Adversarial Attacks in Banknote Recognition Systems

Hangyu Liu
stab2021@hotmail.com
St. Anne's- Belfield School
Charlottesville, Virginia, USA

Haoyang Cai
willcaiai@gmail.com
St. Anne's- Belfield School
Charlottesville, Virginia, USA

ABSTRACT

Numerous computer vision algorithms have been designed to help detect the values of the paper banknotes for blind and visually impaired individuals. Previous algorithms relied mostly on traditional methods that are less efficient, accurate, and transferable than a convolutional neural network. So in our study, we used the convolutional neural networks to perform banknote image classification. We trained our network with 1,000 Thai banknote images. Our network could classify real-life banknotes used in our study with an accuracy of 100%; however, we have also found out that our convolutional neural network model is vulnerable to adversarial attacks. We then retrained the network using adversarial training which reduced the attack success rate by approximately 50%. Overall, because we did our experiments using MobileNet, which is a relatively small convolutional neural network that can be used in a mobile phone, we have trained an accurate and robust banknote recognition convolutional neural network that could be integrated into a mobile app or wearable device for the visually impaired.

CCS CONCEPTS

• **Social and professional topics** → Financial crime; • **Theory of computation** → *Models of computation*;

KEYWORDS

banknote value recognition, computer vision, convolutional neural network, adversarial attack, Fast Gradient Sign Method, adversarial training

ACM Reference Format:
Hangyu Liu and Haoyang Cai. 2021. Adversarial Attacks in Banknote Recognition Systems. In *2021 4th International Conference on Data Storage and Data Engineering (DSDE '21), February 18–20, 2021, Barcelona, Spain.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3456146.3456153

1 INTRODUCTION

Ackland et al. [1] approximates that there were 36 million blind people around the world in 2015. This number has been fairly stable throughout the past few years, which means that there remains a critical need to provide convenience for blind people around the world. Given that they cannot see, blind people often experience difficulties when having to rely on other ways to acquire visual information [3]. One specific difficulty that blind people often encounter is to know the value of banknotes (paper currency) that they use. Although banknotes can often be distinguished by size and sometimes texture, there are exceptions when the banknotes are being heavily used and lose some of their physical features. Moreover, when carrying or obtaining more than a few banknotes in change, blind people have to organize the banknotes in order of their face values based on their sizes, which is a tedious and error-prone process.

To address blind people's inconvenience of knowing the value of banknotes, researchers developed computer vision algorithms that could recognize the face value of banknotes based on the banknotes' RGB images. Earlier banknote recognition systems relied on traditional computer vision algorithms, using feature extraction techniques like SIFT, SURF, etc [13]. But the most recent banknote recognition systems turned to using Convolutional Neural Networks, because CNN is more adaptive to sophisticated real-world scenarios, can be easily applied to classifying different types of banknotes, and performs with higher accuracy.

In our study, we have made three contributions to address the problem of banknote recognition: I. We verified the effectiveness of classifying banknote images with CNN. II. We proved that the current CNN banknote recognition models are vulnerable to an adversarial attack, which could be potentially exploited by individuals with bad intention. III. We managed to increase the model's robustness against a malicious attack with adversarial training.

2 LITERATURE REVIEW

2.1 Banknote Recognition Systems

In 2011, Solymár et al. [13] utilized an image-filtering method that detects the interest point on Hungarian forints and then calculates the spatial relationship between those interest points to make classifications.

In 2012, Hasanuzzaman et al. [5] utilized a more complex component based framework to recognize banknote values. This method extracts local features of the banknote with Speed Up Robust Feature (SURF) and compares them with the reference regions of the ground truth image in each banknote category. It improved on previous methods by classifying banknotes with higher accuracy when encountering partially occluded banknotes, noisy image background, and images with no banknotes (camera not pointed to banknote).

To minimize the error of identification, Yeh et al. [18] employed a machine learning algorithm called the Support Vector Machine with multiple kernels. Similar to deep learning methods, traditional machine learning algorithms also takes the divided parts of images

DSDE '21, February 18–20, 2021, Barcelona, Spain

ACM ISBN 978-1-4503-8930-3/21/02…$15.00
https://doi.org/10.1145/3456146.3456153

as input, and process each part with the weighted kernel, and combined the output of kernels into a matrix as the output. This specific machine learning method has succeeded in detecting counterfeit banknotes.

Lee et al. [7] also reviewed many banknote recognition methods associate with certain types of sensors. Conventional sensors like the visible-light sensor can be used to obtain RGB or black and white images for employing neural network methods, and there are UV sensors that collect lights from non-visible wavelengths, usually to extract the counterfeit banknotes' features, and different methods are used according to the type of sensor that collects the data.

In an AI Summit in 2017, Microsoft announced their version of banknote recognition, which uses CNN and Transfer Learning. It achieved an accuracy of 85% [8]. Later, a real-time banknote recognition method was developed base on a similar neural network model by Ren et al. [11]; it identified the seventh series New Zealand currency using RGB images, and employed methods like LBP-histogram to extract colour and texture features from the images. This artificial neural network model differed from our CNN model by its architecture; all of the neurons in ANN are connected, but only the last layer(s) of CNN are fully connected.

2.2 Adversarial Attacks

As novel machine learning methods gained popularity, their attributes were also heavily investigated. Szegedy et al. [15] made an interesting discovery: several machine learning models, including the more complicated deep neural networks that generalize well on an object recognition task, are vulnerable to adversarial examples. These deep learning models tend to misclassify images and perform with significantly lower accuracy under adversarial attacks. The followings are some common attack methods.

(1) Fast Gradient Sign Method (FGSM): This adversarial attack method is developed by Goodfellow et al. [4]. It is an image-specific one-shot method that requires minimal calculations. There are multiple variations of this method, including iterative-FGSM and targeted-FGSM.
(2) Basic & Least-Likely-Class Iterative Methods (BIM): This is a method similar to FGSM mentioned in Kurakin et al. [6]. While the FGSM only perturbs the image once, the BIM Algorithm perturbs the image for many iterations and adjusts the direction of perturbation after each step.
(3) Jacobian-based Saliency Map Attack (JSMA): This is a targeted attack method developed by Papernot et al. [10], It intends to minimize the perturbation by modifying the pixels one at a time and monitoring how much it changes classification. This change is recorded for every pixel to create a saliency map for the image, and only pixels at higher values of the saliency map are perturbed.
(4) One Pixel Attack: This method perturbs the image and alter the classification only by changing the value of one pixel in the image[14]. It can achieve that by producing multiple perturbation vectors, and modify the vectors by the criterion given by the model's prediction probability; eventually, the vector that has the high potential to alter the class will be used for perturbation.

2.3 Defence Against Adversarial Attacks

Adversarial training methods were developed to defend a CNN model against adversarial attacks. Generally, adversarial training uses some perturbed samples to approximate the inner maximization of the threat model and change model weights consequently.

(1) "Free" adversarial training method: Shafahi et al. [12] proposed the "Free" adversarial training method as a defence to FGSM perturbed samples. This method has m steps in each minibatch. In each step, it first calculates the updated perturbation with stochastic gradient descent then uses the gradient to calculate the minimal step to update the model parameters, thus iteratively updating the parameters. This method had limited application because it was designed specifically for L-BFGS.
(2) FGSM Adversarial training: This adversarial training method was developed by Goodfellow et al. [4]. In the experiment mentioned in the literature, the maxout network trained by this function with α of 0.5 successfully reduced the error rate from 0.94 % to 0.84 %.
(3) Smooth adversarial training: Xie et al. [17] introduced a method that abandoned the previously used reLU activation function during the training process, other training methods believe that reLU can significantly increase the robustness of the model because its rectifying nature; however, this literature thinks the non-smooth nature of reLU leads to the decrease of accuracy during adversarial training. So, this method employed a smooth activation function, and without any further computational change of the model, it not only increases the robustness of Resnet-50 from 33.0% to 42.3%, but also improves the accuracy by 0.9% using the Imagenet dataset.

3 METHODOLOGY

In this section, we will discuss our design of a banknote recognition system in the following steps: bench-mark data collection, data splitting, and data augmentation. Then, we will discuss the CNN we uses - MobileNet - and its training process. After that, we will discuss our method (FGSM) for generating adversarial images with adversarial attack. Finally, we will discuss how to make our banknote recognition system more robust against adversarial attacks by using adversarial training (Figure 1).

3.1 Bench-mark Dataset Collection

To collect the best training data for our CNN model, we are looking for banknote image datasets that meet the following requirements:

(1) The dataset should contain more than 1000 images in each banknote value class. There are approximately even amounts of images in each banknote value class so that the trained model will not be biased toward certain classes.
(2) Most banknote images are taken in real-life scenarios. This means that the banknotes may be partially occluded, folded, rippled, angled away from the camera, or have bad lighting conditions. Moreover, some images should have noisy backgrounds.
(3) Every original image (before augmentation) captures an entire banknote even if the banknote is folded.

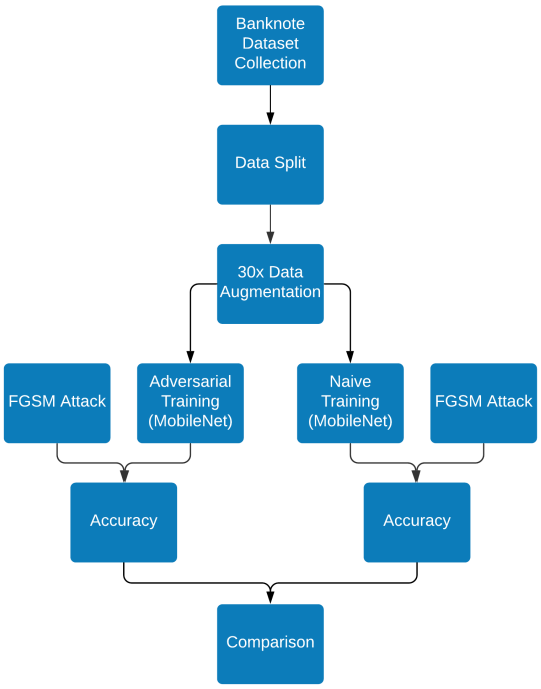

Figure 1: Workflow

Table 1: An Overview of Available Data Sets

Data Set Name	Source	Size	Augmented
Hong Kong	Kaggle [2]	12300	True
India 1	Github [16]	2578	False
India 2	IEEE [9]	1900	False
Thailand	IEEE DataPort	1000	False

Among the datasets we have collected (table 1), the Hong Kong datasets has the most images, so we previously decided to use it to train our CNN model. However, we later found out that it uses a 30-time augmentation as described in its Kaggle documentation, so it has only 410 original images in total, which is not the largest dataset that we can find. Both India datasets have two series of currencies - an old series consisting of 11 different classes and a new series consisting of 9 classes, resulting in too many classes with insufficient data in each. So eventually we decided to use the Thailand dataset which has five value categories (20, 50, 100, 500, 1000) with 200 images in each. It has the largest class sizes among all four datasets. Figure 2 shows 9 examples from the Thailand dataset. It includes banknotes images taken in various conditions (partial occlusion, noisy background, etc.) and shows that the Thailand dataset meets the second dataset requirement mentioned earlier.

Banknote Images.png

Figure 2: Thailand Dataset Samples

3.2 CNN Model - MobileNet

With data prepared, we can now proceed to model selection. According to Goodfellow et al. [4], adversarial attacks generalizes well on a variety of neural networks. So, theoretically, we can choose to perform our experiments on any of the following CNN models: LeNet, VGG, Resnet, AlexNet, and MobileNet. Initial experiment was performed on Resnet18, which produced 100% training and testing accuracy in 50-100 epochs. However, we realized that MobileNet might be a better choice because it had a smaller architecture and could fit into a Mobile App more easily [2], where most of the banknote recognition systems will be deployed. So eventually we decided to use MobileNet in our study.

3.3 Adversarial Attack - FGSM

After selecting our model, we can proceed to the discussion of adversarial attacks in this section. We decide to test our trained MobileNet against FGSM [4]. FGSM attack could be implemented in two ways: targeted and non-targeted attacks.

3.3.1 *Non-targeted Attack.*

Non-targeted adversarial attack aims to fool the classifier to predict false prediction of an arbitrary class label.

$$\boldsymbol{x}^{adv} = \boldsymbol{x} + \varepsilon \cdot \text{sign}\left(\nabla_{\boldsymbol{x}} J\left(\boldsymbol{\theta}, \boldsymbol{x}, y_{true}\right)\right) \tag{1}$$

In formula 1, $\boldsymbol{x}^{adv}$ is the perturbed output, $\boldsymbol{x}$ is the original input, ε is a scalar parameter adjusting the magnitude of perturbation, J is the loss function, $\boldsymbol{\theta}$ is model parameters, and y_{true} is the true label of the original input $\boldsymbol{x}$. Using this formula, the FGSM method perturbs the image by adding a small perturbation to every pixel. The sign of perturbation is directed toward the gradient of the loss function calculated with input $\boldsymbol{x}$ and the true label y_{true}. The magnitude of perturbation is controlled by the coefficient ε.

3.3.2 *Targeted Attack.*

Targeted attack fools the model to falsely predict inputs of any class into a certain target class.

$$\boldsymbol{x}^{\boldsymbol{adv}} = \boldsymbol{x} - \varepsilon \cdot \text{sign}\left(\nabla_{\boldsymbol{x}} J\left(\boldsymbol{\theta}, \boldsymbol{x}, y_{target}\right)\right) \tag{2}$$

Formula 2, which is for targeted attack, is slightly different from formula 1. The loss function J calculates the loss between input $\boldsymbol{x}$ and the targeted label class y_{target} instead of y_{true}. Moreover, the perturbation term is subtracted rather than added to the input $\boldsymbol{x}$.

3.4 Training

Having explained our method for adversarial attacks, we will then explain the two types of training that we are going to use: naive training and adversarial training. We are going to train two models with the two types of training and compare their robustness against adversarial attacks.

3.4.1 *Data Split.*

We used stratified data splitting to split the dataset into training set, validation set, and test set. The proportion for each set in order is 8:1:1. So we end up with 800 images in training set, 100 images in validation set, and 100 images in test set.

Table 2: Details of augmentation

Operation	Parameters[1]	Augmentation
Rotate without crop	-180 to 180	10
Crop	0.6, 0.65, 0.7, 0.75, 0.8	5
Shear	-0.25 to 0.25	5
Skew[2]	1	4
Flip	Vertical & Horizontal	2
Zoom	1.1 to 1.5	4
		Total: 30x

3.4.2 *Data augmentation.*

To increase the robustness and generalization ability of our CNN model, we decide to use data augmentation on the training set to increase its diversity. According to table 2, we use 6 augmentation operations, including rotation(without cropping), cropping, zooming, shearing, flipping and skewing. We also randomize the parameters each time for more variations. In total, the training set is augmented 30 times, which yields 24000 new augmented images. After the augmentation, the training set includes 24800 images. Samples of an image after augmentation are shown in figure 3.

3.4.3 *Naive Training.*

Our definition of naive training refers to the normal training of CNN, which uses an unmodified loss function.

In our experiment, we will use cross-entropy loss and Adam optimizer throughout our naive training process. We will use a learning rate of $1 \cdot 10^{-3}$ from epoch 0 to 40 to speed up training. Then, we will lower the learning rate to $5 \cdot 10^{-4}$ from epoch 40 to 60 to prevent overfitting.

[1]The actual parameters are randomly chosen from the range by the Augmentor library unless specified.

[2]Skew has a randomly chosen direction parameter: either left to right, top to bottom, or one of 8 corner directions.

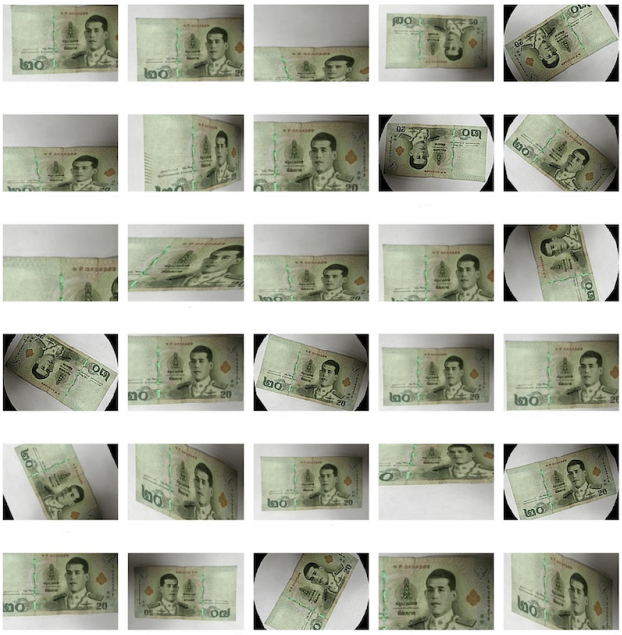

Figure 3: 30-time Augmentation of A Single Banknote Image.

3.4.4 *Adversarial Training.*

Adversarial training allows the network to be more robust against adversarial attacks. Goodfellow et al. [4] mentions a method that uses the FGSM as an adversarial objective function that regularizes the loss function.

$$\tilde{J}(\boldsymbol{\theta}, \boldsymbol{x}, y) = \alpha J(\boldsymbol{\theta}, \boldsymbol{x}, y) + (1 - \alpha) J\left(\boldsymbol{\theta}, \boldsymbol{x}^{adv}, y\right) \tag{3}$$

This adversarial training formula defines a new loss function $\tilde{J}$ as the sum of two loss functions balanced by weight α. The term with coefficient α is the original loss function and the other term with coefficient (1-α) is the loss function whose inputs $\boldsymbol{x}^{adv}$ are perturbed with FGSM as in formula 1. In our experiment, we will use $\alpha = 0$, which means we will train the classifier entirely with adversarial images. In other words, our adversarial training algorithm is the same as a ordinary CNN/ MobileNet network, and the only change we make is feeding the network with adversarial images. We believe that setting $\alpha = 0$ will speed up adversarial training. We will also set $\varepsilon = 1, 2$ for constructing $\boldsymbol{x}^{adv}$. We choose 1 and 2 because they are close to the ε used in adversarial attacks in our experiments. We would have increased ε if training with $\varepsilon = 1, 2$ does not make the model robust enough.

4 RESULTS

4.1 Naive Training and Testing Results

4.1.1 *Naive Training.*

The validation loss is very unstable during the first 20 epochs of training, fluctuating between 0.05 and 1.4 (figure 4). However, they stabilize after 20 epochs and remain between 0 and 0.2. The highest validation accuracy of 100% occurs at epoch 57, so we take

that as our final model. We test the final model on the test set and achieve 100% accuracy on the test set as well.

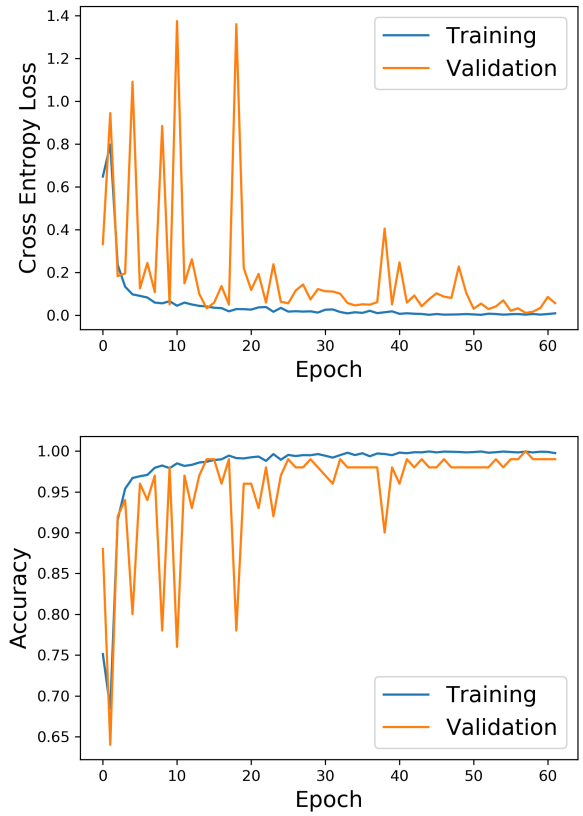

Figure 4: Loss and Accuracy Curves During 60 Epochs of Naive Training

4.1.2 Non-targeted Attack on Naive Model.

We then test our model with adversarial images. As mentioned earlier in formula 1, epsilon is the coefficient that controls the magnitude of perturbation. Larger epsilons generates more visible adversarial patterns on the banknote (figure 5). The goal is to maximize epsilon while not making it too visible to human eyes. So eventually we choose to set epsilon = 2 in both targeted and non-targeted attacks. A non-targeted attack of our model lowers its testing accuracy from 100% to 36%, which produces a fooling rate of 64% (figure 6). The highest single-class fooling rate of 90% was with class 1000, and the lowest rate of 45% was with class 20 and 500.

4.1.3 Targeted Attack on Naive Model.

Table 3 shows the targeted attack success rate of false predicting each true label class to be each target class. In the table, values below 20% are bold. These values shows instances where the targeted attacks are not very strong. For example, the 0 in the third column of the second row is bold, which means that the success rate of a targeted adversarial attack falsely predicting 100 as 50 is 0%.

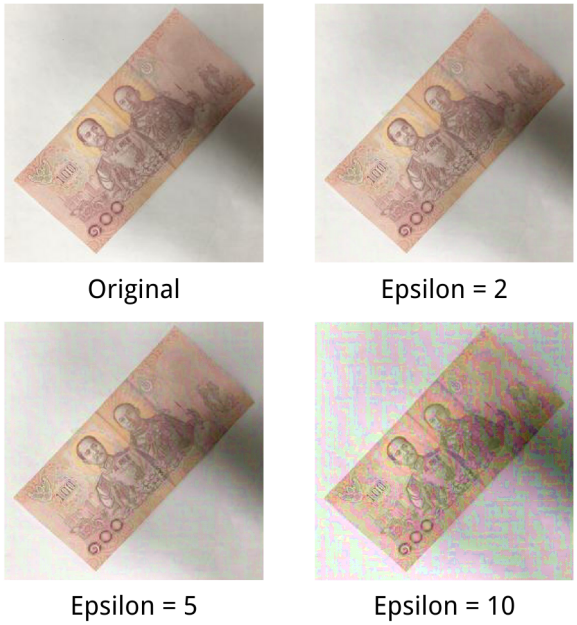

Figure 5: Adversarial Images Generated with Three Different Epsilons

Table 3: Targeted Attack Success Rate By Class on Naive Model ($\epsilon = 2$)

Label/Target	20	50	100	500	1000	Average
20	100%	40%	**5%**	**10%**	**10%**	35%
50	70 %	100%	**0%**	70%	50%	58%
100	40%	**0%**	25%	85%	75%	45%
500	**5%**	75%	20%	35%	55%	38%
1000	65%	35%	45%	75%	100%	64%
Average	56%	50%	**19%**	55%	58%	

Classes (50 & 100) and (20 & 500) are two-way non-attackable, because both converting one class to another or the other way around have a success rate of below 20%. Classes (500 & 100) and (20 & 500) are one-way non-attackable, because only attacking the former true label class to be the latter target class have a success rate of below 20%.

The attack to target classes 20, 50, 500, and 1000 have average success rates from 50% - 60%. However, the attack to target class 100 only has an average success rate of 19% (bottom row).

On average, it is more difficult to attack true label classes 20 and 500, because the average attack success rates of converting them to some other target class is between 35% - 38%. The other true label classes are easier to attack and have attack success rates of 45%-64% (rightest column).

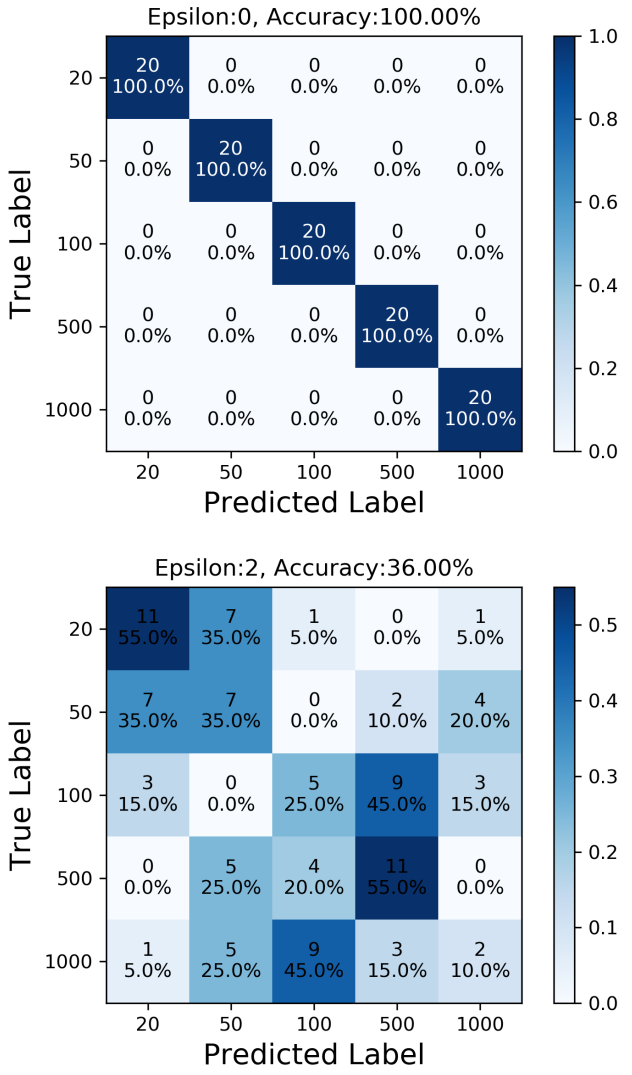

Figure 6: Naive Model's Confusion Matrices Before and After Non-targeted Adversarial Attack.

4.2 Adversarial Training and Testing Results

4.2.1 Adversarial Training.

We begin adversarial training by loading the final naive model trained with 60 epochs, which means the model we start with is already trained to perform with 100% accuracy with clean images.

In the first 10 epochs of adversarial training, the validation loss of 0.7 is consistently lower than the training loss of 0.12, because only the training images are perturbed with adversarial attack. As adversarial training proceeds beyond 20 epochs, both the validation loss and the training loss converge to the low range from 0 to 0.15. The final model after adversarial training achieves an accuracy of 99% on the test set.

4.2.2 Non-targeted Attack after Adversarial Training.

Adversarial training has significantly increased the model's robustness against non-targeted FGSM adversarial attack. Figure 7 show the testing accuracy of the classifier when attacked with epsilon from 0 to 3. Adversarial training with $\varepsilon = 1$ increases the model's testing accuracy by 32% (38%⟶ 70%) when the epsilon used for attack is 2. Moreover, adversarial training with $\varepsilon = 2$ increases that accuracy by 55% (38% ⟶ 93%).

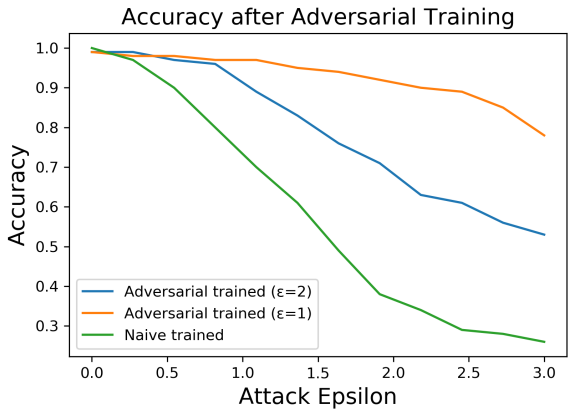

Figure 7: Three Different Model's Performance When Attacked by FGSM Adversarial Attacks with Epsilons from 0 to 3.

4.2.3 Targeted Attack after Adversarial Training.

Table 4 suggests a reduction of targeted attack success rate after adversarial training. The average success rate for attacking each target class drops by 10% - 40%. Overall, adversarial training with $\varepsilon = 2$ reduces the average success rate of targeted attack by 26%, thus improving the model's robustness against targeted attack.

Table 4: Average Targeted Attack Success Rate Before and After dversarial Training with $\varepsilon = 2$

Target Class	20	50	100	500	1000	Average
Naive training	56%	50%	19%	55%	58%	48%
Adversarial training	16%	33%	9%	18%	34%	22%
Reduction	40%	17%	10%	37%	24%	26%

5 DISCUSSION

5.1 Naive Training

Mobilenet model was successfully trained for banknote classification with naive training. As shown in figure 4, despite a few oscillations in the validation curve due to the unaugmented validation set, the two curves displayed a very minimal gap since the training proceeded beyond the 20th epoch, showing that neither overfit or underfit occurred. Our final model achieved 100 % accuracy on the test set with 200 images, meaning that the predicted class of the sample images matched with all the labels in the dataset; therefore, we can conclude that the Mobilenet model was successfully trained for banknote classification.

5.2 Adversarial Attack

Our non-targeted FGSM attack on the naive model achieved high fooling rate, proving that the naive model is vulnerable to adversarial attacks. Kurakin et al. [6] used FGSM to attack an Inception v3 model trained on ImageNet dataset, achieving fooling rates from 63-69% with ϵ in the range [2,32]. In our experiment, we achieved a fooling rate of 64% using the non-targeted FGSM, but did so with $\epsilon = 2$. So we achieved approximately equal fooling rate in our experiment with smaller and less visible perturbation, likely caused by the fact that banknote classification is a simpler task than ImageNet classification.

The important banknote features to our model might include banknote color, special markings and symbols, digit value, etc. From the confusion matrix in figure 6, we discover two possible features in Thai banknote that are most heavily weighted in our model.

(1) Numbers of digits: value classes with the same number of digits have close decision boundaries. This is derived from the fact that the values in the confusion matrix are relatively high between 20 and 50 and between 100 and 500.
(2) Silver mark: figure 8 shows that Thai banknotes 20 and 50 do not have the silver mark that other banknote values have. This pattern difference correlates with the existence of the 2x2 deeper colored region in the top left corner of the confusion matrix. It shows that not having the silver mark might increase the distance from 20 and 50 to other classes. So it is more difficult for an adversarial attack to push 20 and 50 across the decision boundaries of other classes. This discovery could have an impact on future banknote design, which will be discussed in the conclusion section.

Figure 8: Lack of Silver Marks from 20 and 50 Banknotes

For the targeted attack, the overall fooling rate was slightly lower but was enough to show that the perturbation succeeded. However, the targeted attack for target class 100 was less successful because its overall success rate was only 19%. We hypothesize that the failure in targeting 100 was caused by the fact that 100 looked similar to both 500 and 1000. Thus, the distance between 100 and those two classes was small. As a result, when the adversarial attack was trying to push any input into the decision boundary of 100, it was easy for it to arrive at 500 and 1000 instead. However, this hypothesis is contested by the fact that lowering the value of attack epsilon did not raise the success rate of attacking target class 100. Thus, at this point, we cannot offer a definitive explanation for the failure in attacking target class 100 without more experiments.

5.3 Adversarial Training

Our adversarial training successfully increased our model's robustness against FGSM adversarial attack. As expected, in figure 7, we also found a positive correlation between training ϵ and model's robustness against FGSM attack: greater ϵ caused the model to perform with a higher accuracy and a lower rate of being fooled by FGSM attack.

6 CONCLUSION AND OUTLOOK

6.1 Areas for improvements

We have demonstrated the possibility for the adversarial attack in banknote recognition systems, whose effect could be partly alleviated by adversarial training. Here, we will enumerate a few areas for further improvement:

6.1.1 *Adversarial Attack (Representation learning).*

For adversarial attack, as we generally used large epsilon as parameters in FGSM, the step added by the attack might be too large that it altered the class to an excessive level, employing iterative methods might improve the overall fooling rate especially in certain classes of targeted attacks. To conclude the pattern of fooling the classifier, we also should compute a saliency map that visualizes how each pixel in the image is weighted, this will help us understand how are the models reacting to perturbations.

6.1.2 *Defence.*

Our defence method for the single-shot adversarial training was significant but was not bullet-proof. We could experiment with more types of defence methods and employ those that will reduce the fooling rate; By using multiple defence methods at the same time, we not only have more robustness to the simple FGSM because the model can identify noises better but also will perform better in real-world situations because of the unpredictability of the attack type.

6.1.3 *Detection of Adversarial Examples.*

When encountering strong attack with large attack epsilon on the images, we can employ detection only defence methods. These methods only detect whether perturbation exists instead of identifying the original class of the image. They generally have higher success rates comparing to their complete defence counterparts. Normally, blind people cannot verify the result of the recognition algorithm, so it is better to notifying them when the banknote is susceptible to malicious attacks.

6.2 Social Implications

6.2.1 *Banknote Design.*

Policymakers should consider the results of classifying algorithms when designing the features of different banknotes for visually impaired people. From our previous discussion, silver marks and digits seemed to be playing a significant role in our MobileNet's classification; therefore, adding more silver marks with distinct shape and location might increase the overall identification rate significantly and make the CNN model less vulnerable to adversarial attacks. Moreover, the RGB colours and font of number digits should not be too similar, which many banknotes already employ.

6.2.2 *Real-world deployment.*

Many apps that originally employ banknote recognition algorithms could potentially update them to CNN models with the adversarial training. As we tested with MobileNet, currency can be identified with high accuracy even when banknote photos are taken in a variety of conditions, however, their use must be carefully monitored. The nice thing of using CNN models is that one can develop one recognition system that can be used for all different types of currencies, whereas, for the traditional methods that relied on more hand-crafted features, one needs to develop the set of rules specifically for different currency types which could be more time-consuming even though might be less prone to adversarial attacks.

REFERENCES

[1] Peter Ackland, Serge Resnikoff, and Rupert Bourne. 2017. World blindness and visual impairment: despite many successes, the problem is growing. *Community eye health* 30, 100 (2017), 71–73. 29483748[pmid].

[2] Willon Chun Yin. 2019. Recognition system for visually impaired people in Hong Kong by computer vision. https://www.kaggle.com/chunyinlai1997/hong-kong-banknotes-for-object-detection

[3] Arzu Gurdal Dursin. 2012. Information Design and Education for Visually Impaired and Blind People. *Procedia - Social and Behavioral Sciences* 46 (2012), 5568 – 5572. https://doi.org/10.1016/j.sbspro.2012.06.477 4th WORLD CONFERENCE ON EDUCATIONAL SCIENCES (WCES-2012) 02-05 February 2012 Barcelona, Spain.

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. arXiv:1412.6572 [stat.ML]

[5] F. M. Hasanuzzaman, X. Yang, and Y. Tian. 2012. Robust and Effective Component-Based Banknote Recognition for the Blind. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42, 6 (2012), 1021–1030.

[6] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial Machine Learning at Scale. *CoRR* abs/1611.01236 (2016). arXiv:1611.01236 http://arxiv.org/abs/1611.01236

[7] Ji Woo Lee, Hyung Gil Hong, Ki Wan Kim, and Kang Ryoung Park. 2017. A Survey on Banknote Recognition Methods by Various Sensors. *Sensors* 17, 2 (2017). https://doi.org/10.3390/s17020313

[8] Reporter Microsoft. 2017. Seeing AI can now tell people with blindness which banknote they are holding. https://news.microsoft.com/en-gb/2017/12/13/seeing-ai-can-now-tell-people-with-blindness-which-banknote-they-are-holding/

[9] S. Mittal. 2018. Indian Banknote Recognition using Convolutional Neural Network. , 6 pages.

[10] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2015. The Limitations of Deep Learning in Adversarial Settings. *CoRR* abs/1511.07528 (2015). arXiv:1511.07528 http://arxiv.org/abs/1511.07528

[11] Yueqiu Ren, Minh Nguyen, and Wei Qi Yan. 2018. Real-Time Recognition of Series Seven New Zealand Banknotes. *International Journal of Digital Crime and Forensics* 10, 3 (2018), 50–65. https://doi.org/10.4018/ijdcf.2018070105

[12] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial Training for Free! arXiv:1904.12843 [cs.LG]

[13] Z. Solymár, A. Stubendek, M. Radványi, and K. Karacs. 2011. Banknote recognition for visually impaired. In *2011 20th European Conference on Circuit Theory and Design (ECCTD)*. 841–844.

[14] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation* 23, 5 (Oct 2019), 828–841. https://doi.org/10.1109/tevc.2019.2890858

[15] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. arXiv:1312.6199 [cs.CV]

[16] Tenzin. 2018. Indian-Currency-Recognition. https://github.com/10zinten/Indian-Currency-Recognition

[17] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V. Le. 2020. Smooth Adversarial Training. arXiv:2006.14536 [cs.LG]

[18] Chi-Yuan Yeh, Wen-Pin Su, and Shie-Jue Lee. 2011. Employing multiple-kernel support vector machines for counterfeit banknote recognition. *Applied Soft Computing* 11, 1 (2011), 1439 – 1447. https://doi.org/10.1016/j.asoc.2010.04.015