ADVERSARIAL PATCH ATTACKS ON VGGFACE2

Matthew Li

Advisor: Dr. Konrad Kording

Presented in Completion of the Requirements for the Master of Science in Engineering (MSE) in Data Science

Background

Neural networks are a more recent scientific discovery that have changed the way we approach machine learning problems. Deep neural networks are popular machine learning models that perform well on a variety of complex tasks, given adequate amounts of training data and computational power. Convolutional Neural Nets (CNNs) are used predominantly for image processing tasks. By using convolutional kernels, CNNs are able to provide translation-equivariant responses that are scale invariant feature maps of the desired target. While the size of the convolutional filters and kernels are fixed, the weights are learned through gradient descent. Recent and more advanced models have achieved better-than-human performance on facial recognition, object classification and language processing. While CNNs are exceptional in various tasks, it has been shown that they can easily be fooled by adversarial examples.

Adversarial attacks

Adversarial machine learning attempts to exploit models by taking advantage of obtainable parameters. The field originated in "evasion attacks" on spam filters, as malicious actors realized they could insert "good words" into their email to get them to pass. The field has rapidly evolved since then, with attacks on advanced learning models (Ekyholt 2018) showing that some strips of tape on a stop sign can cause self-driving cars to ignore the stop sign entirely or even accelerate to dangerous speeds.

Figure 1. An adversarial attack on machine learning algorithms for self-driving cars

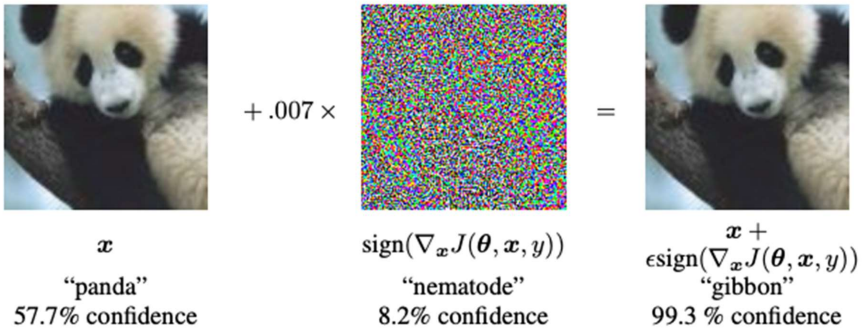
Ekyholt 2018

Other examples of adversarial attacks include a 3-D printed turtle that causes Google's object detection AI to classify it as a rifle from any angle (Athalye 2017), and t-shirts that cause one to be invisible to object detection software (Xu 2020).

Adversarial attacks can be classified into white box and black box attacks. White box attacks assume knowledge and full information of the weights and biases of the target network, while black box attacks resemble real life scenarios with the adversary having no knowledge of the model to be attacked. Information about the system has to be gained from querying the target network, which is both slow and computationally expensive. For the purposes of this paper, I decided to proceed with the white box approach as it was more feasible given the resources I had access to.

One commonly used algorithm in Adversarial Learning is the Fast Gradient Sign Method (Goodfellow 2014). It exploits the gradients of a loss function (MSE loss or Cross Entropy Loss) with respect to the input image and then uses the sign of the gradients to create a new image. An epsilon factor is included to determine the strength of the perturbation. Goodfellow showed that it was possible to trick the target network into an incredibly confident answer.

Figure 2. The Fast Gradient Sign Method



Goodfellow 2014

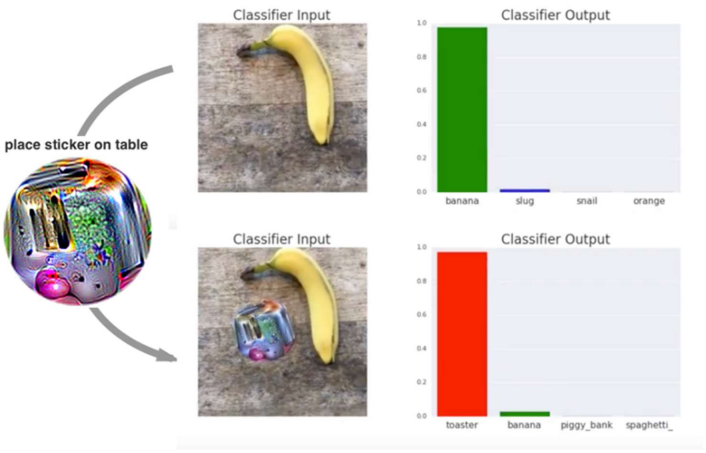
Related Works

For the target of my thesis, I wanted to apply adversarial learning to facial recognition software. In terms of the scope of the project, there has been literature published on adversarial attack on object classification.

My interest lie in combatting mass surveillance, in the hopes that my research could help political dissidents escape monitoring or help ensure differential privacy in the information age. Being invisible to facial recognition software online could ensure privacy in online databases would act as a form of digital camouflage. Attacking facial recognition software on videos proved challenging as the computational cost was high, and dealing with real world lighting conditions proved tricky. I wanted an attack that was generalizable to a large corpus of images and modified the picture to a lesser extent that a human could see what the correct class of the image was.

Adversarial Patch attacks distort only a small portion of the image, rather than modifying the input on the entire image (Brown 18).

Figure 3. The effects of an adversarial patch on classifier output



Brown 2018

$$\hat{p} = \arg\max_{p} \mathbb{E}_{x \sim X, t \sim T, l \sim L} \left[\log \Pr(\hat{y} | A(p, x, l, t)) \right]$$

For the training of the path, we used a variation of the Expectation over Transformation framework (Athalye 2018). In the above formula, p-hat is tasked to optimize the above function. X is over the training set of images, T is transformations of the image, and L is all possible locations for the patch. As the expectation is over different images, the patch should work, regardless of the background. As I did not optimize for transformations, the distribution did not include distribution T.

These attacks exploit the way that these images are created. As images can contain multiple items, there is only one label in the ground truth. The network then learns to detect the "brightest" and most salient image. Thus, the network can create images that create items that are much more salient than real world objects as the patch remains constant through all images that it superimposed upon.

Dataset Selection

VGGFace2 contains 3.31 million images of 9131 subjects, with an average of 362.6 images for each subject. Images are pulled from Google Images, and most curated images are of celebrities

and superstars (Cao 2017). VGGFace2 is quite commonly used for Facial Recognition model training, and these models have the following accuracy:

Figure 4. Accuracy of different facial recognition models (Cao 2017)

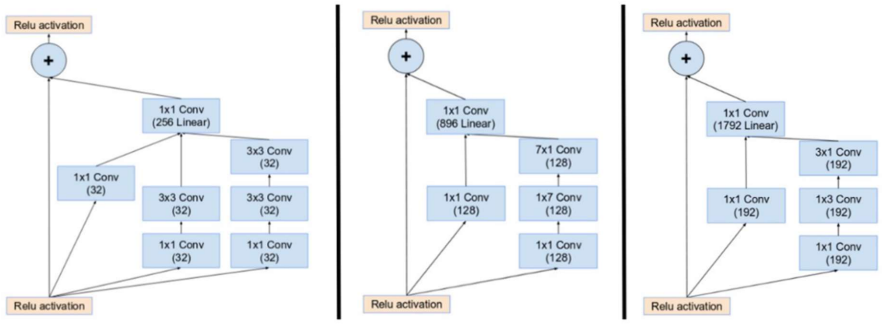
Architecture	Feat dim	Pretrain	TAR@FAR = 0.001	TAR@FAR = 0.01	Model Link
ResNet-50	2048	N	0.878	0.938	Caffe, MatConvNet, PyTorch
ResNet-50	2048	Y	0.891	0.947	Caffe, MatConvNet, PyTorch
SE-ResNet-50	2048	N	0.888	0.949	Caffe, MatConvNet, PyTorch
SE-ResNet-50	2048	Y	0.908	0.956	Caffe, MatConvNet, PyTorch
ResNet-50-256D	256	Y	0.898	0.956	Caffe, MatConvNet, PyTorch
ResNet-50-128D	128	Y	0.904	0.956	Caffe, MatConvNet, PyTorch
SE-ResNet-50-256D	256	Y	0.912	0.965	Caffe, MatConvNet, PyTorch
SE-ResNet-50-128D	128	Y	0.910	0.959	Caffe, MatConvNet, PyTorch

Model Architecture

When considering facial recognition models to fool I had two main criteria – computational speed and accuracy. While training one pass of the FGSM is not as time intensive as learning the whole weights and parameters of an entire model, I wanted it to still be able to run relatively quickly on my local machine.

Inception nets draw on the idea that while deeper CNNs are generally more accurate but slower, one can achieve similar levels of accuracy at a lower computational cost by passing it through smaller 1x1 filters.

Figure 5. Stem Modules

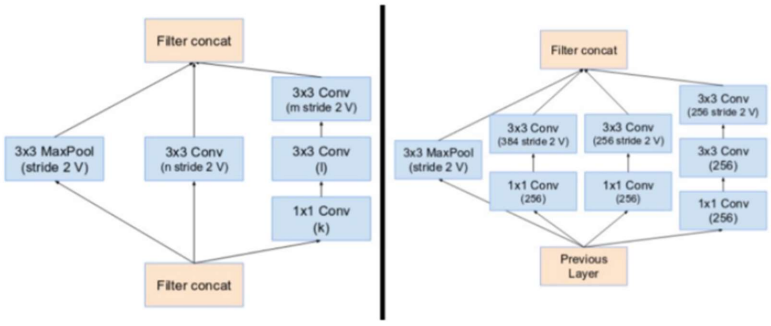


Szegedy 2016

Computation is more efficient when the dimensions of the input data from the previous layer are not significantly altered, as too much factorization results in loss of information. Reducing the size of the filters reduces computational cost, as a 5x5 convolutional node can be broken down into 2 layers of 3x3 nodes, reducing computational cost by 2.78x (Szegedy 2016).

Another factorization method replaces nodes of size nxn with 1xn and nx1. A 5x5 node would be replaced with a 1x5 and a 5x1 and produces a longer network, reducing computational cost and reducing the chance of information loss. The newest innovation of Inception ResNet was make the network more uniform as a whole. The first layer of data processing before the inception modules are reduction blocks which regulate the breadth and depth of the network.

Figure 6. Reduction Blocks

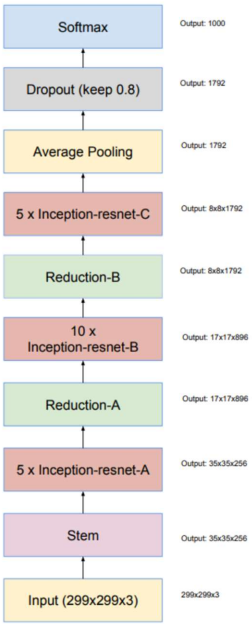


Szegedy 2016

Other than the stem blocks, reduction blocks, and inception modules, the architecture is relatively straight forward. There is an average pooling layer for downsampling, and a dropout

layer to prevent overfitting. This is then activated with a softmax layer which then determines the ending probabilities of each class.

Figure 6. Inception ResNet v1 Architecture



Szegedy 2016

Results

I leveraged similar attack framework of that of the original patch attack and applied the same logic to facial recognition attacks. While the original attack mislead object classifiers, I wanted to mislead facial recognition technology. My target was "alex trebek", the beloved host of Jeopardy. While the original patch attack proposed by Google included rotations, different lighting conditions, as well as other translations during training to make it more robust, my primary objective was the misclassification of images so I omitted those steps. As avenues for further research, I would like to make my patch attack more resistant in real life lighting conditions. Running through an FGSM, I obtained the following 64x64 patch:

Commented [LMM1]: What is the original patch attack, be more clear about that/who did it/how does it work

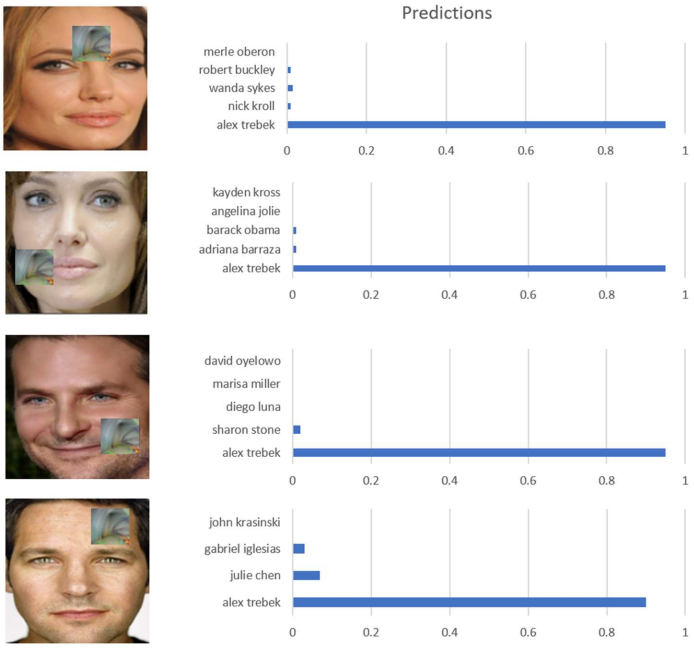
Commented [LMM2R1]: Would go into more detail about how the google patch attack works + how yours builds on it/differs

Commented [LMM3]: I would put this in a section at the end and expand on what you mean by resistant in real life lighting

Figure 7. 64x64 adversarial patch designed to target "alex trebek"

After superimposing the patch onto training images, I then tasked a black box Inception ResNet to make the following predictions:

Figure 8. Image predictions after applying “alex trebek”-targeted adversarial patch to various training images



Data produced by author.

Running it through a cohort of 2000 images, we achieved 96.7% of the time for Top-1 predictions and 99.2% for Top-5 predictions (Alex Trebek was chosen as the most likely class 96.7% of the time, and was included in the Top 5 99.2% of the time). Much like Goodfellow’s work, the network displayed higher confidence on the perturbed images than the original. By exploiting patterns that are incredible salient to our target network as the patterns and features that Inception ResNet learns are exploited by FGSM to create these adversarial images.

Limitations and Avenues for Further Research

VGGFace2 is by no means a perfectly equitable dataset. It is not representative of the racial demographics of the US, let alone the world, and should only be used as a corpus of training images and not for any commercial applications. While the creators of VGGFace2 made an effort to include as many demographics as possible in this dataset, all images were pulled from Google Images and may encode implicit biases on who we consider celebrities and superstars. Further testing should be done on datasets that are proven to be more equitable.

Bibliography

Athalye, Anish, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. “Synthesizing Robust Adversarial Examples.” ArXiv:1707.07397 [Cs], June 7, 2018. http://arxiv.org/abs/1707.07397.

Bhambri, Siddhant, Sumanyu Muku, Avinash Tulasi, and Arun Balaji Buduru. “A Survey of Black-Box Adversarial Attacks on Computer Vision Models.” ArXiv:1912.01667 [Cs, Stat], February 7, 2020. http://arxiv.org/abs/1912.01667.

Brown, Tom B., Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. “Adversarial Patch.” ArXiv:1712.09665 [Cs], May 16, 2018. http://arxiv.org/abs/1712.09665.

“Convolutional Neural Network.” In Wikipedia, April 28, 2022. https://en.wikipedia.org/w/index.php?title=Convolutional_neural_network&oldid=1085146109.

Eykholt, Kevin, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. “Robust Physical-World Attacks on Deep Learning Models.” ArXiv:1707.08945 [Cs], April 10, 2018. http://arxiv.org/abs/1707.08945.

labsix. “Fooling Neural Networks in the Physical World.” Accessed May 10, 2022. https://www.labsix.org/physical-objects-that-fool-neural-nets/.

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. “Explaining and Harnessing Adversarial Examples.” ArXiv:1412.6572 [Cs, Stat], March 20, 2015. http://arxiv.org/abs/1412.6572.

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning.” ArXiv:1602.07261 [Cs], August 23, 2016. http://arxiv.org/abs/1602.07261.

VGGFace2 Dataset for Face Recognition (Website). MATLAB. 2018. Reprint, VGG@Oxford, 2022. https://github.com/ox-vgg/vgg_face2.

Xu, Kaidi, Gaoyuan Zhang, Sijia Liu, Quanfu Fan, Mengshu Sun, Hongge Chen, Pin-Yu Chen, Yanzhi Wang, and Xue Lin. “Adversarial T-Shirt! Evading Person Detectors in A Physical World.” ArXiv:1910.11099 [Cs], July 5, 2020. http://arxiv.org/abs/1910.11099.