Review of Independent Study on Adversarial Machine Learning - Spring 2021

Wes Robbins, Seth Bassetti

April 28, 2021

1 Introduction

Throughout the semester of spring 2021 we completed an Independent Study on Adversarial Machine Learning (AML). This was completed at Montana State University and advised by Travis Peters.

In this paper we give a summary of the topic of AML and an overview of the work we completed for the study. The purpose of this paper is three fold. The first is reflection. By thoroughly looking back over our work we aim to give ourselves a realistic picture of what we learned, where we made progress, and where we didn't. Secondly, we try to make this paper resourceful for someone who is 1) interested in an introduction to Adversarial Machine Learning or 2) is interested in doing a similar independent study. Lastly, this paper serves as our final graded assignment for the semester.

The rest of the paper is organized into five sections. In the remainder of this section we discuss our collaboration process and introductory work. In the Section 2 we give a summary of the the topic of AML along with several examples. In Section 3 we review the work we completed and what we learned during our 'Examination of Attacks and Defenses' section of our study. In Section 4 we discuss different ideas we considered for a final research problem. Then, in Section 5 we talk in-depth about *Feature Attribution for Defending Adversarial Examples*, which is the research topic we focused on the last few weeks of the semester. In Section 6 we present our conclusions.

1.1 Collaboration

Collaboration was important throughout the independent study. This includes collaboration between the authors and collaboration with our advisor and other peers. All following following work discussed in this review was aided by the collaboration processes discussed here.

During the semester we had weekly meetings with our advisor. These meetings were invaluable for getting feedback and talking through our ideas. An outside perspective was very helpful for examining and improving our work. Additionally, we had weekly stand-up meetings with the computer security group. In these meetings we discussed our progress and were able to get inspiration from the projects of the other group members. These meetings also offered another opportunity to talk through out work.

Throughout the semester the authors met once or two times a week to work on AML, either through zoom or in person. During these With only two people, we were able to be flexible on these work times which was beneficial during busier weeks. We used GitHub and GoogleColab for collaborating on implementations. Additionally, we used slack and text messages to share papers and other ideas.

1.2 Introductory Work

Our first work for the semester was submitting an independent study application which included a rough schedule for the semester. This schedule included three primary sections: an introduction, a more detailed look at attacks and defenses, and a final research project. Although we changed many details on the schedule to follow specific topics of interest, we kept our schedule aligned with these broad sections. This was helpful for preventing us from spending too much time in one area. After completing the independent study application, we started work for our introduction section.

Our goal for the the introduction section was to gain a broad understanding of adversarial machine learning and also get up to speed on tools that could help us implement AML techniques throughout the semester. To get started, we watched tutorial series on several machine learning libraries. These included Pytorch, TensorFlow, and Keras which we ended up using in almost all implementations throughout the semester. While implementing practice models with these libraries we were able to review machine learning concepts including loss functions, optimization functions, and model architectures. We also started using GoogleColab which is a Jupyter Notebook based environment. We had limited prior experience with Jupyter Notebooks so this was very helpful. All these concepts and tools proved be helpful go forward with the semester.

In order to gain a general understanding of AML we used several resources. In the first three weeks of the semester we read the first three chapters of an Adversarial Machine Learning Textbook [11]. These introduction chapters offered some valuable information for defining and classifying attack scenarios. They also gave several real world examples of AML. We also watched a few AML lectures that were available online.

Our deliverable work from this section was an AML Summary (link here). This offered us a good chance to reinforce what we learned about different categories of adversarial machine learning attacks. The following summary in section 2 is an update of this original summary.

2 Adversarial Machine Learning

2.1 What is Adversarial Machine Learning?

Adversarial Machine Learning is the study of attacks on machine learning systems and how to defend these attacks. More specifically, the field looks at attacks by changing machine learning model inputs (2.2.1). AML could be considered a subtopic of computer security or machine learning. It is a subtopic of computer security because it discusses defending an attack surface in a system that uses a machine learning model. AML becomes especially intertwined with security in cases where machine learning models provide security-related functionality. One popular example of this is a machine learning based Intrusion Detection System [11].

AML could also be considered a subtopic of machine learning for several reasons. AML is not just related to machine learning because it defends machine learning systems. It also utilizes machine learning techniques to create attacks and defenses. Furthermore, recent literature has suggested ways to use AML techniques to create better performing models [32] and also to increase model interpretability [5].

2.2 Attacking Machine Learning Systems

2.2.1 The AML Threat Model

An adversarial machine learning attack occurs when an adversary gives input to a model with the intent of causing unnatural behavior in the model. Specifically, the adversary aims to either 1) deceive the model at evaluation time or 2) cause the model to learn a desired pattern during training.

It is an important distinction that the adversary's control is limited to inputs. Any direct control of the the model itself or the output falls outside the scope of adversarial machine learning. Although this may seem limiting, research has shown an adversary can have a strong influence on the behavior of a model with well crafted inputs or manipulated data [9, 29, 13]. In fact, it is these findings that have prompted a proliferation of research into AML. Furthermore, this threat model is not arbitrary. It is common for a potential adversary to have access to model inputs. On the other hand, for an adversary to directly make changes to the model or model output, they must either have insider access or have hacked the system in which the model is deployed. While these are valid issues, they are beyond the scope of AML.

2.2.2 A Taxonomy of Attacks

In AML, there are several settings an attack can be deployed in. As with other domains of computer security, the abilities and limitations of the attacker vastly differ from situation to situation. Due to

this, there has been significant research into defining a a taxonomy of attacks [11]. The taxonomy we present here is adopted from the textbook by Joseph et al. [11].

The taxonomy has four different axes on which an attack can be classified on. These are: security violation, target range, model access, and data access. Each one of these axes is able to represent different characteristics of an attack and we dedicate a subsection to each. Although all of these can be valuable for classifying an attack, we found the most important distinction to be data access. Therefore we discuss this axis (Sections 2.2.2.4-2.2.2.6) in more length than the other three.

2.2.2.1 Security Violation

Different AML attacks result in different security violations. Three established security violations in computer security are privacy attacks, integrity attacks, or availability attacks. The line between availability attacks and integrity attacks is somewhat blurred in AML. Both attacks aim to induce misclassification by a model. In an integrity attack, the model fails by allowing malignant instances to pass through a model. An attack that goes undetected by an intrusion detection system would be an example of an integrity violation. An integrity violation can also be described as instances of false positives. An availability violation occurs when a model is preventing use of a normal use of a system to expected users. For example, an intrusion detection system that denies normal users access to a system would be an example of an availability violation. This violation can also be described as instances of false negatives.

In a privacy attack on a machine learning model, an adversary aims to glean private information from the training dataset. Scenarios where there may be an incentive includes models trained on medical records or financial information. The nature of these attacks are different from integrity or availability attacks because the attacker's goal is not to cause misclassification.

2.2.2.2 Target Range: Targeted vs. Indiscriminate

In a *targeted* attack an adversary is concerned with misclassifying a specific subset of input data. On the other hand, an adversary performs an *indiscriminate attack* if they are not concerned with which inputs the model misclassifies, but rather that the general model performance is degraded. This distinction is primarily used in the causative setting because a causative attack has an influence on *all* inputs at evaluation time.

2.2.2.3 Model Access: White-Box vs. Black-Box

A *white-box* scenario is when the adversary has access to all information about the model. This includes algorithm type and internal parameters. A *black-box* scenario occurs when an adversary does not have this information. In a *black-box* attack an adversary may attempt to reverse engineer the model. Given access to a sufficiently similar dataset a *black-box* is comparably effective to the *white-box* counterpart.

2.2.2.4 Data Access: Causative vs. Exploratory

The last axis, and the most important in our opinion, is data access. Given that an adversarial machine learning attack occurs from an adversary manipulating input data, we can further break attacks into two categories: *causative attacks* and *exploratory attacks*.[1]

2.2.2.5 Causative Attacks

Causative attacks can occur when the adversary has some control over the training data. By controlling certain parts of the training data the adversary can guide the machine learning algorithm to learn desired patterns. When given arbitrary control over the training data an adversary can easily get the model to learn a pattern based on only a few of the input features [31]. This pattern can then act as back door for the adversary once the model is deployed. Figure 1 gives an example of this effect. Another example would be if an adversary put sunset backgrounds in all stop sign

[1]We adopt these terms from Joseph et al. [11], which have been widely adopted by others in the AML research community. Note, however, that many synonyms can be found in the literature. Causative attacks are also often referred to as *data poisoning attacks*, *back door attacks*, and *training time attacks*. Exploratory attacks are often referred to as *test time attacks*, *evaluation time attacks*, and *adversarial examples*.

photos in a road-sign data set. In this theoretical example, a model trained on this dataset would be at risk of classifying *all* signs as stop-signs during sunsets.

In general, causative attacks are effective. The literature has shown theoretical limitations on model accuracy given the presence of a causative adversary [11]. Figure 1 is an example of the complete influence an adversary can have over a model with access to the entire training set. However, it is not even necessary for the adversary to manipulate the entire dataset to influence the model. In some cases poisoning less than 3% of the training data can result in a successful attack [12].

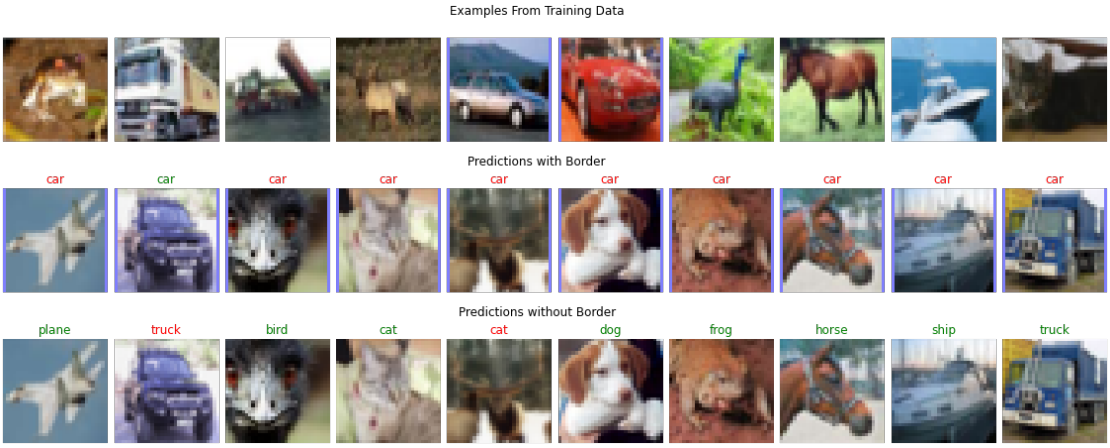

Figure 1: An example of the influence of a causative attack on the Cifar 10 dataset. In the training data we put vertical blue borders *only* on the cars. The first row show samples from the training set. The second row shows 10 images that were ran through the classifier. Even though most of these images look nothing like cars, they were classified as cars because they had the blue vertical borders. The last row shows the same images without the vertical borders. The same classifier classified most of these images correctly. The code for creating this figure and the related model can found in the code linked here.

2.2.2.6 Exploratory Attacks

In an exploratory attack, the attacker can only give inputs to the model once it has finished training. Unlike a causative attack, the attacker in an exploratory attack does not have the ability to modify training data. Without access to the training data the adversary has no ability to change what the model learns. Instead, the attacker aims to deceive the model. In this setting an attacker can still craft inputs that cause inaccuracies with high probability. Furthermore, the inputs may appear very similar to natural examples [9]. A gradient based perturbation on an image, which is imperceptible to the human eye, can cause the model to have a vastly different output (see Figure 2). There is significant discussion in the literature on why these inaccuracies occur on such small perturbations. A common theory is that their are many blind spots in the solution space of a model. To trick a classifier an adversary has to craft an input that falls into one of these blind spots. Furthermore, these blind spots are ubiquitous in the solution space and every input is close to some blind spot [7]. Furthermore, literature suggests that these blind spots are fundamental to machine learning algorithms and that there is a fundamental trade-off between robustness and classification accuracy [7, 11, 16].

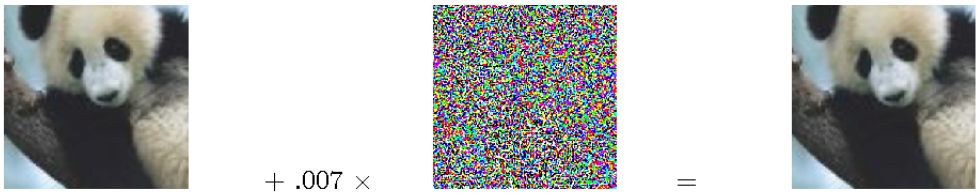

Figure 2: Adversarial example created with FGSM. A small gradient based perturbation results in drastically different model output. Figure adopted from Goodfellow et al. [9].

Depending on the setting, there are a couple ways an adversary can carry out an exploratory attack. The strongest and quickest exploratory attack can be created if the adversary has a copy of the model that they are attacking. This copy can often be obtained by reverse engineering the model [21]. With a copy of the model the adversary can employ model-based attacks. These attacks use information from the model to find optimal perturbations on the input. The seminal attack in this class is FGSM [9], which is a one step gradient-based attack. Other prominent model-based attacks include the Carlini-Wagner Attack [4], DeepFool [15], BIM [13], and BPDA [2].

Even without a copy of the model an adversary can still carry out an exploratory attack. By querying the model while making subtle changes to the input an adversary can eventually find a subtle change that causes misclassification. There are theoretical guarantees that such an adversarial example can be found in polynomial queries with respect to the input dimensionality [11].

2.3 Defenses Against Adversarial Attacks

With an understanding of how a machine learning model can be attacked we can now look at strategies that can be used to defend these attacks. Defense methods for causative and exploratory attacks are separate so we discuss them separately below.

2.3.1 Defenses Against Causative Attacks

In order to defend against causative attacks the defender must determine which sections of the training data have been poisoned. If the defender does not filter out poisoned data, the model can either degrade in performance or learn a pattern desired by the attacker. The latter can act as a backdoor after training for the attacker. There are two possible approaches to detecting poisoned data. The first method is statistical examination of the dataset. Poisoned data is more likely to have outliers from the dataset. Traditional outlier detection can be applied to this problem [23]. A second method is to examine how the model reacted after training on some example. The *Reject on Negative Impact* defense rejects an example if the model update surpasses a threshold for negative effect on model performance [17]. An abnormally high loss score can also be used to detect poisoned data [33]. A more thorough defense can be created by combining outlier removal and model based detection [33]

2.3.2 Defenses Against Exploratory Attacks

Contrary to causative attacks, exploratory attacks assume the attacker does not have access to the training data, and thus must adopt different techniques to attack the model. These changes are also reflected in the methods employed to defend against these attacks. There are two main tracks that researchers have explored to defend against exploratory attacks. The first is simply making the model more robust to adversarial input, while the second one involves detecting adversarial examples and either reforming those examples closer to the training distribution, or discarding those inputs [1]. These approaches are discussed in more detail below.

2.3.2.1 Robustness Defenses

Robustness defenses encompass a a wide range of methods that have proven themselves to be useful against exploratory attacks. These styles of defenses generally involve some changes done to the target classifier model before the testing phase. One such example of this is adversarial training [30]. This defense expands the distribution that the model can perform well on by creating adversarial examples to train the model on. A variety of techniques have been tested with in adversarial training that have shown promising increases in robustness. This is an especially effective method when the adversarial examples are similar to those previously seen by the model. However, adversarial training shows shortcomings when different attack techniques are used. Another type of robustness defense is defensive distillation [22]. Defensive distillation uses one target classifier to generate probability vectors for the training set, and then trains a different "distilled" model using the probability vector as each image's label. The theory behind this is that the second model can gain much more information from the probability vectors and would thus be able to generalize more effectively to data outside of the training distribution. In practice, this technique has been shown to be highly effective and can reduce adversarial image misclassification from 95% on MNIST to less than 0.45% [22]. On the other end of exploratory defenses are those that deal with catching and fixing adversarial inputs once the model is already trained.

2.3.2.2 Detection Defenses

Detection defenses deal with identifying adversarial inputs and either discarding those inputs or reforming them so they can be classified correctly. There are a variety of techniques that have been proposed. One of the techniques we focused heavily on was a defensive architecture called MagNet [14]. MagNet utilized an autoencoder that learned the distribution of the training data to detect adversarial inputs. Inputs that were far away from the training manifold (measures via reconstruction error) were discarded, whilst all others went through the autoencoder in an attempt to reform them towards the training distribution. The reformation would then act as a secondary technique that would improve classification on adversarial inputs that avoided detection. Another example of a detection defense is DefenseGAN [24]. DefenseGAN functions similarly to MagNet in that it trains a Generative Adversarial Network (GAN) on the training distribution to create a representation of the training data. When it is time to predict a class, the input is fed into the GAN, which produces an input similar to the original, but without noise or adversarial perturbations. This generated input is then fed into the original classifier. Techniques like MagNet and DefenseGAN come across as highly effective due to their "attack-agnosticism"; they do not rely on being trained on any specific type of attack, which makes them highly desirable for use where an attacker could theoretically use one or multiple methods of generating adversarial perturbations.

3 Examination of Attacks and Defenses

After completing the introduction section and turning in our original AML summary, we started on a section to examine proposed AML attacks and defenses. Our goal for this section was to gain a more practical understanding of AML attacks and defenses. We started off with the intention to study both attacks and defenses in both the causative and exploratory setting. However, we ended up focusing primarily in the exploratory setting. Our approach for learning was to implement the prominent attacks and defenses.

Throughout this section we almost entirely focused on image classification. There are several reasons for this. Getting started there were the most resources available for creating models and attacks on image classification datasets. We originally intended to move our focus to other domains including intrusion detection. However, as we got further into the semester image classification remained appealing. One reason for this is that images are high dimensional inputs. High dimensionality means an easier attack surface and therefore the influence of attacks become more pronounced [11]. Also, using images were fun to work with because we got see a visualize how much(or how little) an attack changed an input. Although it feels like getting to look at images is trivial, I actually feel like the easy visualization played a role in helping us gain intuition. Furthermore, all image inputs are continuous. Other domains such as natural language processing or intrusion detection system have discrete features which adds complexity to creating adversarial attacks. By limiting

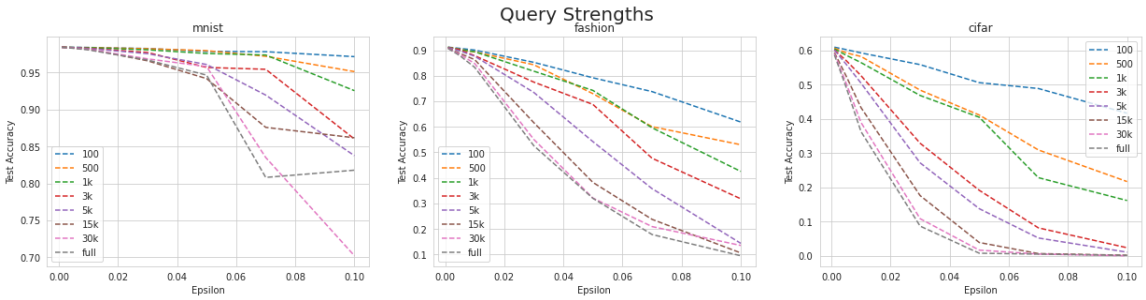

Figure 3: Classifier accuracy against black-box attacks with different query strengths. Tested on the MNIST, Fashion MNIST, and CIFAR 10 datasets.

ourselves to image classification we bypassed the chance to get familiar to problems associated with adversaries in other common domains. However, because of this we were able to direct our focus on implementing attacks and defenses. We think much of the understanding we gained for AML with image classification could carry over to other domains in future work.

For the first week of this section (week 4 in the semester) we focused on attacks. At the end of week 4 we completed a notebook overviewing several attacks from the causative setting and one attack from the exploratory setting (notebook link). The causative attacks in this notebook were similar to the attack shown in figure 1. We added a certain feature to one class in the training data. Then, we applied this feature to all classes in the evaluation dataset. We saw up to 99% of the evaluation data classified as the chosen class. The features we added to the data were squares of different colors and sizes. This led us to believe that a model can learn to put importance on a small number of features. Furthermore, we even saw this effect when the pixel colors within the square were random.

For the exploratory attack we implemented FGSM. We tested the attack under different perturbation strengths. As the strength of the attack increased we saw greater rates of misclassification. All the experimentation in this notebook was done using the Cifar 10 dataset.

There were several important takeaways for us from creating the first notebook. The methods we used to implement attacks were used several times down throughout the semester. Also, we gained some intuition on how machine learning models learned. We found that models can learn to put a lot of weight on a small percentage of the features. This intuition meshes well with the effect of adversarial examples. A good causative attack creates low hanging fruit for the model to learn.

For weeks 5-7 we focused on exploratory attacks and defenses. For a deliverable we completed a GoogleColab notebook which is linked here. In this second notebook we implemented FGSM and rand+FGSM attacks [30]. We tested these attacks on classifiers for each of three common image datasets. Additionally, we tested the strength of the attacks when constrained by limited queries. Our results from this can be seen in Figure 3.

The defenses we implemented for this notebook were adversarial training [30], MagNet [14], and Defense-GAN [24]. Overall, the implementations of these defenses were challenging and we spent the better part of two weeks working on them. Adversarial training was the easiest to implement. Adversarial training is achieved by adding adversarial examples into the training set. Each iteration of training new adversarial examples examples are created and added to the dataset. The intuition behind this is that you can train a model on adversarial examples to cover up blind spots. Each iteration of adversarial examples exposes new blind spots for the model to learn. In our simple We tested between 1 and 20 iteration of adversarial training. We saw significant increased performance against the FGSM attack after 10-20 adversarial training iterations.

The second defense we implemented was MagNet. MagNet is to be considered a prepossessing defense because the model is not affected by the defense. The defense works by prepending an autoencoder to the original model. The autoencoder works by compressing an input into a lower dimensional latent space. Subsequently, the autoencoder reconstructs the image from the latent space. The motivation for the autoencoder is that it is able to learn a representation of the data. When the autoencoder takes in examples that are in the distribution of the training data it will reconstruct the image with little reconstruction error. However, when an input is outside of the

training data distribution (e.g adversarial examples) the autoencoder reconstructs the image with more error. MagNet classifies an input as adversarial if the reconstruction error is above a threshold. This defense fits in with a class of defense papers that argue that adversarial examples fall off the manifold of the training data. Under this explanation, an input can be determined to be adversarial if it does not fall on the manifold. The authors of MagNet suggest the autoencoder is able to learn to represent the manifold and therefore has the ability to detect adversarial examples.

Our implementation of MagNet used a five layer convolutional network that had a latent space 4x smaller that the input. This defense showed a significant robustness improvement against the vanilla FGSM attack (see Figure 4).

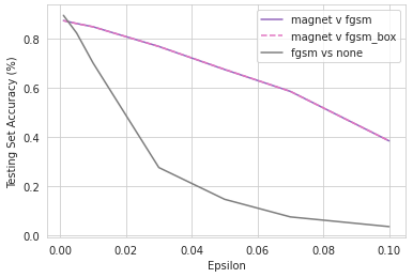

Figure 4: MagNet vs FGSM.

The last defense we implemented for this section was Defense-GAN. This defense works by using a Generative Adversarial Network (GAN) [8] to recreate images without adversarial perturbations. The motivation for this is similar to MagNet. The defense aims to learn a representation of the data and leverage the representation to find and reform adversarial inputs. The authors present state-of-the-art results from the technique citing the the unequaled representation power of GANs.

The first step in implementing Defense-GAN was training a GAN by itself. We had difficulty with this step. In retrospect, we think our GAN implementation suffered from mode collapse as described in Srivastava et al. [27]. With our marginal GAN in hand, we were able to integrate it into a Defense-GAN implementation. Guided by figures from the original paper (shown in Figure 5 and Figure 6) we created a running implementation. We saw no improvement in robustness from our implementation. We think this was primarily due to how bad the original GAN was. After turning in our exploratory attacks we defenses notebook in week 7, we had time for one topic before we wanted to move on to the next section. We were originally planning on doing a similar 2 to 3 week section on causative attacks. However, we ended up deciding to continue to focus on the exploratory setting. The decision was based primarily on that we were interested and that we would have to focus less on model training.

For the last topic of this section we looked at exploratory attacks based on queries to find an adversarial example instead of first reverse engineering a model. We read Chapter 8 of the AML textbook which focuses on this topic. The chapter provided several algorithms for searching for an adversarial input without having a copy of the model. They also discussed why this was a realistic threat model and proved some theoretical bounds on the ability of an attacker in this setting. We alloted one week for this topic before completing this section.

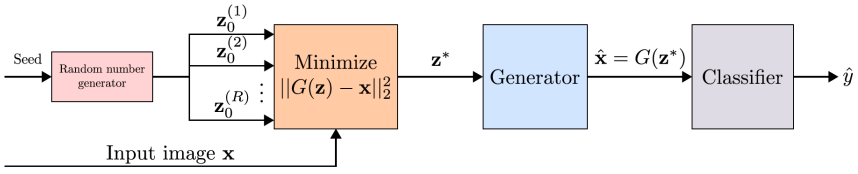

Figure 5: A high level diagram Defense-GAN. An input vector z is optimized to minimize reconstruction of the input by the pretrained GAN. The output from the generator is fed to the original classifier. Figure adopted from Samangouei et al. [24]

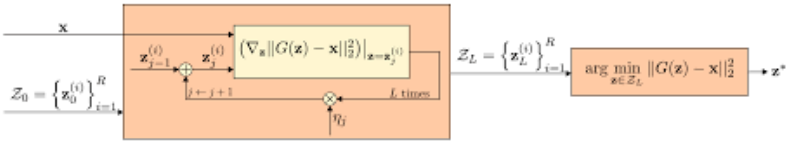

Figure 6: A more detailed look at Defense-GAN's optimization mechanism. Vector z is optimized by many gradient descent steps that aim to reduce reconstruction error. Figure adopted from Samangouei et al. [24]

4 Other Candidate Research Problem

At week 9 we decided to move on to the section where we would look into finding a research topic to pursue. Specifically, we wanted to work on a defense strategy. We spent a good portion of this section trying to brainstorm a defense to work on. We eventually settled on *Using Feature Attribution for Adversarial Defense* as a research topic. We spent the last weeks of the semester doing researching more background on this topic and starting to implement experiments. We dedicate Section 5 to our future work for this topic. In the rest of this subsection we discuss the other ideas we looked at in the weeks before we decided on a feature attribution as a defense.

4.1 Adversarial Transformation Model for Adversarial Training

Training a machine learning model to create adversarial input for adversarial training was the first idea we seriously considered pursuing. We spent a week testing the idea. When we further looked through the literature we found several papers from 2017 that looked thoroughly at this idea [3, 18]. We were unable to think of a meaningful iteration/improvement from these works and decided to keep looking at other options for defenses. Still, it was interesting learning about this strategy and we thought it would be worth reviewing.

The idea was that you could train one model to create adversarial examples and then simultaneously train the main model on these examples. It would work as a sort of dynamic adversarial training. The intuition was that the Adversarial Transformation Model would learn to make examples in the blind spot of the model. Meanwhile, the real model would be trained on these examples and effectively learn to cover up these blind spots. In the next iteration, the Adversarial Model would find new blind spots and so on. Furthermore, the universal approximation property of the transformation network could learn a to exploit a broader distribution of blind spots than a standard attack. After many iterations, the hypothesized result would be a more robust classifier.

One of the interesting things we learned from trying this was that a neural network can handle two loss functions. The whole idea was contingent on the idea that we could change train a network to make transformations that caused misclassification without significant changes to the image. We attempted to do this by training the adversarial transformation model on minimizing both reverse cross entropy (to cause misclassification) and mean squared error (to keep images similar). Even, though these cost functions have seemingly opposite goals, a network was still able to optimize both.

4.2 Looking at Current Defenses in Different Security Contexts

We brainstormed ideas based on ideas suggested in a paper titled *Motivating the Rules of The Game for Adversarial Examples* [6]. The paper suggests that a lot of the literature in adversarial machine learning claims to be directly motivated by security concerns, but in reality does a poor job addressing the breadth of practical concerns. Specifically, lots of work in the literature has addressed attacks limited to minuscule perturbations, when in reality lots of settings do not have this limitation. The paper discusses an image classification attack scenario where the attackers are only limited by *content preservation*. The paper cited above explicitly defines this:

> Content-preserving perturbation: The attacker does not get to choose the starting point, but is handed a draw from the data distribution. However, the attacker may make any

perturbation to the example they want, as long as the content is preserved. That is, if it is a picture of a particular person, it must clearly still be that person.

As a research idea we considered choosing a defense that was proposed to defend imperceptible perturbations and then adapting it to defend the class of content-preserving attacks. Although we did not pursue this topic, we have kept in mind the the idea of defending a spectrum of attack classes.

4.3 Deciding on Using Feature Attribution as a Defense

We ran into this idea from the papers [34, 10]. Both pf these papers look at using feature attribution for defenses. However, these works were recent and there seemed to be lots of variations that had not been looked into. We brainstormed a couple different ways feature attribution could be used that we though would be worth at least trying out. We were also interested in this topic because we wanted to look more at general model interpretation and feature attribution methods. We also feel like this category of interpretation based defense may not have the extent of theoretical limitations that has been shown for other defense strategies like adversarial training.

5 Future Research Problem: Feature Attribution for Defending Adversarial Against Examples

5.1 Background

Feature attribution assigns an attribution value to each feature in an example after it has been ran through a classifier. The attribution assigned to each feature represents how important that feature was to the classifier's decision. A high attribution value means that the feature was important in the classifier's decision. A low attribution value means the feature was unimportant to the classifier's decision. Feature attribution has been researched under the topic of model interpretability and is an open research problem in its own right. However, there are now several prominent feature attribution methods. A few of these are *Integrated Gradients* [28], *Saliency Maps* [26], and *DeepLIFT* [25].

Recent work has proposed using feature attribution as a means to defending adversarial examples [34, 10]. *ML-LOO* is an algorithm proposed by Yang et al. [34] that uses the sparseness of a feature attribution map to determine whether an image input is adversarial. They find that clean images have a sparse feature attribution map (i.e a few features have high attribution where as most features have near zero attributions). Inversely, adversarial images have a less-sparse feature attribution map. Figure 7 shows their findings for attributions of a clean image vs an adversarial image.

Jha et al. [10] propose another method to find adversarial examples. They suggest masking the top 10% of features based on their attribution values and then reevaluating the decision of the classifier. They find that a classifier will change its classification on an adversarial image after the mask has been applied. On the other hand, a clean image is highly likely to receive the same classification even after the top 10% of features have been masked. Therefore an image can be determined to be adversarial based on whether the classification changes after masking.

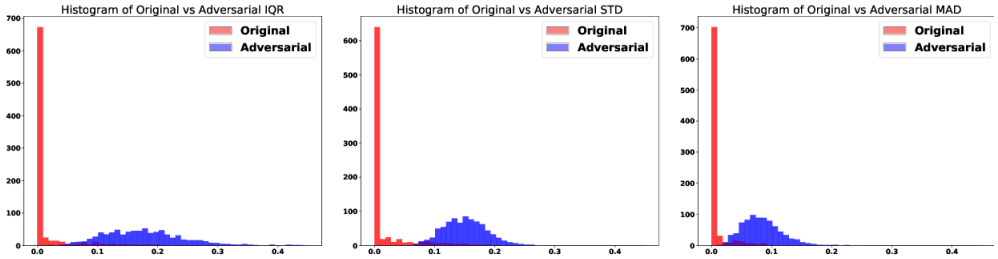

Figure 7: A small percentage of the features are very important in a clean image. A higher percentage of the features are medially important in an adversarial image. Figure adopted from Yang et al [34].

5.2 Future Work

In future work we want to continue to explore feature attribution as a method for defending adversarial examples. The first idea we want to implement is testing the algorithms discussed in Section 5.1 on an array of feature attribution methods. The authors of both papers suggest that the algorithms should perform similarly regardless of the feature attribution method used. We want to see if we find that different attribution methods work better with either of the algorithms. This will also give us background using feature attribution libraries.

The next step would be to work on a new algorithm using feature attribution to detect adversarial inputs. One idea we have includes comparing feature attributions at each layer. This concept is outlined in rough pseudo-code in Algorithm 1. Our cursory hypothesis is that the attribution maps from a clean images would change consistently between subsequent layers where as attribution maps from an adversarial image the would be random. If this does not work we could look in to a variation off of Jha et al. or Yang et al.

Algorithm 1: Layer-wise Attribution Analysis

Result: return decision
$L \leftarrow$ layers in network;
$Feature\_Attributions \leftarrow []$;
for $l \in L$ **do**
  $Feature\_map \leftarrow feature\_attribution(l)$;
  $Feature\_Attributions.append(Feature\_Map)$;
end
$gradients \leftarrow \nabla Feature\_Attributions$;
if *gradients are similar* **then**
  $decision \leftarrow clean$;
else
  $decision \leftarrow adversarial$;
end

6 Conclusion

We would like to end our independent study review paper by sharing concluding thoughts. In Section 6.1 we give our final remarks on the topic of AML. Finally, in Section 6.2 we reflect on what we learned through out the semester.

6.1 AML Summary

Adversarial Machine Learning research addresses how to make machine learning models more robust against adversaries. The literature shows that small manipulations to data at both training and evaluation time can have drastic effects on model behavior. These findings have prompted increasing study into why this behavior exists and how machine learning models can be defended. With increasing adoption of ML for real-world scenarios, there is more demand for ML interpretability and robustness. AML research attempts to makes progress on the these problems. There has been significant increases in robustness is the last 5-10 years. However, almost all defenses can be broken by some alteration of an attack. Furthermore, some of these defenses have provable limitations. This means defending adversarial examples is still largely an open problem.

6.2 Reflection

Looking back at the semester, both authors think the independent study was successful. We were able to learn about a topic of ongoing research that we are both interested in. Furthermore, we got to build skills related to both machine learning and computer security. For machine learning we got practice with different frameworks like Tensorflow and PyTorch and learned new concepts like Autoencoders and GANs. Related to computer security, we got to examine the importance of threat modeling when defending an attack scenario.

Still, there were things that could have been improved throughout the semester. In the introduction we fell short of the goal of gaining a broad understanding of AML. This is in part due to there not being a lot of centralized information. However, we could have done a more enough job researching the breadth of the topic and using more resources. The introduction chapters in the textbook were valuable for understanding different attack settings, but since this was the only resource we used to start off with we were left with gaps of understanding in approaches to defenses. Furthermore, we had no idea on what the state-of-the-start performances were for attacks or defenses were. We think if we had spent more time scanning literature in the beginning we would have saved time later on in the semester.

Throughout the semester we set deadlines for ourselves for our main deliverables, however we were inconsistent with setting smaller week to week goals. Similarly, the work times with both authors was not always consistent. This was nice for alleviating stress during busier weeks for each of us throughout the semester. However, in general having a more structured week to week schedule for would have likely resulted in a faster and more consistent pace.

In the future, both authors aim to to apply what we learned from this semester. This includes the technical tools and concepts, but also the takeaways from a self-directed study. These skills and takeaways will hopefully be used to benefit future work.

Acknowledgements

We thank Travis Peters for suggesting AML as an independent study topic, advising the study, and for his consistent feedback throughout the semester. We also thank other members of the MSU Security & Privacy Research Group, Madison Tandberg and Reese Pearsall, for helpful tips and interesting security discussions throughout the semester. We are indebted to the open source community that has developed AML libraries used throughout this work, including the Adversarial-Robustness-Toolbox [19] and CleverHans [20].

References

[1] Naveed Akhtar and Ajmal Mian. Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey, 2 2018.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. *35th International Conference on Machine Learning, ICML 2018*, 1:436–448, 2 2018.

[3] Shumeet Baluja and Ian Fischer. Adversarial Transformation Networks: Learning to Generate Adversarial Examples. *arXiv*, 3 2017.

[4] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. Technical report.

[5] Prasad Chalasani, Jiefeng Chen, Amrita Roy Chowdhury, Somesh Jha, and Xi Wu. Concise Explanations of Neural Networks using Adversarial Training. Technical report, 11 2020.

[6] Justin Gilmer, Ryan P. Adams, Ian Goodfellow, David Andersen, and George E. Dahl. Motivating the Rules of the Game for Adversarial Example Research. *arXiv*, 7 2018.

[7] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial Spheres. *6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings*, 1 2018.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 10 2020.

[9] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015 -*

Conference Track Proceedings. International Conference on Learning Representations, ICLR, 12 2015.

[10] Susmit Jha, Sunny Raj, Steven Lawrence Fernandes, Sumit Kumar Jha, Somesh Jha, Gunjan Verma, Brian Jalaian, and Ananthram Swami. Attribution-driven Causal Analysis for Detection of Adversarial Examples. *arXiv*, 3 2019.

[11] Anthony D. Joseph, Blaine Nelson, Blaine Nelson, and J. D. Tygar. *Adversarial machine learning*. Cambridge University Press, 1 2019.

[12] Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger Data Poisoning Attacks Break Data Sanitization Defenses. *arXiv*, 11 2018.

[13] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings*. International Conference on Learning Representations, ICLR, 7 2019.

[14] Dongyu Meng and Hao Chen. MagNet: A Two-Pronged defense against adversarial examples. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 135–147, New York, NY, USA, 10 2017. Association for Computing Machinery.

[15] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: a simple and accurate method to fool deep neural networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:2574–2582, 11 2015.

[16] Preetum Nakkiran. Adversarial Robustness May Be at Odds With Simplicity. Technical report, 2019.

[17] Blaine Nelson, Marco Barreno, Fuching Jack, Chi Anthony, D Joseph, Benjamin I P Rubinstein, Udam Saini, Charles Sutton, J D Tygar, and Kai Xia. Exploiting Machine Learning to Subvert Your Spam Filter. Technical report.

[18] Linh Nguyen, Sky Wang, and Arunesh Sinha. A learning and masking approach to secure learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11199 LNCS, pages 453–464. Springer Verlag, 10 2018.

[19] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian M. Molloy, and Ben Edwards. Adversarial Robustness Toolbox v1.0.0. *arXiv*, 7 2018.

[20] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel. Technical Report on the CleverHans v2.1.0 Adversarial Examples Library. 10 2016.

[21] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. 5 2016.

[22] Nicolas Papernot, Patrick Mcdaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. Technical report.

[23] Andrea Paudice, Luis Muñoz-González, András György, and Emil C. Lupu. Detection of adversarial training examples in poisoning attacks through anomaly detection, 2 2018.

[24] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models, 5 2018.

[25] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. Technical report, 7 2017.

[26] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings*, 12 2013.

[27] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning. Technical report, 2017.

[28] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. Technical report, 7 2017.

[29] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, 12 2014.

[30] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble Adversarial Training: Attacks and Defenses. *arXiv*, 5 2017.

[31] Loc Truong, Chace Jones, Brian Hutchinson, Andrew August, Brenda Praggastis, Robert Jasper, Nicole Nichols, and Aaron Tuor. Systematic Evaluation of Backdoor Data Poisoning Attacks on Image Classifiers. Technical report, 2020.

[32] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan Yuille, and Quoc V Le. Adversarial Examples Improve Image Recognition. Technical report, 2020.

[33] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative Poisoning Attack Method Against Neural Networks. Technical report.

[34] Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I. Jordan. ML-LOO: Detecting Adversarial Examples with Feature Attribution. *arXiv*, 6 2019.