Spring 5-22-2019

Breaking Audio Captcha using Machine Learning/Deep Learning and Related Defense Mechanism

Heemany Shekhar
San Jose State University

Breaking Audio Captcha using Machine Learning/Deep Learning and Related Defense Mechanism

By: Heemany Shekhar
Advisor: Melody Moh
CS 298 Report

Department of Computer Science
San Jose State University
San Jose, CA, U.S.A.

Spring 2019

The Designated Project Committee Approves the Project Titled

Breaking Audio Captcha using Machine Learning/Deep Learning and Related Defense Mechanism

By
Heemany Shekhar

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2019

Dr. Melody Moh	Department of Computer Science, SJSU
Dr. Teng Moh	Department of Computer Science, SJSU
Dr. Chris Pollett	Department of Computer Science, SJSU

ABSTRACT

CAPTCHA is a web-based authentication method used by websites to distinguish between humans (valid users) and bots(attackers). Audio captcha is an accessible captcha meant for the visually disabled section of users such as color-blind, blind, near-sighted users. In this project, I analyzed the security of audio captchas from attacks that employ machine learning and deep learning models. Audio captchas of varying lengths (5, 7 and 10) and varying background noise (no noise, medium noise or high noise) were analyzed. I found that audio captchas with no background noise or medium background noise were easily attacked with 99% - 100% accuracy. Whereas, audio captchas with high noise were relatively more secure with breaking accuracy of 85%. I also propose that adversarial example attacks can be used in favor of audio captcha, that is, adversarial example attacks can be used to defend audio captcha from attackers. I explored two adversarial examples attack algorithms: Basic Iterative Method (BIM) and DeepFool method to create new adversarial audio captcha. Finally, I analyzed the security of these newly created adversarial audio captcha by simulating Level I and Level II defense scenarios. Level I defense is a defense against pre-trained models that have never seen adversarial examples before. Whereas a Level II defense is a defense against models that have been re-trained on adversarial examples. My experiments show that Level I defense can prevent nearly 100% of attacks from pre-trained models. It also proves that Level II defense increases security of audio captcha by 57% to 67%. Real world scenarios such as multi-retries are also studied and related defense mechanism are suggested.

ACKNOWLEDGEMENTS

I would like to thank Dr. Melody Moh for guiding me and helping me identify my master's project. She helped me through times when I was overwhelmed with different topics and was not able to decide my path to graduation. She was always patient and kind when she saw me confused or struggling with any topic; I am very grateful that she agreed to be my advisor.

I am thankful to Dr. Teng Moh for his technical guidance on the project. Through the master's project meetings, he helped me understand ways of performing new research experiments and best ways to analyze the results. He often inspired students to get quality work done which motivated me.

I would also like to thank Dr. Chris Pollett for being my committee member. His classes helped me gain interest in various branches of computer science. He has been very kind and helpful throughout my master's degree education and I am very grateful for his time and support.

Finally, I would like to thank my family who supported and encouraged me to pursue a degree in computer science. I am thankful to them for their moral support and their belief in me.

TABLE OF CONTENTS

Contents

ABSTRACT **4**

TABLE OF CONTENTS **6**

ACRONYMS **7**

LIST OF FIGURES **8**

LIST OF TABLES **9**

CHAPTER 1. Introduction **10**

CHAPTER 2. Related Work **13**

CHAPTER 3. Dataset and Feature Extraction **15**
- ***3.1 Dataset*** ***15***
- ***3.2 Feature Extraction*** ***16***

CHAPTER 4. Attack Models **18**
- ***4.1 Machine Learning Attack Models*** ***18***
- ***4.2 Deep Learning Attack Models*** ***19***

CHAPTER 5. Adversarial Examples **21**

CHAPTER 6. Proposed Solution **22**
- ***6.1 Level I Defense Mechanism*** ***22***
- ***6.2 Level II Defense Mechanism*** ***23***

CHAPTER 7. Performance Evaluation **27**
- ***7.1 Experiment Setup*** ***27***
- ***7.2 Attack Results*** ***27***
- ***7.3 Defense Results*** ***29***

CHAPTER 8. Defense Mechanism in Realistic Scenarios **31**
- ***8.1 Defense Mechanism : Doubling the wait time*** ***33***
- ***8.2 Defense Mechanism : Round Robin / Random captcha generation*** ***34***
- ***8.3 Defense Mechanism : Combination Method*** ***35***

CHAPTER 9. Conclusion **36**

CHAPTER 10. Future Work **37**

References **38**

ACRONYMS

AE: Adversarial Examples

CAPTCHA: Completely Automated Public Turing test to tell Computers and Humans Apart

CNN: Convolutional Neural Network

DL: Deep Learning

DTC: Decision Tree Classifier

GradBoost: Gradient Boosting

ML: Machine Learning

FE: Feature Extraction

kNN: k-Nearest Neighbors

RFC: Random Forest Classifier

SVM: Support Vector Machine

AdaBoost: Adaptive Boosting

BIM: Basic Iteration Method

LIST OF FIGURES

Figure 1: Audio CAPTCHA 11
Figure 2: Feature Extraction 17
Figure 3: Proposed Level I Defense Architecture 23
Figure 4: Proposed Level II Defense Architecture (Re-Training Phase) – Scenario 1 25
Figure 5: Proposed Level II Defense Architecture (Attack/Test Phase) – Scenario 1 25
Figure 6: Proposed Level II Defense Architecture (Re-train with samples from each dataset) – Scenario 2 26
Figure 7: CAPTCHA attacking accuracy for machine learning models 28
Figure 8: CAPTCHA attacking accuracy for deep learning models 28
Figure 9: Attack accuracy on Level I defense 29
Figure 10: Attack accuracy on Level II defense (Scenario 1) 30
Figure 11: Attack accuracy on Level II defense (Scenario 2) 30
Figure 12: Multi Trial Attack Accuracy of Machine Learning Models 32
Figure 13: Multi Trial Attack Accuracy of Deep Learning Models 32
Figure 14: Multi trial attack accuracy on adversarial audio captcha 33
Figure 15: Attack Accuracy on using adversarial datasets in round robin to create audio captcha 35

LIST OF TABLES

Table 1: AUDIO CAPTCHA DATASET 16

CHAPTER 1. Introduction

CAPTCHA stands for Completely Automated Public Turing test to tell Computers and Humans Apart. Captcha is a computer-generated challenge that is easy to solve by humans but difficult to solve by current computing systems. This characteristic of the captcha is often referred to as the captcha paradox, where a machine creates and scores a test that it itself should not pass [1]. It is a web-based authentication method and also the first line of defense used by websites to prevent bots from creating fake accounts or spamming. A bot is a malicious program that can perform repeated tasks automatically over the internet and thus creating problems in the network.

The web is a universal medium for communication and sharing that should be accessible by every human alike. Deploying only visual captchas creates a considerable obstacle to a certain group of users, such as the visually impaired, color-blind, near sighted users [2]. This necessitates the need for more accessible captchas such as audio captcha shown in Figure 1. But, with the advancements in machine learning and deep learning it is now possible to train speech to text models that can attack audio captcha and solve them like humans. The attackers can train their own attack models or use publicly available models to create DoS, DDoS attacks that compromise the availability of the websites that use audio captchas. Hence, it is important to study and analyze the different models and techniques that can be used to break audio captcha and the ways in which these attacks can be prevented.

Breaking audio captcha means that an attack model (could be a machine learning model or deep learning model) is able to identify what digits are spoken in the audio captcha. For example, if the digits 3 4 5 2 1 are spoken in the audio captcha, the attack model should identify the digits 3 4 5 2 1. If the attack models are correctly able to identify each digit , then , we can say that the attack model has broken the audio captcha. We can also say that the attack model was successful at attacking or breaking the audio captcha. In this report, attacking and breaking have been used with the same meaning throughout.

Adversarial Examples Attacks are algorithms that create well-crafted noise intended to fool a deep learning or machine learning model. Currently, these adversarial examples are used to attack models, hence the name Adversarial Examples Attack. These adversarial examples are crafted in a way such that the perturbations or changes to the image or audio is imperceptible to humans, but they can fool the models into classifying them incorrectly. The main contributions of this project are as follows:

- Audio captcha of varying lengths and background noise are created. This allows us to analyze a variety of audio captcha and understand what features make an audio captcha more secure.
- Multiple machine learning, and deep learning models are trained to attack the audio captcha. Experiment results prove that audio captchas are not secure.
- A novel method of creating audio captcha is proposed. This method uses adversarial audio to create audio captcha. Normal audio captcha is generated using recorded human voice (reading out numbers or alphabets) and adding random noise to it.
- Security of the newly generated adversarial audio captcha is studied and analyzed. Resilience of adversarial audio captcha against machine learning and deep learning attacks is analyzed by simulating Level I and Level II defense scenarios. Defense against pre-trained attack models that were trained on normal audio datasets and have never encountered adversarial examples before is referred to as Level I defense mechanism in this project. Level II defense is simulated by re-training attack models on adversarial audio captcha as well.

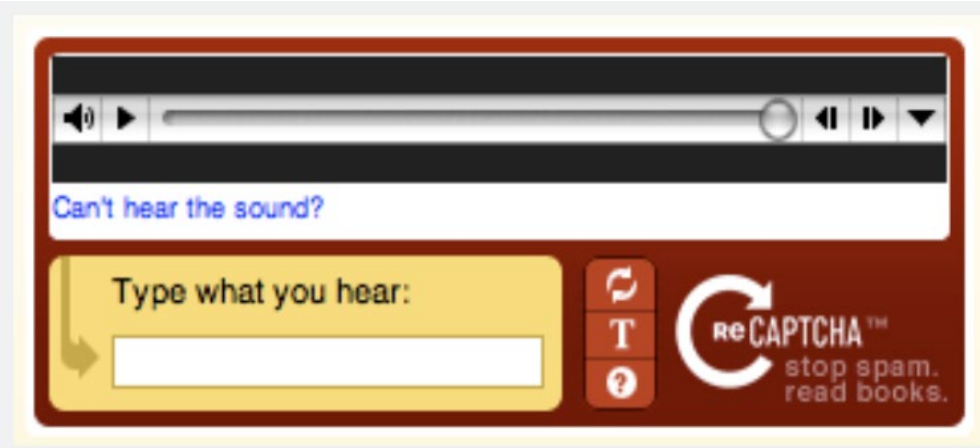

Figure 1: Audio CAPTCHA

Section II briefly describes the background and previous work in the field of attacking audio captcha that helped in getting started with the project. Section III describes the datasets used and their feature extraction process. In Section IV, details of the attack models and their architecture are discussed. Section V reviews the concepts of adversarial examples. In Section VI, the proposed solution of this project is explained. In Section VII, the experimental setup and results from attacking audio captcha are described. Also, the attack results on Level I and Level II defense mechanism are discussed here.

CHAPTER 2. Related Work

This section reviews research work that has previously been performed for attacking audio captcha. In reference [3] the authors break the task of solving an entire captcha into sub tasks of first segmenting the audio file and then predicting spoken digits/alphabets. Segmentation and labelling are performed using manual effort for training dataset whereas a fixed size window is used for segmenting the test data audio. The project implements AdaBoost with decision stumps, SVM with RBF kernel and K Nearest Neighbors (K-NN) to predict the captchas. For an exact match condition SVM gives the highest accuracy of 67%. Whereas, for one mistake passing condition, SVM gives the highest accuracy of 92% among all models. K-NN gives the lowest accuracy for both one mistake and exact match passing conditions.

In reference [4], the authors aimed to break eBay's audio captcha using their prototype *Decaptcha*. At the time of the experiments eBay used audio captcha with 6 digits between 0 to 9 spoken by multiple people with different genders, accents and background noise levels. Using Decaptcha the paper was able to break 75% of eBay's audio captchas.

Reference [5] investigates the security of Google's continuous audio captcha using Hidden Markov Models. The continuous audio captcha contained two kinds of distortion: *challenge distortion* and *digit distortion*. Challenge distortion is a static noise that is added for the entire duration of the audio, whereas digit distortion is a noise applied to each digit. The audio captcha contained nine spoken digits clustered in three groups of three digits each. The results are evaluated off-by-one accuracy, strict accuracy (all digits labelled correctly), per-cluster accuracy, per-digit accuracy and N-segment accuracy. The off-by-one accuracy for new test data is 52% but the strict accuracy stands at 17%. The per-digit classifier does better at 76%. The authors also experimented by providing the model with the ground truth number of digits in each cluster, this increased off-by-one accuracy for new test data to 58% and strict accuracy to 27%.

Reference [2] proposes that off-the-shelf (OTS) speech recognition systems can be misused by

attackers to break audio captchas with very high accuracy. The paper tested the speech recognition models on various popular captcha service providers such as Recaptcha v2.0, Recaptcha v2.1, Apple, etc. Google's Cloud Speech to text (US accent) performs the best with an accuracy of 98.3% on its own Recaptcha v2.0 and with an accuracy of 83.9% on Recaptcha v2.1. Recaptcha audio captcha have no noise in the background and are generally 10 digits long.

CHAPTER 3. Dataset and Feature Extraction

3.1 Dataset

To completely analyze security of audio captcha, 9 different audio captcha datasets are generated as shown in Table I. To create these datasets, I collected audio of people speaking out the digits [0-9]. Six different speakers are used with different genders and accents. I implemented algorithms that joined these audio files to create audio captcha of varying lengths. The audio files are mixed at random intensities and random speeds to create realistic audio captcha. Background noise such as static noise, bell or beeping noise, etc. are scattered throughout the length of the audio captcha. All audio captchas are made up of spoken digits between 0 to 9.

Broadly, the 9 datasets can be placed into three major buckets based on the kind of background noise present in the dataset. These three buckets are: no background noise datasets, medium background noise datasets and high background noise datasets.

No Noise Datasets: Three datasets with no noise in the background are referred to as no noise datasets. These three datasets contain audio captchas of length 5, 7 and 10 respectively.

Medium Noise Datasets: Another three datasets of length 5, 7 and 10 with some noise are referred to as medium background noise datasets. Medium background noise refers to addition of static noise and few bells/beeps in the entire duration of the audio captchas.

High Noise Datasets: Lastly, three datasets of length 5, 7 and 10 are created with high noise in the background. High noise refers to static white noise, bell/beep sound in addition to audio files played in reverse at random speeds in background. The combination of these noise is spread across the entire duration of the audio captcha.

Irrespective of the intensity of background noise, it is ensured that the audio captcha is clearly understandable to human ears. To ensure legibility, samples from each dataset is manually played and tested.

Table 1: AUDIO CAPTCHA DATASET

Dataset_Names	Audio captcha datasets				
	Length	*Background Noise*	*Accents*	*Random Speed*	*Random Intensity*
NoNoise_5	5	None	True	True	True
NoNoise_7	7	None	True	True	True
NoNoise_10	10	None	True	True	True
MediumNoise_5	5	Medium	True	True	True
MediumNoise_7	7	Medium	True	True	True
MediumNoise_10	10	Medium	True	True	True
HighNoise_5	5	High	True	True	True
HighNoise_7	7	High	True	True	True
HighNoise_10	10	High	True	True	True

3.2 Feature Extraction

Mel Frequency Cepstral Coefficient (MFCC) is considered a standard method to extract features from audio and is widely used in automatic speech-to-text models [4, 6]. The Mel Frequency uses a non-linear scale, which is approximately linear for frequencies below 1 KHz and logarithmic for frequencies above 1 KHz [7]. The motivation for such scale comes from analyzing how human auditory systems are selective towards sound waves of different frequencies. The Mel-scale relates perceived frequency to the actual measured frequency. The formula to convert a frequency to Mel-Scale is as below:

$$F_{Mel}(f) = 1125.log_{10}\left(1 + \frac{f}{700}\right) \qquad (1)$$

The discrete Fourier Transform of a frame is obtained using the following formula below where 1≤ k ≤ K, K is the length of the discrete Fourier Transform and h(n) is an N sample long analysis window.

$$S_i(k) = \sum_{n=1}^{N} s_i(n)h(n)\mathrm{e}^{-j2\pi kn/N} \quad (2)$$

MFCC is used for extracting features in this project. Figure 2 shows the process followed to extract MFCC features from audio files.

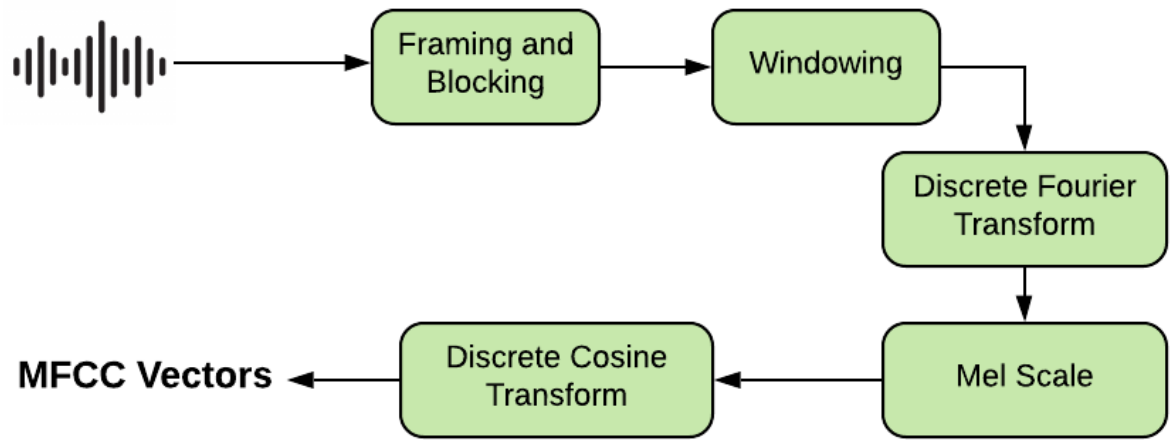

Figure 2: Feature Extraction

CHAPTER 4. Attack Models

4.1 Machine Learning Attack Models

I used four different machine learning models to attack audio captchas. These models were Support Vector Classifiers (SVC), Random Forest Classifier (RFC), AdaBoost with RFC as base classifier, and Gradient Boosting algorithm.

Support Vector Classifier – SVC is a powerful supervised learning algorithm that is used for classification, regression and novelty detection. The goal of SVC is twofold, it first aims to find a hyperplane that linearly separates the training data. Then, it aims to maximize the minimum distance between the hyperplane and any data of the training set, that is, it tries to maximize the margin [8]. Often to achieve this goal, a kernel function is used that moves data to a higher dimension. SVCs have been used to differentiate between noise and target audio data earlier [9] implements SVC for audio stream recorded in vehicles during different driving conditions. I used SVC for multi-class classification of audio data.

Random Forest Classifier (RFC) – Random Forest Classifier algorithm is an ensemble method for classification or regression that uses decision tree as base classifier. [10] uses RFC for wildlife intruder detection by classifying audio consisting of birds, gunshots, chainsaw, tractors and human voice. For this project, I also applied RFC to our audio captcha dataset to classify digits from 0 to 9.

Adaptive Boosting Algorithm – Also referred to as AdaBoost algorithm, it is used in conjunction with other machine learning models (often referred to as weak learners). AdaBoost is an adaptive learning model that uses weak learners to classify data points. The misclassified data points are assigned higher weights in the subsequent iterations till the algorithm learns to best classify all data points. [11] uses AdaBoost to classify audio data in complex audio environments. In this project, I use AdaBoost in conjunction with RFC to classify audio captcha data.

Gradient Boosting Algorithm – Similar to RFC and AdaBoost, gradient boosting algorithm also is an ensemble method that uses weak base learners, generally decision tree with fixed size, to create

stronger prediction models. This method works in stages to optimize the arbitrary loss functions that are differentiable. I experimented with gradient boost algorithms to classify audio captcha dataset.

4.2 Deep Learning Attack Models

Convolutional Neural Networks (CNNs) have proven to perform extremely well in the field of image classification. Recent work in audio classification [12] shows promise in the field of audio classification as well. I used four different CNN models to attack audio captchas. I did not use Recurrent Neural Networks (RNNs) since RNNs are meant for time-series data where there is a relation between datapoints. In case of audio captchas, the digits are spoken randomly and have no relation among each other. Hence, I experimented with below four CNN models: cnn-1, cnn-2, cnn-3, vgg.

cnn-1 - [13] uses cnn models for small footprint keywork spotting task in audio. I implemented the cnn architecture which is referred to as cnn-one-fpool and call it cnn-1. This architecture consists of one cnn layer and two dense layers followed by a SoftMax layer.

Conv2D(54, kernel_size=(2, 2), activation='relu')
Dense(32, activation='relu')
Dense(128, activation='relu')
Dense(128, activation='relu')
Dense(10, activation='softmax')
loss=categorical_crossentropy
optimizer=Adadelta

cnn-2 – I used a second cnn model again from reference [13] which consists of two convolution layers and two dense layers followed by a SoftMax layer. This model is similar to the model named cnn-trad-fpool3 except that I have added an additional dense layer to the architecture.

Conv2D(32, kernel_size=(2, 2), activation='relu')
Conv2D(64, kernel_size=(2, 2), activation='relu'))
MaxPooling2D(pool_size=(2, 2)))
Dense(32, activation='relu'))
Dense(128, activation='relu'))
Dense(10, activation='softmax'))
loss=categorical_crossentropy
optimizer=Adadelta

cnn-3 – I created my own cnn architecture which is an extension of cnn-2 model. cnn-3 model consists of three convolution layers and two dense layers followed by a SoftMax layer. Below is a representation of the model.

Conv2D(32, kernel_size=(2, 2), activation='relu')
BatchNormalization()
Conv2D(48, kernel_size=(2, 2), activation='relu'))
BatchNormalization()
Conv2D(120, kernel_size=(2, 2), activation='relu'))
BatchNormalization()
MaxPooling2D(pool_size=(2, 2)))
Dropout(0.25))
Dense(128, activation='relu'))
Dropout(0.25))
Dense(64, activation='relu'))
Dropout(0.4))
Dense(10, activation='softmax'))
loss=categorical_crossentropy
optimizer= Adadelta

vgg – To experiment with much larger cnn models I chose the VGG classifier from [12] which consists of 16 layers (13 convolution layers and 3 dense layers) followed by a SoftMax layer. This is the standard VGG model with no modifications.

CHAPTER 5. Adversarial Examples

Machine learning, and deep learning models are highly expressive models that learn from data. But studies in [13] show that the models learn from the space rather than the individual units. Also, the input-output mapping is quite discontinuous which leaves room for introducing small imperceptible perturbation that can result in misclassification of data. Utilizing this understanding [14, 15] created adversarial examples attack algorithms that generate adversarial examples. Adversarial examples are created by adding well-crafted noise/perturbations to original/clean data which are not interpretable by humans. But, these adversarial examples can fool machine learning and deep learning models to misclassify them.

In our use case, deep learning models or machine learning models are used to attack audio captcha, hence, it makes sense to explore the use of adversarial examples to defend audio captcha from attacks. I created adversarial audio using two popular adversarial examples attack algorithms, Basic Iterative Method (BIM) and DeepFool method.

Basic Iterative Method (BIM): BIM method iteratively implements Fast Gradient Sign Method [15] to create adversarial audio. The original data is updated based on the gradient value of the loss. In equation 3, e stands for epsilon.

$$audio_{adv} = audio_{original} + e * sign(grad) \qquad (3)$$

Deepfool Method: To create adversarial examples, deep fool method aims to minimize L2 distance but also switch classes. It iteratively finds the minimum perturbation that will result in class switch. In each iteration the method calculates L2 distance between the adversary and the original sample as shown in equation 4.

$$NextIter_{inp} = inp_{curr} + perturbation_{curr} \qquad (4)$$

CHAPTER 6. Proposed Solution

By performing attack experiments on audio captcha this project proves that no noise and medium noise datasets are very easy to attack, whereas high noise audio captcha must be longer to ensure some security. The findings of the project show that there is need for more secure forms of audio captcha. To do this, the following is proposed:

- A Level I defense mechanism is proposed as shown in Figure 3, that defends against pre-trained machine learning or deep learning attack models that were trained on normal audio datasets and have never encountered adversarial examples before.
- Also, a Level II defense mechanism is proposed that is a defense against re-trained machine learning or deep learning models as shown in Figure 4 and Figure 5. These attack models have retrained themselves on adversarial examples.

6.1 Level I Defense Mechanism

I use Adversarial Examples attack algorithms such as BIM and Deep Fool to create new adversarial audio. These adversarial audios are joined together to form audio captcha of varying length. The pre-trained attack models such as SVM, RFC, CNNs have not been trained on any adversarial examples. The hypothesize is that when such pre-trained models attack the newly generated adversarial audio captcha, the attacks should fail. To simulate these experiments, I used all eight pre-trained attack models that were trained on normal audio datasets to attack the newly generated adversarial audio captcha.

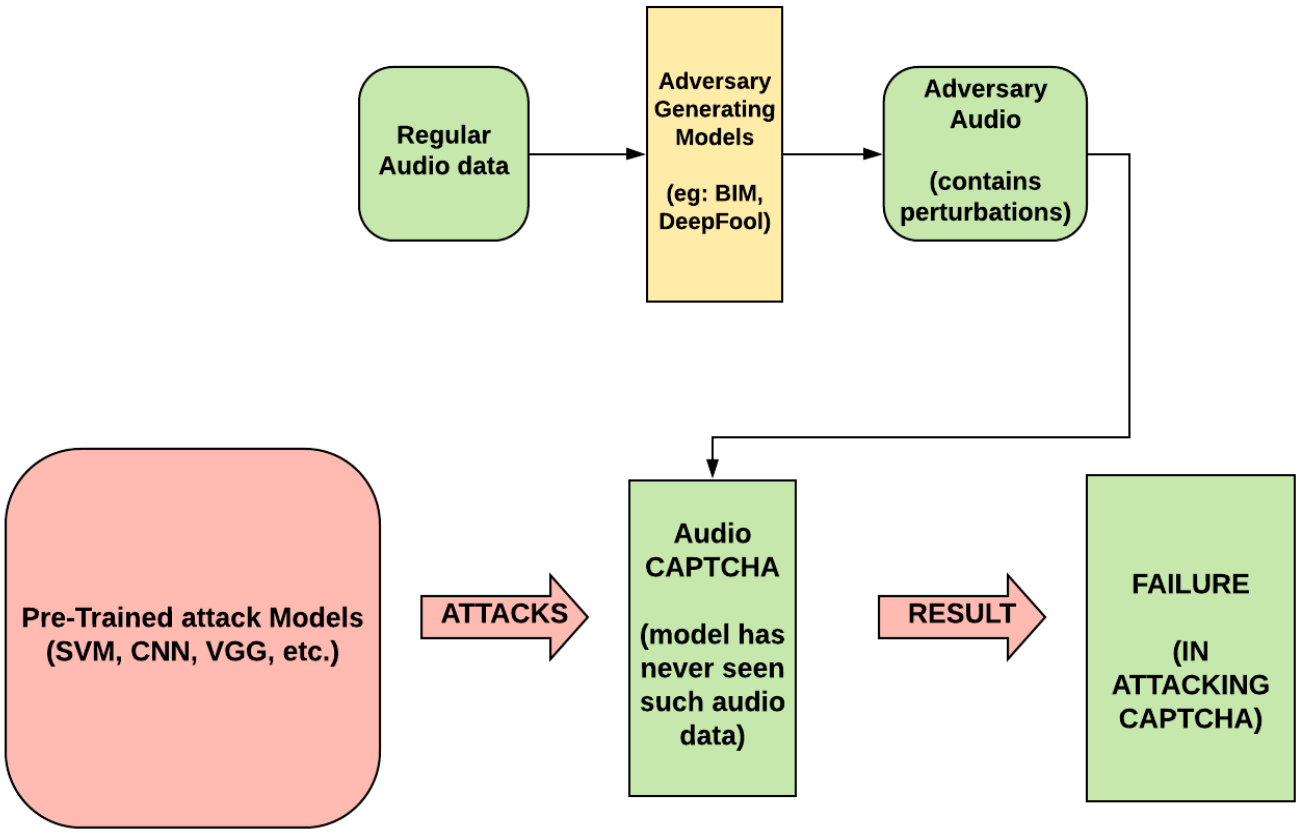

Figure 3: Proposed Level I Defense Architecture

6.2 Level II Defense Mechanism

Reference [16] shows that retraining deep learning models or machine learning models on all adversarial examples is very costly and not completely possible. This is because infinite number of adversarial examples can be created with extremely small perturbations to the original data. This means that retraining on such data is both costly and never fully possible. On the contrary, creating adversarial audio is not costly, hence we can create multiple sets of adversarial audios using different adversary creation techniques. This concept is used to create a level II defense strategy.

To create a level II defense, multiple sets of adversarial examples should be created using different adversary generation techniques (e.g.: BIM, DeepFool) and different amounts of perturbations. Audio captchas of length 5, 7, or 10 can be created by randomly selecting from any of these adversarial audio

sets.

To analyze the security of level II defense, I set up the experiments in two different ways as mentioned below in Scenario 1 and Scenario 2.

Scenario 1: The experiment is set up as shown in Figure 4 and Figure 5. This scenario assumes that the attacker does not have access to all versions/types of adversarial audio. Hence, the attacker re-trains its attack model on few adversarial audio datasets. Figure 4 shows the re-training phase for the attacker. The attacker can retrain their attack models on normal audio and a subset of adversarial audio datasets. In the experiments, I chose two adversarial audio sets at random for re-training. A five-fold cross validation method was used, where I performed the experiments five times. Each time a different subset of adversary audio datasets was used for retraining along with the normal audio datasets.

Figure 5 shows the attack phase when, the attack model from Figure 4, is used to attack audio captcha. To create the test audio captcha, the algorithm randomly select audio from any of the adversarial sets and join them to form audio captcha of length 5, 7 or 10.

Scenario 2: This scenario simulates a real-world situation where the attacker accumulates the adversarial audio captcha over time and retrains their model on these adversarial audio data as well. In this scenario, we believe that the attacker can retrain their attack model on all available audio datasets as shown in Figure 6. To do this, the attacker either collects the adversarial audio samples or creates its own adversarial audio samples of all possible types. This scenario can give us a concrete proof of how secure adversarial audio captcha are. To understand the robustness of adversarial audio captcha against attacks, the experiment is set up as following: The attacker collects/creates samples of all adversarial audio datasets. The sample size ranges from 5% of the audio dataset up to 80% of the audio dataset. The remaining 20% is used for testing, the test process is same as shown in Figure 5. To create the test audio captcha, the algorithm randomly select audio from any of the adversarial sets and join them to form audio captcha of length 5, 7 or 10.

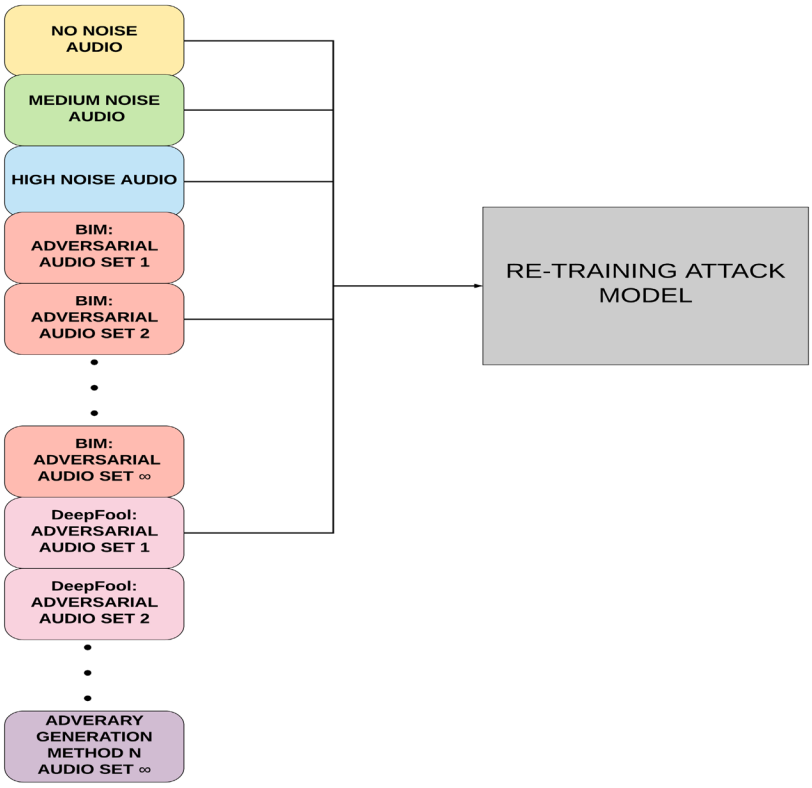

Figure 4: Proposed Level II Defense Architecture (Re-Training Phase) – Scenario 1

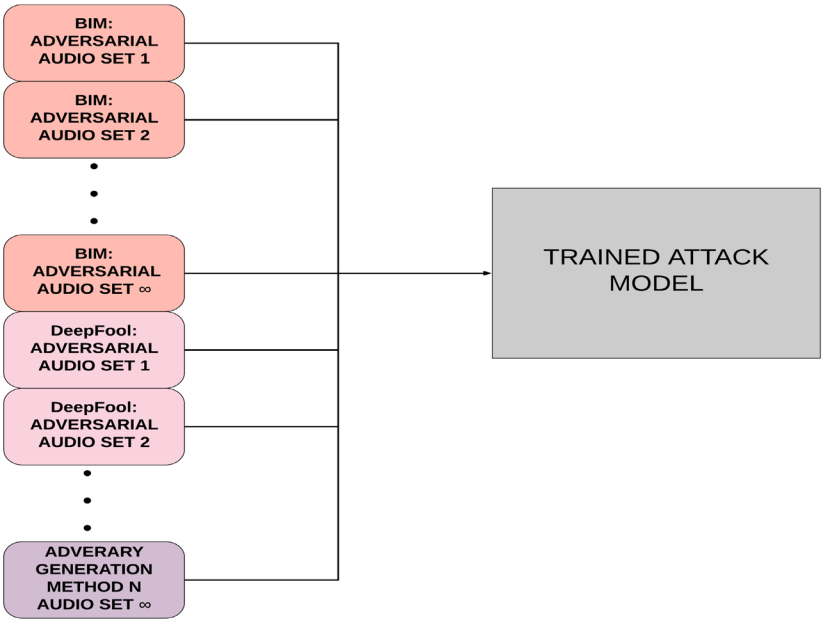

Figure 5: Proposed Level II Defense Architecture (Attack/Test Phase) – Scenario 1

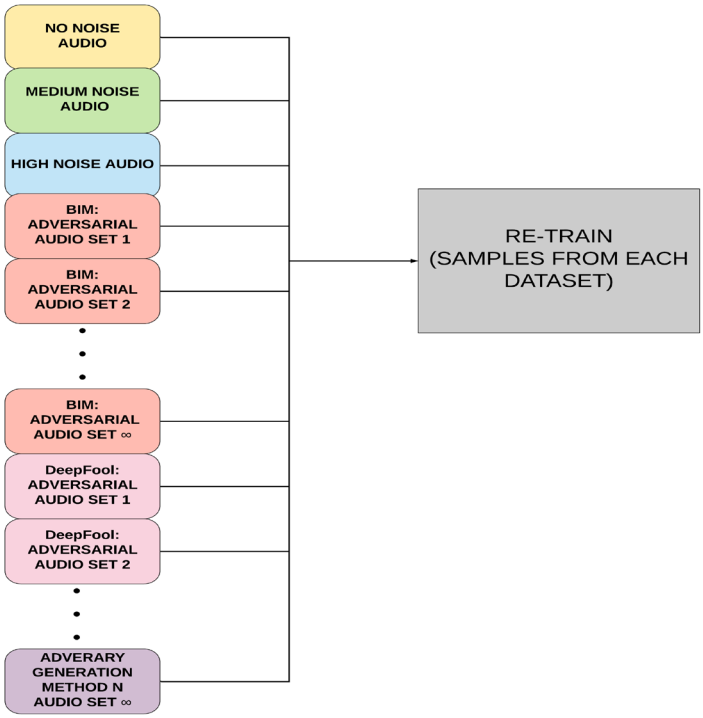

Figure 6: Proposed Level II Defense Architecture (Re-train with samples from each dataset) – Scenario 2

CHAPTER 7. Performance Evaluation

In this section, the experiment results from attacking and defending audio captcha are discussed

7.1 Experiment Setup

For attacks I use SVM, RFC, AdaBoost, Gradient Boost, three versions of CNN models as discussed in Section IV and VGG model. For defense, I use adversarial generation methods BIM and DeepFool. The architecture of defense is shown in Figures 3, 4, 5 and 6. To run the experiments I used python, keras and pytorch for creating both attack and defense models. I used Intel® Core™ i7 with 2.80 GHz CPU and 16 GB RAM.

7.2 Attack Results

Figure 7 displays the attacking accuracies of machine learning models on audio captchas. No noise audio captchas of any length can be attacked with 100% accuracy. Medium noise audio captcha can be attacked with 98% (length 5) to 95.64 % (length 10) accuracy. High Noise audio captcha can be attacked with highest accuracy of 62.61% (length 5) to 29.96 % (length 10).

Figure 8 displays the attacking accuracies of deep learning models on audio captchas. No noise audio captchas of any length can be attacked with 100% accuracy. Medium noise audio captcha can be attacked with 98.40% (length 5) to 97 % (length 10) accuracy. High Noise audio captcha can be attacked with highest accuracy of 84.80% (length 5) to 75.25 % (length 10).

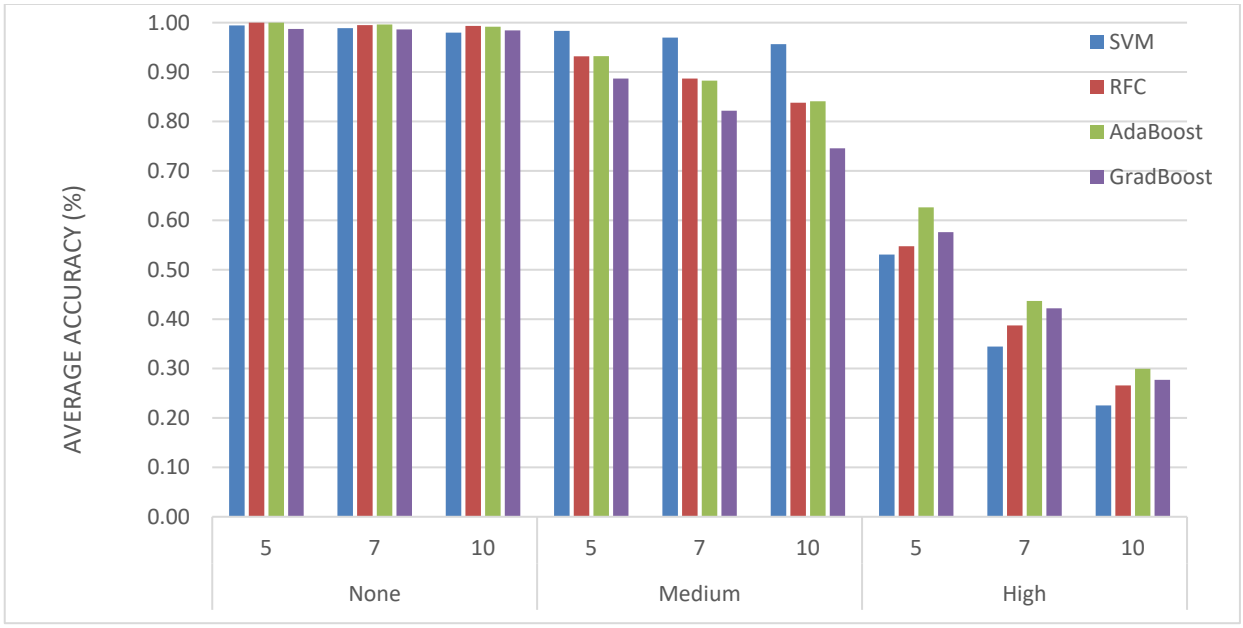

Figure 7: CAPTCHA attacking accuracy for machine learning models

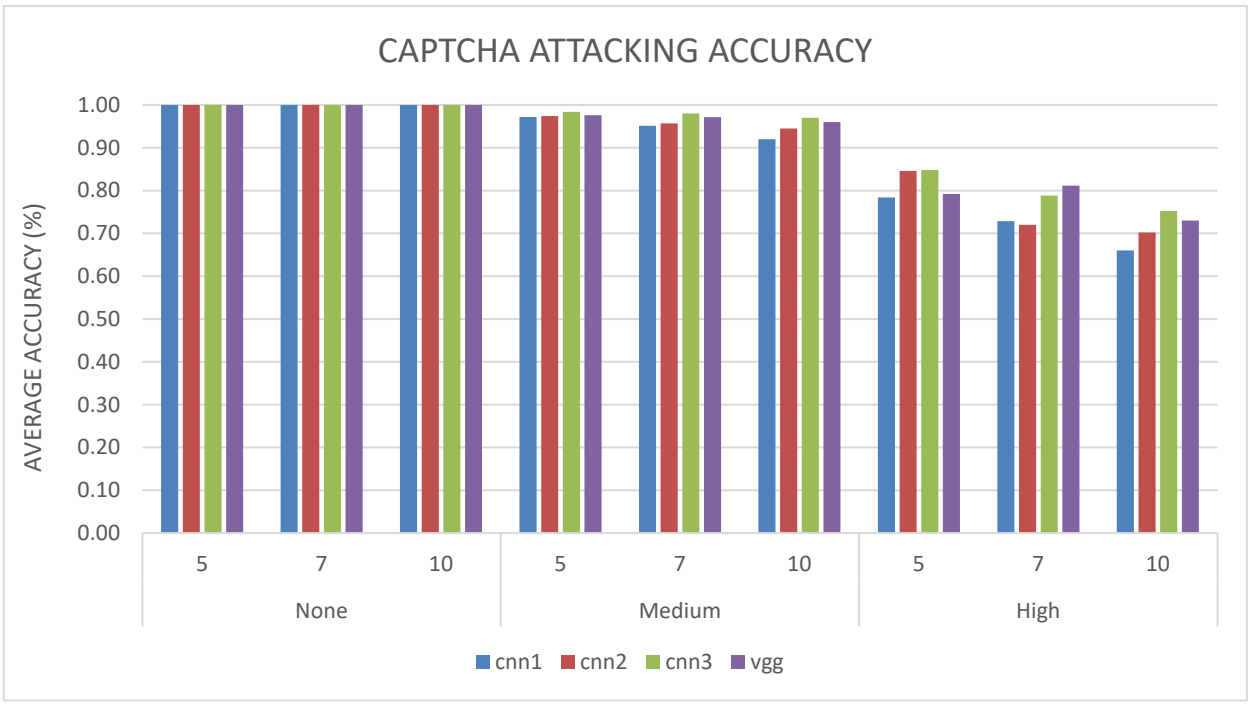

Figure 8: CAPTCHA attacking accuracy for deep learning models

7.3 *Defense Results*

Below are the attack accuracies after applying Level I and Level II defense mechanism. Figure 9 shows the attack results after Level I defense mechanism has been applied to audio captcha, that is, the attacks have been performed on adversarial audio captcha generated using BIM and DeepFool methods.

Figure 10 shows the attack results on Level II defense – Scenario 1. For audio captcha of length 5 the attacking models could break the captcha with average accuracy of 36%. Audio captcha of length 7 could be broken with average accuracy of 31%. Whereas audio captcha of length 10 could be broken with average accuracy of 25%

Figure 11 shows that with 5% samples of each adversarial audio dataset the attack accuracy is only 20% for captcha of length 10 and 40% for captcha of length 5. The attack accuracy increases as the sample space increases to 35%, where the attack accuracy is 65.11% for length 10 captcha and 78% for length 5 captcha. After this, the increase in attack accuracy stagnates and even decreases slightly after 55%. Even after all available audio datasets (80% of each audio dataset) are used to re-train the attack models, the attack accuracy is lower (72% for len 10, 82% for len 5) than the attack accuracy on normal audio datasets (100% on no/medium noise captcha of any length and 75% for length 10 high noise, 85% for length 5 high noise). The results show that adversarial audio captcha is more robust as compared to normal audio.

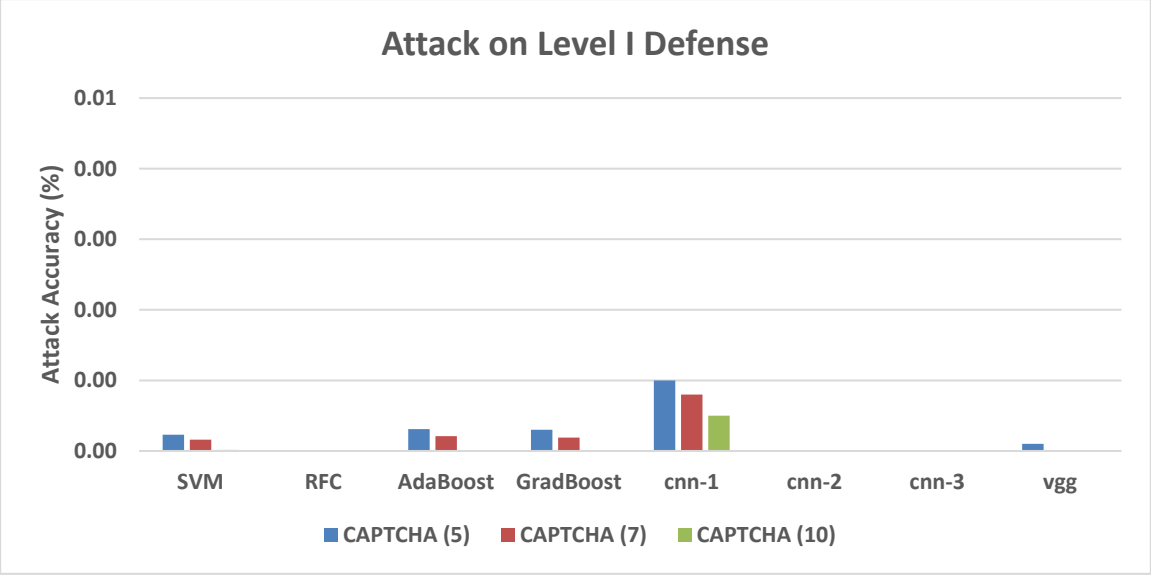

Figure 9: Attack accuracy on Level I defense

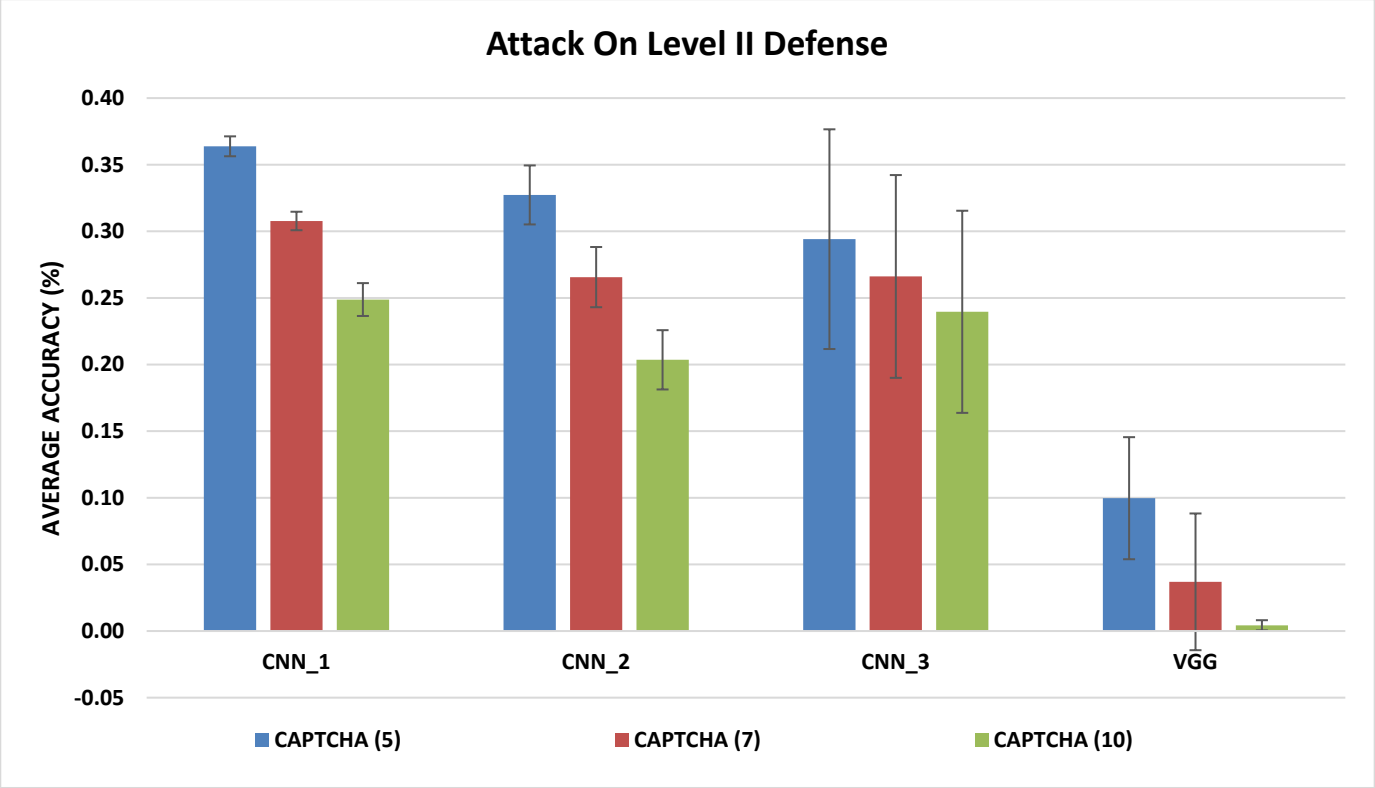

Figure 10: Attack accuracy on Level II defense (Scenario 1)

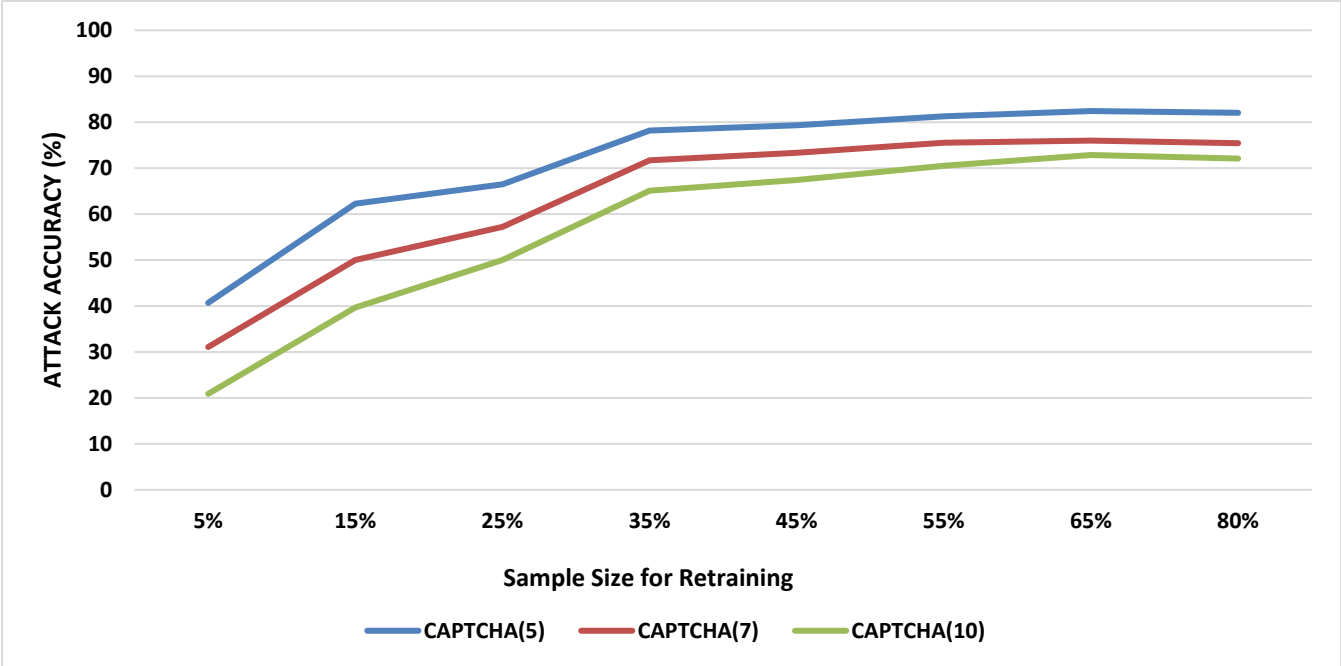

Figure 11: Attack accuracy on Level II defense (Scenario 2)

CHAPTER 8. Defense Mechanism in Realistic Scenarios

In the real-world, audio captchas, specially the captchas with background noise are tuff to understand for humans as well. To ensure, that humans do not suffer due to such difficulties, most websites offer up to three chances to correctly identify the digits in an audio captcha. This feature can have implications for breaking audio captcha as well, since now the attacker too gets up to three trials to break the audio captcha. In this section we discuss the attack accuracy of models considering the multiple retry feature and propose few related defense mechanisms.

Before moving further, we discuss the definitions of independent and dependent events and explore some basic understanding of probability related to the project. Two events are independent of each other if, the probability that one occurs does not impact the probability of the other event. For example, rolling a die and flipping a coin. Both events are completely independent of each other. Whereas, two events are called dependent events if the probability of one event's occurrence is dependent on another event. For example, if we draw two cards from a deck of 52 cards without replacing them, then, the probability of drawing an ace on the second draw depends on whether an ace was drawn on the first try. In the scenario of an audio captcha with three retries, we can say that the three retries are independent of each other. To be successful at breaking audio captcha, the attacker can get three trials to identify the captcha correctly. Hence, if the probability of breaking an audio captcha in one trial is x, then the probability of breaking the audio captcha in at least one of the three trials can be calculated as following.

Probability of not being successful in all three trials = $(1 - x)^3$

Probability of being successful in at least one of three trials = 1 - $(1 - x)^3$

Using this information, we can recalculate the attack accuracy of all machine learning and deep learning models. We get the attack accuracy as shown in Figure 12 and 13. Figure 12 shows the attack accuracy using machine learning models and Figure 13 shows the attack accuracy using deep learning models. The results show that, given three trials all audio captcha including no noise, medium noise and high noise can be broken with near perfect (almost 100%) accuracy. Figure 14 shows the impact of three trials on the

newly generated adversarial audio captcha. The results prove that given three trials adversarial audio captcha are also not strong defense. Hence, the need to handle the issue of multi-retries. In the following sub-sections, we discuss defense strategies for multi-retries.

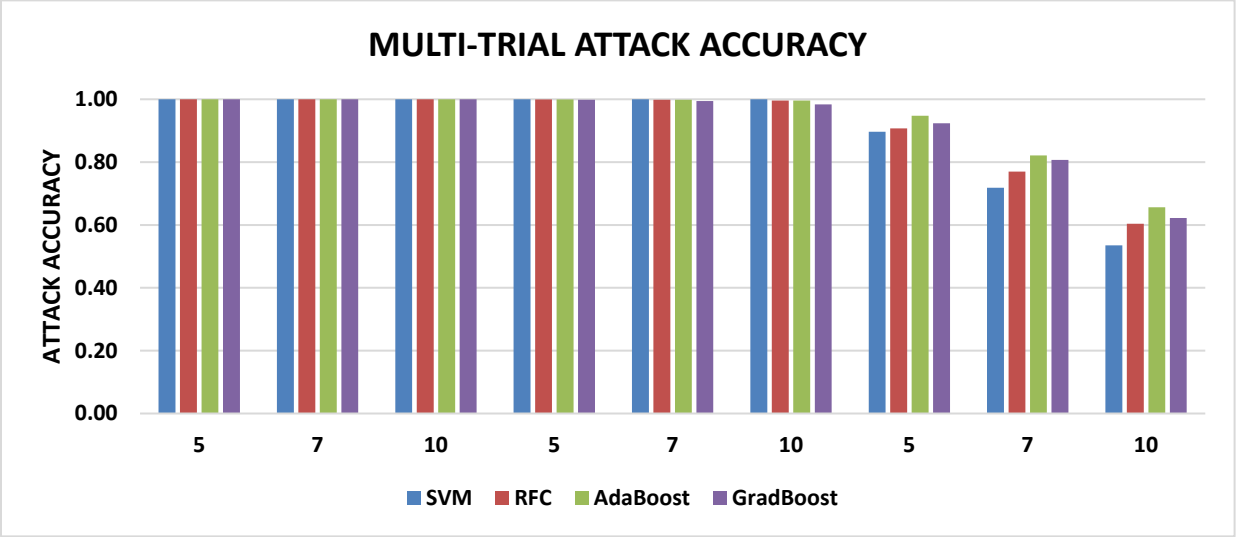

Figure 12: Multi Trial Attack Accuracy of Machine Learning Models

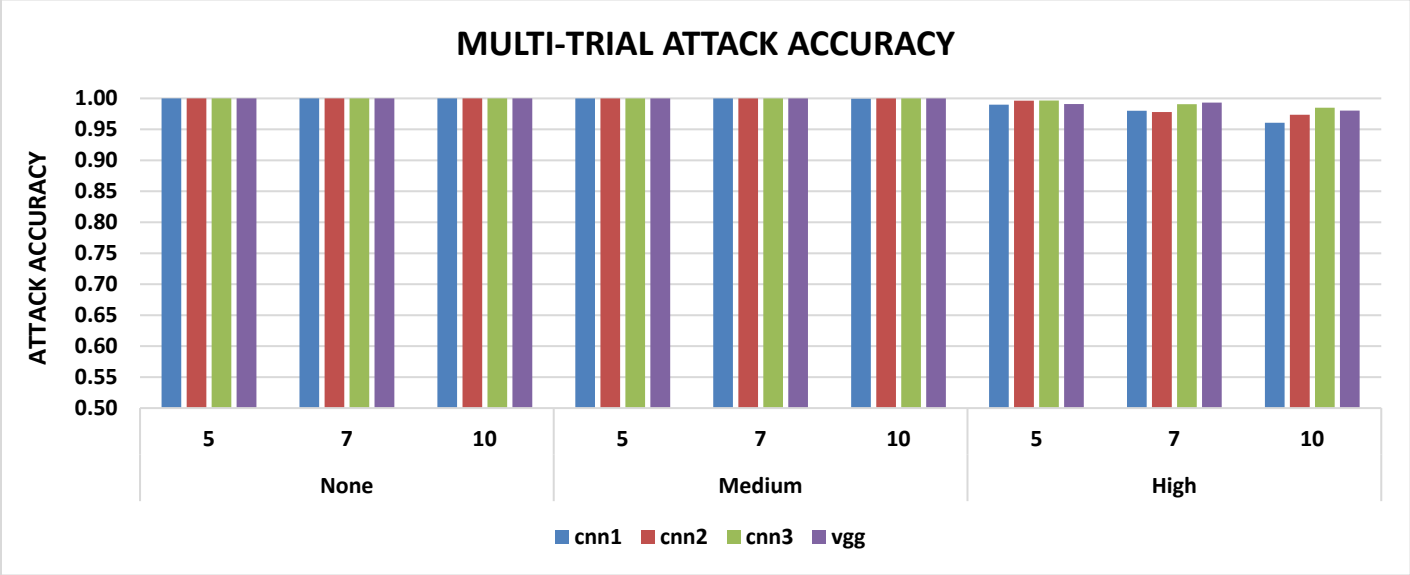

Figure 13: Multi Trial Attack Accuracy of Deep Learning Models

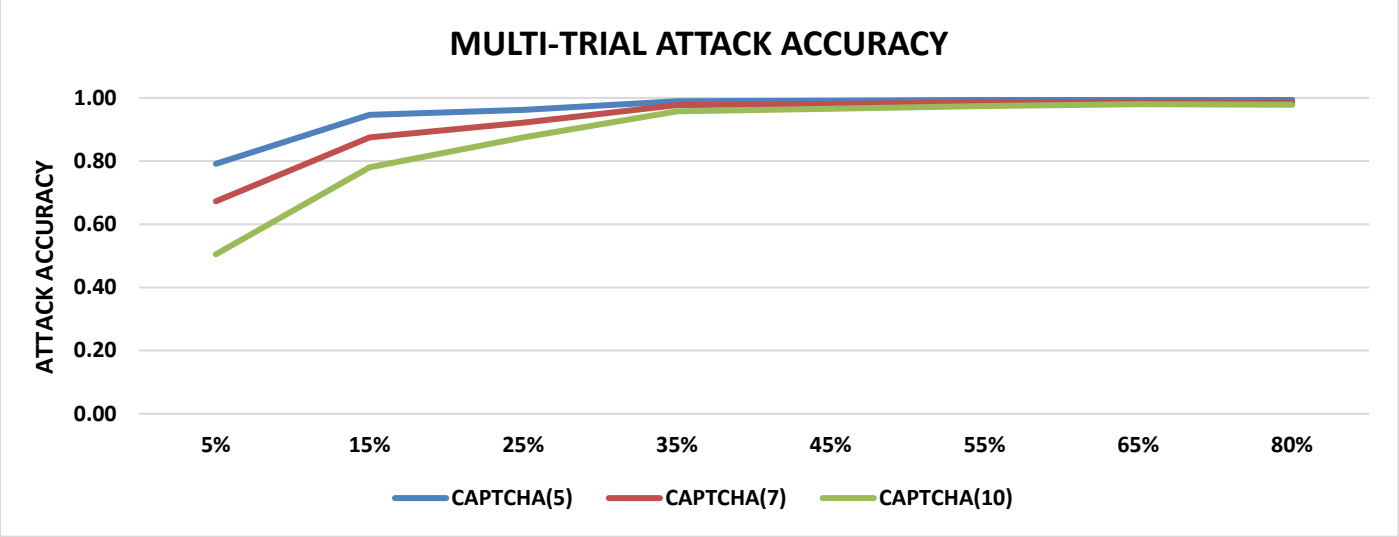

Figure 14: Multi trial attack accuracy on adversarial audio captcha

8.1 Defense Mechanism : Doubling the wait time

As seen in figure 12, 13 and 14, providing the option of three retries increases the attack accuracy of both machine learning and deep learning models. It also means that attacker has more opportunity to collect the adversarial audio captchas for retraining. To prevent easier collection and higher attack accuracy following is proposed. Websites can track the potential customer's IP address. After the first unsuccessful trial, the wait time for the next trial should be doubled. For example, if using a certain IP address, a user starts creating an account and is unsuccessful at the first try, the wait time should be doubled for each extra trial. This wait time should increase even when the next account is being created using the same IP address. Hence, if the general wait time was 5 seconds, on the second trial the wait time becomes 10 seconds, followed by 20, 40, 80 seconds and so on. Increasing the wait time will prevent the attacker from collecting huge samples of adversary data. If the attacker can collect only 5% of the adversary data, the attack accuracy will be around 45% for audio captcha of length 10 and 80% for audio captcha of length 5. But, there is still a scope to create stronger defense mechanism. In the following section we explore a round-robin method of creating audio captcha.

8.2 Defense Mechanism : Round Robin / Random captcha generation

To prevent attackers from accumulating huge samples of adversarial audio data, websites can create adversarial audio captcha by using adversarial audio datasets in a round robin or random fashion. In previous sections, we had proposed that audio captcha should be created by joining audio at random from any of the 12 adversarial audio datasets. In this defense mechanism we propose that instead of giving away on all adversarial sets at once, the captcha can be created by using subsets. At periodic interval these subsets can be changed. In this case, even if the attacker accumulates adversarial audio captcha and retrains on them, after a period, the datasets using which captcha is generated will be changed. At this point, the attackers retraining will become futile. To understand the results of such a scenario, we setup the experiment as follows: In iteration 1, we choose two adversarial sets at random or in round robin fashion to create the audio captcha for test. For the next iteration, we assume that the attacker has accumulated these adversarial audio datasets (the ones used for test) and retrained on them. But, we now create audio captcha using 2 different adversarial audio datasets. These can be chosen at random from the remaining datasets or in a round robin manner. Performing this for 5 iterations we get results as shown in Figure 15. This method proves to be successful at preventing attacks, with only one trial the attack accuracy is 30% for audio captcha of length 5 and 21% for audio captcha of length 10. Even with three trials the attack accuracy is still 65% for audio captcha of length 5 and 50% for audio captcha of length 10. As we can see that rotating the audio datasets after regular period is extremely helpful in creating successful defense mechanisms. Rotating datasets is an automatic work and needs no extra effort in implementation at regular intervals. Hence, this is the best defense mechanism till now.

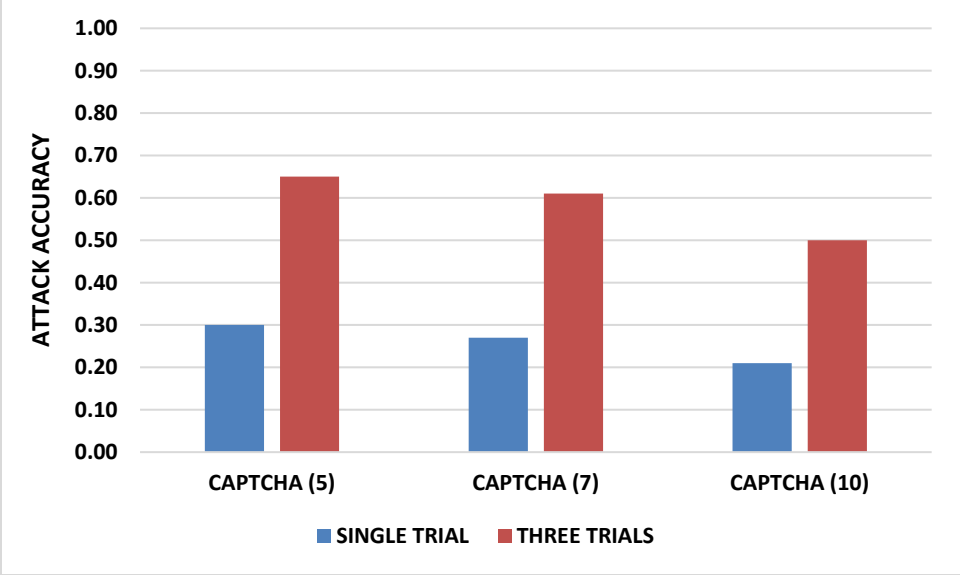

Figure 15: Attack Accuracy on using adversarial datasets in round robin to create audio captcha

8.3 Defense Mechanism : Combination Method

Given that both above defense mechanisms help in reducing the attack accuracy, it makes sense to combine the two. By combining the two defense mechanisms, following should be performed:

1. IP address of users should be tracked and after each attempt to break audio captcha the wait time should be doubled. This will ensure lower collection of datasets which will in turn result in lower attack accuracy.
2. To create audio captchas a subset of adversarial audio dataset should be used, after regular periods, the subset datasets should be changed. This choice of subset datasets can be random or in a round robin fashion.

CHAPTER 9. Conclusion

Audio captcha is an accessible form of captcha that is deployed by websites to prevent abuse by spammers and bots. Though audio captchas are meant for the visually disabled section of users, they are available to all users alike. An attacker can target these audio captchas to create fake accounts, spam users, etc. Hence, it is important to study the security of these audio captchas from attacks. Through this work I show that audio captcha is not secure and recognize the need for more secure forms of audio captcha. I proposed a new architecture that uses adversarial audio to make audio captcha secure against deep learning and machine learning attack models. On performing, Level I and Level II defense against attacks, I observed that L1 defense strategy prevents nearly 99.9% attacks from pre-trained models. Whereas L2 defense prevents 64% attacks from re-trained models (on adversarial examples) for audio captcha of length 5, 69% attacks from re-trained models for audio captcha of length 7 and 75% attacks from re-trained models for audio captcha of length 10. Even when the attacker retrains its model on samples from all adversarial audio datasets, the attacking accuracy is still lower than that of normal audio captcha. This proves that adversarial audio captcha is more robust and resilient towards attacks from ML/DL models. I also explored real world scenarios where the users are given multiple chances to break the audio captcha. In such scenarios defense mechanisms such as tracking IP addresses, increasing wait time after each attempt at breaking the captcha, round robin selection of adversarial datasets for captcha generation provide good security to audio captcha.

CHAPTER 10. Future Work

Security of audio captcha requires constant research. New and different methods of creating audio captcha need to be studied and applied to ensure security over longer period of time. Adversarial examples in general are used to attack a model, with increasing academic work in the field of adversarial examples, researchers are working to create more robust models that are not fooled by adversarial examples. Thus, there is a constant need to keep updating the methodology of generating audio captcha to keep them safe.

References

[1] L. Von Ahn, M. Blum and J. Langford, "Telling humans and computers apart automatically," Communications of the ACM, 47, 56-60 (2004).

[2] S. Solanki, G. Krishnan, V. Sampath and J. Polakis, "In (Cyber) Space Bots can Hear You Speak: Breaking Audio CAPTCHAs using OTS Speech Recognition," Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 69-80.

[3] J. Tam, J. Simsa, S. Hyde and L.V. Ahn, "Breaking Audio Captchas," Advances in Neural Information Processing Systems, 2009, pp. 1625-1632.

[4] E. Bursztein and S. Bethard, "Decaptcha: Breaking 75% of eBay Audio CAPTCHAs," Proceedings of the 3rd USENIX Conference on Offensive Technologies, 2009, pp. 8.

[5] S. Sano, T. Otsuka and H.G. Okuno, "Solving Google's Continuous Audio CAPTCHA with HMM-Based Automatic Speech Recognition," International Workshop on Security, 2013, pp. 36-52.

[6] A. Hagen, D.A. Connors and B.L. Pellom, "The Analysis and Design of Architecture Systems for Speech Recognition on Modern Handheld-Computing Devices," Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, 2003, pp. 65-70.

[7] K.P. Bennett and C. Campbell, "Support vector machines: Hype or hallelujah?" Acm Sigkdd Explorations Newsletter, 2, 1-13 (2000).

[8] M. Won, H. Alsaadan and Y. Eun, "Adaptive multi-class audio classification in noisy in-vehicle environment," arXiv Preprint arXiv:1703.07065, (2017).

[9] L. Grama and C. Rusu, "Audio Signal Classification using Linear Predictive Coding and Random Forests," 2017 International Conference on Speech Technology and Human-Computer Dialogue (SpeD), 2017, pp. 1-9.

[10] D. Wu, "An Audio Classification Approach Based on Machine Learning," 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2019, pp. 626-629.

[11] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss and K. Wilson, "CNN Architectures for Large-Scale Audio Classification," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 131-135.

[12] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," (2015).

[13] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, "Intriguing

properties of neural networks," arXiv Preprint arXiv:1312.6199, (2013).

[14] A. Kurakin, I. Goodfellow and S. Bengio, "Adversarial examples in the physical world," arXiv Preprint arXiv:1607.02533, (2016).

[15] I.J. Goodfellow, J. Shlens and C. Szegedy, "Explaining and harnessing adversarial examples," arXiv Preprint arXiv:1412.6572, (2014).

[16] K. Pei, Y. Cao, J. Yang and S. Jana, "Deepxplore: Automated Whitebox Testing of Deep Learning Systems," Proceedings of the 26th Symposium on Operating Systems Principles, 2017, pp. 1-1