PROJECT REPORT

Visual Cryptography & Defense against Adversarial attacks.

Course: Cyber Security (CSE4003)

BY

Siddharth Das	18BIT0379
Shruti Varsha Venkatraman	18BIT0405
Harida P K	18BIT0411
Yadhu Anand K J	18BIT0373

SLOT: C1+ TC1

NAME OF FACULTY: NAVAMANI T

ABSTRACT

In this quick world of networking, secure image transmission of information whether or not the image having some information or some text, is one of the real focus for security.
Visual Cryptography could be a special secret encryption technique to cover info in pictures in such the simplest way that it can be decrypted by the human sensory system. The advantage of the visual secret sharing scheme is in its decryption method where with none complicated cryptographic computation encrypted knowledge is decrypted using Human sensory system (HVS).

We want to question whether neural networks can learn to use secret keys to protect data from other neural networks. Specifically, we focus on ensuring confidentiality properties in a multi network system i.e. we would like to show those properties in terms of an adversary. Thus, a system may consist of neural networks named A and B, and we aim to restrict what a third neural network an adversary named E learns from eavesdropping on the communication between two individuals. We want to train end-to-end, adversarial
There is more to cryptography than encryption and decryption. In this spirit, further work may consider other tasks, for example visual cryptography combined with access control and explore adversarial attacks by animating the classification scores and CAM heatmap visualization by tuning in the strength of integration checks applied in real-time. While it seems improbable that neural networks would become great at cryptanalysis, they may be quite effective in making sense of metadata and in traffic analysis.

MOTIVATION

The importance of secured network with integrity is increasing day by day due to various malicious attacks like spam or malware. But there is a very little or no knowledge about the adversarial learning which tells about different adversarial attacks.

In recent years, artificial intelligence technologies have been widely used in computer vision, natural language processing, automatic driving, and other fields. However, artificial intelligence systems are vulnerable to adversarial attacks, which limit the applications of artificial intelligence (AI) technologies in key security fields. Therefore, improving the robustness of AI systems against adversarial attacks has played an increasingly important role in the further development of AI. According to the target model's different stages where the adversarial attack occurred, this paper expounds the adversarial attack methods in the training stage and testing stage respectively. Then, we sort out the applications of adversarial attack technologies in computer vision, natural language processing, cyberspace security, and the physical world. Finally, we describe the existing adversarial defence methods respectively in three main categories, i.e., modifying data, modifying models and using auxiliary tools.

Main motivation to carry out this work is to analyse various threat models and attacks in the network which encrypting the image and the text and to provide a suitable countermeasure for the problem as any network without security is of a total waste which should be governed.

We are henceforth carrying out this work like other surveys and recent researches but without restricting to a certain application. There are a lot of attacks aiming on learning the algorithm and the models which should be prevented. Since the security for any model is based on the adversarial capability, we should classify the threat models to strengthen the adversary.

INTRODUCTION

Visual cryptography is a cryptographic procedure which permits visual data (Picture, content, and so on) to be encoded so that the decryption can be performed by the human visual framework without the guide of our personal computers. Picture is a mixed media part detected by human. The smallest component of a computerized picture is pixel. In a 32-bit computerized image every pixel comprises of 32 bits as in Figure 1, further divided into four sections, to be specific Alpha, Red, Green and Blue; each with 8 bits. Alpha is a part which specifically represents degree of transparency. On the off chance that all bits of Alpha part are '0', at that point the picture is completely transparent. This is spoken to in the accompanying figure.

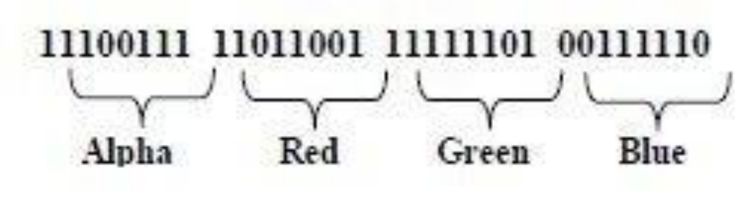


***Figure 1**. Structure of a 32-bit pixel*

Human visual System goes exactly same as an OR function, as depicted in Figure 2. On the off chance that two transparent items are stacked together, the last heap of articles will be transparent as well. However, on the off chance that any of them is not transparent, at that point the last heap of items will be non-transparent. Like OR, 0 OR 0 = 0, thinking about 0 as transparent and 1 OR 0 = 1, 0 OR 1 =1, 1 OR 1=1, considering 1 as non-transparent.

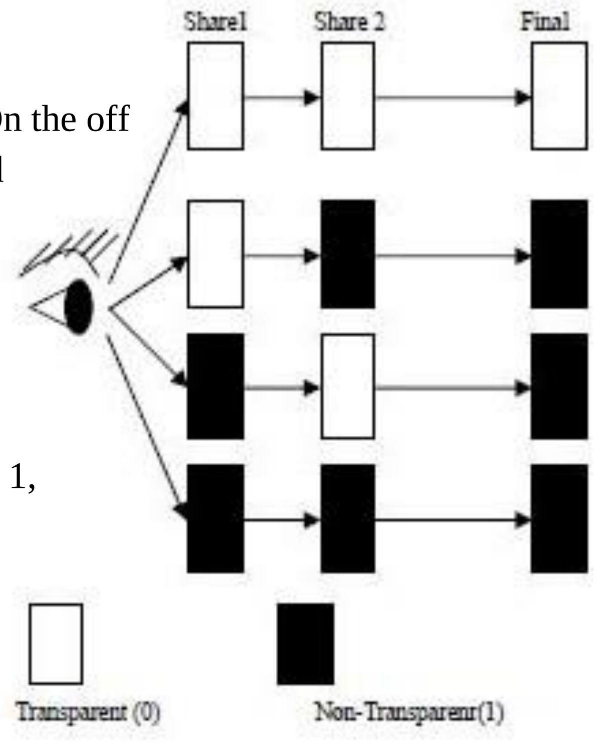


***Figure 2**. Human Visual system as OR function*

Several machine learning models, including neural networks, consistently misclassify adversarial examples—inputs formed by applying small but intentionally worst-case perturbations to examples from the dataset, such that the perturbed input results in the model outputting an incorrect answer with high confidence. Early attempts at explaining this phenomenon focused on nonlinearity and overfitting. We argue instead that the primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature.

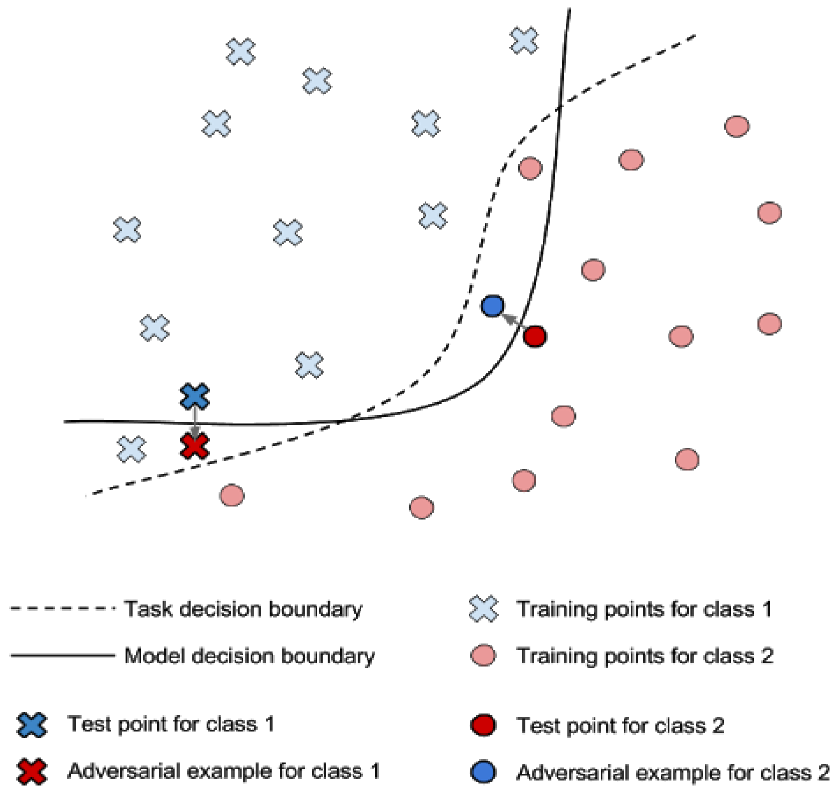

Task decision boundary
Model decision boundary
Training points for class 1
Training points for class 2
Test point for class 1
Adversarial example for class 1
Test point for class 2
Adversarial example for class 2

Figure 3. *Understanding the Gradient Descent function by training data points and modelling the decision boundary between separate classes*

The cause of these adversarial examples was a mystery, and speculative explanations have suggested it is due to extreme nonlinearity of deep neural networks, perhaps combined with insufficient model averaging and insufficient regularization of the purely supervised learning problem. We show that these speculative hypotheses are unnecessary. Linear behavior in high-dimensional spaces is sufficient to cause adversarial examples. This view enables us to design a fast method of generating adversarial examples that makes adversarial training practical.

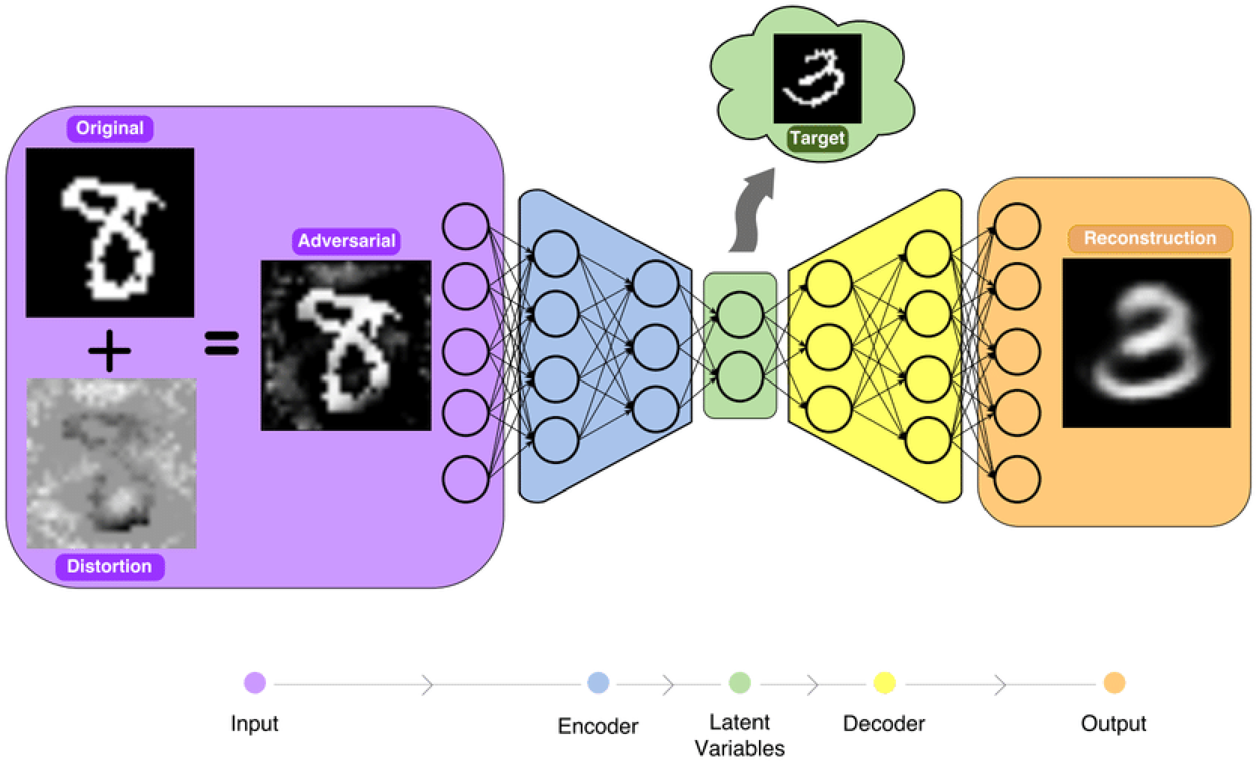


Figure 4. *Functional Structure of Adversarial Attacks*

Generic regularization strategies such as dropout, pretraining, and model averaging do not confer a significant reduction in a model's vulnerability to adversarial examples, but changing to nonlinear model families such as RBF networks can do so.

Our explanation suggests a fundamental tension between designing models that are easy to train due to their linearity and designing models that use nonlinear effects to resist adversarial perturbation, as showed in Figure 4. In the long run, it may be possible to escape this tradeoff by designing more powerful optimization methods that can successfully train more nonlinear models.

LITERATURE SURVEY

[1] To explain and understand the working of adversarial neural cryptography, the proposed system did not prescribe specific cryptographic algorithms to these neural networks but factually, it showed how to perform forms of encryption and decryption successfully denying the third-party any information about the message. However, the system did not show any solution learned by the networks in their work.

[2] A related work explained the importance of cryptography and highlighted severe security issues of the communication channel using private key cryptography. It has been observed that the proposed method generated a reliable connection and increases confidentiality of messages.

[3] In a renowned work focused on cryptography and security in communicators, wireless and IP network security, as well as optical network security, quantum cryptography and quantum-key distribution processes specific to optical networks is discussed.

[5] The security of any machine learning model is measured on the basis of adversarial goals and capabilities. Here, the threat models are taxonomized in machine learning systems keeping in mind the strength of the adversary. The identification of threat surface of systems built on machine learning models is to identify how and where an adversary may attempt to subvert the system under attack.

[4] [6] Samangouei et al. proposed a mechanism to leverage the power of Generative Adversarial Networks to reduce the efficiency of adversarial perturbations, which works both for white box and black box attacks. In a typical GAN, a generative model which emulated the data distribution, and a discriminative model that differentiates between original input and perturbed input, are trained simultaneously. Although Defense-GAN was quite effective against adversarial attacks, its success relied mostly on the expressiveness and generative power of the GAN. Moreover, the performance of Defense-GAN can significantly degrade if not properly trained.

[7] Despite their high accuracy and performance, machine learning algorithms have been found to be vulnerable to subtle perturbations that can have catastrophic consequences in security related environments. Since the threat becomes more grave when the applications operate in adversarial environment, it has become an immediate necessity to devise robust learning techniques resilient to adversarial attacks. A number of research papers on adversarial attacks as well as their countermeasures has surfaced since Szegedy et al demonstrated the vulnerability of machine learning algorithms. In this paper we have tried to explore some of the well-known attacks and proposed defense strategies. We have also tried to provide a taxonomy on topics related to adversarial learning. After the review we can conclude that adversarial learning is a real threat to application of machine learning in physical world. Even though there exist certain countermeasures, but none of them can act as a solution for all challenges as it remains as an open problem for the machine learning community to come up with a considerably robust design against these adversarial attack

[8] Megha B.Goel , Vaishali B.Bhagat , Veena K. Katankar has proposed their work on Authentication Framework using Visual Cryptography. Since in the existing system there is an need to remember password for different websites in this proposed system instead of password they use visual cryptography by hiding the answer of secret question in the image.Valid and invalid users are easily identified using this system and blocks the unauthorised users to log-in.

[9] Reem Ibrahim Hasan and Huda Adil Abdulghafoor has proposed that Cipher Secret Image using Hybrid Visual Cryptography uses many statistical attacking measures to cover the encrypt images.They found that the result is obtained with minimum computational time, storage space and recovered the images based on chaos diffusion and a share masking structure. This method provides better results in resistance of data loss and cropping effect.The flat image histogram is used in the encryption to improve the system.

[10] Megha R.Chaudhari, Neha D.Chaudhari, Shubhangi S.Kanade, Sumedh G.Bhadre, Dhiraj D.Bhagat has proposed about the Phishing Attack Prevention Using Visual Cryptography. They have proved about approaches that provides authentication of website by using OTP on e-mail id.User can log in only if the OTP is matched.This method is based on Anti-Phishing Captcha validation using visual cryptography and prevents phishing attacks. It prevents security and hence to overcome this attack by sharing the user share to server database.

[11]Sankar Das, Sandipan Chowdary and Dibya Chakraborty has proposed the System of Visual Cryptography using Three Independent Shares in Color Images.This method encrypts the secret message in three shares as they are fully secure and no information can be retrieved and info can be accessed only when three shares are used together. As the quality of the decrypted message gets degraded it can be improved by removing the unwanted colors from them.

[12] Deng yuqiao, song Ge has proposed a method on A verifiable Visual Cryptography scheme using Neural Networks using pi-sigma neural networks and is highly secure and is one of the effective verification method used. This method needs small weights compared to others. This method also results in lower communication rate and is more flexible.When we input the shares into the input layer the output image is recovered as a partial image.Again after connecting all the sub-mages and decrypting them we get back the original image.If the user is new then his/her share will be generated randomly.The redundancy id small related to the size of the image.

[13] This paper introduces a threshold visual cryptography scheme, denoted as (k,n)-VCS as short. This technique can encrypt the target image into n share images. Later, stacking the any k share images can reveal the encrypted image. If the number of share images are less than k, no content is leaked. Building a normal form hyper star access structure VCS and building a general access structure VCS from several small normal form hyper star access structure VCSs are explained in the paper.

[14] In secret sharing technique, the image is broken down to multiple unreadable format types. Thus the image wont be revealed until it is combined again using some mathematical computation. In the previous schemes introduced, there were flaws regarding security and loss of image data during reconstruction causing the resultant image differing from the actual one. In this paper, CMY colour model is implemented using (n,n-1) secret sharing scheme. The model introduced in this paper is found to be better than the existing schemes and the RGB colour space.

[15] This paper discusses about a new cryptography scheme which will modify the visualization of images. This wont affect the metadata of the image. The intensity of each of three colour layers (RGB) is changed using the RC4 algorithm. The fuzziness of the image is dependant on the number of colour layers included. The resulting encrypted image will be noisy and unrecognizable. Since RC4 has three level options, this cryptographic method is found to provide high security.

[16] In this paper the authors researched about a lot of cryptography schemes such as “pixel expansion” and “relative difference”, analyzed them and put down their respective advantages and disadvantages. After more analysis of various methods of reconstruction and visual cryptography, they characterized and examined visual cryptography plans for dark level pictures. We gave a fundamental and adequate condition for such plans to exist. They demonstrated the optimality of .k; k;m;g/ - GVCS. An intriguing open issue which merits further examination is the encoding of dark level pictures for various models of VCS.

[17] In contrast to most investigations of visual cryptography, which focus on high contrast pictures, this paper misuses the strategies of halftone innovation and shading disintegration to build three techniques that can manage both dark level and shading visual cryptography. In light of the hypothesis of shading decay, each shading on a shading picture can be disintegrated into three essential hues: C, M, and Y. With the halftone innovation, we can change a dark level picture into a double one reasonable for creating visual cryptography. As the customary plans for highly contrasting visual cryptography, their techniques grow each pixel of a shading mystery picture into a 2×2 square in the sharing pictures and keep two shading and two straightforward pixels in the square. Their investigation is the first one that misuses the color decomposition and halftone innovation to create visual cryptograms for both gray-level and shading pictures. Their techniques can likewise be handily applied to the plans created in past examinations, such as the ‘t’ out of ‘n’ edge conspire and the all-inclusive plans for visual cryptography. Despite the fact that their paper concerns shading figures, the standards can be surely known by utilizing high contrast ones. Since shading plates are costly, the figures in this paper are imprinted in highly contrasting organization.

[18] In this paper, they have proposed another strategy which joins two significant parts of picture sharing: Visual Cryptography (VC) and polynomial-style sharing (PSS). In the unraveling issue, this new technique is more adaptable than applying VC or PSS autonomously, since their strategy gives a "two-choices" deciphering.
To clarify why utilizing huge squares typically suggests a superior PSNR, we may examine the connection between the picture pressure proportion and the size of the square. When all is said in done, the compacted picture S(comp) will have a superior PSNR if the picture pressure proportion (a worth at least 1) is littler (more like 1).

[19] Broadened Visual Cryptography is a kind of cryptography which encodes various pictures in the manner that when the pictures on transparencies are stacked together, the shrouded message shows up without any unique pictures. The decoding is done legitimately by the human visual framework with no exceptional cryptographic estimations. This paper presents a framework which accepts three pictures as an information and creates two pictures which relate to two of the three info pictures. The third picture is remade by printing the two yield pictures onto transparencies and stacking them together. While the past explores essentially handle just paired pictures, this paper builds up the all-inclusive visual cryptography plot appropriate for common pictures. By and large, visual cryptography experiences the decay of the picture quality. This paper additionally depicts the strategy to improve the nature of the yield pictures. The compromise between the picture quality and the security are examined and surveyed by watching the real consequences of this strategy. Moreover, the streamlining of the picture quality is talked about.

[20] In this paper, an overall system of halftone visual cryptography is proposed. Applying the rich hypothesis of blue clamor halftoning into the development system of ordinary VC, the proposed technique produces outwardly satisfying halftone shares conveying huge visual data. The got visual quality is better than that accomplished by some other accessible VC strategy known to date. The new technique can be comprehensively utilized in various visual mystery sharing applications which require excellent visual pictures, for example, watermarking, electronic money, and so forth.

PROBLEM STATEMENT:

The real problem here is that our machine learning models exhibit unpredictable and overly confident behavior outside of the training distribution. Adversarial examples are just a subset of this broader problem. We would like our models to be able to exhibit appropriately low confidence when they're operating in regions they have not seen before.

Many of the most important problems still remain open, both in terms of theory and in terms of applications. We do not yet know whether defending against adversarial examples is a theoretically hopeless endeavor or if an optimal strategy would give the defender an upper ground. On the applied side, no one has yet designed a truly powerful defense algorithm that can resist a wide variety of adversarial example attack algorithms.

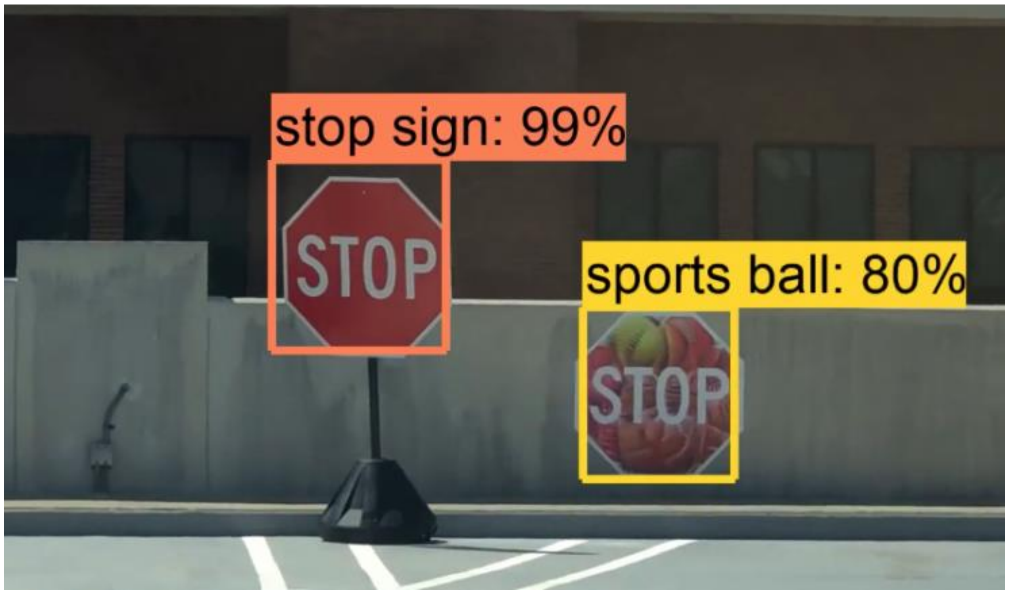


Figure 5*. Misclassification of data by image detection system used in self-driving cars*

If nothing else, the topic of adversarial examples gives us an insight into what most researchers have been saying for a while. Despite the breakthroughs, we are still in the infancy of machine learning and still have a long way to go here. Machine Learning is just another tool, susceptible to adversarial attacks which can have huge implications in a world where we trust them with human lives via self-driving cars and other automation, as shown in Figure 5.

PROPOSED SYSTEM & ITS APPLICATIONS:

- **Java Cryptographic Architecture:**

Java security technology includes a large set of APIs, tools, and implementations of commonly used security algorithms, mechanisms, and protocols. The Java security APIs span a wide range of areas, including cryptography, public key infrastructure, secure communication, authentication, and access control. Java security technology provides the developer with a comprehensive security framework for writing applications, and also provides the user or administrator with a set of tools to securely manage applications.

Encryption and decryption are fundamental requirements of every secure-aware application; therefore, the Java platform provides strong support for encryption and decryption through its Java Cryptographic Extension (JCE) framework (Figure 6) which implements the standard cryptographic algorithms such as AES, DES, DESede and RSA.

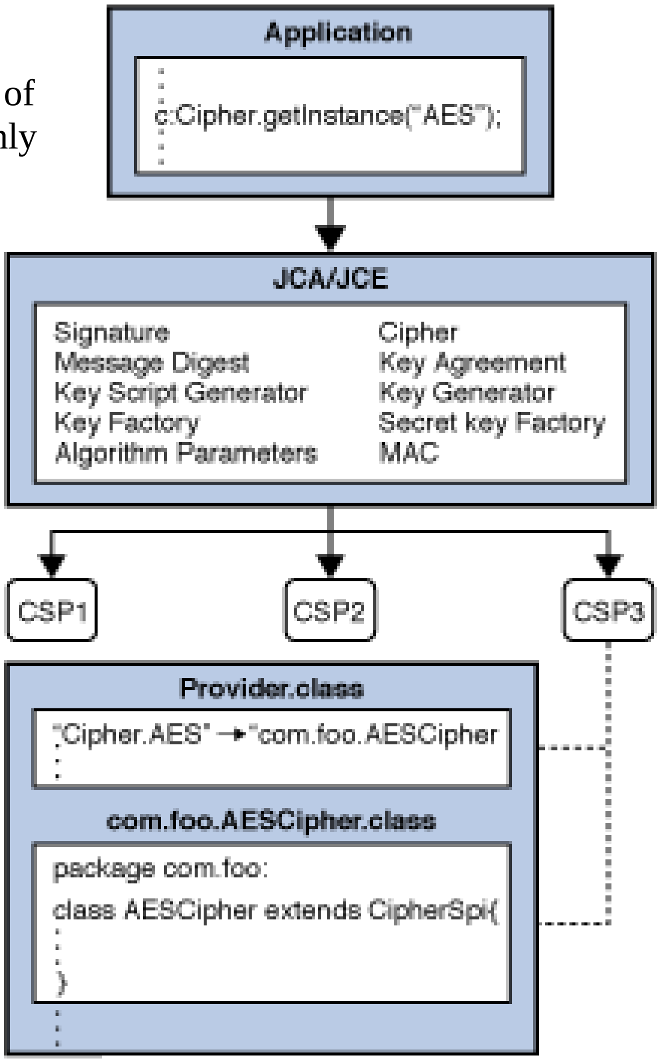


Figure 6. *JCE Framework and System Architecture*

General steps to encrypt/decrypt a file in Java:

- Create a **Key** from a given byte array for a given algorithm.
- Get an instance of **Cipher** class for a given algorithm transformation. See document of the Cipher class for more information regarding supported algorithms and transformations.
- Initialize the **Cipher** with an appropriate mode (encrypt or decrypt) and the given **Key**.
- Invoke **doFinal(input_bytes)** method of the **Cipher** class to perform encryption or decryption on the **input_bytes**, which returns an encrypted or decrypted byte array.
- Read an input file to a byte array and write the encrypted/decrypted byte array to an output file accordingly.

- **React.js and Tensorflow.js: Adversarial Attacks Visualization**

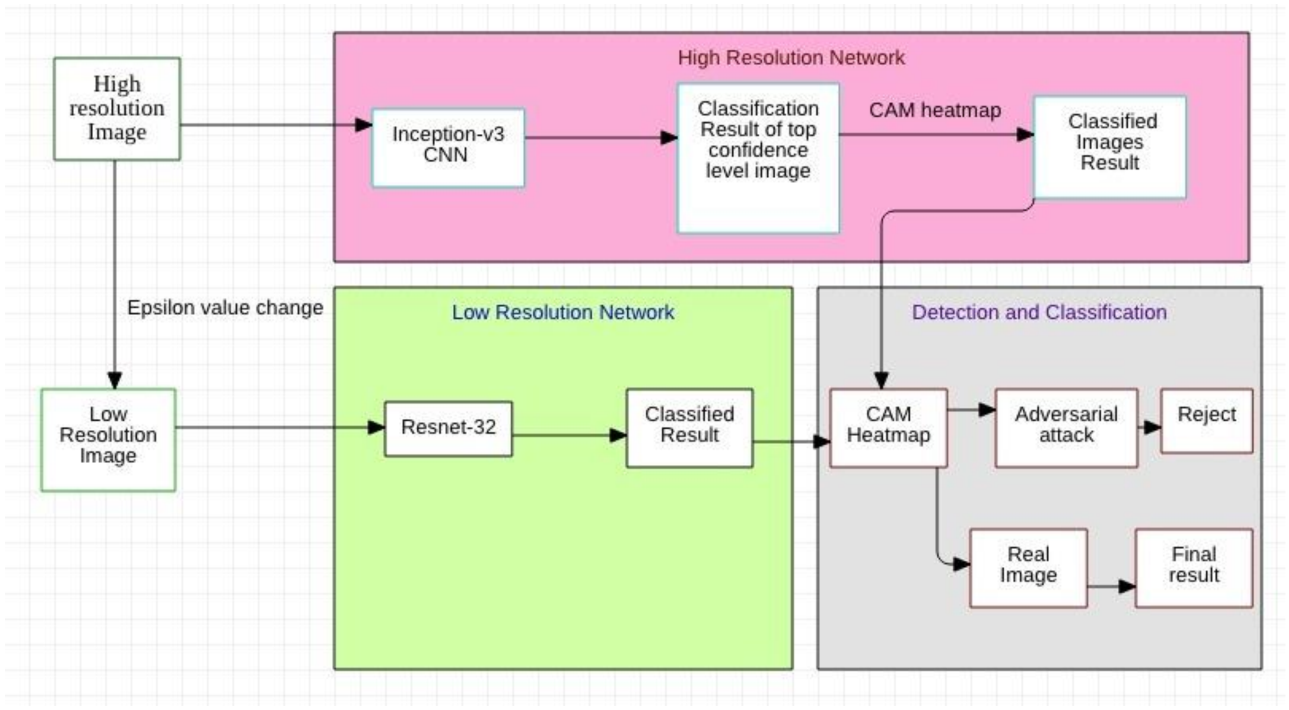


Figure 6.1 *System architecture of the proposed system*

When you see a corrupted image of, let's say, a panda - you recognize it. Probably by the colorful noise. But for the machine it's not a noisy photo of a panda, it's a chihuahua. And it's so sure about it, that it doesn't make sense to question its own decisions.

As you can see in Figure 6.1, using these Web-GL accelerated, browser-based JavaScript libraries, it can deploy and train CAM (Class Activation Mapping), a technique that visualizes the regions of input that are "important" for predictions from these models - or visual explanations.

It lets you explore adversarial attacks by animating the classification scores and CAM heatmap visualization as you tune the strength of perturbation applied in real-time. CAM uses the class-specific gradient information flowing into the final convolutional layer of a CNN to produce a coarse localization map of the discriminative regions in the image.

Why does this matter? (Real-world Applications)

- Computer vision is used everywhere!
- Facial Recognition
- Self-Driving Cars (change speed limit sign)
- Biometric Recognition
- Text Applications too! (Think CNNs for Sentiment Analysis or Text classification)
- Ad-blockers can incorrectly classify.
- Spam Classifiers can incorrectly identify malicious mails

Although it's hard to defend adversarial examples, limited effort has been made on practical adversarial learning. The reason is that the attacker usually can only change the input on a limited degree to the system, which accounts for attacker's little access to the system device.

However, it is easy to imagine that it would be very dangerous if real world systems can be compromised by attackers with adversarial examples, if only the systems employ ML models, especially when the attacker didn't break into the system.

For instance, attackers may freely pass face authentication-based entrance access doors if the face authentication models were compromised. Autonomous vehicles may overspeed if the road sign recognition models inside were compromised.

An adversarial example generated by the aforementioned methods cannot be directly used to attack a real-world system, because of their utterly different threat models. In this section, we compare their models in detail and highlight the model used for theoretical and practical adversarial example attacks.

Threat model for theoretical attacks

White box attacks: The attack is thought to be successful when the output of the model is indeed the target instead of the attacking object, regardless of how the perturbation is added into the image of attacking object. Therefore, an attacker can generate perturbations by minimizing the loss over the model between the target (the cat) and the sum of the attacking object (the dog) and the perturbation.

Black box attacks: In some cases, the model is not transparently exposed to attackers, in which case the attacker can only query the model with images input and get the result returned. In this setting, the attacker still only needs to work out a perturbation for the target, such that the model outputs the target.

Untargeted attacks: For some scenarios, attackers do not have a specific target that must be output by the victim model. Instead, they only want the output is not correct, i.e., output whatever rather than dogs. This kind of attack was called untargeted attacks. In this case, attackers need only maximize the distance between the perturbed output and the authentic label.

Practical adversarial example attacks threat model

For this scenario, attackers even face more challenges: the system may impose detection modules between the model and the front-end input camera to detect potential attacks. For instance, in a face authentication system, there exists liveness detection modules to examine if the object in front of the camera is a live human being or a printed photo. Considering those mechanisms, attackers usually firstly place an object that can pass the detection and then apply small perturbations that won't fail the front-end examinations.

Similar to theoretical attacks, practical attacks may also differ in black box and white box settings. usually, researchers assume white box settings first where the model structure and weights are known to attackers and come up with supporting black box extension to ease the assumption.

For instance, attackers may firstly train a substitutional model by querying the black box. Then they can generate adversarial examples for the substitutional mode, which by expectation will also be valid for the model inside the black box

WHY DEFENDING NEURAL NETWORK IS SO HARD?

Let’s try to develop an intuition behind what’s going on here. Most of the time, machine learning models work very well but only work on a very small amount of all the many possible inputs they might encounter. In a high-dimensional space, a very small perturbation in each individual input pixel can be enough to cause a dramatic change in the dot products down the neural network. So, it’s very easy to nudge the input image to a point in high-dimensional space that our networks have never seen before. This is a key point to keep in mind: the high dimensional spaces are so sparse that most of our training data is concentrated in a very small region known as the *manifold*. Although our neural networks are nonlinear by definition, the most common activation function we use to train them, the Rectifier Linear Unit, or ReLu, is linear for inputs greater than 0.

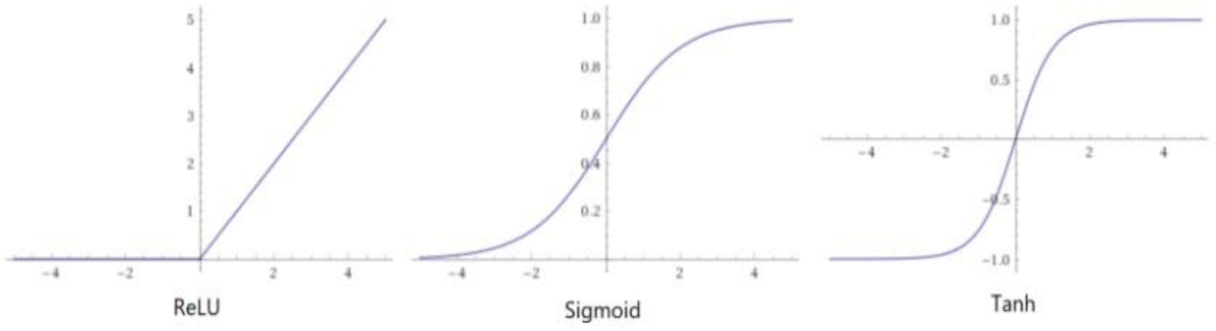


Figure 7. *Comparing different activation functions based on the easiness of data trainability.*

ReLu, as shown in Figure 7, became the preferred activations function due to its ease of trainability. Compared to sigmoid or tanh activation functions that simply saturate to a capped value at high activations and thus have gradients getting “stuck” very close to 0, the ReLu has a non-zero gradient everywhere to the right of 0, making it much more stable and faster to train. But, that also makes it possible to push the ReLu activation function to arbitrarily high values.

Looking at this trade-off between trainability and robustness to adversarial attacks, we can conclude that the neural network models we have been using are intrinsically flawed. Ease of optimization has come at the cost of models that are easily misled.

DEFENSE AGAINST ADVERSARIAL ATTACKS

- **Adversarial training**:

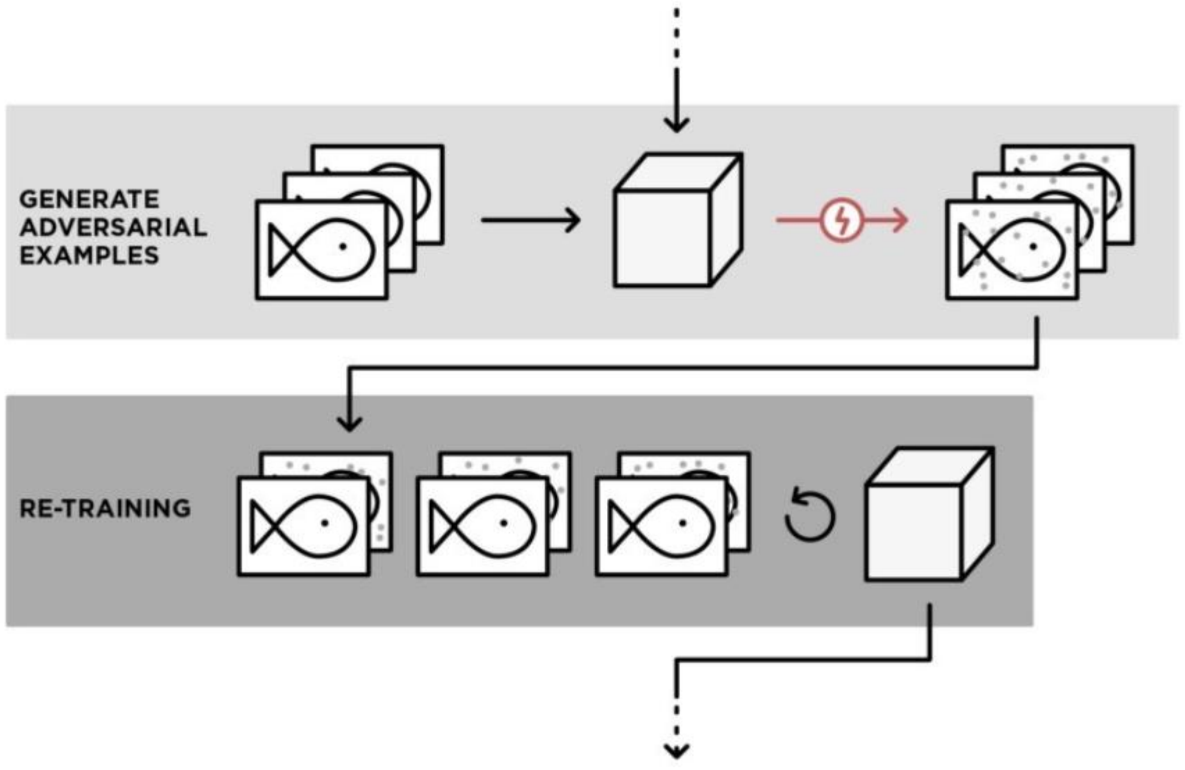


Figure 8. *General training process of adversarial examples*

This is a brute force solution where we simply generate a lot of adversarial examples and explicitly train the model not to be fooled by each of them. The aim for adversarial training is to proactively generate adversarial examples as part of the training procedure. The model as shown in Figure 8 is then trained to assign the same label to the adversarial example as to the original example (i.e. generate a cat, perturb that image to be misclassified, label it a cat regardless)

Thus, Adversarial training is a standard brute force approach where the defender simply generates a lot of adversarial examples and augments these perturbed data while training the targeted model, which is useful only on adversarial examples which are crafted on the original model.

Defensive distillation:

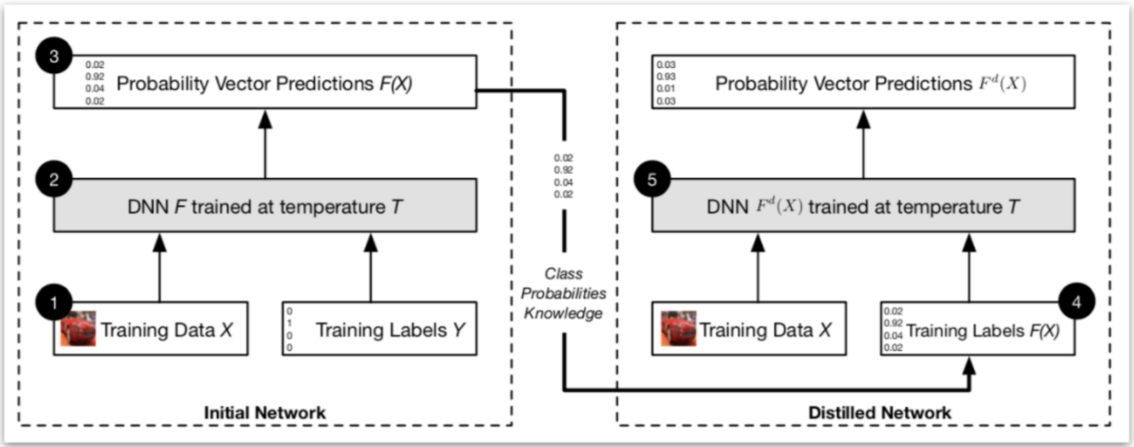


Figure 9. *An overview of defense mechanism based on a transfer of knowledge contained in probability vectors through distillation*

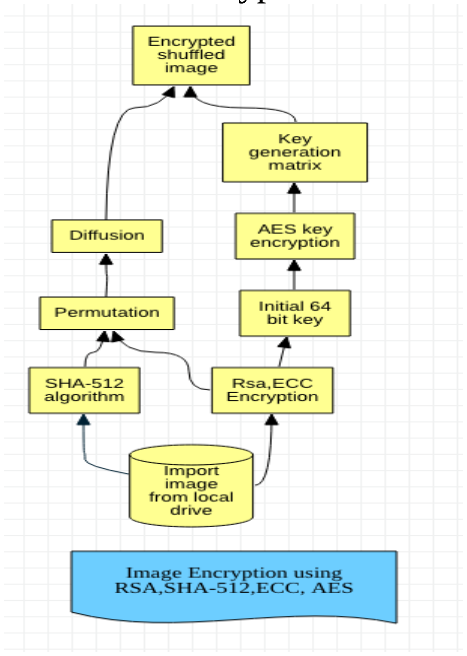
Defensive distillation is an adversarial training technique that adds flexibility to an algorithm's classification process so the model is less susceptible to exploitation. In distillation training, as shown in Figure 9, one model is trained to predict the output probabilities of another model that was trained on an earlier, baseline standard to emphasize accuracy.

The first model is trained with "hard" labels to achieve maximum accuracy, for example requiring a 100% probability threshold that the biometric scan matches the fingerprint on record. The problem is, the algorithm doesn't match every single pixel, since that would take too much time. If and when an attacker learns what features and parameters the system is scanning for, the scammer can send a fake fingerprint image with just a handful of the right pixels that meet the system's programming, which generates a false positive match.

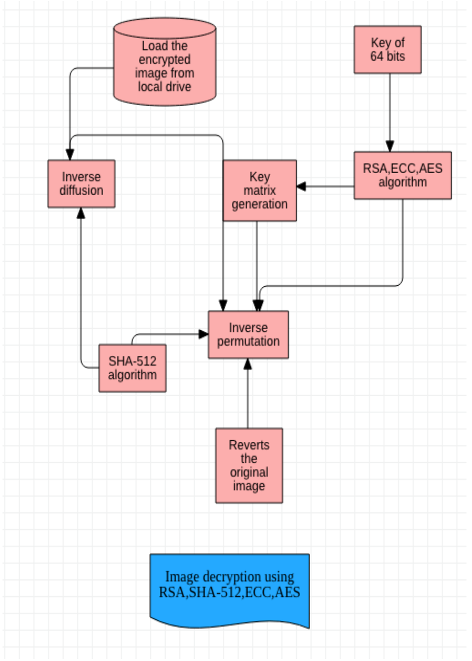
The first model then provides "soft" labels with a 95% probability that a fingerprint matches the biometric scan on record. This uncertainty is used to train the second model to act as an additional filter. Since now there's an element of randomness to gaining a perfect match, the second or "distilled" algorithm is far more robust and can spot spoofing attempts easier. It's now far more difficult for a scammer to "game the system" and artificially create a perfect match for both algorithms by just mimicking the first model's training scheme.

MODULAR DESIGN

Module 1: Encryption



Module 2: Decryption


Load the encrypted image from local drive
Key of 64 bits
Inverse diffusion
Key matrix generation
RSA,ECC,AES algorithm
Inverse permutation
SHA-512 algorithm
Reverts the original image
Image decryption using RSA,SHA-512,ECC,AES

Module 3: Adversarial Attacks Visualization

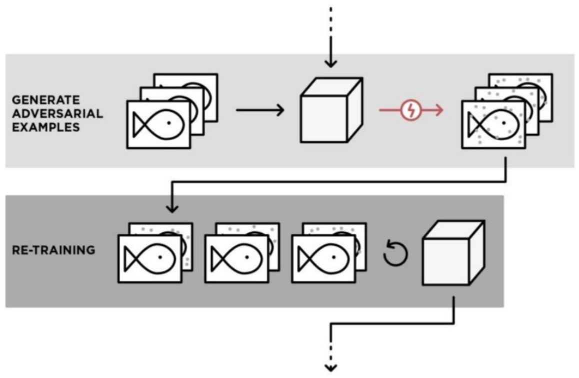


Figure 8. *General training process of adversarial examples*

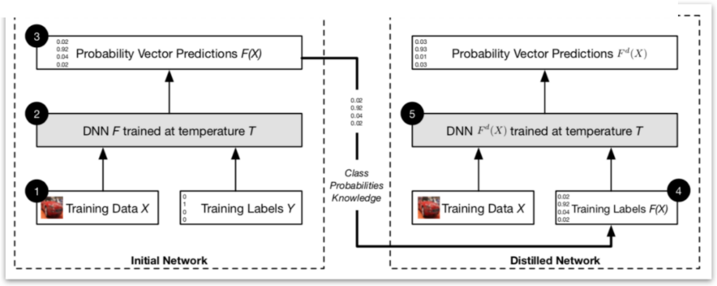

Probability Vector Predictions F(X)
DNN F trained at temperature T
Training Data X
Training Labels Y
Initial Network
Class Probabilities Knowledge
Probability Vector Predictions $F^d(X)$
DNN $F^d(X)$ trained at temperature T
Training Data X
Training Labels F(X)
Distilled Network

Figure 9. *An overview of defense mechanism based on a transfer of knowledge contained in probability vectors through distillation*

NOVELTY OF THE PROPOSED SYSTEM:

Proposed Defense Strategy:

Adversarial Training and Defensive distillation through amalgamation of random transforms culminated in the epsilon toggle.

These specialized algorithms like defensive distillation and adversarial training can easily be broken by giving more computational firepower to the attacker.

In the game of security, both attacker and defender rely on gradient descent to learn. The gradient gives the defender information about how to improve the model; the same gradient tells the attacker how to trick the model. Gradient masking occurs when the defender attempts to hide the gradient from the attacker, often by using a non-differentiable transform or technique like a decision tree. If you can't differentiate, there is no gradient for the attacker to use. A particular type of gradient masking, known as shattered gradients, makes the defense gradient either non-differentiable, non-existent, or just wrong.

But the gradient masking is not fool-proof.

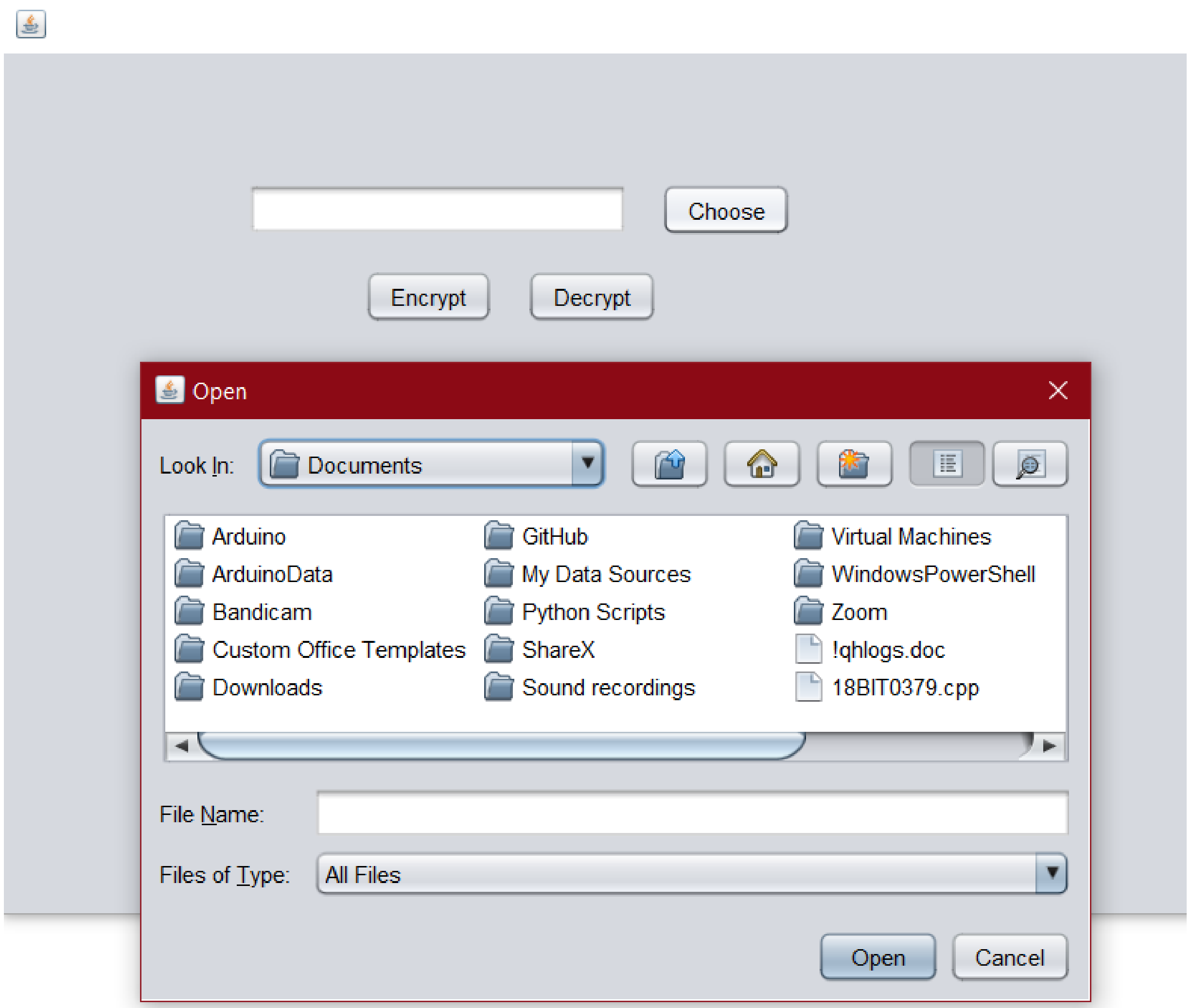
The attackers can also recreate the network by training the adversarial data of the pre-defined gradient descent function assigned by the defender and create a model of observations to classify the masked perturbations and manipulate the changes in CAM heatmap to generate a sturdy attack.

So, we can stop the manipulation by injecting random transforms like color precision reduction, noise injection etc. While the randomness hurts the accuracy when not under attack, it gives more robustness in the worst-case scenario of a strong and powerful adversary.

So, to maintain the accuracy, we have embedded some random transforms in the epsilon toggle. The attacker now has the much more challenging task of finding a single perturbation that could survive all of the possible transformations.

IMPLEMENTATION

Our project is about Visual Cryptography and Defense against Adversarial attacks.
First, we will present a scenario-based understanding of the adversarial attack through visual cryptography. We will encrypt and decrypt some images using a Java platform.
Encryption and decryption are fundamental requirements of every secure-aware application; therefore, the Java platform provides strong support for encryption and decryption through its Java Cryptographic Extension (JCE) framework which implements the standard cryptographic algorithms such as AES, DES, DESede and RSA.



Look In: Documents

Files of Type: All Files

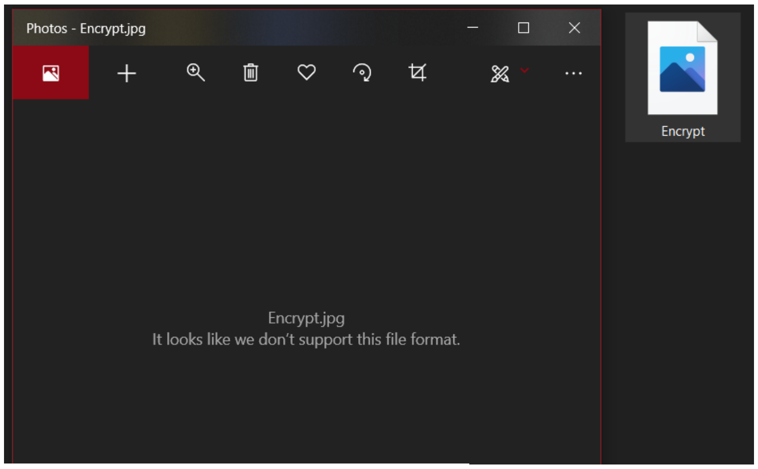
Figure 10. *Java platform for image cryptography*

SAMPLE CODE:

```
package crypto;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class ImageCrypto extends javax.swing.JFrame {

    public ImageCrypto() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    private void initComponents() {

        file_path = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jButton1.setText("Choose");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jButton2.setText("Encrypt");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jButton3.setText("Decrypt");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(130, 130, 130)
                        .addComponent(file_path, javax.swing.GroupLayout.PREFERRED_SIZE, 201, jav
ax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jButton1))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(192, 192, 192)
                        .addComponent(jButton2)
                        .addGap(18, 18, 18)
                        .addComponent(jButton3)))
                .addContainerGap(349, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(69, 69, 69)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(file_path, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton1))
                .addGap(18, 18, 18)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jButton2)
                    .addComponent(jButton3))
                .addContainerGap(313, Short.MAX_VALUE))
        );

        pack();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        JFileChooser chooser = new JFileChooser();
        chooser.showOpenDialog(null);
        File f = chooser.getSelectedFile();
        file_path.setText(f.getAbsolutePath());
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)
        try{
            FileInputStream file = new FileInputStream(file_path.getText());
            FileOutputStream outStream = new FileOutputStream("Encrypt.jpg");
            byte k[]="CooL2116NiTh5252".getBytes();
            SecretKeySpec key = new SecretKeySpec(k, "AES");
            Cipher enc = Cipher.getInstance("AES");
            enc.init(Cipher.ENCRYPT_MODE, key);
            CipherOutputStream cos = new CipherOutputStream(outStream, enc);
            byte[] buf = new byte[1024];
            int read;
            while((read=file.read(buf))!=-1){
                cos.write(buf,0,read);
            }
            file.close();
            outStream.flush();
            cos.close();
            JOptionPane.showMessageDialog(null, "The file encrypted Successfully");
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            FileInputStream file = new FileInputStream(file_path.getText());
            FileOutputStream outStream = new FileOutputStream("Decrypt.jpg");
            byte k[]="CooL2116NiTh5252".getBytes();
            SecretKeySpec key = new SecretKeySpec(k, "AES");
            Cipher enc = Cipher.getInstance("AES");
            enc.init(Cipher.DECRYPT_MODE, key);
            CipherOutputStream cos = new CipherOutputStream(outStream, enc);
            byte[] buf = new byte[1024];
            int read;
            while((read=file.read(buf))!=-1){
                cos.write(buf,0,read);
            }
            file.close();
            outStream.flush();
            cos.close();
            JOptionPane.showMessageDialog(null, "The image was decrypted successfully");
            Runtime.getRuntime().exec("rundll32 url.dll, FileProtocolHandler "+"Decrypt.jpg");
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledL
ookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(ImageCrypto.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(ImageCrypto.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(ImageCrypto.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(ImageCrypto.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new ImageCrypto().setVisible(true);
            }
        });
    }
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JTextField file_path;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    // End of variables declaration//GEN-END:variables
}
```

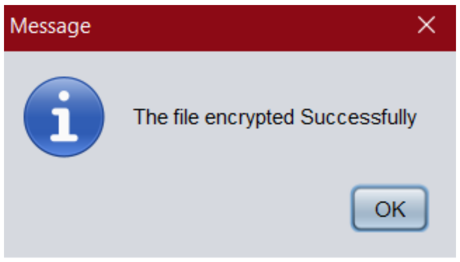
OUTPUT:

Figure 11. *Original Image*



Encrypt.jpg
It looks like we don't support this file format.

Figure 12. *Image Encryption*

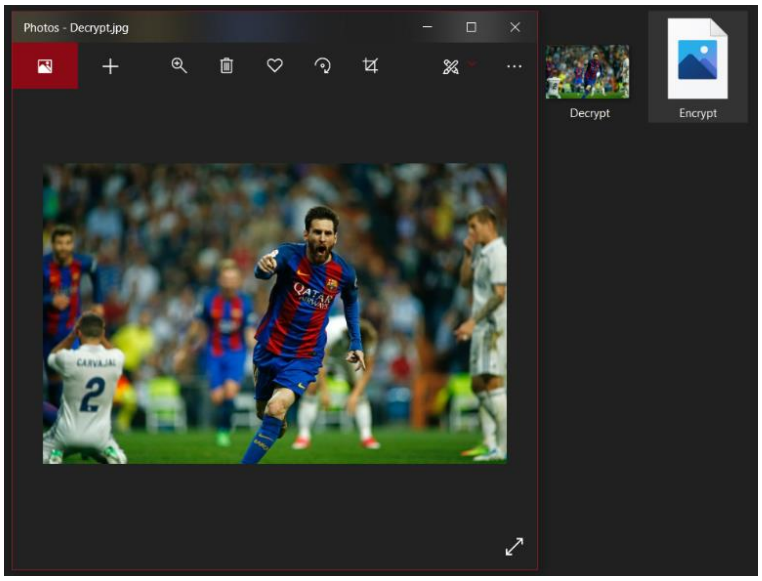


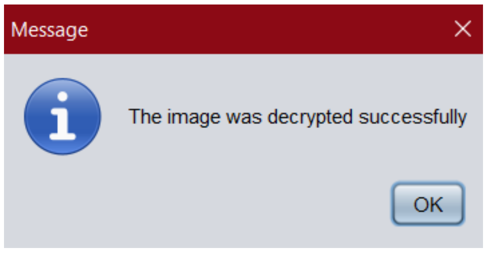
Message

The file encrypted Successfully

OK

Figure 13. *Pop-up for successful encryption*





Message

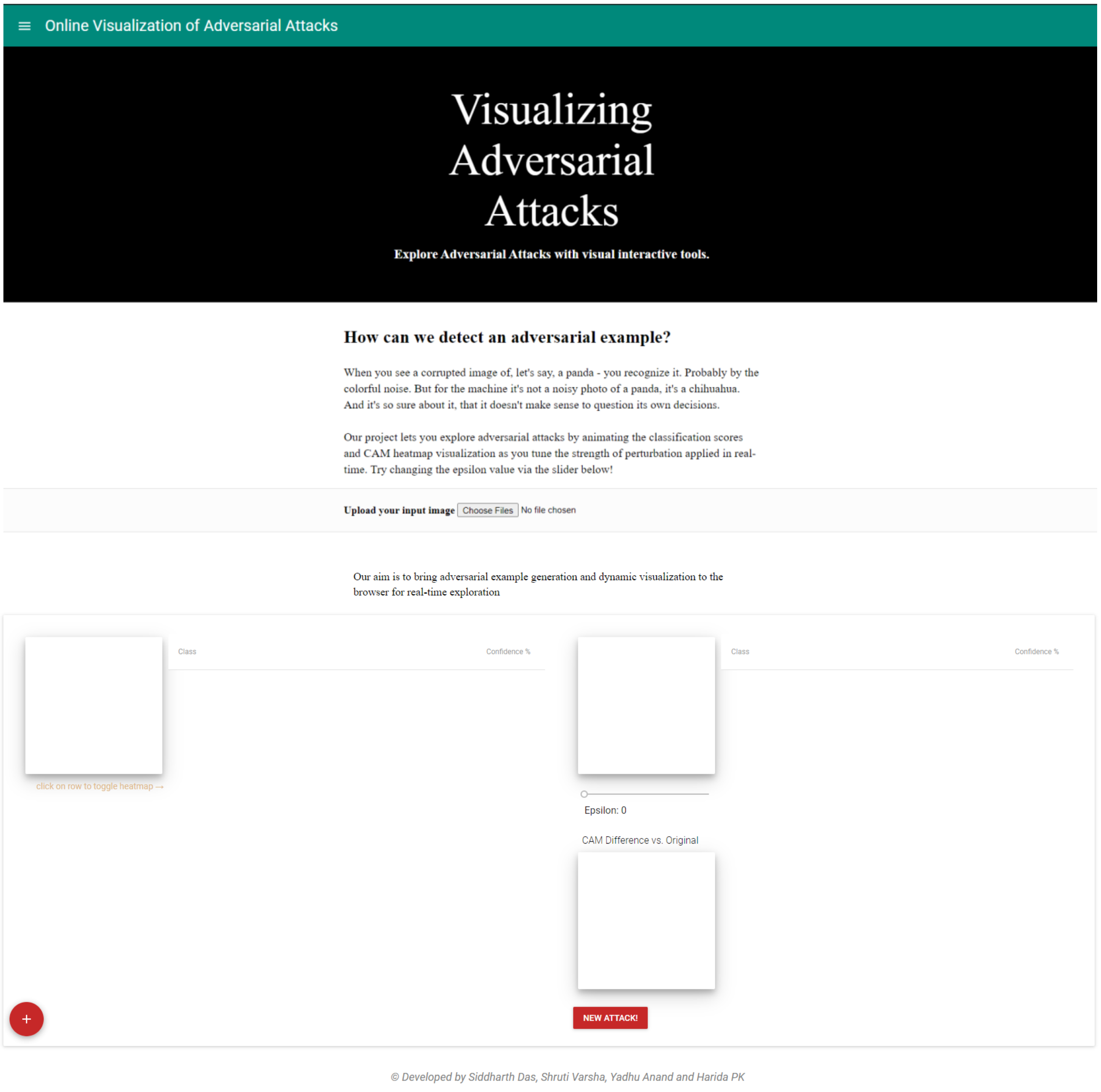
The image was decrypted successfully

OK

Figure 15. *Pop-up for successful decryption*

Figure 14. *Image Decryption*

Though we understand the process of Visual Cryptography, there is always a chance of Man-in-the-middle (MITM) attack which can hamper the data integrity of the image and subsequently violate the CIA triad. Although minor changes are not eye-catching for a human but for an automated machine or AI, it might be different. So, we have tried to create a react webapp to visualize adversarial attacks and exploit those transformations to build a defense against these attacks, as shown in Figure 16.


Online Visualization of Adversarial Attacks
Class
Confidence %
click on row to toggle heatmap →
Epsilon: 0
CAM Difference vs. Original
NEW ATTACK!
+

Visualizing Adversarial Attacks

Explore Adversarial Attacks with visual interactive tools.

How can we detect an adversarial example?

When you see a corrupted image of, let's say, a panda - you recognize it. Probably by the colorful noise. But for the machine it's not a noisy photo of a panda, it's a chihuahua. And it's so sure about it, that it doesn't make sense to question its own decisions.

Our project lets you explore adversarial attacks by animating the classification scores and CAM heatmap visualization as you tune the strength of perturbation applied in real-time. Try changing the epsilon value via the slider below!

Upload your input image Choose Files No file chosen

Our aim is to bring adversarial example generation and dynamic visualization to the browser for real-time exploration

© Developed by Siddharth Das, Shruti Varsha, Yadhu Anand and Harida PK

Figure 16. *Webapp GUI for online visualization of adversarial attacks*

SAMPLE CODE:

- **Index.js**

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import registerServiceWorker from './registerServiceWorker';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
```

- **App.js**

```
import React, { Component } from 'react';

// UI imports
import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
import { AppBar } from 'material-ui';
import getMuiTheme from 'material-ui/styles/getMuiTheme';
import { blueGrey700, teal600, tealA700, red800 } from 'material-ui/styles/colors';

// Tools
import CAMTool from './tools/CAMTool.js';

// Explanations
import {
  IntroExplanation,
  CAMExplanation,
  DeepDreamExplanation
} from './explanations/Explanations.js'


class App extends Component {
  constructor(props) {
    super(props);
    this.muiTheme = getMuiTheme({
      palette: {
        primary1Color: teal600,
        accent1Color: red800,
        image: 'panda.jpg'
      },
    });

    this.state = {
      image: null,
    };
  }

  setCroppedImage = e => {
    this.child.uploadCroppedImage(e);
  }


  render() {
    return (
      <MuiThemeProvider muiTheme={this.muiTheme}>
        <AppBar title="Online Visualization of Adversarial Attacks"></AppBar>
        <div className="banner-cover" id="banner">
          <div className="Page-intro-title">
            <span class="advis-shine"> Visualizing Adversarial Attacks </span><br />
          </div>
          <div className="Page-intro-description">
            Explore Adversarial Attacks with visual interactive tools. <br />
          </div>
        </div>

        <div className="Explanation-intro">
          <IntroExplanation setCroppedImage={this.setCroppedImage} />
        </div>

        <div className="toolBox">
          <CAMTool onRef={ref => (this.child = ref)} srcImage={this.state.image}
attackDisplays={this.state.attackDisplays} />
        </div>

        <div className="Explanation-center">
          <DeepDreamExplanation />
        </div>

      </MuiThemeProvider>
    );
  }
}
export default App;
```

- **Explanation.js**

```
import React, { Component } from 'react';
import 'typeface-roboto';
import {RaisedButton, Divider, Paper} from 'material-ui';
import ReactCrop, {makeAspectCrop} from 'react-image-crop';
import 'react-image-crop/dist/ReactCrop.css';
import '../code.css';

export class IntroExplanation extends Component {
  constructor(props) {
    super(props);

    this.state = {
      src: null,
      crop: null
    };
  }

  onImageLoaded = image => {
    this.setState({
      crop: makeAspectCrop({
        x: 25,
        y: 25,
        aspect: 1 / 1,
        width: 227,
      },
      image.width / image.height),
      image: image,
    });
  }

  onSelectFile = e => {
    if (e.target.files && e.target.files.length > 0) {
      const reader = new FileReader();
      reader.addEventListener(
        'load',
        () => {
          this.setState({
            src: reader.result,
          });

          let parent = this;
          let image = new Image();
          image.src = reader.result;
          image.onload = function() {
            // cache raw image size here
            parent.setState({
              origWidth: image.width,
              origHeight: image.height
            });
          };
        },
        false
      )
      reader.readAsDataURL(e.target.files[0]);
    }
  }

  onCropChange = crop => {
    this.setState({ crop });
  }

  onCropClicked = () => {
    this.getCroppedImg(this.state.image, this.state.crop);
  }

  getCroppedImg(image, pixelCrop) {
    // Print crops to canvas
    const canvas = document.createElement('canvas');
    canvas.width = this.state.origWidth;
    canvas.height = this.state.origHeight;
    const ctx = canvas.getContext('2d');

    // ctx.drawImage(image, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight) API:
    // https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/drawImage
    ctx.drawImage(
      image,
      (pixelCrop.x/100)*this.state.origWidth,
      (pixelCrop.y/100)*this.state.origHeight,
      (pixelCrop.width/100)*this.state.origWidth,
      (pixelCrop.height/100)*this.state.origHeight,
      0,
      0,
      227,
      227
    );

    // As Base64 string
    const base64Image = canvas.toDataURL('image/jpeg');
    this.props.setCroppedImage(base64Image);
  }

  render() {
    return (
      <div>
      <div className="Explanation-center">
        <h2>How can we detect an adversarial example?</h2>
        <p>
          When you see a corrupted image of, let's say, a panda - you recognize it. Probably by
the colorful noise. But for the machine it's not a noisy photo of a panda, it's a chihuahua. And
it's so sure about it, that it doesn't make sense to question its own decisions. <br /><br />Our
project lets you explore adversarial attacks by animating the classification scores and CAM
heatmap visualization as you tune the strength of perturbation applied in real-time. Try changing
the epsilon value via the slider below!
        </p>
      </div>

      <Divider />
      <div style={{backgroundColor: 'hsl(0, 0%, 99%)'}}>
        <br />
        <div id="StickyPicker" style={{gridColumn: 'screen', margin:'auto'}}>
          <div svelte-1277576141 class="root">
            <div class="sticky base-grid" style={{margin: 'auto', maxWidth: '640px'}}>
              <div class="container" style={{margin: 'auto'}}>


                  <h4 style={{display: "inline"}}>Upload your input image </h4>
                  <input style={{marginTop: "0px", marginBottom: "0px", display: "inline"}}
onChange={this.onSelectFile} type="file" id="files" name="files[]" multiple/>

                  {this.state.src && (
                  <div>
                    <ReactCrop
                      src={this.state.src}
                      crop={this.state.crop}
                      onImageLoaded={this.onImageLoaded}
                      onChange={this.onCropChange}
                      style={{marginTop: "6px", marginBottom: "4px", maxHeight: "150px"}}
                    />
                    <br />
                    <RaisedButton
                      label="Crop"
                      secondary={true}
                      onClick={this.onCropClicked}
                    />
                    <br />
                  </div>
                )}

              </div>
            </div>
          </div>
        </div>
        <br />
        </div>

      <Divider />
      <div className="Explanation-center">
        <p> Our aim is to bring adversarial example generation and dynamic visualization to the
browser for real-time exploration </p>
      </div>
      </div>
    );
  }
}
```

```
export class CAMExplanation extends Component {
  constructor(props) {
    super(props);
  }

  render() {
    return (
      <div style={{fontFamily: "Roboto"}}>

      </div>
    );
  }
}

export class DeepDreamExplanation extends Component {
  constructor(props) {
    super(props);
  }
  render() {
    return (
      <div style={{fontFamily: "Roboto"}}>
        <p style={{color:'gray', textAlign:'right'}}><em> © Developed by Siddharth Das, Shruti
Varsha, Yadhu Anand and Harida PK</em></p>
        <br /><br /><br />
      </div>
    )
  }
}
```

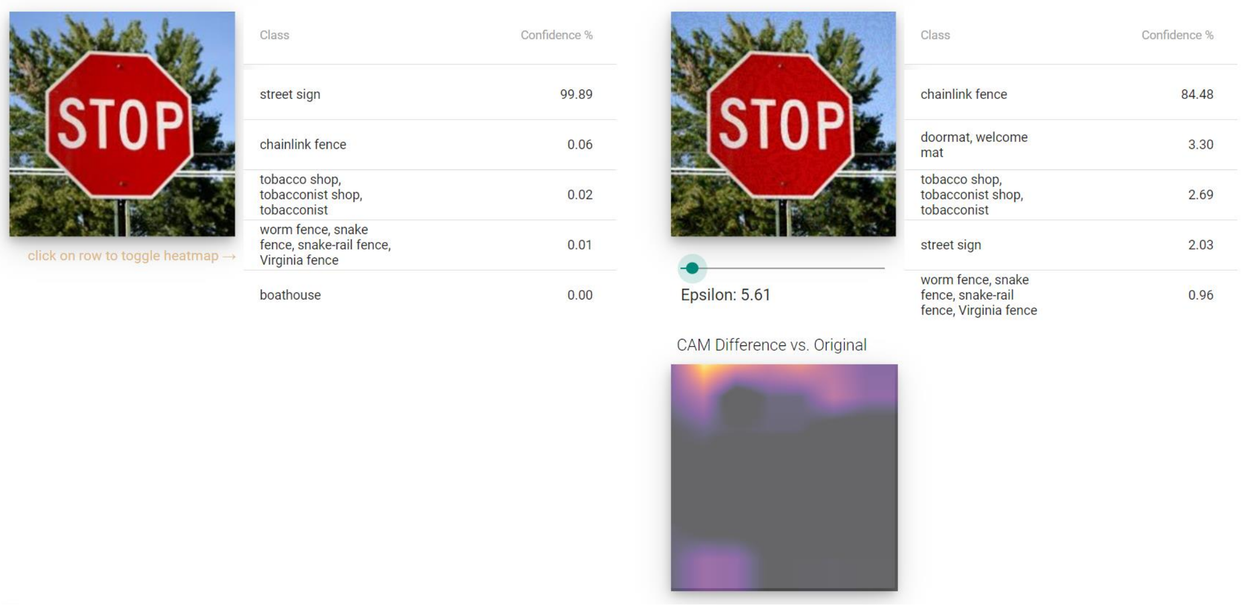
- **registerServiceWorker.js**

```
const isLocalhost = Boolean(
  window.location.hostname === 'localhost' ||
    // [::1] is the IPv6 localhost address.
    window.location.hostname === '[::1]' ||
    // 127.0.0.1/8 is considered localhost for IPv4.
    window.location.hostname.match(
      /^127(?:\.(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)){3}$/
    )
);

export default function register() {
  if (process.env.NODE_ENV === 'production' && 'serviceWorker' in navigator) {
    const publicUrl = new URL(process.env.PUBLIC_URL, window.location);
    if (publicUrl.origin !== window.location.origin) {
      return;
    }

    window.addEventListener('load', () => {
      const swUrl = `${process.env.PUBLIC_URL}/service-worker.js`;

      if (isLocalhost) {
        checkValidServiceWorker(swUrl);

        navigator.serviceWorker.ready.then(() => {
          console.log(
            'This web app is being served cache-first by a service ' +
              'worker. To learn more, visit https://goo.gl/SC7cgQ'
          );
        });
      } else {
        registerValidSW(swUrl);
      }
    });
  }
}

function registerValidSW(swUrl) {
  navigator.serviceWorker
    .register(swUrl)
    .then(registration => {
      registration.onupdatefound = () => {
        const installingWorker = registration.installing;
        installingWorker.onstatechange = () => {
          if (installingWorker.state === 'installed') {
            if (navigator.serviceWorker.controller) {
              console.log('New content is available; please refresh.');
            } else {
              console.log('Content is cached for offline use.');
            }
          }
        };
      };
    })
    .catch(error => {
      console.error('Error during service worker registration:', error);
    });
}

function checkValidServiceWorker(swUrl) {
  // Check if the service worker can be found. If it can't reload the page.
  fetch(swUrl)
    .then(response => {
      // Ensure service worker exists, and that we really are getting a JS file.
      if (
        response.status === 404 ||
        response.headers.get('content-type').indexOf('javascript') === -1
      ) {
        // No service worker found. Probably a different app. Reload the page.
        navigator.serviceWorker.ready.then(registration => {
          registration.unregister().then(() => {
            window.location.reload();
          });
        });
      } else {
        // Service worker found. Proceed as normal.
        registerValidSW(swUrl);
      }
    })
    .catch(() => {
      console.log(
        'No internet connection found. App is running in offline mode.'
      );
    });
}

export function unregister() {
  if ('serviceWorker' in navigator) {
    navigator.serviceWorker.ready.then(registration => {
      registration.unregister();
    });
  }
}
```

Adversarial Example #1

Initially it will detect the image as a street sign with epsilon set to zero, as shown in Figure 17.
A normal AI will detect the same unless it is adversarially attacked.

As we increase the epsilon toggle, the CAM heatmap changes accordingly which subsequently changes the gradient descent function causing a misclassification of the data cluster.
Thus, adversarial attacks happen where the attacker hides itself in the plain sight.

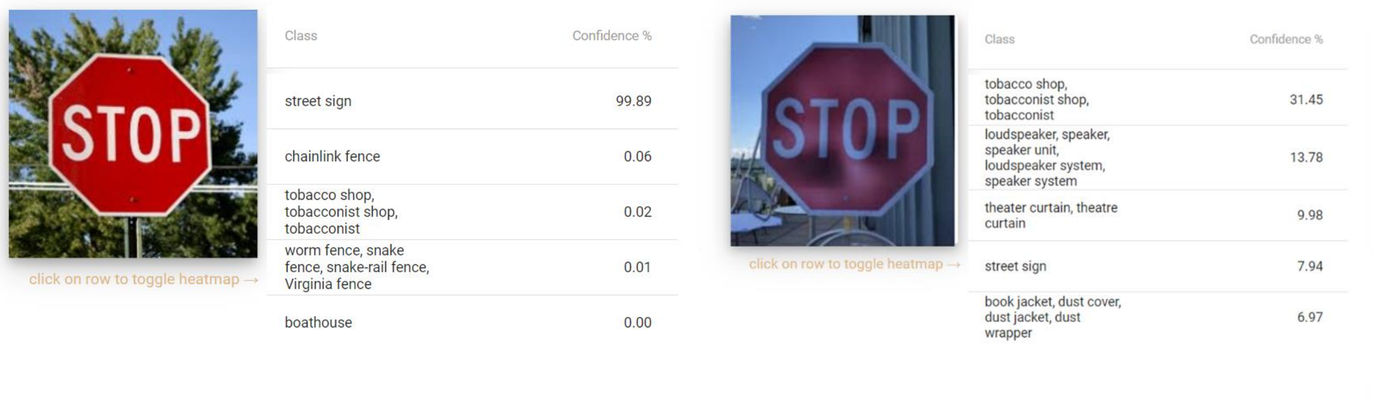


Class	Confidence %
street sign	99.89
chainlink fence	0.06
tobacco shop, tobacconist shop, tobacconist	0.02
worm fence, snake fence, snake-rail fence, Virginia fence	0.01
boathouse	0.00

Class	Confidence %
chainlink fence	84.48
doormat, welcome mat	3.30
tobacco shop, tobacconist shop, tobacconist	2.69
street sign	2.03
worm fence, snake fence, snake-rail fence, Virginia fence	0.96

Figure 17. *Online Visualization of Adversarial Attacks*

Adversarial attacks can also be conducted externally also if the attacker recreates the AI's decision tree network and tests the adversarial examples by performing external changes.
Here in the first, it shows correctly as a street sign whereas the in the second it shows incorrectly as a tobacco shop. Though a human eye will not get a sense of delusion by these changes,
But an AI can be fooled through adversarial attacks, as in Figure 18.

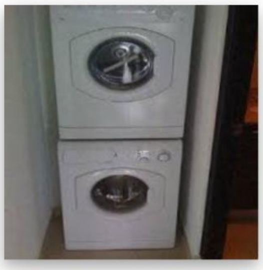

STOP
click on row to toggle heatmap →
STOP
click on row to toggle heatmap →

Class	Confidence %
street sign	99.89
chainlink fence	0.06
tobacco shop, tobacconist shop, tobacconist	0.02
worm fence, snake fence, snake-rail fence, Virginia fence	0.01
boathouse	0.00

Class	Confidence %
tobacco shop, tobacconist shop, tobacconist	31.45
loudspeaker, speaker, speaker unit, loudspeaker system, speaker system	13.78
theater curtain, theatre curtain	9.98
street sign	7.94
book jacket, dust cover, dust jacket, dust wrapper	6.97

Figure 18. *Comparing same street signs with different external appearance and color texture*

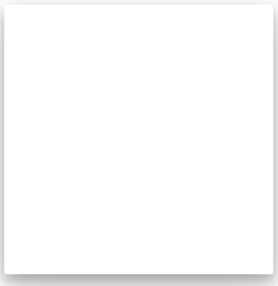
Adversarial Example #2

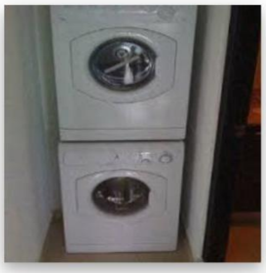


Epsilon: 0

Class	Confidence %
washer, automatic washer, washing machine	76.07
stove	13.09
safe	2.61
space heater	2.43
microwave, microwave oven	2.06

CAM Difference vs. Original

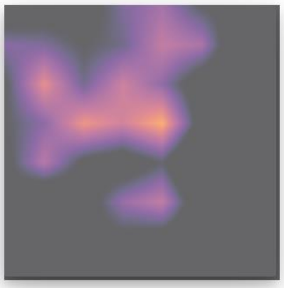


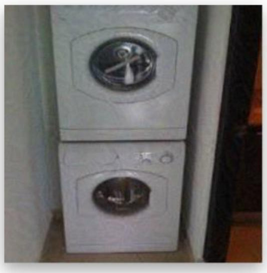


Epsilon: 0.5

Class	Confidence %
stove	80.17
space heater	8.71
soap dispenser	6.09
safe	1.53
loudspeaker, speaker, speaker unit, loudspeaker system, speaker	0.52

CAM Difference vs. Original

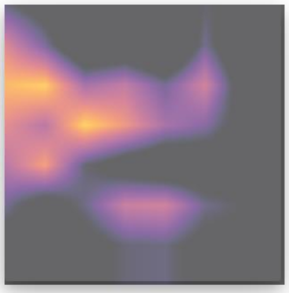


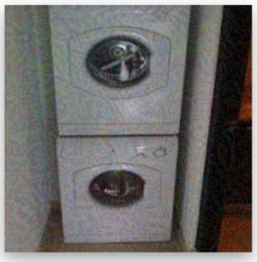


Epsilon: 2.67

Class	Confidence %
soap dispenser	44.43
stove	32.77
space heater	10.97
safe	3.20
loudspeaker, speaker, speaker unit, loudspeaker system, speaker	2.55

CAM Difference vs. Original





Epsilon: 7.57

Class	Confidence %
space heater	29.87
soap dispenser	23.89
switch, electric switch, electrical switch	17.20
loudspeaker, speaker, speaker unit, loudspeaker system, speaker system	7.71
modem	4.99

CAM Difference vs. Original

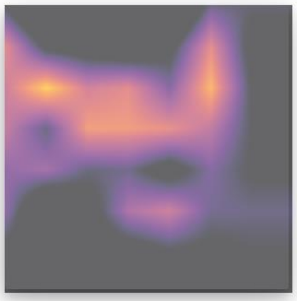


Figure 19. *Classification of data according to various epsilon values and corresponding visualization of CAM heatmap*

Adversarial Attack #3

We will study both of these images and will show the online visualization of adversarial attack Both are the images of a Siamese cat but the test results will differ because of the adversarial attack on the second image(Figure 20).

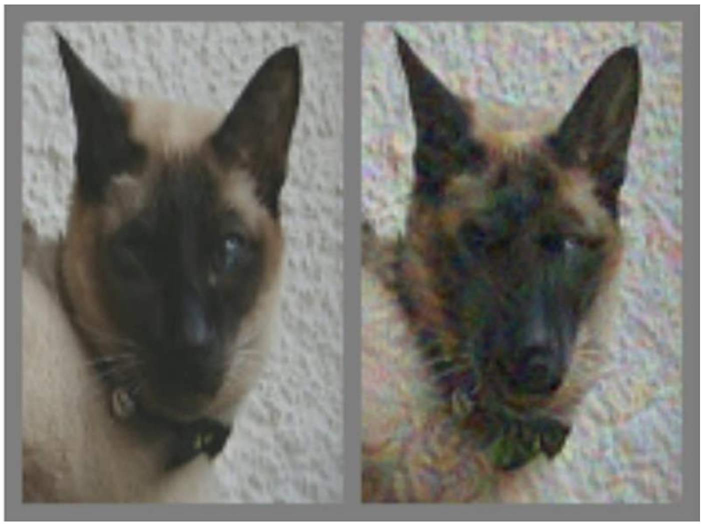
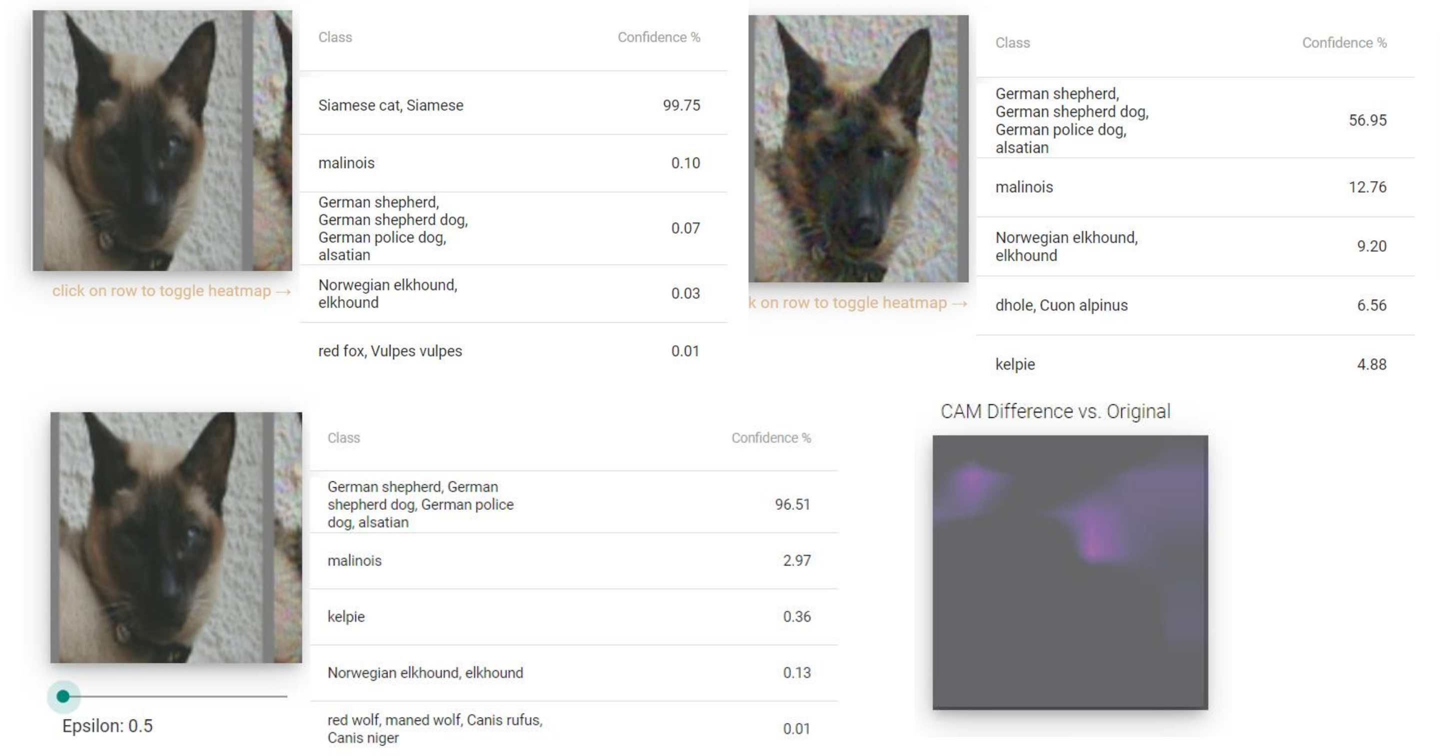


Figure 20. *Image of Siamese Cat*

First one shows correctly as Siamese Cat while the other image shows it as German Shepherd Dog. This is digitally engineered attack which creates a misclassification of two different set of data due to the manipulation of the gradient descent function by the attacker , as in Figure 21.



click on row to toggle heatmap →
k on row to toggle heatmap →
Epsilon: 0.5
CAM Difference vs. Original

Class	Confidence %
Siamese cat, Siamese	99.75
malinois	0.10
German shepherd, German shepherd dog, German police dog, alsatian	0.07
Norwegian elkhound, elkhound	0.03
red fox, Vulpes vulpes	0.01

Class	Confidence %
German shepherd, German shepherd dog, German police dog, alsatian	56.95
malinois	12.76
Norwegian elkhound, elkhound	9.20
dhole, Cuon alpinus	6.56
kelpie	4.88

Class	Confidence %
German shepherd, German shepherd dog, German police dog, alsatian	96.51
malinois	2.97
kelpie	0.36
Norwegian elkhound, elkhound	0.13
red wolf, maned wolf, Canis rufus, Canis niger	0.01

Figure 21. *Visualizing difference in epsilon values to misclassify Siamese Cat as German Shepherd Dog.*

PERFORMANCE ANALYSIS:

- **Tabular analysis**

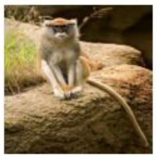
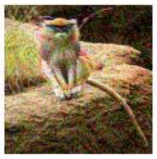
original image	true label	Clarifai.com results of original image	target label	targeted adversarial example	Clarifai.com results of targeted adversarial example
	viaduct	bridge, sight, arch, river, sky	window screen		window, wall, old, decoration, design
	hip, rose hip, rosehip	fruit, fall, food, little, wildlife	stupa, tope		Buddha, gold, temple, celebration, artistic
	dogsled, dog sled, dog sleigh	group together, four, sledge, sled, enjoyment	hip, rose hip, rosehip		cherry, branch, fruit, food, season
	pug, pug-dog	pug, friendship, adorable, purebred, sit	sea lion		sea seal, ocean, head, sea, cute
	Old English sheep-dog, bobtail	poodle, retriever, loyalty, sit, two	abaya		veil, spirituality, religion, people, illustration
	maillot, tank suit	beach, woman, adult, wear, portrait	amphib-ian, amphibi-ous vehicle		transportation system, vehicle, man, print, retro
	patas, hussar monkey, Erythro-cebus patas	primate, monkey, safari, sit, looking	bee eater		ornithology, avian, beak, wing, feather

***Figure 22**. Original images and adversarial images evaluated over Clarifai.com. For labels returned from Clarifai.com, the authors sort the labels firstly by rareness: how many times a label appears in the Clarifai.com results for all adversarial images and original images, and secondly by confidence. Only the top five labels are provided.*

- **Graphical Analysis**

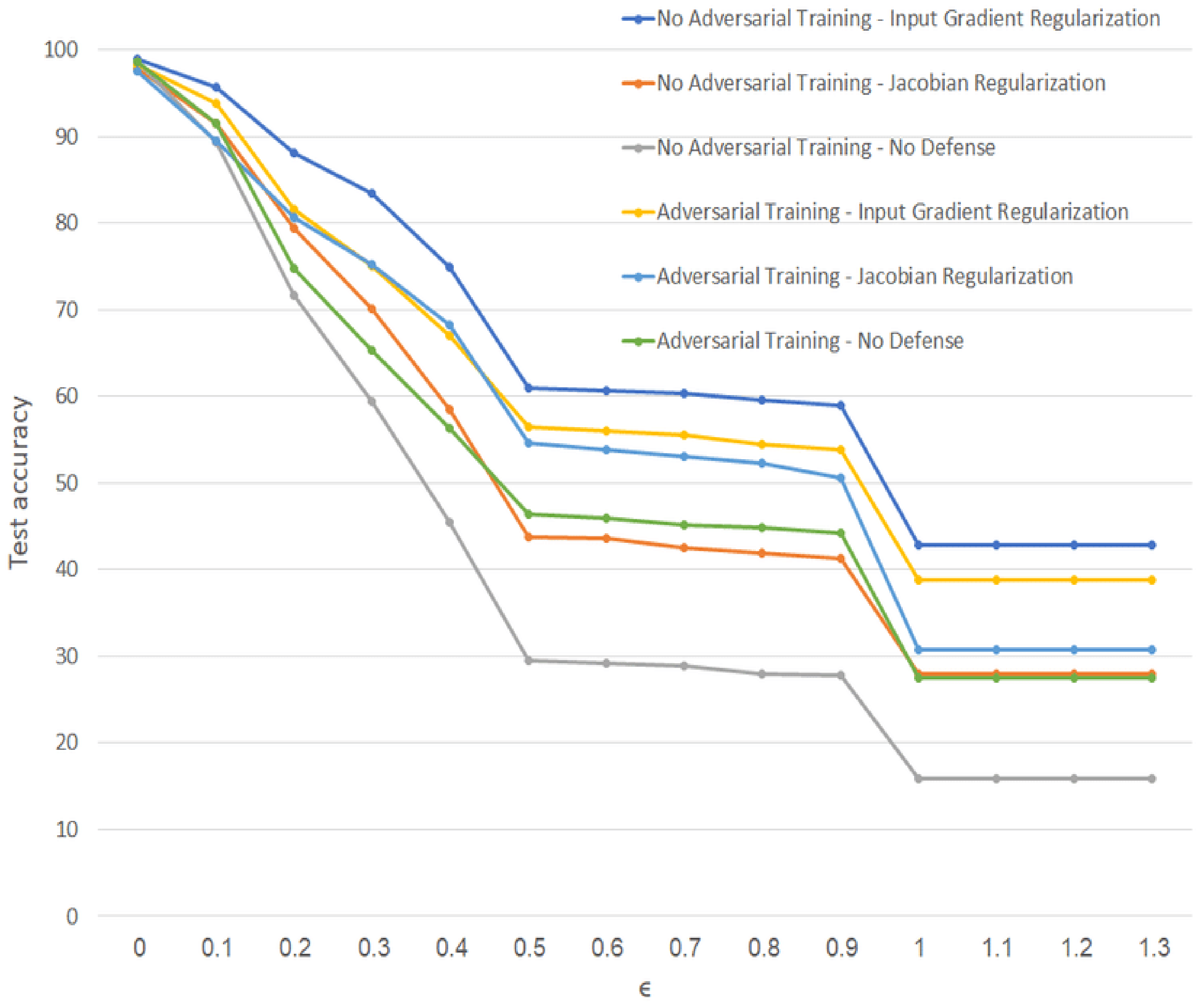


Figure 23. *Test accuracy results under an Adversarial Attack with and without Adversarial Training.*

CONCLUSION

In this project, we have focused on test-time inputs intended to confuse a machine learning model, but many other kinds of attacks are possible, such as attacks based on surreptitiously modifying the training data to cause the model to learn to behave the way the attacker wishes it to behave.

One bright spot in adversarial machine learning is differential privacy, where we actually have theoretical arguments that certain training algorithms can prevent attackers from recovering sensitive information about the training set from a trained model. It is interesting to compare machine learning to other scenarios where attacks and defences are both possible.

In cryptography, the defender seems to have the advantage. Given a set of reasonable assumptions, such as that the cryptographic algorithm is implemented correctly, the defender can reliably send a message that the attacker cannot decrypt.

In physical conflict, attackers seem to have the advantage. It is much easier to build a nuclear bomb than to build a city that is able to withstand a nuclear explosion. The second law of thermodynamics seems to imply that, if defending requires maintaining entropy below some threshold, then the defender must eventually lose as entropy increases over time, even if there is no explicit adversary intentionally causing this increase in entropy.

The no free lunch theorem for supervised learning says that, averaged over all possible datasets, no machine learning algorithm does better on new points at test time than any other algorithm. At first glance, this seems to suggest that all algorithms are equally vulnerable to adversarial examples. However, the no free lunch theorem applies only when we make no assumption about the structure of the problem. When we study adversarial examples, we assume that small perturbations of the input should not change the output class, so the no free lunch theorem in its typical form does not apply.

The study of adversarial examples is exciting because many of the most important problems remain open, both in terms of theory and in terms of applications. On the theoretical side, no one yet knows whether defending against adversarial examples is a theoretically hopeless endeavour (like trying to find a universal machine learning algorithm) or if an optimal strategy would give the defender the upper ground (like in cryptography and differential privacy). On the applied side, no one has yet designed a truly powerful defence algorithm that can resist a wide variety of adversarial example attack algorithms. That’s what makes this area of research more exciting.

References

[1] Abadi, M.; Andersen, D.G. Learning to protect communications with adversarial neural cryptography

[2] NehaTyagi, Ashish Agarwal, Anurag Katiyar, Shubham Garg, Shudhanshu Yadav, 2017 “Methods for Protection of Key in Private Key Cryptography”, International Journal of Innovative Research in Computer Science & Technology (IJIRCST), Volume-5, Issue-2.

[3] Subhrajit Mondal, Tania Khatun Mollah, ArindamSamanta, Soumya Paul, 2016 “A Survey on Network Security Using Genetic Algorithm “, International Journal of Innovative Research in Science, Engineering and Technology (A High Impact Factor, Monthly Peer Reviewed Journal) Vol. 5, Issue 1. A.O. Isah, J.K Alhassan, S.S Olanrewaju, Enesi Femi Aminu, 2017 “Enhancing AES with Time-Bound and Feedback Artificial Agent Algorithms for Security and Tracking of Multimedia Data on Transition”, International Journal of Cyber-Security and Digital Forensics (IJCSDF) 6(4): 162- 178162 The Society of Digital Information and Wireless Communications (SDIWC), ISSN:2305-0012.

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. CoRR abs/1406.2661 (2014). arXiv:1406.2661

[5] Nicolas Papernot, Patrick D. McDaniel, Arunesh Sinha, and Michael P. Wellman. 2016. Towards the Science of Security and Privacy in Machine Learning. CoRR abs/1611.03814 (2016).

[6] Maya Kabkab Pouya Samangouei and Rama Chellappa. 2018. Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. arXiv preprint arXiv:1805.06605 (2018).

[7] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. CoRR abs/1312.6199 (2013).

[8] Megha B.Goel , Vaishali B.Bhagat , Veena K. Katankar , Authentication Framework using Visual Cryptography

[9] Reem Ibrahim Hasan and Huda Adil Abdulghafoor , Cipher Secret Image using Hybrid Visual Cryptography

[10] Megha R.Chaudhari, Neha D.Chaudhari, Shubhangi S.Kanade, Sumedh G.Bhadre, Dhiraj D.Bhagat , Attack Prevention Using Visual Cryptography

[11]Sankar Das, Sandipan Chowdary and Dibya Chakraborty , System of Visual Cryptography using Three Independent Shares in Color Images

[12] Deng yuqiao, song Ge , A verifiable Visual Cryptography scheme using Neural Networks

[13] Guo, T., & Zhou, L. (2018). Constructing visual cryptography scheme by hypergraph decomposition. *Procedia computer science*, *131*, 336-343.

[14] Dahat, A. V., & Chavan, P. V. (2016). Secret sharing based visual cryptography scheme using CMY color space. *Procedia Computer Science*, *78*(C), 563-570.

[15] Siahaan, A. P. U. (2017). RC4 Technique in Visual Cryptography RGB Image Encryption.

[16] Visual cryptography for grey level images : Carlo Blundo , Alfredo De Santis a,Moni Naor

[17] Visual cryptography for color images : Young-Chang Hou

[18] VCPSS:Atwo-in-one two-decoding-options image sharing method combining visual cryptography (VC) and polynomial-style sharing (PSS) approaches : Sian-Jheng Lin, Ja-Chen Lin

[19] Extended visual cryptography for natural images : Nakajima, Mizuko,Yamaguchi, Yasushi

[20] Halftone Visual Cryptography : Zhi Zhou, Member, IEEE, Gonzalo R. Arce, Fellow, IEEE, and Giovanni Di Crescenzo