Temasek Polytechnic

School of Informatics & IT

Diploma in Applied Artificial Intelligence

AICS

AY2022/2023 April Semester

Ensemble Modelling of Hybrid CNN, Autoencoders, and DCT with AAEs for Steganalysis

C21U03

Maximilian See Tze Jie

2102869A

**Table of Contents**

[I. Introduction 3](#_Toc136222799)

[II. Hardware & Dataset 3](#_Toc136222800)

[III. Model Ideology & Tactics 4](#_Toc136222801)

[IV. Model Architecture 5](#_Toc136222802)

[V. Model Fine-Tuning 5](#_Toc136222803)

[VI. Model Evaluation & Analysis Part 1 6](#_Toc136222804)

[VII. Model Evaluation & Analysis Part 2 7](#_Toc136222805)

[VIII. Attack Strategy Plan 8](#_Toc136222806)

[Model Weaknesses 8](#_Toc136222807)

[Possible Attacks Scenarios 9](#_Toc136222808)

[Attacks Scenarios 1: White Box 10](#_Toc136222809)

[Attacks Scenario 2: Grey/Black Box 11](#_Toc136222810)

[IX. MITRE ATLAS Framework 14](#_Toc136222811)

[Attack Scenario 1 14](#_Toc136222812)

[Attack Scenario 2 15](#_Toc136222813)

[X. Mitigations 18](#_Toc136222814)

[XI. Conclusion 21](#_Toc136222815)

[References 22](#_Toc136222816)

Abstract

In this report, we introduce an innovative strategy that integrates a modified CNN and autoencoder model for steganalysis & a Hybrid Adversarial Auto encoder for ensembling. Our approach comprises three essential stages: utilizing EfficientNetB2 as the underlying CNN model, extracting DCT features from the images, and constructing a dedicated steganalysis model. Alongside the hybrid model, we explore the utilization of unsupervised learning deep learning models, such as Adversarial Auto Encoders (AAEs), to facilitate model ensembling techniques (Hybrid CNN + AAEs) that enhance adversarial robustness against adversarial examples.

# Introduction

Steganography is the art of hiding information within other data, such as an image, audio, or video file. It plays a crucial role maintaining privacy and confidentiality in communication, especially in the digital age where data can easily be intercepted and analysed. By hiding sensitive information within innocent-looking data, steganography allows people to communicate covertly without arousing suspicion. For companies this might mean a leakage of confidential data like technology blueprints etc. For government’s stegnography in images might mean espionage and for day-to-day users such stegnograph images might hide executables which can introduce malware into our systems.

**DARKCOMET**

The "DarkComet" malware campaign, which involved steganographically hiding malicious payloads within image files to evade detection, serves as a notable case study.

To ensure a comprehensive analysis, we employ two CNN models that examine different spatial dimensions: RGB and YCCBBR colour spaces. The EfficientNetB2 model, pretrained on ImageNet, operates on the RGB colour space, while the SRNET model focuses on the YCCBBR colour space. Additionally, the DCT features are extracted specifically from the YCCBBR colour space for a different perspective.

Despite the use of smaller image sizes (256x256), the proposed model manages to achieve comparable or superior performance compared to the standard CNN model, EfficientNetB2, which operates on larger images (512x512). This suggests that our approach maintains the model's effectiveness even with reduced image dimensions.

In addition to the CNN model, we incorporate the use of unsupervised learning AAEs to detect stegnograph images. The parameters for this would be images of size (512 X 512) and 1000 image from each class.

# Hardware & Dataset

The dataset used in this report is the "ALASKA" dataset, which contains digital images that may contain hidden messages. The dataset was created for a competition aimed at developing efficient and reliable methods to detect secret data hidden within digital images. The dataset includes images acquired from up to 50 different cameras, from smartphones to high-end cameras, and processed in various ways. The goal of the competition is to create robust detection algorithms with minimal false positives, which can help law enforcement officers combat criminals using hidden messages. The competition is organized by IEEE WIFS in collaboration with several universities and research labs. Successful entries will contribute to more accurate steganalysis and help catch criminals whose communications are hidden in plain sight.

This dataset includes unaltered images (Cover) and images in which information has been hidden using one of three steganography algorithms (JMiPOD, JUNIWARD, UERD). The objective of the competition is to identify which images in the Test set have hidden messages. The payload length is adjusted so that the difficulty is comparable for all images regardless of their content, and each algorithm is used with the same likelihood. The images are compressed using one of three JPEG quality factors (95, 90, or 75), and the average message length is 0.4 bit per non-zero DCT coefficient. The Cover folder contains 75k unaltered images for training, and the JMiPOD, JUNIWARD, and UERD folders each contain 75k images with hidden messages. The Test folder contains 5k images for prediction, and the sample\_submission.csv file contains an example submission in the correct format. The dataset is hand crafted by experts applying the algorithms to 150000 images, and thus is more immune to data poisoning or a misclassification of data in the training set.

The hardware that we would be using is a 16 GB P100 & a 2 core Intel Xeon CPU(R). Due to the limited hardware available, we would only be using a 0.75% of the total number of images in each class.

# Model Ideology & Tactics

1. What is Stenography & How it Works.

Steganography is the art and science of hiding information within another file, such as an image or audio file, in a way that does not affect the original file's functionality. The goal is to make the hidden message difficult to detect and extract by an attacker who might intercept the file.

In terms of images, steganography works by slightly modifying the pixel values of the original image to embed the secret message. For example, consider an image with a blue-sky background. The steganography algorithm might alter the least significant bits of the blue pixel values to encode the secret message. The modifications might not be noticeable to the human eye, but they can be detected and extracted by the appropriate software.

Suppose we want to hide the binary message "110110" in the image. We can do this by modifying the least significant bit of each red component of the image pixels. For each pixel, we check the binary representation of the red component and modify the least significant bit to match the next bit of the secret message. This can be done using bitwise operations.

For example, suppose we have the pixel (240, 128, 65) in the image. The binary representation of 240 is 11110000, and we want to hide the first bit of the secret message "110110", which is "1". We can modify the least significant bit of the red component to get 11110001, which represents the decimal value 241. Similarly, we can modify the least significant bit of the red components of other pixels to hide the remaining bits of the secret message.

1. DCT Discrete Cosine Transformation

DCT (Discrete Cosine Transform) is a transformation technique used in steganalysis, the detection of hidden information within images. It helps in steganalysis by analysing the frequency components, energy distribution, and statistical properties of images. DCT transforms an image into frequency components, and deviations from expected energy distribution and statistical properties can indicate the presence of hidden data.

1. Objective

The primary goal of integrating feature extractors and CNN models is to identify steganography in images effectively through multiple perspectives without the need for an enormous amount of data The feature extractors will be responsible for extracting the DCT features from the images. These extracted features will then be fed into an Auto Encoder model, which will learn to extract informative features from the images. Similarly, the CNN models will act as feature extractors to capture significant features from the images.

By incorporating both colour spaces and frequency domains, we aim to create a robust and accurate model for steganography detection. Exploring multiple colour spaces allows us to uncover hidden information that may be more apparent in specific colour representations. Additionally, analysing the frequency domain through DCT features enables a comprehensive examination of the image content. This approach ensures a more reliable output and enhances the overall performance of the model in identifying steganography in images.

In addition to CNN, Adversarial Auto Encoders will be used to detect anomalous images. Together both models would be put to the test and evaluated. The chosen model would serve as the main model and later both models would be used together for model ensembling.

The primary goal of the mode is to reduce the number of data needed for training. It is of note that with more complex models, the convergence and inference speed would be affected. In this report we will go for accuracy instead of optimization. This is since with our small dataset our accuracy is low, thus prioritizing accuracy over optimization would be the goal. If given more hardware resources to work with then optimization would be of greater or equal importance.

# Model Architecture

The architecture of this model consists of EfficientNetB2 combined with an auto encoder and SRNET. The features are then concatenated and fed into a dense layer to make the final predictions, as seen in the diagram. Provided in the zip file. The model predictions would then later be used together with AAE as ensemblin tactics to increase robustness.

The model architecture is inspired by GUANSHUO XU: <https://www.kaggle.com/competitions/alaska2-image-steganalysis/discussion/168548>

1. Auto Encoder

Auto Encoders are unsupervised Neural network that efficiently compress and encode data and reconstructs the data back from the encoded version to a representation as closed to that of the original. By design it reduces the dimensionality of data by learning how to ignore the noise in the data. The idea is to use auto encoders to extract feature representations and compress them into a smaller dimension. This is in hopes of speeding up inference process and reducing the “Curse of Dimensionality”. The input to the auto encoder is DCT (Discrete Cosine Features) in the Frequency domain. The frequency domain or specifically DCT are how JPEG images are made from or compressed.

1. EfficientNetB2 & SRNET

The proposed approach involves utilizing EfficientNetB2 and SRNET to extract features from images. These extracted features will serve as inputs to a multi-layer perceptron, which acts as our second-level model. While the consensus leans towards using EfficientNetB2, a widely recognized CNN architecture developed by Google, certain studies have also demonstrated the effectiveness of SRNET, a model specifically designed for steganography detection.

SRNET adopts a unique strategy by preserving the original input size of the images for as long as possible. This approach aims to maximize the learning potential from the images without down sampling. It is important to note that SRNET operates on images in the YCCBR colour format. This colour space selection facilitates the detection of hidden information, as certain steganographic techniques are more easily identifiable in specific colour spaces.

1. Multi-Layered Perceptron

After both feature extraction from the two CNN model, and the features extracted from the autoencoder from the extracted features will be concatenated and used as inputs to a MLP to learn and finally decide which images are tampered and which are not.

1. AAEs

Adversarial Autoencoders (AAEs) are a type of neural network architecture used in steganalysis, which combines the principles of autoencoders and adversarial training. AAEs consist of an encoder, a decoder, and a discriminator network. The encoder maps the input data into a compressed latent space, the decoder reconstructs the original input from the latent space, and the discriminator tries to distinguish between the reconstructed samples and the real ones.

# Model Fine-Tuning

Due to limited RAM and time, the Gaussian Naive Bayes tuning will be performed using 256 X 256 images. The results from this process will then be used to set the parameters for our model.

1. Call-backs & Label Smoothing, & Early Stopping

For model fine tuning we consider the approach of using the Early Stopping to prevent overfitting and reduce on plateau to lower the learning rate upon validation loss plateauing. We also include the addition of save model checkpoint for best weights. Additionally, Label Smoothing will be used. Label smoothing is a regularization technique used in deep learning, to prevent models from becoming overconfident or overfitting to the training data by altering their target values into probability values. Think of it as giving the incorrect labels the “Benefit of The Doubt” rather than being completely certain that the correct label is the only possible choice. We will set a smoothing value of 0.05.

1. Unfreezing the Layers

For fine-tuning the model, we unfreeze the layers in a similar manner to Phase 1. The reason for this is because the Imagenet dataset does not accurately reflect the data we are currently working with. Our decision is because although the Imagenet dataset has over 1000 classes, it was not designed to handle stenograph data. The dataset primarily consists of everyday objects or animals, which do not reflect the hidden nature of our target images. Thus, the unfreezing of all layers

1. Gaussian Naive Bayes Tuning

Bayesian optimization is a method for finding the best set of parameters for a machine learning model. It works by creating a probability model of how the model's performance is affected by changes in its parameters. For example, suppose we have a machine learning model that we want to train on a dataset, and it has several parameters that can be adjusted, such as the learning rate or the number of hidden layers. Bayesian optimization can help us find the best combination of parameter values that will give us the highest accuracy on our test set. The method works by iteratively selecting parameter values to test based on their probability of leading to an improvement in the model's performance. As more iterations are performed, the probability model is updated to incorporate the results of previous tests, allowing the optimization process to focus on the most promising parameter values. Overall, Bayesian optimization can help us save time and resources by efficiently searching through a large space of parameter values to find the best combination for our model.

# Model Evaluation & Analysis Part 1

**Metrics & Optimizer**

The metric as used in the evaluation of the model is a custom metric made by Max Jeblik for the ALASKA competition: <https://www.kaggle.com/competitions/alaska2-image-steganalysis/discussion/147182>

The selection of the AdamW optimizer provides several advantages compared to SGD or regular Adam optimizers. AdamW incorporates adaptive learning rate techniques, which adjust the learning rate during training to optimize convergence. Additionally, it integrates momentum, which aids in faster convergence by accumulating gradient information from previous steps. Moreover, AdamW incorporates weight decay, a form of regularization that helps prevent overfitting by adding a penalty term to the loss function. One notable advantage of AdamW is its robustness to different learning rate choices, making it less sensitive to manual tuning.

**EfficientNetB2 + Auto Encoder + SRNET + MLP**

**A picture containing text, screenshot, plot, line

Description automatically generated** **A picture containing text, screenshot, plot, diagram

Description automatically generated**

Figure 1. The graph above shows the Validation Accuracy of the model.

As we lack the resources to train on more images, we can only try to optimise the model to work better with lesser data.

The model utilizes the strength of transfer learning and specialised Deep learning models with the addition of DCT features extracted from images. The model outperforms models with higher resolution as well as images when compared to standard transfer learning models at an average of ~62% accuracy when trained on 1000 images of each class “Tampered” & ‘Untampered’. This means that as the model scales in terms of images and resolution we can garner higher accuracy from the increased number of images.

However, although the graph above shows considerable accuracy with lesser data than conventional models, it also shows that the learning process is unstable. This can be seen in the deviation between each epoch where the model showed unstable validation accuracy of over 14%. This may be due to several reasons such as the complexity and hence sensitivity of the model, or the lack of validation data like specific number of classes in each split. These issues might cause the model weights to deviate significantly upon each update due to the lack of specific classes in the validation data. Thus, causing inaccuracy and imprecision.

**Optimization Algorithm : Bayesian**

By tuning the dropout rate, learning rate and batch size, the optimization algorithm improved the accuracy of the validation set by a considerable 2 - 3%. The optimization algorithm although improves the validation accuracy, still fails to solve the issue of unstable validation accuracy and unstable learning process.

**Status**

The model is not suitable for deployment as the main algorithm fails to consistently achieve a stable validation accuracy. This indicates that the training performed may not be sufficient for reliable performance. Only when the model receives larger input training data and validation data will we re-consider the main model. In

# Model Evaluation & Analysis Part 2

**Adversarial Auto Encoders**

Adversarial Autoencoders (AAEs) are a class of generative models that combine the concepts of autoencoders and generative adversarial networks (GANs). They were introduced to learn disentangled representations of data in an unsupervised manner. Autoencoders are neural networks that are trained to encode an input data point into a lower-dimensional representation (encoder), and then decode it back to its original form (decoder). The goal of an autoencoder is to reconstruct the input data as accurately as possible, forcing the encoder to learn a meaningful representation in the process. In AAEs, an additional component is introduced called the "adversary" or "discriminator," which is inspired by GANs. The discriminator is trained to differentiate between the learned representations (latent codes) generated by the encoder and a set of prior distributions (usually Gaussian). The encoder and decoder are trained in tandem to fool the discriminator into believing that the reconstructed samples are drawn from the prior distributions.

The key idea behind AAEs is to encourage the encoder to generate representations that are indistinguishable from the prior distributions, thus forcing it to learn a meaningful and disentangled latent space. By doing so, AAEs can capture the underlying structure and variations in the data, enabling tasks like anomaly detection.

**Performance**

The AAEs performance metrics are heavily dependent on the threshold value . For our threshold value we set a percentile of 60.5%. after which if the Mean Squared Error lies outside the 60th percentile then we consider the image as malicious. Upon initial testing we were able to garner a validation accuracy of 79% or 158 / 200. This means that only 42 images were misclassified. The percentile or threshold value can be further optimized for further accuracy and precision.

As compared to our CNN, the Adversarial Auto Encoder has a accuracy rate of at least 7% higher than that of the CNN. In addition to the higher accuracy the model also is more efficient in terms of inference and scalability. For our Adversarial Auto Encoder, we were able to use 1000 images from each class with resolution of 512 by 512. (Scaling down would mean a loss of data)

**Optimization Algorithm: Bayesian**

In this section, we define a search space using Bayesian Optimization to specify the hyperparameters we want to tune.

**Status & Caveats**

The model is ready for deployment and is the primary model being used. It outperforms the standard CNN model. However, there is a challenge when it comes to deciding the appropriate threshold values. It is difficult to determine the number and types of images the threat actor might send. This might mean increased maintenance.

# Attack Strategy Plan

## Model Weaknesses

When developing a deep learning model to detect steganography in images, it's important to consider potential weaknesses and attacks that can be used against the model. One weakness of such a model is its susceptibility to adversarial attacks, which can exploit vulnerabilities and deceive the model into misclassifying images. For instance, pixel shifting attacks can manipulate the pixel values in an image, making subtle changes that are imperceptible to humans but can trick the model into misinterpreting the image. Similarly, the one-pixel attack involves modifying a single pixel in an image to mislead the model's classification. To attack the steganography detection model, an adversary might attempt to create images with embedded hidden information that can bypass the model's detection mechanism. By carefully designing the steganographic payload and embedding technique, the adversary can try to evade detection and make the hidden information imperceptible to the model. The attacker may also explore other methods like adversarial perturbations or generative adversarial networks to generate images that contain steganographic content but appear benign to the model.

Below list 2 known vulnerabilities of the model, we have trained.

1. **Robustness**

As, mentioned above, Model robustness refers to the ability of a machine learning model to maintain its performance and generalization capabilities even when faced with various types of perturbations or deviations in the input data. A robust model exhibits resilience and stability, performing consistently well across different conditions, such as variations in the input data, noise, outliers, or adversarial attacks. In the context of model robustness, a robust model is less sensitive to changes in the input data and can handle unexpected or challenging scenarios without significant degradation in its performance. It can effectively generalize and make accurate predictions on unseen or slightly different data points compared to the training set. Ensuring model robustness is crucial for real-world applications where the input data may vary or be subject to noise, errors, or intentional manipulations. Robust models are more reliable, trustworthy, and capable of maintaining their performance in diverse and dynamic environments.

However, within the framework of our model, it exhibits limitations in detecting steganographic images with minor augmentations. Even small augmentations, such as shifting images by one pixel or altering their resolution, have a notable impact on the accuracy and precision of the model. This drawback arises from the absence of data augmentation techniques employed during the training phase, hindering the development of a more robust model. To address this issue, future iterations of the model, particularly when more powerful hardware resources become available, will incorporate the utilization of data augmentation techniques. By doing so, the model will be trained on augmented data, enabling it to better handle various types of image alterations and improve its accuracy and precision in detecting steganography.

1. **Generalization To Newer or More Complex Steganography Algorithm**

The ability of a model to generalize to newer or more complex steganography algorithms refers to its capability to effectively detect hidden information in images even when confronted with advanced or previously unseen steganographic techniques. To ensure generalization to newer or more complex steganography algorithms, the model needs to exhibit a high level of adaptability and robustness. This means that it should be able to identify patterns and features indicative of steganography across a wide range of algorithms, regardless of their intricacy or novelty.

However, it is important to note that our current training dataset primarily consists of images steganographed with algorithms such as JUNIward, JIMIpod, and UERD. Consequently, if other algorithms like OutGuess are used, the model's performance might be compromised, leading to incorrect classification of the data.To address this limitation, future iterations of the model should incorporate training data that encompasses a wider array of steganography algorithms. By diversifying the training dataset to include samples with various steganographic techniques, the model can develop a broader understanding of steganographic patterns and improve its ability to generalize to newer or more complex algorithms.

## Possible Attacks Scenarios

S**cenarios & Assumptions**

To ensure the security and integrity of our implemented models, it is crucial to establish a comprehensive and well-defined attack strategic plan. This plan will help us identify and understand any weaknesses in our model and outline mitigation measures in case the model is misused or compromised. The strategic plan will include a sequence of steps for executing an attack, with a focus on preventing further vulnerabilities and minimizing risks. In different scenarios, our approach will consider the assumption that the model is not publicly accessible.

* In the first scenario, we will adopt the role of a discontented employee who possesses insider knowledge of the model development and data gathering processes. This will enable us to conduct a White box attack, where we have complete knowledge and access to the model's internal workings.
* In the second scenario, we will assume the perspective of an external actor with limited familiarity regarding the model's internal mechanisms. This situation calls for a Black box attack, where we must strategize and attempt to compromise the model without much information.
* We will explore the scenario of a disgruntled employee within the company who is not directly involved in the creation of the model. This attacker's actions will be considered a Gray or Black box attack, as they possess partial knowledge or no access to certain aspects or the totality of the model due to its high confidentiality status.

By considering these different attack scenarios and strategizing accordingly, we can proactively identify potential vulnerabilities, strengthen our model's defences, and develop countermeasures to mitigate risks associated with misuse or compromise.

**Assumed Model Location on Network**

In addition to the type of attack, we also consider the steg analysis algorithm to be held at various points in a network. The points mentioned below are some of the considered areas:

* **Network Perimeter**: Steganography detection will be performed at the network perimeter, where traffic enters or exits the company's network. This can include deploying detection algorithms on firewalls, intrusion detection systems (IDS), or intrusion prevention systems (IPS) that inspect network packets for signs of steganographic techniques.
* **Network Monitoring Systems**: Steganography detection algorithms will be integrated into network monitoring systems, such as network traffic analysers or packet capture appliances. These systems capture and analyse network traffic in real-time or from stored packet captures, looking for anomalies or suspicious patterns that may indicate the presence of steganography.
* **Email Gateways:** Since steganography can be used to hide information in email attachments, steganography detection algorithms will be implemented within email gateways. These algorithms can scan email attachments for steganographic content, helping to prevent the transmission of hidden information through email communications.

**Notes**

Note, these systems as assumed to be in the company network will not be publicly accessible. System is highly confidential, and location of the model is not known to employees or outsiders but only to those working on the model or those working together with the model developers. This is done so to reduce information on the model to reduce the frequency of both black and grey box attacks. Note that in section 2 we utilise this assumption mentioned above.

Note that the likelihood of Scenario 1 happening is low but never zero. Since the employee is working together with a development team note that all possible attack mentioned in the section “Types of Attack” will be difficult to carry out without anyone noticing. Thus, we will highlight two possible attacks that are of difficulty to spot and detect.

Note that The MITRE Attack plan would utilise the steps in “Attack Strategy” and further elaborate upon the steps.

## Attacks Scenarios 1: White Box

**Attack Strategies**

|  |  |
| --- | --- |
| **Attack Strategies Info** | |
| **High Effectiveness Low Probability** | |
| **Nature** | White/Working on Dev of Model/Insider |
| **Objective** | Present Model with Erroneous Data, Edit Model Architecture for Backdoor |
| **Type** | Reverse Engineering, Model Poisoning, Dataset Poisoning Backdoor Attacks |
| **Expected Output** | Evasion Through Confusing/Poisoning Model/Poisoning Data, active persistence |

**Nature of Attack**

In a white box setting, the attacker has full knowledge of the model architecture, parameters, and internal workings. This allows them to craft targeted attacks by directly manipulating the input data based on their understanding of the model's vulnerabilities.

**Objective**

The objective of our attack strategic plan is two folded. Firstly, we anticipate successfully deceiving the steganography detection model, resulting in misclassifications or erroneous predictions when presented with adversarial examples. We will fool the model by presenting the model with erroneous data to be trained on in the development stage, we will also query and test the model performance to see what the model fails to detect and what it does successfully detect during the development stage. This two-prong approach would be discussed further below.

**Type of Possible Attacks**

Under Evasion Attacks when an insider threat actor is involved in the model, Dataset poisoning becomes a direct method to compromise the model's integrity by manipulating its foundational knowledge. This is accomplished by introducing inaccurate or mislabelled data into the dataset, thereby disrupting the entire learning process. Such attacks can take place during the data collection or curation phases, and their detection can be challenging due to the typically extensive size of training datasets, often obtained from diverse distributed sources.

Additionally, Reverse Engineering plays a significant role in generating deceptive data that evades model detection. The process of reverse engineering may involve using a GANs network to learn the model's classification failures and carefully modifying the threat actor's GANs model to generate images that deceive the models into perceiving them as non-erroneous. This tactic complements the data poisoning attack as the attacker can input such manipulated data into the models to poison the dataset.

Furthermore, a backdoor attack can be initialised. In this type of attack, the attacker establishes a hidden pathway or mechanism within an AI model, enabling them to manipulate the model's behaviour according to their intentions. This manipulation often involves classifying specific inputs in a predetermined manner. The backdoor itself can be introduced by tampering with the training data used during the model's training phase as mentioned above or by injecting malicious code into the training process. This means that if the attacker knows the model well enough, the attacker might plant a section of code, where for example if the steganographed image is paired with a certain input or characteristic and when the model receives the image paired with that specific input , the model might instead use that specific input or characteristic to ignore the detection process and therefore bypass the model.

Lastly, Model Poisoning occurs when the entire operational model is replaced with a different model. This form of attack shares similarities with conventional cyber-attacks, where the digital files comprising the model can be modified or substituted. If the model is deployed, and if a threat actor was working with another threat actor from outside the company, the company network might be compromised, such that stegnographed images containing executables might be able to infiltrate the companies email gateway, network perimeter as well as network monitoring systems.

**Attack Plan : Assuming a Data poisoning + Backdoor Attacks + Evasion**

Reconnaissance Phase: In the reconnaissance phase, the insider with their unique insight gained from being involved in the model's creation would leverage their understanding of its weaknesses and vulnerabilities. This knowledge would allow them to identify potential points of failure within the model's architecture, algorithms, or training process. They would carefully analyse the model's components, such as its data sources, pre-processing techniques, and decision-making mechanisms. By examining these aspects, they could pinpoint specific areas where the model may exhibit vulnerabilities or have limited robustness.

Creation of Adversarial Data & Model: Once the potential weaknesses are identified, the disgruntled employee would proceed to craft adversarial data aimed at exploiting these vulnerabilities. They would strategically manipulate the input data to deceive the model and induce it to produce inaccurate or undesirable outputs. Various techniques could be employed to generate adversarial examples that effectively fool the model. Input perturbations involve making slight modifications to input data, such as adding imperceptible noise or altering pixel values, to cause significant changes in the model's predictions. Generative methods, such as generative adversarial networks (GANs), could be utilized to generate synthetic data that appears innocuous but leads to misclassifications or erroneous predictions. Furthermore, in addition to manipulating the data, the disgruntled employee might go a step further by developing an adversarial model or modifying the existing one. This could involve tampering with the code or parameters of the model to enhance its susceptibility to adversarial attacks. By subtly altering the steganalysis model's code or adjusting certain parameters, they could diminish its ability to detect or defend against adversarial examples. This malicious modification would make the model more vulnerable to the crafted adversarial data and increase the chances of successful exploitation. Throughout the process, the disgruntled employee would meticulously test the generated adversarial data against the target model. They would carefully evaluate the effectiveness of the adversarial examples in evading detection and causing the desired misclassification or erroneous predictions. This testing phase ensures that the crafted attacks remain stealthy and successfully bypass any defence mechanisms implemented by the model. By continuously iterating and refining their adversarial data and techniques, the insider would strive to stay one step ahead of the model's detection capabilities.

Overall, the insider's knowledge and access to the model's inner workings grant them a significant advantage in strategically exploiting its weaknesses. By leveraging their understanding of the model's vulnerabilities, manipulating data inputs, and potentially modifying the model itself, they can compromise its reliability and integrity. This insider's ability to carefully plan and execute targeted attacks can have detrimental consequences for the model's performance, trustworthiness, and the systems integrity.

**List of steps taken – Summary**

1. Gain a understanding of the company network infrastructure through the projects done
2. Gain an insider's understanding & Assess model weaknesses: The employee carefully examines the model's code, parameters, and decision-making mechanisms to identify areas that can be exploited. They may identify ways to modify the steganalysis model's code or adjust parameters to make it less effective in detecting adversarial examples.
3. As the attacker would like to better cover the tracks, he/she might need to acquire their GPUs or CPUs to craft the adversarial models and generate the adversarial data.
4. Develop an adversarial model (optional): In addition to manipulating the data, the employee may choose to develop an adversarial model or modify the existing model. This involves altering the code or parameters to enhance its susceptibility to adversarial attacks.
5. Craft adversarial data: The employee strategically manipulates the input data to deceive the model. They employ techniques such as input perturbations, gradient-based optimization, or generative methods to generate adversarial examples that can effectively fool the model.
6. Test adversarial data: The employee rigorously tests the crafted adversarial data against the target model. They evaluate the effectiveness of the adversarial examples in evading detection and causing misclassification or erroneous predictions.
7. Refine and iterate: Based on the testing results, the employee iteratively refines the adversarial data and techniques to enhance their success rate and evade detection.
8. Test adversarial data: The employee rigorously tests the crafted adversarial data against the target model. They evaluate the effectiveness of the adversarial examples in evading detection and causing misclassification or erroneous predictions.
9. After careful testing the employee will use the created proxy or adversarial model to generate erroneous data on mass to poison the training dataset, to further erode the model’s integrity and therefore accuracy.
10. Repeat step 4 – 9 till succession
11. The attacker can also create a backdoor ML model to facilitate for easier evasion.
12. Maintain stealth: Throughout the process, the employee ensures that the crafted attacks remain stealthy and do not trigger any defence mechanisms implemented by the model.
13. While the employee has access to the model and training and testing data, he/she may be able to exfiltrate information and sell it on the Dark Web to make a profit. The attacker can also team up with other malicious threat actors to further find exploitations within the model.
14. Once the attacker or insider has sold the information to outsiders, the attacks become more severe. Depending on the threat actor and their goals they might be able to do the following below or more.
15. Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
16. Erosion of Model Integrity: If the attackers want to continue to erode the ML system. Attackers can continue to add in erroneous data to confuse the model and further compromise the integrity of the model.
17. Sell Information: The Attacker might sell information on the Dark Web for profit and attract more unwanted threat actors who might take more advantage and cause more damage to the system. This might cause economic harm to the company as data might be lost, stolen, or destroyed.
18. False Alarm: Spamming or generating false alarms which wastes the time of analyst at the organization.

**Expected Output**

The anticipated outcome of the attack is for the steganography detection model to exhibit misclassifications of stenographic images and be evaded by the generated adversarial networks. Through the utilization of artificial intelligence, the attacker aims to create deceptive images that confuse and misguide the model, thereby compromising its accuracy and reliability. These manipulated images are strategically introduced into the dataset used for training the model, effectively poisoning its foundational knowledge. If the attacker successfully executes the attack without detection, the potential consequences could include the leakage of sensitive company information or trade secrets. Such data breaches have the potential to severely damage the company's reputation, customer trust, and ultimately impact its sales and business operations.

## Attacks Scenario 2: Grey/Black Box

|  |  |
| --- | --- |
| **Attack Strategies Info** | |
| **Moderate/Low or High Effectiveness Depending on skill of threat actor, Increased Probability** | |
| **Nature** | Black or Grey/ RED TEAM/Pentesters/Outsider |
| **Objective** | Present Model with Erroneous Data, Evade model Through Adversarial Examples |
| **Type** | Active persistence, Evasion Attacks Adversarial Example ,Data Poisoning |
| **Expected Output** | Evasion Through Confusing/Poisoning Model/Poisoning Data, active persistence |

**Nature of attack**

A grey/black box attack is a type of attack where the attacker has no prior knowledge or access to the internal workings of a target system or model. In a black box attack, the attacker can only interact with the system through its inputs and observe its outputs. They have limited or no knowledge about the internal algorithms, parameters, or architecture of the system. The goal of a black box attack is to exploit vulnerabilities and manipulate the system's behaviour without any specific knowledge of its internal mechanisms. The attacker typically relies on techniques such as input manipulation, probing, and analysis of system responses to achieve their objectives. As such for our scenario the red team(I) will have knowledge of such a system in place, but no other information.

**Objective**

Due to the limited knowledge of the network & hence the security monitoring system. We are unable to assess the true weakness of the system, we are only left with speculation, and experience. Ideally, we must employ traditional cybersecurity techniques found in the MITRE Attack plan. The objective is to attack a steganography detection model, gather information, testing its weaknesses using phishing emails with tampered images, and potentially deploying poisoned datasets for long-term exploitation of the system.That would be the easiest way to compromise the system assuming the attackers knew about the new AI system in place through insiders.

**Type of Attacks : Reverse Engineering + Adversarial Attacks/Evasion Attack Approach**

In a black/Gray box attack, the first step is to gather relevant information about the target's steganography detection model or dataset. The attacker may analyse publicly available information, research papers, or employ social engineering techniques to contact the author or individuals familiar with the model. The objective is to gain insights into the model's functionality, current vulnerabilities, and the data it is trained on. By understanding these aspects, the attacker can develop a strategic plan to exploit the model's weaknesses effectively. If the steganography detection model is publicly accessible, the attacker can acquire it to conduct further investigations.

This may involve reverse engineering the model to uncover its inner workings and assess its effectiveness against known adversarial techniques. Through careful analysis, the attacker can identify potential vulnerabilities and design attack strategies accordingly. However, as direct access to the model is not possible, the attacker can still leverage similar models or research to gain a better understanding of the underlying principles and potential vulnerabilities that may apply to the target model.

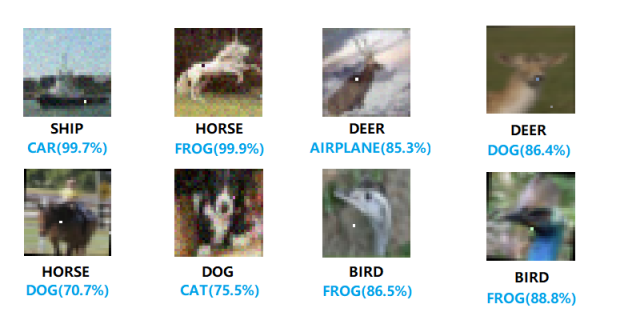


Figure 2: One Pixel Attack Example

Considering that the red team is sufficiently skilled and can mask or change their IP Address at will, we can expect the red team to adopt the conventional method of sending phishing emails tampered with images that contain malware or executables. The attacker can then use strategies like the one-pixel attack to test the robustness of the model. If the image gets through and successfully downloads into the client system, then sends a signal that the process was successful then the attacker will be able to test the model and create adversarial examples. If the attacker fails and his IP gets block, then we know that the perturbed image is not successful. This method although time , intensive and risky can allow a good understanding of what perturb images work and what does not. Due to the nature of the attack information on the model or where it is located might be difficult for the attacker to assess, not only does the attacker need to do port scanning to find the weakness of the system the attacker might also require some social engineering to gain more info on the model.

This will present a path of attack. If the model is found to be robust then the attacker might have to play the long game and published poisoned datasets to trick the development team into using the attacker’s poisoned dataset. This will allow the attackers to successfully carry out data poisoning. The poisoned dataset will then serve as a persistent measurement to allow the attackers to infiltrate or exploit the system for a longer period.

**List of Steps - Summary**

1. Gather information: The attacker needs to collect relevant information about the target's steganography detection model or dataset. This can be done by analysing publicly available information, research papers, or employing social engineering techniques to contact individuals familiar with the model. Search the Dark Web if necessary.
2. Understand model functionality and vulnerabilities: By analysing the gathered information, the attacker aims to gain insights into the model's functionality, current vulnerabilities, and the data it is trained on. This understanding helps the attacker develop a strategic plan to exploit the model's weaknesses effectively.
3. Analyse similar models or research: If direct access to the model is not possible, the attacker can leverage similar models or research to gain a better understanding of the underlying principles and potential vulnerabilities that may apply to the target model.
4. Reverse engineering the model researched online to uncover its inner workings and assess its effectiveness against known adversarial techniques.
5. Acquire hardware like GPU to build advance adversarial models.
6. Developing Adversarial models based on the research done in Reconnaissance.
7. Develop & send phishing emails with tampered images to target the email gateways: The attacker can adopt the conventional method of sending phishing emails that contain malware or executables disguised as images. These images are tampered to test the robustness of the steganography detection model.
8. Test the model and create adversarial examples: If the tampered image successfully bypasses the model's detection and downloads into the client system, the attacker can further develop their model and create adversarial examples. This step helps the attacker understand which perturbed images work and which do not. Assess IP blocking: If the attacker's IP gets blocked during the attack, it indicates that the perturbed image was not successful. This information helps the attacker evaluate the effectiveness of their attack and adjust if necessary.
9. Assess the robustness of the model: Based on the results obtained from testing and creating adversarial examples, the attacker determines the robustness of the steganography detection model. If the model is found to be robust, the attacker may need to proceed with a different approach.
10. Publish poisoned datasets: If the model is robust, the attacker can play the long game and publish poisoned datasets. These datasets are designed to trick the development team into using them, allowing the attacker to carry out data poisoning.
11. Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
12. If the attacker wishes to gain access or privilege escalate or infect the companies email gateway or gain control of the domain or gain access to an administrator/ root account, the attacker can encode an exe or a execute a download file over terminal when the file is downloaded into the system.
13. If the attacker manages to bypass the NMS (Network Monitoring Systems) though privilege escalation the attacker might be able to get full model access and further improve , edit, and create a Backdoor ML model without the team noticing. Since the attacker has full access to the companies network the attack will be considered successful.
14. The attacker now must cover his footprints to exploit the system as long as possible.
15. Once the attacker has gained access to the network the attacker could send data via traditional cyber means or transfer data to cloud accounts of which they have access to to avoid typical file transfer and network-based exfiltration detection.
16. The attacker could then gain more information on the company network systems and the contents within it.
17. Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
18. Erosion of Model Integrity: If the attackers want to continue to erode the ML system. Attackers can continue to add in erroneous data to confuse the model and further compromise the integrity of the model.
19. Sell Information: The Attacker might sell information on the Dark Web for profit and attract more unwanted threat actors who might take more advantage and cause more damage to the system. This might cause economic harm to the company as data might be lost, stolen, or destroyed.
20. False Alarm: Spamming or generating false alarms which wastes the time of analyst at the organization.

**Expected Output: Black Box**

The expected output of the attacks described in the previous steps would be to compromise the effectiveness and reliability of the steganography detection model. The specific outcomes can vary depending on the nature and success of the attack, but some possible expected outputs include:

* Misclassification of tampered and untampered images: The manipulated model may incorrectly classify tampered images as genuine or vice versa, leading to a loss of accuracy and integrity in the steganalysis process.
* Evasion of detection: The generated adversarial data and techniques aim to deceive the steganography detection model, causing it to overlook hidden information or fail to detect subtle alterations in images that would typically trigger detection.
* Reduction in model robustness: The successful attacks can undermine the model's robustness, making it more susceptible to adversarial manipulation in general. This can impact the model's performance on future unseen data.
* Compromised trust in the model: The attacks can erode trust in the steganography detection model, as its ability to accurately identify hidden information or detect tampered images may be called into question.
* Potential impact on downstream systems: If the compromised model is used as a part of a larger system or application, the inaccurate outputs and misclassifications can have cascading effects on the overall system's functionality and reliability.

# MITRE ATLAS Framework

MITRE ATLAS™ (Adversarial Threat Landscape for Artificial-Intelligence Systems) is a comprehensive knowledge base that focuses on adversary tactics, techniques, and case studies specifically designed for machine learning (ML) systems. It is built upon real-world observations, demonstrations conducted by ML red teams and security groups, and the latest findings from academic research. ATLAS follows a similar framework to MITRE ATT&CK®, and its tactics and techniques complement those found in ATT&CK. By leveraging ATLAS, organizations can gain valuable insights into potential threats and enhance their defence against adversarial attacks targeting ML systems.

## Attack Scenario 1

Reconnaissance & Resource Development

**Tactics:** Gather Victim Network Information , Gather Victim Host Information, Search Closed Source Repo, Search Application Repo, Search for Publicly Available Adversarial Vulnerability Analysis

* Gain a understanding of the company network infrastructure through the projects done
* Gain an insider's understanding & Assess model weaknesses: The employee carefully examines the model's code, parameters, and decision-making mechanisms to identify areas that can be exploited. They may identify ways to modify the steganalysis model's code or adjust parameters to make it less effective in detecting adversarial examples.

These steps fall under "Reconnaissance & resource development" as they involve gathering information and resources to exploit vulnerabilities in a system. The employee gains an insider's understanding by examining the steganalysis model's code, parameters, and decision-making mechanisms. This knowledge allows them to identify areas that can be exploited and modify the model to make it less effective in detecting adversarial examples. These steps are crucial for planning an attack or developing countermeasures.

Resource Dev & Initial Access, ML Model Access & Execution

**Tactics:** Acquire Infrastructure, ML Development workspace GPUs & CPUs, Develop Adversarial ML Attack Capabilities, Evade ML Model, Obtain Capabilities, ML Supply Chain Compromise

* As the attacker would like to better cover the tracks, he/she might need to acquire their GPUs or CPUs or ML Software to craft the adversarial models and generate the adversarial data.
* Craft adversarial data: The employee strategically manipulates the input data to deceive the model. They employ techniques such as input perturbations, gradient-based optimization, or generative methods to generate adversarial examples that can effectively fool the model.
* Test adversarial data: The employee rigorously tests the crafted adversarial data against the target model since he has access. They evaluate the effectiveness of the adversarial examples in evading detection and causing misclassification or erroneous predictions.
* Refine and iterate: Based on the testing results, the employee iteratively refines the adversarial data and techniques to enhance their success rate and evade detection.
* Develop an adversarial model (optional): In addition to manipulating the data, the employee may choose to develop an adversarial model or modify the existing model. This involves altering the code or parameters to enhance its susceptibility to adversarial attacks.
* Test adversarial data: The employee rigorously tests the crafted adversarial data against the target model. They evaluate the effectiveness of the adversarial examples in evading detection and causing misclassification or erroneous predictions.
* Refine and iterate: Based on the testing results, the employee iteratively refines the adversarial data and techniques to enhance their success rate and evade detection.

These steps are classified under the "Resource Dev & Initial Access and Execution" category as they involve the development of resources and techniques necessary to gain initial access and execute an attack on the target model. The employee strategically crafts adversarial data by manipulating input data to deceive the model, employing techniques like input perturbations, gradient-based optimization, or generative methods. They rigorously test the crafted adversarial data against the target model to assess its effectiveness in evading detection and causing misclassification or erroneous predictions. Based on the testing results, they refine and iterate the adversarial data and techniques to enhance their success rate and evade detection. Additionally, the employee may choose to develop or modify an adversarial model to make it more susceptible to attacks. The repeated testing, refining, and iterating steps highlight the iterative nature of the process. These steps are categorized as "Resource Dev & Initial Access and Execution" because they involve developing and refining resources, such as adversarial data and techniques, to gain initial access to the target model and execute the attack.

Execution & Persistence

**Tactics:** Poison Training data, Create or Modify System Process, Inter-Process Communication

* After careful testing the employee will use the created proxy or adversarial model to generate erroneous data on mass to poison the training dataset, to further erode the model’s integrity and therefore accuracy.
* The attacker can also create a backdoor ML model to facilitate for easier evasion.

These points are classified as Execution & Persistence within the MITRE ATT&CK framework because they involve the implementation of specific actions by the attacker to execute their attack and persistently maintain their presence within the system. The first point states that after thorough testing, the attacker will utilize the created proxy or adversarial model to generate a large amount of erroneous data. This data is injected into the training dataset, with the intention of poisoning it. By doing so, the attacker aims to further degrade the integrity and accuracy of the targeted ML model. This step falls under Execution as it involves actively carrying out the attack and Persistence as the impact is long-lasting, affecting the model's future performance. The second point highlights that the attacker can create a backdoor ML model. This refers to the development of an alternative model or component that acts as a hidden entry point for the attacker. This backdoor model facilitates easier evasion, allowing the attacker to bypass detection mechanisms and maintain access to the system. This action falls under Execution as it involves the creation of a specific component to aid in the attack and Persistence as the backdoor allows for continuous unauthorized access.

Defence Evasion, Discovery Collection & Initial Access & ML Attack Staging

**Tactics**: Evade ML Model, Impair Defense, Full Model Access, Create Proxy Model, Verify Attack, Craft Adversarial Data, ML Artifact Collection…

* Maintain stealth: Throughout the process, the employee ensures that the crafted attacks remain stealthy and do not trigger any defence mechanisms implemented by the model.
* After careful testing the employee will use the created proxy or adversarial model to generate erroneous data on mass to poison the training dataset, to further erode the model’s integrity and therefore accuracy.

According to MITRE The process of stealing data from a corporate system is also known as exfiltration, Throughout the process, the employee responsible for the attack takes great care to ensure that the crafted attacks remain stealthy and do not trigger any defence mechanisms implemented by the targeted ML model. By avoiding detection and bypassing security measures, the attacker aims to maintain their unauthorized access and manipulate the system without raising alarms.

Another step which falls under the categories of Discovery Collection, Initial Access, and ML Attack Staging. It begins with the careful testing conducted by the attacker, which allows them to understand the behaviour and vulnerabilities of the ML model. This process involves collecting information and assessing the model's response to various inputs to identify weaknesses. Once the attacker gains initial access to the system, they utilize a created proxy or adversarial model. This proxy or model serves as a tool to generate a large volume of erroneous data, which is then injected into the training dataset. The primary goal here is to poison the dataset, further eroding the integrity and accuracy of the ML model.

By combining these steps, the attacker successfully evades defences, discovers vulnerabilities, gains initial access, and stages an ML attack by manipulating the training dataset. These actions highlight the attacker's efforts to compromise the ML model and undermine its reliability and effectiveness.

Exfiltration

**Tactics:** Automated Exfiltration, Transfer Data to Cloud Account, Exfiltration via Cyber Means, Exfiltration Over Alternative Protocol

* While the employee has access to the model and training and testing data, he/she may be able to exfiltrate information and sell it on the Dark Web to make a profit. The attacker can also team up with other malicious threat actors to further find exploitations within the model.

This step falls under the category of exfiltration within the context of the MITRE ATT&CK framework. The employee, who has access to the ML model, as well as the training and testing data, may engage in the exfiltration of information for personal gain. By exfiltrating this sensitive data, the attacker can then sell it on the Dark Web, where there is a market for such illicit activities.

The exfiltrated information can be valuable to various parties, including competitors, organized crime networks, or even nation-state actors. They may be interested in obtaining insights into the ML model, training data, or proprietary algorithms. By selling this information, the attacker can potentially make a significant profit.

Additionally, the attacker might collaborate with other malicious threat actors to exploit the ML model further. By teaming up, they can pool their expertise and resources to identify and exploit vulnerabilities within the model, its implementation, or the associated infrastructure. This collaborative effort enables the attackers to maximize their impact and potential gains.

In summary, in this context, exfiltration involves the unauthorized extraction and dissemination of sensitive information by the employee with access to the ML model and associated data. The motive behind this activity is primarily financial gain, and the attacker may also collaborate with other threat actors to exploit the model for their own purposes.

Impact

**Tactics:** Erode ML Model Integrity, Evade ML Model, ML Intellectual Property Theft, Spamming ML System with Chaff Data, Dos

* Once the attacker or insider has sold the information to outsiders, the attacks become more severe. Depending on the threat actor and their goals they might be able to do the following below or more.
* Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
* Erosion of Model Integrity: If the attackers want to continue to erode the ML system. Attackers can continue to add in erroneous data to confuse the model and further compromise the integrity of the model.
* Sell Information: The Attacker might sell information on the Dark Web for profit and attract more unwanted threat actors who might take more advantage and cause more damage to the system. This might cause economic harm to the company as data might be lost, stolen, or destroyed.
* False Alarm: Spamming or generating false alarms which wastes the time of analyst at the organization.

The steps specifically apply to the Impact category within the MITRE ATT&CK framework because they encompass a range of techniques and actions that can result in significant consequences for both the targeted system and the organization. The primary objective of the attacker is to degrade the integrity of the ML model. This is accomplished through the injection of erroneous data, manipulation of the model to induce confusion, and compromising its accuracy. These actions undermine the reliability and effectiveness of the ML system, potentially leading to incorrect predictions and compromised outcomes.

In addition to the model degradation and compromised security of the companies’ network, the development team of the model would be heavily screened and questioned for their involvement.

## Attack Scenario 2

Reconnaissance & Resource Development

**Tactics:** Research for publicly available adversarial vulnerability, Search Closed Sources, Acquire Public ML Artifacts

* Gather information: The attacker needs to collect relevant information about the target's steganography detection model or dataset. This can be done by analysing publicly available information, research papers, or employing social engineering techniques to contact individuals familiar with the model. Search the Dark Web if necessary.
* Understand model functionality and vulnerabilities: By analysing the gathered information, the attacker aims to gain insights into the model's functionality, current vulnerabilities, and the data it is trained on. This understanding helps the attacker develop a strategic plan to exploit the model's weaknesses effectively.
* Analyse similar models or research: If direct access to the model is not possible, the attacker can leverage similar models or research to gain a better understanding of the underlying principles and potential vulnerabilities that may apply to the target model.
* Reverse engineering the model researched online to uncover its inner workings and assess its effectiveness against known adversarial techniques.

The above steps are classified as reconnaissance because it involves the gathering of information about the target's steganography detection model or dataset. This includes analysing publicly available information, research papers, and potentially employing social engineering techniques to contact individuals familiar with the model. The goal of this reconnaissance is to understand the model's functionality, vulnerabilities, and the data it is trained on, which allows the attacker to develop a strategic plan to exploit its weaknesses effectively. Additionally, the text mentions reverse engineering the model researched online to uncover its inner workings and assess its effectiveness against known adversarial techniques, which further emphasizes the reconnaissance aspect.

Resource Dev & Initial Access & Execution

**Tactics:** Acquire Infrastructure, ML Development workspace GPUs & CPUs, Develop Adversarial ML Attack Capabilities, Phishing, Evade ML Model, User Execution

* Acquire hardware like GPU to build advance adversarial models.
* Developing Adversarial models based on the research done in Reconnaissance.
* Develop & send phishing emails with tampered images to target the email gateways: The attacker can adopt the conventional method of sending phishing emails that contain malware or executables disguised as images. These images are tampered to test the robustness of the steganography detection model.
* Test the model and create adversarial examples: If the tampered image successfully bypasses the model's detection and downloads into the client system, the attacker can further develop their model and create adversarial examples. This step helps the attacker understand which perturbed images work and which do not. Assess IP blocking: If the attacker's IP gets blocked during the attack, it indicates that the perturbed image was not successful. This information helps the attacker evaluate the effectiveness of their attack and adjust if necessary.
* Assess the robustness of the model: Based on the results obtained from testing and creating adversarial examples, the attacker determines the robustness of the steganography detection model. If the model is found to be robust, the attacker may need to proceed with a different approach.

During the Resource Development and Initial Access phase of the attack, the perpetrator engages in several activities to lay the groundwork for their malicious intentions. First and foremost, they focus on acquiring the necessary resources to carry out their plan. This often involves obtaining hardware such as a Graphics Processing Unit (GPU), which enables the building and utilization of advanced adversarial models. Building upon the knowledge gained during the reconnaissance phase, the attacker develops adversarial models. These models are based on extensive research and serve as powerful tools for their intended purposes. By leveraging the insights gathered during the reconnaissance phase, the attacker gains a deeper understanding of the target's steganography detection model, its functionality, vulnerabilities, and the dataset it operates on. This information is instrumental in formulating a strategic plan to exploit the weaknesses of the model effectively. To target email gateways and assess the robustness of the steganography detection model, the attacker employs a conventional method: phishing emails. These deceptive emails contain tampered images that are skillfully manipulated to evaluate the effectiveness of the model's detection capabilities. By studying the outcomes of these tests, the attacker gains valuable insights into the strengths and weaknesses of the steganography detection model.

Additionally, the attacker engages in further activities to evaluate the model's robustness and adjust their approach accordingly. They create adversarial examples by developing perturbed images that can potentially evade the detection mechanisms of the model. Successful evasion indicates potential vulnerabilities, while blocked access or detection signifies the model's resilience. This knowledge helps the attacker refine their attack strategy.

Overall, the Resource Development and Initial Access phase of the attack lays the foundation for subsequent stages, combining resource acquisition, model development, testing, and potential data poisoning. These activities empower the attacker to gain initial access, exploit vulnerabilities, and pave the way for further infiltration and exploitation of the targeted system.

Execution & Persistence

**Tactics**: User Execution, Poison Training data, Create or Modify System Process, Inter-Process Communication

* Publish poisoned datasets: If the model is robust, the attacker can play the long game and publish poisoned datasets. These datasets are designed to trick the development team into using them, allowing the attacker to carry out data poisoning.
* Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
* If the attacker wishes to gain access or privilege escalate or infect the companies email gateway or gain control of the domain. The attacker can encode an exe or a execute a download file over terminal when the file is downloaded into the system.

During the Execution & Persistence phase, the attacker employs specific tactics to execute their attack and maintain a lasting presence by compromising more systems. One such tactic is the publication of poisoned datasets. If the steganography detection model is found to be robust, the attacker takes a patient approach and releases datasets intentionally crafted to deceive the development team. These poisoned datasets are designed to trick the team into utilizing them, leading to data poisoning within the system.

Once the poisoned datasets are in place, the attacker gains the means to infiltrate and exploit the system over an extended period. By leveraging the persistence provided by the poisoned datasets, the attacker can maintain their presence within the compromised environment, facilitating further attacks and exploitation.

In addition to data poisoning, the attacker may target specific objectives, such as gaining access to sensitive information, escalating privileges, infecting the company's email gateway, or assuming control of the domain. To achieve these goals, the attacker may utilize techniques such as encoding malicious executables or executing download files when accessed by the system. These actions help the attacker establish a foothold and maintain persistence within the compromised system, allowing for continued exploitation and control.

In summary, the activities described in the text align with the Execution & Persistence phase as they involve the deployment of poisoned datasets, leveraging their persistence to infiltrate and exploit the compromised system over an extended period, and carrying out actions to gain control or escalate privileges within the environment.

Defence Evasion, Discovery Collection & Initial Access & ML Attack Staging

**Tactics**: Evade ML Model, Impair Defence, Abuse Elevation Control, Create or Modify System Process, Modify Registry, Full Model Access, Create Proxy Model, Verify Attack, Craft Adversarial Data, ML Artifact Collection…

* If the attacker manages to evade the NMS (Network Monitoring Systems) though privilege escalation the attacker might be able to get full model access and further improve , edit, and create a Backdoor ML model without the team noticing. Since the attacker has full access to the companies network the attack will be considered successful.
* Once the attacker manages to get full access to the system the attacker can collect data from the employees’ or steal data from the development team on the AI model.
* The attacker now must cover his footprints to exploit the system as long as possible.

The steps above can be classified under the category of Defense Evasion in the MITRE ATT&CK framework.as it includes various techniques used by attackers to avoid detection and bypass security measures.

Firstly, the attacker plans to gain access or escalate privileges by encoding an executable or executing a download file over the terminal, aiming to infect the company's email gateway, gain control of the domain, or obtain administrator/root account access if the attacker manages to bypass the Network Monitoring Systems (NMS) through privilege escalation, they can achieve full access to the company's network. This enables them to manipulate and create a backdoor machine learning model possibly without raising suspicion. By evading the NMS, the attacker can continue their activities unnoticed. Lastly, the attacker aims to cover their tracks and maintain access for an extended period. They employ techniques and tools to evade detection by security measures, ensuring their unauthorized access remains undetected. Overall, these actions align with the objectives of Defense Evasion, focusing on evading defensive controls and remaining hidden within the compromised system.

Exfiltration

**Tactics**: Automated Exfiltration, Transfer Data to Cloud Account, Exfiltration via Cyber Means, Exfiltration Over Alternative Protocol

* Once the attacker has gained access to the network the attacker could send data via traditional cyber means or transfer data to cloud accounts of which they have access to to avoid typical file transfer and network-based exfiltration detection.
* The attacker could then gain more information on the company network systems and the contents within it.

The steps above can be classified under the category of Exfiltration in the MITRE ATT&CK framework. Once the attacker gains access to the network, they employ various techniques to move sensitive data out of the target environment while evading detection. Firstly, they utilize traditional cyber means such as email or file transfer protocols to send the data externally. By leveraging established communication channels, the attacker aims to bypass network-based exfiltration detection mechanisms. Secondly, they transfer the data to cloud accounts that they have access to, leveraging the trust associated with legitimate cloud services. This approach adds an additional layer of obfuscation and makes it more challenging to detect the exfiltration attempt. Additionally, the attacker seeks to gather more information about the company's network systems and the contents within them. This involves conducting reconnaissance activities, scanning for vulnerabilities, and performing lateral movement to identify valuable targets. Understanding the network's structure and content provides the attacker with valuable insights to facilitate further exfiltration or exploit opportunities within the compromised environment. Overall, these actions align with the objectives of Exfiltration in the MITRE ATT&CK framework, which focuses on techniques used by attackers to remove sensitive data from a compromised network while remaining undetected.

Impact

**Tactics**: Erode ML Model Integrity, Evade ML Model, ML Intellectual Property Theft, Spamming ML System with Chaff Data, Dos

* Exploit the system: With the poisoned dataset in place, the attacker can infiltrate or exploit the system over a longer period. The poisoned dataset serves as a persistent measure to facilitate further attacks and exploitation.
* Erosion of Model Integrity: If the attackers want to continue to erode the ML system. Attackers can continue to add in erroneous data to confuse the model and further compromise the integrity of the model.
* Sell Information: The Attacker might sell information on the Dark Web for profit and attract more unwanted threat actors who might take more advantage and cause more damage to the system. This might cause economic harm to the company as data might be lost, stolen, or destroyed.
* False Alarm: Spamming or generating false alarms which wastes the time of analyst at the organization.

The steps above pertain to the Impact category in the MITRE ATT&CK framework because it encompasses various techniques and actions that can have significant consequences on the targeted system and organization. The attacker's primary objective is to erode the integrity of the ML model. They achieve this by injecting erroneous data, confusing the model, and compromising its accuracy. Such actions undermine the reliability and effectiveness of the ML system, potentially leading to incorrect predictions and compromised outcomes. Exploiting the system is another key goal for the attacker. By infiltrating the targeted system and leveraging a poisoned dataset, they can persistently undermine the ML model while evading detection and defensive measures. This compromises the overall security and operational efficiency of the system.

Furthermore, the attacker may engage in ML intellectual property theft. They may sell information obtained from the compromised ML system on the Dark Web, seeking to profit from stolen or compromised data. This poses a significant threat to the organization's intellectual property and can result in economic harm, including financial losses and damage to the company's competitive advantage. The text also mentions the tactic of spamming the ML system with chaff data or generating false alarms. By overwhelming the system with irrelevant or misleading information, the attacker aims to waste the time and resources of the organization's analysts. This disrupts normal operations, diverts attention from genuine threats, and hampers the efficiency of the organization's security operations.

In summary, the actions described in the text have a substantial impact on ML systems and the targeted organization. They erode the integrity of the ML model, exploit the system, pose a threat to intellectual property, and introduce disruptions that waste valuable resources. These impacts align with the objectives of the Impact category in the MITRE ATT&CK framework, which focuses on the consequences and harm caused by malicious actions.

# Mitigations

To mitigate against evolving threats, continuous maintenance of datasets and model complexity is essential for long-term effectiveness in detecting advanced threats. By actively maintaining and adapting mitigation techniques, companies can enhance their cyber defence and proactively address emerging risks. Ongoing monitoring and updates ensure that models remain effective in an ever-changing threat landscape, enabling organizations to stay ahead of potential vulnerabilities. By prioritizing continuous maintenance and adaptation, companies can bolster their cybersecurity measures and mitigate risks more effectively. For scenario 1 & 2 we will try to mitigate the most important steps or crucial points where the attacker is able infiltrate or trick our model. For Scenario 1, we will cover the mitigation measures for steps 1 – 11,13 For Scenario 2 we will cover the mitigation measures for steps 6 -10.

**Both Scenario 1 & 2**

Cybersecurity systems are susceptible to adversarial attacks, and when designing ensemble methods, adversarial robustness techniques can be utilized to refine the ensemble’s ability to detect and mitigate attacks aimed at exploiting vulnerabilities in the individual models.

To mitigate adversarial attacks on our Deep Learning steganalysis model, we can employ a range of effective methods. Some of the steps involve retraining the model on diverse datasets as it enables it to adapt to evolving adversarial techniques. Additionally, incorporating noise into inputs and introducing randomness in the model design helps disrupt the patterns exploited by attackers. Adversarial retraining could also play a crucial role in exposing the model to adversarial examples, thus enhancing its ability to detect and mitigate attacks. However, to further illustrate our dataset and model and to use what we have done (develop multiple models) we will be utilizing multiple models created by ourselves and employing model ensembling.

**Ensemble Methods**

Ensemble Method refers to a machine learning approach that enhances predictive accuracy and mitigates overfitting by combining multiple models, such as decision trees or neural networks. In an ensemble, each model is trained on a distinct subset of the training data or employs a different algorithm. During the prediction phase, the ensemble harmonizes the outputs of each model to minimize errors and enhance overall accuracy. Prominent examples of ensemble methods include bagging, boosting, and stacking. This technique proves valuable in harnessing the strengths of individual models and leveraging their collective power to achieve superior performance and robust predictions.

For our models we consider the application of the Standard CNN Transfer Learning methods, The Unsupervised AAE, and the Hybrid CNN + Auto Encoder + MLP Methods. When presented with an image, the three models would be used to infer if the image is indeed tampered with or not. Since we have three models, we will have the AAE as well as the other two CNNs will each weigh in at 33% of the total score. The outputs of both the unsupervised and supervised models can then be combined using ensemble techniques such as averaging, voting, or stacking. This fusion of predictions from multiple models can often result in improved performance, as the different models may capture complementary aspects of the data.

To further increase the reliability of our predictions we can use a simple Neural Network to model the relationship between what is a tampered image and what is not through the acquired datapoints. Since we have the outputs of the model specifically three outputs, we can do simple modelling with the data and build an additional Neural Network to determine the true nature of the image. This is a more reliable method than manually counting the score of each model as this method might capture the relationship between each column or datapoint.

|  |  |  |  |
| --- | --- | --- | --- |
| **CNN + AE + MLP** | **CNN EfficientNetB3** | **Adversarial Auto Encoder** | **Nature of the image** |
| 79% | 59% | 0.035 | Tampered |
| 81% | 57% | 0.200 | Untampered |
| 49% | 50% | 0.775 | Tampered |

**Note: The table above shows how the datatable might look like. Refer to the code to see pseudo code implementation.**

The table above shows three outputs, all three describes the probability of likeliness of the image being Tampered or untampered. This data can then be trained on another neural network (Level 3) to enhance the robustness of the overall model by reducing false positives and false negatives.

**Feature Squeezing**

Feature squeezing operates by compressing the value range of specific features within the input data, effectively "squeezing" them into a narrower span. As a result, the potential set of adversarial examples shrinks, rendering it more challenging for an attacker to discover inputs capable of inducing misclassification. Feature squeezing is a valuable technique employed in steganalysis, specifically in the context of adversarial autoencoders, to enhance their performance and resilience. Steganalysis aims to detect hidden information within digital media, and feature squeezing plays a crucial role in mitigating the impact of adversarial perturbations and improving the model's ability to identify steganographic content.

There are several reasons why feature squeezing is beneficial in an adversarial autoencoder for steganalysis. Firstly, it reduces the noise present in the input data by employing methods like quantization, rounding, or bit-depth reduction. This noise reduction helps the model focus on relevant features and discard unnecessary variations or perturbations introduced by steganographic algorithms.

Secondly, feature squeezing acts as a form of regularization by imposing constraints on the input data. By simplifying the input space, the technique prevents overfitting and encourages the model to learn more general features that are robust across different steganography methods. This ensures that the model performs well on unseen data and does not become too specialized in detecting specific steganographic techniques. Moreover, feature squeezing enhances the model's resilience against adversarial perturbations. Adversarial attacks aim to deceive machine learning models by introducing carefully crafted perturbations that can significantly impact the model's predictions. By reducing the input's dimensionality and sensitivity to minor changes, feature squeezing makes it harder for adversarial perturbations to fool the model, improving its robustness. By reducing the dimensionality of the feature space in images and implementing it within our specific scenario, we effectively decrease the likelihood of perturbed images infiltrating our systems and minimize the risk of our system being compromised by the execution of malicious code. This reduction in the feature space ensures that the model focuses on essential characteristics while disregarding unnecessary variations or perturbations caused by potential steganographic algorithms. Consequently, the chances of adversarial attacks successfully deceiving the model are diminished, thereby bolstering the security and integrity of our system. Moreover, the computational efficiency of the system is also improved, allowing for faster processing and analysis of large datasets.

In Python, there are several libraries that provide functionality for feature squeezing techniques in the context of adversarial machine learning. Here are a few commonly used libraries:

* CleverHans: CleverHans is a powerful library for adversarial machine learning. It provides various utilities and methods for generating adversarial examples and implementing defense mechanisms, including feature squeezing.
* ART (Adversarial Robustness Toolbox): ART is an open-source library for adversarial machine learning developed by IBM. It offers a wide range of tools and techniques for generating and evaluating adversarial attacks and defence, including feature squeezing.
* Foolbox: Foolbox is a Python toolbox for generating and evaluating adversarial attacks. It provides an extensive set of attack methods and defense techniques, including feature squeezing, to improve the robustness of machine learning models.
* Adversarial-robustness-toolbox (ART): Another library with a similar name but different implementation, ART provides a comprehensive set of tools for adversarial machine learning, including feature squeezing as one of the defense mechanisms.

These libraries offer implementations of feature squeezing techniques and provide ready-to-use functions and utilities to apply them to machine learning models. It's important to note that the specific feature squeezing techniques and their implementations may vary across these libraries, so it's recommended to refer to their documentation and examples for more details on usage and integration.

**Adversarial Testing**

Adversarial testing holds significant importance in the field of steganalysis when applied to AI models. The primary objective of such testing is to evaluate the robustness and effectiveness of the AI model in identifying and exposing hidden data.

By crafting adversarial examples specifically tailored to exploit the vulnerabilities or blind spots of the AI model, organizations can assess its ability to withstand sophisticated steganographic techniques. Adversarial testing allows for the identification of weaknesses in the model's feature extraction, classification algorithms, or statistical analysis methods. It serves to uncover potential areas for improvement in the model's ability to accurately detect hidden information.

Furthermore, adversarial testing helps enhance the security and reliability of AI models for steganalysis. By subjecting the model to diverse and challenging adversarial examples, organizations can identify potential evasion techniques or advanced steganographic algorithms that may be used by malicious actors. This knowledge enables the development of countermeasures and the implementation of stronger defense mechanisms to mitigate the risks associated with steganographic attacks.

Adversarial testing also aids in building trust and confidence in AI models for steganalysis. Through rigorous testing and validation, organizations can demonstrate the model's resilience and effectiveness in detecting hidden information, reassuring users, stakeholders, and the wider community about its reliability.

**Exploratory Testing**

Exploratory testing plays a critical role in ensuring the robustness of our AI steganalysis model. By adopting a proactive and immersive approach to security testing, we aim to uncover potential vulnerabilities within the system that could be exploited by malicious actors. The primary objective of this type of testing is to gain deep insights into the inner workings of the steganalysis model, comprehending its behavior, and uncovering any hidden weaknesses.

Through exploratory testing, security testers actively explore the AI steganalysis model, assessing various attack surfaces, potential entry points, and concealed flaws that may expose the system to exploitation. By immersing themselves in the model, testers can proactively identify and address vulnerabilities before they can be leveraged by malicious individuals to hide sensitive information within digital content.

This hands-on and proactive approach enhances the overall security posture of the steganalysis model. By actively identifying and addressing vulnerabilities, we can stay ahead of potential attackers and strengthen the system's defense against steganographic techniques. Exploratory testing helps us gain a deeper understanding of the model's strengths and weaknesses, enabling us to make informed decisions regarding security enhancements and countermeasures.

Through exploratory testing, we actively scrutinize the AI steganalysis model to uncover potential security vulnerabilities. This proactive approach ensures that the model remains robust and effective in detecting and mitigating steganographic content, thereby safeguarding against potential threats posed by hidden information within digital media.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mitigation strategy Framework** | | | |
| **Attack Types** | | **Model Enhancement Mitigation** | **Model-agnostic Mitigation** |
| **Training** | **Poisoning** | * **Enhance data quality.** * **Data sanitization** * **Block poisoning** | * **Output restoration** |
| **Backdoor** | * **Enhance data quality.** * **Data sanitization** * **Trigger detection** * **Model restoration** | * **Trigger detection** * **Trigger deactivation** * **Backdoor detection** |
| **Inference** | **Evasion** | * **Data preprocessing** * **Model hardening** * **Robustness evaluation** | * **AE detection** * **Input restoration** * **Output restoration** |
| **Model stealing** | * **IP management** | * **Limit queries** * **Stealing detection** * **Output obfuscation** * **fingerprinting** |
| **Data extraction** | * **Embed data privacy.** * **Training with privacy** | * **Limit queries** * **Obfuscated confidence scores** |

**Source: ETSI GR SAI 005 V1.1.1 (2021-03)**

The above table describes several techniques to better safeguard and protect our model intellectual property and integrity. The steps mostly apply to our case study and should all be considered when implementing the defense plan. We should not leave any stones unturned and should implement all if possible.

**Mitigating Poisoning Attacks**

To mitigate poisoning attacks on our Deep Learning steganalysis model, we employ a defense-in-depth approach that encompasses several methods. Poisoning attacks involve adding malicious data samples to the training or retraining data of the AI system. To carry out such attacks, attackers typically aim to influence the data collection process or gain access to the data storage to insert poisoned samples.

One important mitigation method is to focus on the security of both the data collection process and general cybersecurity for storage. Following standards such as ISO/IEC 27032 helps ensure that robust security measures are in place to protect against unauthorized access and manipulation of the data.

Another effective approach is data sanitization, which involves thoroughly validating and filtering incoming data samples to identify and remove any potentially poisoned or malicious entries. By scrutinizing the data before it is used for training, we can minimize the risk of including poisoned samples in the dataset.

Robust loss optimization is another method employed to mitigate poisoning attacks. By utilizing loss functions that are less susceptible to the influence of poisoned samples, we can minimize the impact of such malicious data on the training process. This helps to maintain the integrity and accuracy of the steganalysis model.

Additionally, outlier detection techniques are utilized to identify and handle data samples that deviate significantly from the expected patterns or distributions. By flagging and treating potential outliers appropriately, we can minimize the effect of poisoned samples on the training process and reduce the risk of compromising the steganalysis model's performance.

By implementing these methods as part of our defense-in-depth strategy, we enhance the resilience of our Deep Learning steganalysis model against poisoning attacks. This multi-layered approach helps protect the integrity of the training data, optimize loss functions, and identify and handle outliers effectively, ensuring the model's reliability in detecting and mitigating steganographic content.

**Mitigating Inference Attacks**

N.A Our model is closed sourced and implemented in Gateways and Network Monitoring systems in companies. Such models should not have any API due to their sensitive nature and is assumed to have none.

# Conclusion

To conclude, this report showcases a unique approach to authenticate digital images by integrating various techniques, including CNNs and Hybrid Models. It also highlights the concept of steganography and its potential consequences. By incorporating Spatial Co-Occurrence Matrices and utilizing DCT across different colour channels, our proposed model outperforms previous methods that solely relied on CNNs, resulting in a 7% improvement in accuracy.

Furthermore, we conducted thorough assessments of our model's effectiveness and devised an attack plan to simulate potential attacks. Subsequently, we proposed and planned defensive measures to mitigate the simulated attack. These measures include employing model ensembling and conducting Exploratory Testing to evaluate our model and network against various attack surfaces, thereby reducing vulnerabilities within our system.

Lastly, we also created pseudo code to illustrate our point on ensemble modelling and its possible effectiveness.

Overall, this research showcases the significance of integrating different techniques to enhance digital image authentication and steganalysis. By improving accuracy and strengthening our defence against adversarial attacks, we contribute to the development of more secure systems and ensure the integrity of digital media in various applications.

# References

Atlassian. (n.d.). Exploratory testing. Atlassian. https://www.atlassian.com/continuous-delivery/software-testing/exploratory-testing

A mathematical theory of Communication - Harvard University. (n.d.). Retrieved February 19, 2023, from

<https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf>

*Alaska2 image steganalysis*. Kaggle. (n.d.). Retrieved April 29, 2023, from

<https://www.kaggle.com/competitions/alaska2-image-steganalysis/data>

*Atlas™*. MITRE. (n.d.). Retrieved April 29, 2023, from

<https://atlas.mitre.org/techniques/AML.T0015>

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, & Yoshua Bengio. (2022). Generative Adversarial Networks. The Science of Deep Learning, 153–173.

https://doi.org/10.1017/9781108891530.013

Barni, M., Kallas, K., Nowroozi, E., & Tondi, B. (2020). CNN Detection of GAN-Synthetic Face Images based on Cross-Band Co-occurrences Analysis. 2020 IEEE International Workshop on Information Forensics and Security (WIFS).

<https://doi.org/10.1109/wifs49906.2020.9360905>

ETSI GR Sai 001 v1.1. (n.d.-a). https://www.etsi.org/deliver/etsi\_gr/SAI/001\_099/001/01.01.01\_60/gr\_SAI001v010101p.pdf

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

<https://doi.org/10.1109/cvpr.2016.90>

Hyperopt: Distributed asynchronous hyper-parameter optimization. Hyperopt Documentation. (n.d.). http://hyperopt.github.io/hyperopt/

Ivey, T. (2022, September 6). Dangerous cybersecurity assumptions: Why Adversarial Testing Matters. Search Secureworks. https://www.secureworks.com/blog/dangerous-assumptions-why-adversarial-testing-matters

Lutins, E. (2017, August 2). Ensemble Methods in Machine Learning: What are they and why use them?. Medium. https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f

Lai, Z. F., & Deng, H. F. (2018, September 12). Medical image classification based on deep features extracted by deep model and statistic feature fusion with Multilayer Perceptron‬. Computational Intelligence and Neuroscience. Retrieved February 20, 2023, from

https://www.hindawi.com/journals/cin/2018/2061516/.

*MITRE ATLAS*. ATT&CK® navigator. (n.d.). https://mitre-atlas.github.io/atlas-navigator/

Nataraj, L., Mohammed, T. M., Manjunath, B. S., Chandrasekaran, S., Flenner, A., Bappy, J. H., & Roy-Chowdhury, A. K. (2019). Detecting GAN Synthetic Fake Images using Co-occurrence Matrices. Electronic Imaging, 31(5), 532–1.

<https://doi.org/10.2352/issn.2470-1173.2019.5.mwsf-532>

Proximal policy optimization. Proximal Policy Optimization. (n.d.). https://openai.com/research/openai-baselines-ppo

V, B. S. (2012). Grey Level Co-Occurrence Matrices: Generalisation and Some New Features. International Journal of Computer Science, Engineering, and Information Technology, 2(2), 151–157.

<https://doi.org/10.5121/ijcseit.2012.2213>

Karen Simonyan, & Andrew Zisserman. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. Computer Vision and Pattern Recognition.

<https://arxiv.org/abs/1409.1556>

Reversing multilayer .net malware - cisco blogs. (n.d.). https://blogs.cisco.com/security/talos/reversing-multilayer-net-malware

Rashad, F. (2020, December 29). Adversarial auto encoder (AAE). Medium. https://medium.com/vitrox-publication/adversarial-auto-encoder-aae-a3fc86f71758#:~:text=Adversarial%20Autoencoder%20(AAE)%20is%20a,KL-divergence%20that%20VAE%20uses.

Su, J., Vargas, D. V., & Kouichi, S. (2019, October 17). One pixel attack for fooling deep neural networks. arXiv.org. https://arxiv.org/abs/1710.08864

Team, K. (n.d.). Simple. flexible. powerful. Keras. https://keras.io/

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

<https://doi.org/10.1109/cvpr.2016.308>

Shah, P. (2021, June 3). *Label smoothing - make your model less (over)confident*. Medium. Retrieved April 29, 2023, from

<https://towardsdatascience.com/label-smoothing-make-your-model-less-over-confident-b12ea6f81a9a#:~:text=Label%20smoothing%20is%20a%20form,attempt%20to%20penalize%20overconfident%20outputs>.

*Thomas J. Watson college of engineering and applied science ... (n.d.-c).* http://ws.binghamton.edu/fridrich/research/SRNet.pdf

Thornbury, B. (2021, February 2). Can reinforcement learning be used for classification? BRTHOR. <https://www.brthor.com/blog/ai/reinforcement-learning/can-reinforcement-learning-be-used-for-classification/>

OpenAI. (n.d.). https://openai.com/

Xu, W., Evans, D., & Qi, Y. (2017, December 5). Feature squeezing: Detecting adversarial examples in deep neural networks. arXiv.org. <https://arxiv.org/abs/1704.01155>

Appendix

**PPO**

PPO is a reinforcement learning algorithm used in the field of artificial intelligence. PPO is designed to train agents or models to make sequential decisions in an environment, maximizing their performance over time. It belongs to the family of policy gradient algorithms and is known for its stability and sample efficiency.

PPO works by iteratively collecting data through interactions with an environment and using that data to update the agent's policy. It employs a technique called "clipped surrogate objective" to limit the policy updates and ensure more stable learning. This technique prevents large policy changes that can lead to poor performance or instability.

In terms of accuracy PPO fails to outperform that of our baseline EfficientNetB2 CNN model, averaging at around 51% accuracy as compared to our CNN models at 56% . The inputs for the models are 256 by 256 by 3.

**Some reason it might not have worked.**

Reinforcement Learning (RL) is not commonly used for image classification tasks because RL is better suited for problems where the optimal actions need to be determined through trial and error in a dynamic environment. Image classification, on the other hand, is a well-defined supervised learning problem where the goal is to assign predefined class labels to input images based on their features.

In image classification, a large amount of labelled training data is available, and the objective is to learn a mapping between input images and their corresponding class labels. This task is typically addressed using supervised learning approaches, such as neural networks, where the model learns to recognize patterns and features in the images that are indicative of specific classes. This is different from the RL setting, where the agent interacts with an environment, receives feedback in the form of rewards or penalties, and learns to make sequential decisions to maximize cumulative rewards.

While RL can be applied to image classification in certain scenarios, such as active learning or reinforcement learning-based data augmentation, it is not the standard approach due to several challenges. RL requires significant exploration and interaction with the environment, which may not be feasible or efficient for large-scale image datasets. Additionally, the image classification problem is typically well-defined and can be effectively addressed using supervised learning algorithms, which have been extensively studied and optimized for this task.

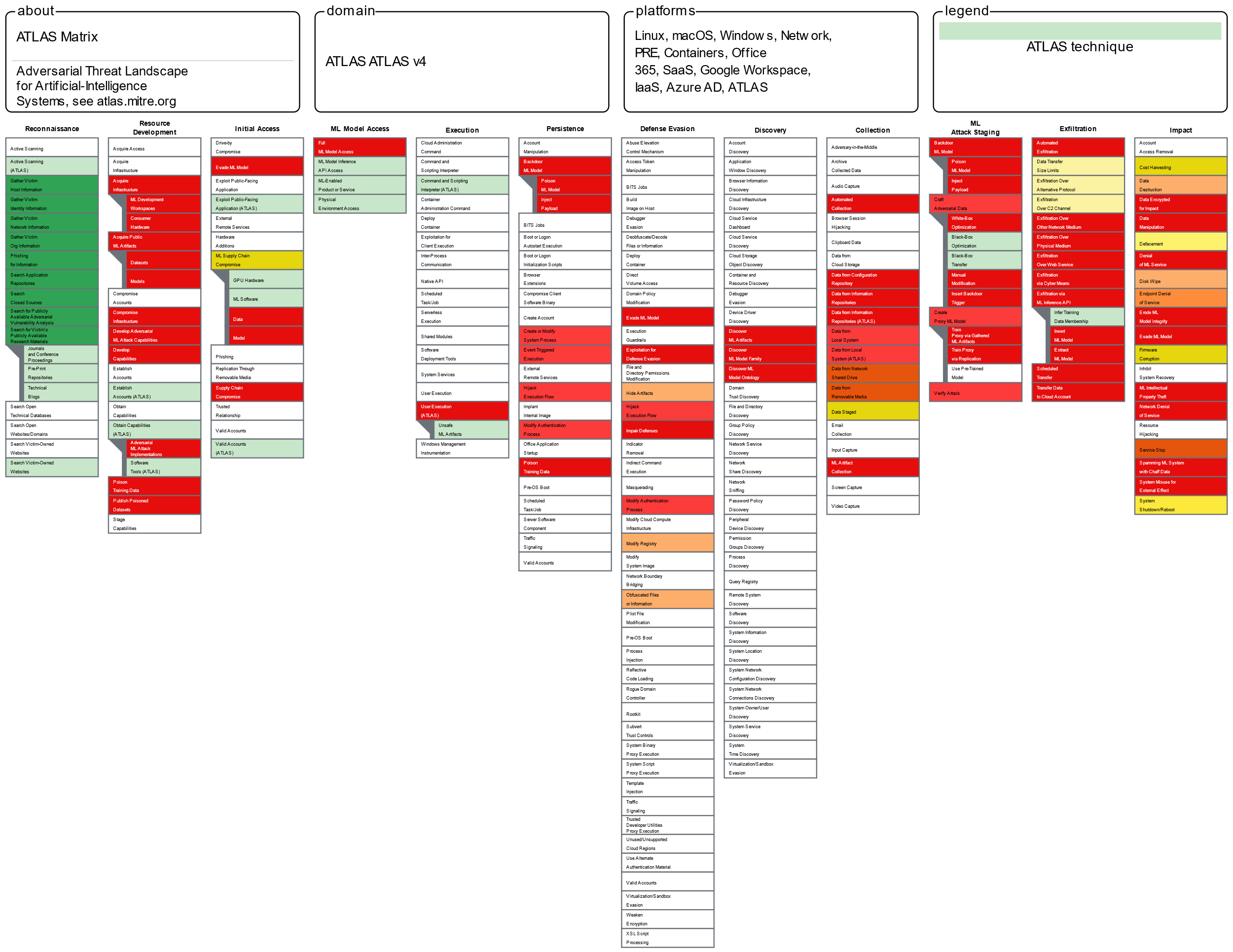
In summary, RL is not commonly used for image classification because supervised learning methods, such as neural networks, are more suitable and effective for this specific task, given the availability of labelled training data and the well-defined nature of the problem.

**CNN: EfficientNetB2**

Standard EfficientNetB2 with 512 by 512 images, averaged at around 56% accuracy as defined by the competition metric given by the organizer of the event. Despite the larger batch size of 32, larger dataset and the larger resolution, the model still fails to beat the hybrid model that we have created above.

White Box

Colour coding represents two things: Probability & Severity. Red can mean High severity, high probability, Green Can mean Low severity , but high probability it will happen



Under **ML Model Access**, edit for white box , physical access should be bright red.

Black box 