networks with the same architecture that were trained on different sets of data. Their work does not necessarily give us any answers on how to build networks that is robust to adversarial attacks, but it does shows that incorporating adversarial examples into training data can improve generalisation on clean data. Their work also serves as the launching point for much of the work that has come afterwards.

3 Robust Optimisation

Rather than explicitly training on adversarial examples as Szegedy et al., later authors incorporate adversarial attacks into the optimisation process [6, 19, 12, 20].

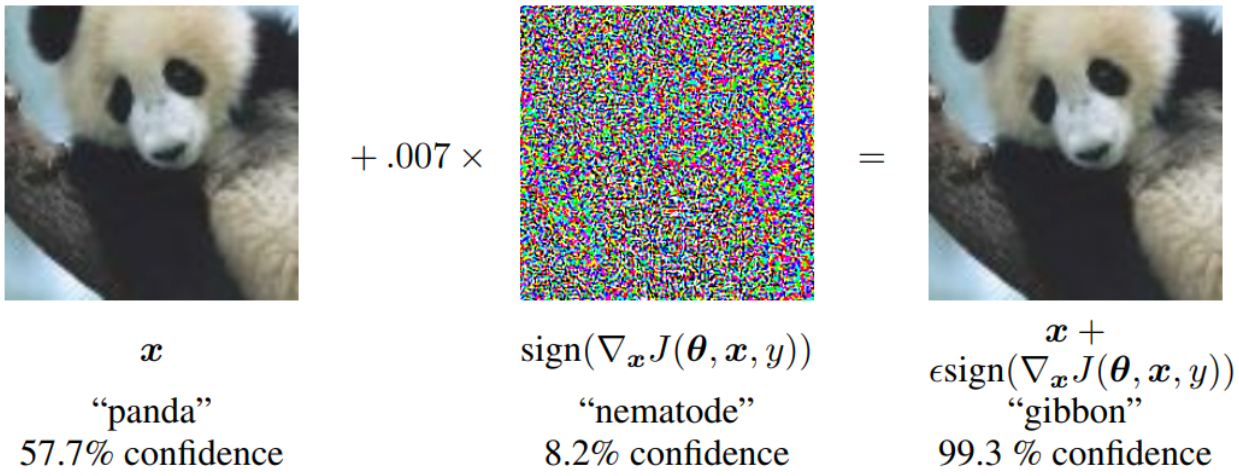

Figure 3: A demonstration of fast adversarial example generation applied to GoogLeNet [21] on ImageNet. By adding an imperceptibly small vector whose elements are equal to the sign of the elements of the gradient of the cost function with respect to the input, we can change GoogLeNet's classification of the image. Here the ϵ of .007 corresponds to the magnitude of the smallest bit of an 8 bit image encoding after GoogLeNet's conversion to real numbers.[4]

3.1 Adversarial Loss as a Regulariser

In [6] not only do Goodfellow et al. show that deep neural networks (DNN) can be fooled easily they also suggest several reasons as to why this happens. One of their main arguments is that DNNs are designed to be linear in order to facilitate efficient training, where they give examples such as the ReLU activation and manipulating activation values to be close to zero in sigmoid layers to avoid the effects of saturation (the sigmoid function near zero acts similar to a linear function).

They introduce a fast method for generating adversarial examples and explore how networks trained on MNIST cope with these. Their method, called the fast gradient sign method (FGSM), is defined as such:

$$\boldsymbol{\eta} = \epsilon \text{sign}(\nabla_{\mathbf{x}} J(\boldsymbol{\theta}, \mathbf{x}, y)). \tag{1}$$

where $\boldsymbol{\eta}$ is the resulting perturbation, ϵ is some small scaling factor, $\boldsymbol{\theta}$ are the parameters of a model, $\mathbf{x}$ the input to the model, y the targets/labels associated with $\mathbf{x}$, $J(\boldsymbol{\theta}, \mathbf{x}, y)$ is the loss to train the model with, and $\nabla_{\mathbf{x}} J(\boldsymbol{\theta}, \mathbf{x}, y)$ is the gradient of the loss function with respect to the input $\mathbf{x}$. Using FGSM they can construct an adversarial example by adding the perturbation $\boldsymbol{\eta}$ to the original input x. The reason why FGSM is fast is that once the gradient of the loss function with respect to the input is known an adversarial example can be made, whereas the method that Szegedy et al. introduced requires an iterative solution to be found via L-BFGS (an alternative optimisation algorithm to gradient descent) which can be slow. While the quality of the solution found by the latter may be higher, FGSM is much faster and is still able to fool the networks consistently[3]. An example of an adversarial example produced with FGSM is shown in Figure 3.

They experiment with adversarial training on the MNIST dataset and with their method they are able to reduce the error rate of their neural networks from 89.4% to about 17.9% on a test set of adversarial examples. They use a loss function that is a weighted sum of the loss on the clean input and the loss on the adversarial example:

$$\tilde{J}(\boldsymbol{\theta}, \mathbf{x}, y) = \alpha J(\boldsymbol{\theta}, \mathbf{x}, y) + (1 - \alpha) J(\boldsymbol{\theta}, \mathbf{x} + \boldsymbol{\eta}).$$

where α is a scaling factor in the interval $[0, 1]$. In their experiments they set $\alpha = 0.5$ which means they weighted the normal loss and the adversarial loss equally. The second term acts sort of like a regularisation term.

Interestingly, while adversarially trained networks exhibit lower error rates, when they misclassified adversarial examples they still did so with a high level of confidence (on average about 81.4%). So even with adversarial training there are still adversarial examples that fool the network with a high level of confidence. This suggests adversarial training does not make models that are impervious to further attacks.

[3] They were able produce a set of adversarial examples that fooled the target with a 100% success rate, whereas FGSM was only able to fool the target network 89.4% of the time.

[4] Figure and caption adapted from [6].

3.2 The Saddle-Point Problem

In their 2017 work [19] Madry et al. take a slightly different view on the problem and investigate adversarial attacks under the lens of optimisation. They cast the problem of training robust models as a saddle-point problem for which if a good solution is found, then a guarantee can be made against the specific set of adversarial attacks that the saddle-point problem addresses. The saddle-point problem they aim to solve is defined as such:

$$\min_{\theta} \rho(\theta) \text{ where } \rho(\theta) = \mathbb{E}_{(x,y)\sim D}\Big[\max_{\delta \in S} L(\theta, x+\delta, y)\Big]$$

where $\rho(.)$ is the outer loss function, θ is the model parameters, δ is perturbation from the set of allowed perturbations S, L(.) is the inner adversarial loss function, and $\mathbb{E}_{(x,y)\sim D}[\dots]$ is the expected value of the worst case adversarial attack for a given input x and label y that are part of the dataset D. The saddle-point problem also goes under the name of minimax in game theory, and we can indeed characterise this formulation as a model playing a sort of game against an adversary. In this imagined game the adversary is playing the role of saboteur and trying to fool the model with subtle modifications to the inputs and the model is trying to correctly label all of the inputs.

They use FSGM (Equation 1) for generating adversarial examples. In their method they aim to train a robust DNN by first performing projected gradient descent (a bounded form of gradient descent) on the inner loss function to find the worst case adversarial attack, and then performing stochastic gradient descent to minimise the expected worst case adversarial loss. The authors remark that this is equivalent to performing SGD on adversarial examples (similar to what Szegedy et al. did in [5]), except that the main difference is that rather than training on a pool of pre-generated adversarial examples we are essentially creating a new adversarial example for each training example. They argue that the above training method is not enough to create a robust model and that a higher model capacity is also required. Adversarial examples are said to create a more complex decision boundary. Thus a higher model capacity is needed to learn the more complex decision boundary in order for adversarial examples to be correctly classified. In further experiments the author show that using the strongest adversary and increased model capacity reduces the ability of adversarial examples to generalise between different models.

In [12] Schmidt et al. build upon previous work including work from [19]. They point out an interesting characteristic of DNNs trained on CIFAR10 compared to DNNs trained on MNIST: DNNs trained on adversarial examples for MNIST generalise well for both the original unperturbed examples and adversarial examples, however for DNNs trained on CIFAR10 there is a noticeable gap in generalisation between the original examples and adversarial examples. Their main argument is that many more examples are needed in order to produce robust classifiers. In their experiments they look at the effect of data set size on adversarial robustness. They show that for Gaussian distributed data (i.e. natural data) the class of model used does not matter (i.e. the adversarial vulnerabilities are more intrinsic to the data rather than the model) and linear models are able to achieve optimal robustness. Furthermore, they stated that "robustness also strongly depends on properties of the underlying data distribution". This statement leads nicely into the next paper that I will talk about.

4 Adversarial Examples Are Not Bugs, They Are Features

Most work to date has viewed adversarial examples as faults in the current systems, something that must be fixed. In recent paper Ilyas et al. take an interesting position on the topic [22]. Their position is that adversarial examples are a feature of the supervised learning paradigm.

In supervised learning the sole objective to minimise some loss function. Thus, a model is going to use any features available to it, even if the features are unintelligible to humans. These models have no inherent preference over which features it uses and they simply use whatever is a good predictor and leads to low loss. This means if we were to have two features, one that is a high-level feature such as a hand and another that is an ambiguous pattern of noise and assuming these features we equally predictive of the true label, then the model would use both.

This means that these models are going to rely on a combination of both robust and non-robust features. Ilyas et al. describe robust features as highly predictive features that are robust to adversarial perturbations and non-robust as features that are highly predictive but not robust against adversarial perturbations. If models rely on a combination of these robust and non-robust features then it follows that they will be vulnerable to adversarial attacks.

Their notion of robust and non-robust features also provide an explanation as to why adversarial examples generalise across models. They posit that these models are all learning similar non-robust features, and that the adversarial examples are taking advantage of these common features. They found that in popular machine learning datasets such as CIFAR10 and ImageNet that these robust and non-robust features are indeed present.

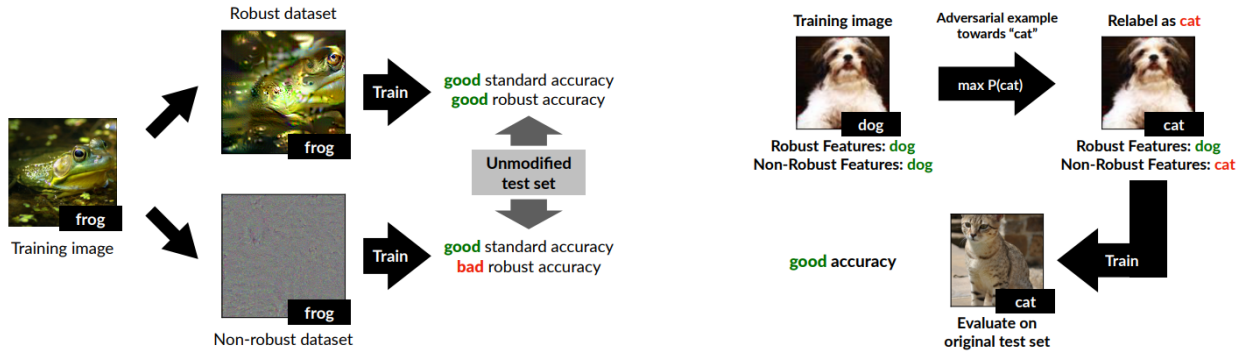

Figure 4: A conceptual diagram of the experiments performed in [22]. In (a) the authors disentangle features into combinations of robust/non-robust features (Section 3.1). In (b) the authors construct a dataset which appears mislabeled to humans (via adversarial examples) but results in good accuracy on the original test set.[5]

In one of their experiments they set out to provide some supporting evidence for the claim that the reliance on non-robust features is a cause of the vulnerability of DNNs to adversarial examples. They take the CIFAR10 dataset and produce two parallel datasets, one that contains solely robust features and another that contains solely non-robust features. An example is shown in Figure 4. As we can see in this example the non-robust feature looks quite similar to the adversarial perturbations. In their experiments both networks trained with the robust and non-robust datasets achieved roughly the same level of generalisation on the clean test set, however on the adversarial test set the network trained on the robust dataset achieved a higher level of accuracy. Perhaps as expected, the adversarially trained networks were more robust against adversarial examples.

4.1 An Interesting Link

There is an interesting link between this paper and a paper [17] that is not directly related to adversarial attacks. In the latter they give evidence that suggests that DNNs trained on the ImageNet dataset are more sensitive to texture than to shape. That is to say that the networks they tested would make decisions/classifications based more on the texture of an image than the shapes present in it (see Figure 5). This runs in contrast to how humans are thought to be more focused on the shape of objects, rather than the texture.

The authors run experiments that make a DNN make decision based more on shapes than texture. They do this by using a technique called style transfer (see Figure 6). The main idea behind this is that it severely weakens the correlation between texture and the class labels, thus forcing the networks to rely on other features instead of texture. An interesting result from their experiments is that forcing these networks to make decision based more on shape lead to them performing more closely to the human participants compared to the networks trained on just the original ImageNet dataset.

So what is the link between this paper and Ilyas et al.'s work? It is possible that the reliance on texture by these models is in fact a reliance on non-robust features, and by decoupling texture from the labels the authors of [17] are actually forcing the networks to learn more robust features. Interestingly, in Ilyas et al.'s work there is a slight drop in generalisation of models trained solely on the robust features, and in Geirhos et al.'s work a similar thing happens when they train networks solely on the Stylized-ImageNet dataset. So it would appear that an interesting link exists between the two pieces work.

5 Discussion &Conclusion

Szegedy et al. brought attention to an intriguing property of neural networks (adversarial examples) in their 2013 work [5] which has prompted much research on the topic. They put forth a method for generating adversarial examples and

[5]Figure and caption adapted from [22].
[6]Figure and caption adapted from [17].
[7]Figure and caption adapted from [17].

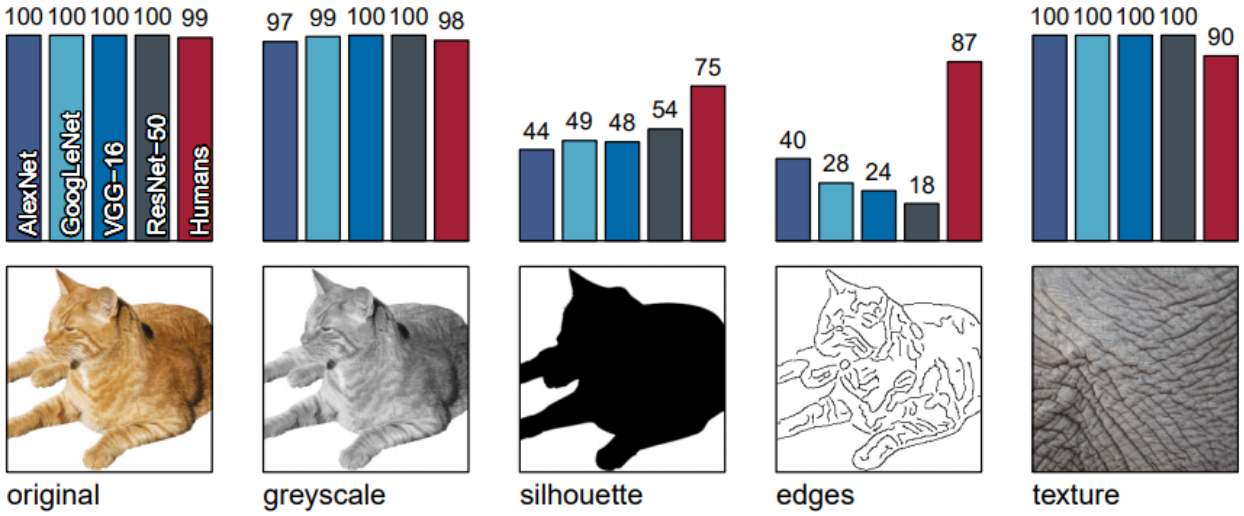

Figure 5: Accuracy of predictions by various DNNs and human participants for varying stimuli. The first four images are of a cat and the last is of an Indian elephant's skin. Notice how on the first four images the human participants remained relatively confident that the image was of a cat, whereas the neural networks did not handle the silhouette or edges very well.[6]

Figure 6: Visualisation of Stylized-ImageNet (SIN), created by applying AdaIN style transfer to ImageNet images. Left: randomly selected ImageNet image of class ring-tailed lemur. Right: ten examples of images with content/shape of left image and style/texture from different paintings. After applying AdaIN style transfer, local texture cues are no longer highly predictive of the target class, while the global shape tends to be retained. Note that within SIN, every source image is stylised only once.[7]

showed that they generalise between different architectures and models trained on different sets of data. They also showed that training jointly on original data and adversarial examples can improve generalisation.

Following on from this paper there has been various attempts at explaining the vulnerability of neural networks to adversarial attacks and various methods for building neural networks that are robust to said attacks have been proposed. The main approach that I covered was the robust optimisation approach where adversarial attacks are built into the optimisation process. The authors of [6] propose a composite loss function that jointly considers the loss on the original input and the loss on the adversarial example. The authors of [19] and [12] cast the problem as solving the saddle-point problem where an inner function modelling adversarial loss is maximised, which essentially considers the worst case attack, and then minimises the loss with respect to this worst case attack. This approach builds robust networks that achieve high accuracy on adversarial examples.

Finally, the authors of [22] give an interesting perspective on adversarial attacks and suggest that they are a feature of the supervised learning process. They show that forcing the network to rely more on robust features results in networks that are more robust to adversarial examples, however the robust optimisation approach appears to be more effective. They also hint at a trade-off between generalisation ability and reliance on solely robust features. There is also an interesting link between this paper and [17] where the latter may provide an alternate high-level explanation as to why the standard training process leaves neural networks vulnerable to adversarial attacks.

One paper that I did not discuss is [23] which proposes giving neural networks the *full Bayesian treatment* in order to make neural networks robust against adversarial attacks.

There is also an interesting similarity between robust optimisation methods for adversarial training and generative adversarial networks (GAN). Ian Goodfellow published the landmark paper on GANs in the same year that he published his other paper on adversarial attacks [6]. So I wonder if these methods were part of what inspired GANs. Furthermore, it would be interesting to see if the discriminator network part of GANs exhibit a similar level of adversarial robustness as networks trained with the robust optimisation methods.

References

[1] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.

[2] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 253–256. IEEE, 2010.

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

[4] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[5] 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.

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

[7] 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*, pages 2574–2582, 2016.

[8] Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016-2016 IEEE Military Communications Conference*, pages 49–54. IEEE, 2016.

[9] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. Did you hear that? adversarial examples against automatic speech recognition. *arXiv preprint arXiv:1801.00554*, 2018.

[10] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

[11] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017.

[12] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pages 5014–5026, 2018.

[13] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

[14] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[15] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[16] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*, 2017.

[17] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[18] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[20] Ziang Yan, Yiwen Guo, and Changshui Zhang. Deep defense: Training dnns with improved adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 419–428, 2018.

[21] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[22] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*, 2019.

[23] Nanyang Ye and Zhanxing Zhu. Bayesian adversarial learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6892–6901. Curran Associates Inc., 2018.