8 or∞: A Dive into Intelligent Character Recognition

Daniel Ye	Keshav Prasath	Shrnyaa Sharma
898875	990401	870529
daniely	kprasath	shrnyaas

Abstract

Intelligent character recognition and its application have been an integral part of image processing models deployed in everyday life. However, research has shown that they are very prone to attacks and can have catastrophic consequences. The objective of this study looks at how different forms of adversarial perturbations can affect character recognition performance and which strategies can be implemented to make the neural network model (CNN) more robust. Methods such as Gaussian Noise, Image Augmentation and FGSM are used to emulate adversarial attacks and it is found these manipulations drastically reduce the performance of the model, and deploying methods such as adversarial training and randomisation increases the robustness towards such perturbations.

1 Introduction

The research in the field of Optical Character Recognition has notably progressed over the past decade. Most research papers utilise deep learning for OCR to avail their ability to partition the image in multiple sections and learn the patterns observed in these partitions. Neural Networks, although perform exceptionally well, impose two primary challenges. Firstly, the ability of the neural network to generalise is highly dependent on the training dataset. In theory, the more diverse the dataset is, better generalisation the model will have on unseen data.[1] Secondly, neural networks, especially CNNs are extremely vulnerable to adversarial examples: inputs with perturbations—often imperceptible—that are crafted to force a convolutional neural network (CNN) to fail. This type of attack can seriously undermine the security of systems utilising CNNs with significant consequences. For example, autonomous vehicles may be crashed, biometric authentication systems may be spoofed, banned or illicit content may be crafted to bypass filters on social networks. This paper aims to highlight the importance of diversifying the training dataset and also quantifies the effects of several untargeted adversarial attacks on a CNN with the aim to compare the efficacy and transferability of naïve attacks to a white-box attack of the fast signed-gradient method. Furthermore, we quantify and compare the efficacy of some proposed defense strategies such adversarial training and randomisation.

2 Dataset

The EMNIST dataset [2] is an extension of the MNIST database which contains grey-scaled images of handwritten alphanumeric characters (see Fig 1.). This report uses the EMNIST Balanced dataset, which contains an equal number of instances for each class,to reduce any underlying bias and to reduce the need of using cross validation while testing.

Balanced EMNIST dataset consists of 47 classes (10 numeric and 37 lowercase and uppercase alphabets). Lowercase alphabets which resembled their uppercase counterparts such as ‘I’ or ‘Z’ were combined to create one class (see Fig. 2). The classifiers use pre-defined splits of training and testing datasets available from the Python emnist library to maintain consistency and reproduction of the same results. The training and testing dataset has 112,800 and 18,800 instances respectively. Each

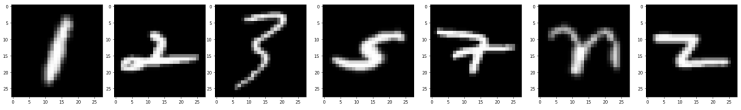

Figure 1: Sample images of instances

image consists of 28 by 28 pixels (784 pixels in total). Each pixel has a value ranging between 0, representing lowest brightness, to 255, representing the highest brightness which is scaled to a [0,1] interval. The original dataset provided as a 28 by 28 2-D array for each image which was reshaped to create one row.

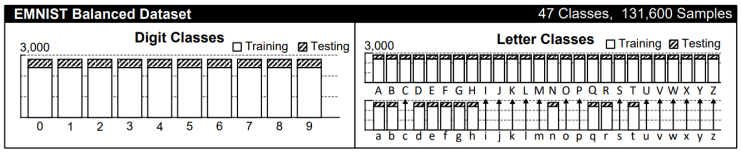

Figure 2: Distribution of classes in the Balanced Data set

The character pairs, (1, I), (O, 0) and (F,f) have the highest overlap amongst their clusters which suggests the presence of similar features, hence it was hypothesised that the characters are most likely to be misclassified. On the other hand, character pairs (1,0), (O,I) and (n,3) have the least amount of overlap (see Fig. 3), therefore will be relatively easier to classify. Characters 1, T, 7 have the least amount of variance in pixel values which can be due to the presence of similar instances in the dataset. This might lead to an easier classification for these characters for this dataset, but reduces the models ability to generalise.

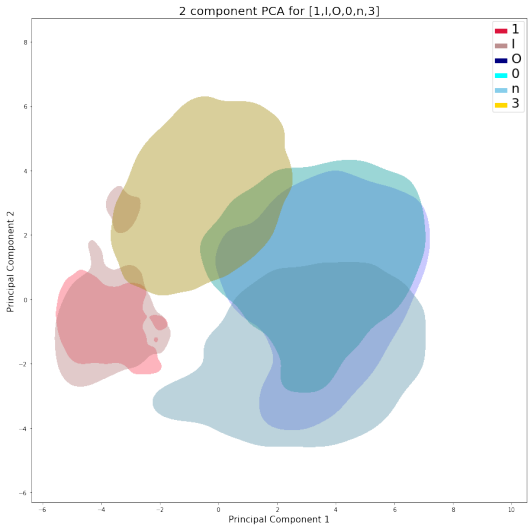

Figure 3: PCA on selected characters

3 Methodology

3.1 Initial Modelling

After extensive research on the different kinds of neural networks used to perform image classification, Perceptron and Convolutional Neural Network (CNN) were chosen as baseline models for neural networks. Both Perceptron and CNN were built with a single hidden layer. Additionally, traditional algorithms such as Decision tree, Multiclass SVM and kNN were also trained to draw comparisons with the neural networks.

Model	Accuracy(%)	Runtime (in mins)
One-R	2.1%	0.3
Multiclass SVM	67.8%	30
kNN	60%	44
Decision Tree	58.9%	0.96
Perceptron	84.1%	1
One layered CNN	85.3%	13

Figure 4: Accuracy scores for the initial models

Deep learning in image processing optimises the features extracted with greater precision as compared to traditional machine learning methods like kNN and SVM[4]. This allows it to perform better. As we had large amounts of data and the computational power to process neural networks, the choice of using deep learning to tackle this problem was eminent.

Both neural network models performed reasonably well with base MLP producing 84% accuracy and CNN producing 85% accuracy. As CNN's base model performed better and CNN's are commonly used for image processing tasks, CNN was chosen as the elementary model which was further refined.

3.2 Neural Network Architecture and Hyperparameter Tuning

The elementary CNN model was further refined by hyperparameter tuning and changes to the architecture. Through hidden and trial, a brute force approach was adopted to design the architecture of the model. The changes consisted of adding layers, adjusting the number of neurons, changing the kernel size and strides. The alterations that resulted in an increase of accuracy of the model were retained. The final model consists of 7 hidden convolutional layers in addition of one input and one output layer. This model performed better than the elementary model with an accuracy of 88%. (See Fig.5)

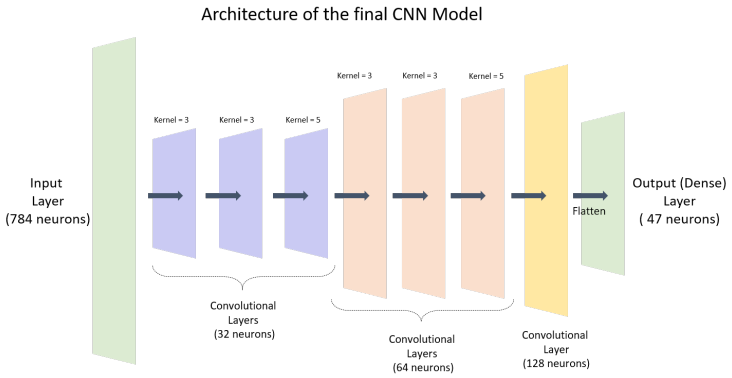

Figure 5: Architecture of the Final Neural Network

To enhance the model performance, hyperparameter tuning with 3 fold cross validation was performed. Randomised search[5] and Bayesian Optimisation was performed instead of grid search as the latter is much more computationally expensive on a neural network. Parameters tuned were: the number of epochs, processing batch size, activation functions, optimizer, optimizer learning rate and dropout percentage. The learning curves showed that the model with 75 epochs, batch size of 100, Leaky RELu activation function[6], ADAM optimizer[7] with a learning rate of 0.001 and a dropout of 0.5 performed the best with minimal overfitting. This final model performed with a test accuracy of 90.3

3.3 Generating Adversarial Examples

The goals of generating the adversarial examples are to introduce diversity and variety in the dataset to improve and test the generalisation of the model and secondly to emulate a potential attack aimed

to purposefully cause the model to misclassify the labels.In our experiment, we consider four different perturbations to the image:

Rotation and Translation Images were randomly shifted up or down from the centre of the image and were rotated left or right by a maximum of 40 degrees.

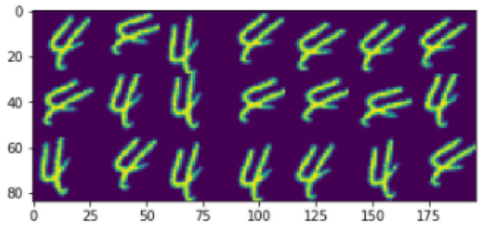

Figure 6: Examples for rotated and translated images

Random Gaussian Noise For each pixel, its brightness level x is randomly perturbed by $\epsilon \sim N\left(0, \sigma^2\right)$, where σ is the standard deviation of the perturbation. This noise is chosen to mimic the type of noise in images when the sensor is placed under low light conditions or extreme temperatures.

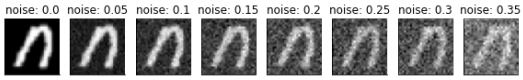

Figure 7: Examples for Gaussian Noise with increase in sigma

Impulse Noise Also otherwise known as salt and pepper noise, a proportion of pixels determined by the signal to noise ratio (SNR) are randomly selected to be perturbed. Out of the selected pixels, a proportion p_w are randomly made white, while $p_b = 1 - p_w$ are made black.

In our experiment, we fix the proportion of white pixels $p_w = 0.5$, and pick $SNR \in \{0.75,\ 0.8,\ 0.85,\ 0.9,\ 0.95\}$. This type of noise is chosen for its similarity to a physical sticker attack [8] (Eykholt et al.).

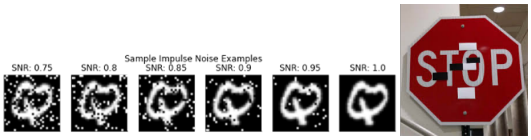

Figure 8: Examples for Impulse Noise

Fast Gradient Sign Method (FGSM)

FGSM [9] is a single-step attack that finds the perturbation which yields the highest increase of the cost function under the l_∞. Let X_n denote the n-th image in a dataset containing N images, and y_{true} denote its corresponding true label. Denote the loss as $L\left(X_n, y_N^{\text{true}}; \theta\right)$ where θ is the model parameters. Then the FGSM adversarial example is

$$X_n^{\text{fgsm}} = X_n \ + \ \epsilon \ \cdot \text{sign}\left(\nabla_{X_n} \ L\left(X_n, y_n^{\text{true}};\ \theta\right)\right) \tag{1}$$

where ϵ controls the magnitude of the perturbation. We try $\epsilon \in \{0.05,\ 0.1,\ 0.15,\ 0.2,\ 0.25,\ 0.3,\ 0.35\}$ for this experiment to quantify the effects of increasing the perturbation magnitude.

Figure 9: Examples for FGSM

3.4 Defending Against Adversarial Examples

The goal of defence is to make the CNN robust against misclassification on adversarial examples, while maintaining good performance on non-adversarial examples. A popular method of defending against adversarial examples is adversarial training, first suggested by [9]. This is the process of re-training the classifier on the adversarial images to boost their performance on them. An alternative approach on mitigating adversarial effects is a randomization procedure proposed by [10]. In the method, a series of randomisation layers are added at the beginning of the classification network. Each input image undergoes random resizing and random padding to create a new set of images, upon which one is randomly and passed to the original CNN for classification. Our experiment implements a variation of their method. First, the input image is transformed with a random zoom within a predefined range. Second, the resulting image undergoes a random shift, with empty pixels being set to a value of 0. The main difference from Xie et al.'s proposed method is the original image may be clipped by this random shift.

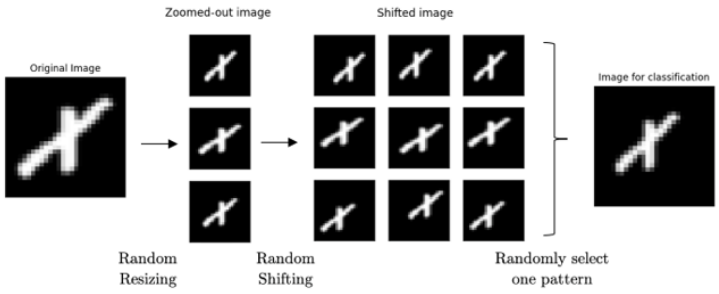

Figure 10: Randomisation: Approach to combat adversarial attacks

4 Experimental Results and Discussion

4.1 Performance of the Model

Overall the model with the final architecture and hyperparameter tuning resulted in the accuracy score of 90.3%. The confusion matrix (see Fig. 11) suggests that the model is generalising quite well. Mostly classes, as observed in the section 2, which had the most overlap in clusters, such as (I,1) and (O,0), were misclassified.

4.2 Image Augmentation(Rotation and Translation)[11]

The training and testing of data was done in two ways,
i) trained on the original dataset and tested on the augmented dataset
ii) trained on the augmented dataset and tested on the augmented dataset
The first method caused the accuracy score of the model to decrease by 20%, which is a significant decrease. The second method, however, only decreased in accuracy score by 2%. These results highlight the importance of diversification of instances in the training dataset. This suggests that the generalisation and the robustness of the model improves when the training dataset includes a variety of instances.

4.3 Adversarial Attacks

To quantify the effect of each type of adversarial example, the performance of the original CNN is evaluated—with no re-training or fine tuning—on each of the three types of adversarial perturbation applied to the held-out test set. Furthermore, to qualify the robustness of an FGSM attack, it is compared to the Gaussian noise adversarial attack. Gaussian and Impulse adversarial examples are presented as baseline attacks, as they are not optimized to fail a particular classifier. Rather, in this naive method, the perturbations are fabricated randomly with hope that it will cause misclassification. On the other hand, FGSM examples are actively optimized against the loss function. The performance

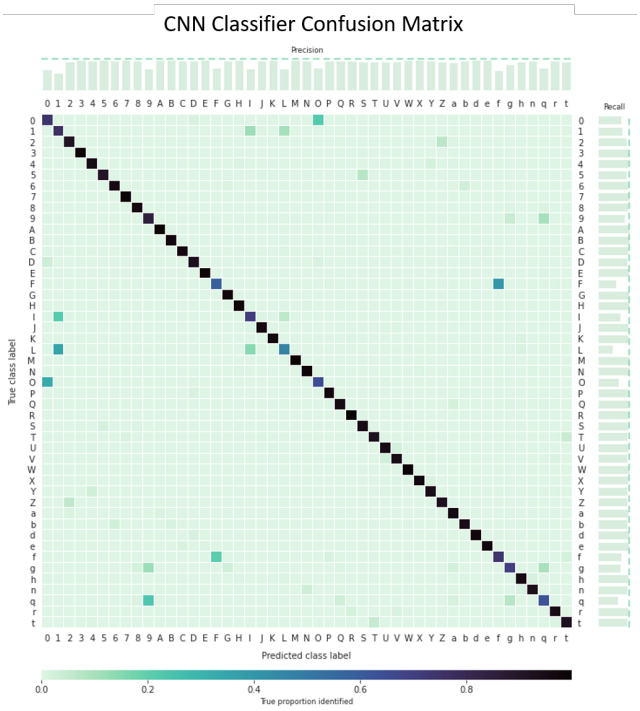

Figure 11: Confusion Matrix for the Final Model

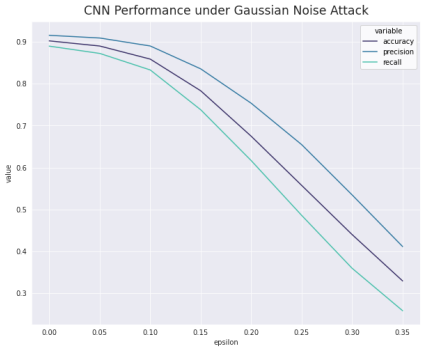

Figure 12: Random Gaussian Noise Attack

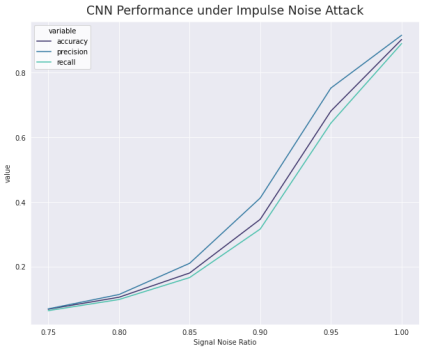

Figure 13: Impulse Noise Attack

of the model degrades at an increasing rate with added intensity of Gaussian noise. (see Fig.12)A similar result is observed with increasing intensity of Impulse noise (see Fig.13), however the degradation levels off when the signal noise ratio falls under 80%. Under the FGSM attack (see Fig.14) , the degradation is drastic for even low values of epsilon. Values of epsilon above 0.1 reduce the performance to near zero. By face value, FGSM seems substantially more effective as an attack.

With the aim of quantifying the improvement of the FGSM against a naive adversarial approach, a comparison is drawn between its performance on Gaussian examples. Let an adversarial pixel value be $x^{adv} = x + \epsilon$, where ϵ is the pixel perturbation. Each set of adversarial examples is normalized so comparisons are made between adversarial sets where the $_\infty$ norm of the standard deviation pixel perturbations ϵ is the same. To do this, we make the assumption that each FGSM pixel perturbation (prior to applying the magnitude factor ϵ) follows a discrete uniform distribution $U \sim (-1, 1)$, and choose the ϵ such that the variance of ϵ is the same.

After applying normalisation, we find the FGSM cripples the classifier with substantially less perturbation than Gaussian noise. FGSM can reach a low accuracy of approximately 30% with an

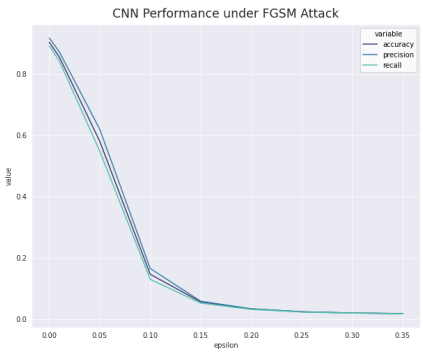

Figure 14: FGSM Attack

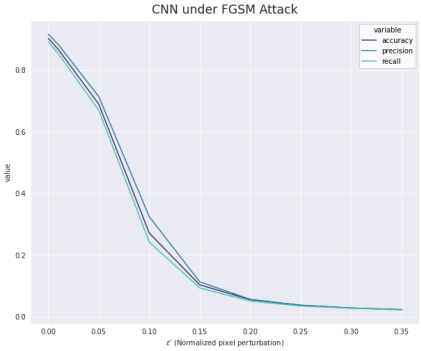

Figure 15: FGSM Attack (Normalised Pixel Perturbations)

epsilon value of 0.1, while Gaussian noise requires over a 3 times larger epsilon value to achieve the same decline in accuracy.

4.4 Defence against Adversarial Attacks

Adversarial Training

As a defence strategy the model is, both, trained and tested on the adversarial examples. This follows from the intuition that the model will learn the adversarial patterns observed in the training dataset, hence, will become robust to the attacks.

For each type of adversarial example, we pick one perturbation magnitude value to generate a set of adversarial training examples due to time and computing restraints. Impulse examples were generated with SNR of 0.75, Gaussian noise with 0.35 standard deviation, FGSM with $\epsilon = 0.1$. In general, each adversarial-training strategy was very good at defending against the examples that it had been trained for, displaying peak performance at that magnitude. Both Gaussian and Impulse strategies also generalised well for increased and decreased magnitude of perturbation. The network has likely learnt the filters necessary to mitigate the noise.

Interestingly, FGSM performed incredibly well (99.09%) at peak, but struggled to generalise to different magnitudes. This could likely be due to the network learning the perturbation pattern instead of the actual underlying character to be classified.

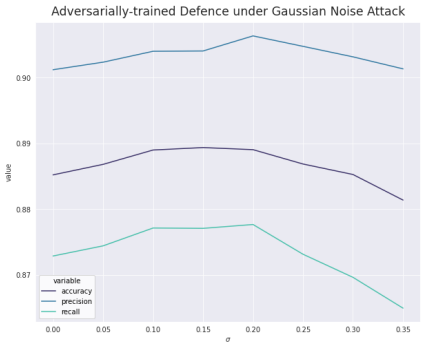

Figure 16: Gaussian Noise Defense (Adversarial Training)

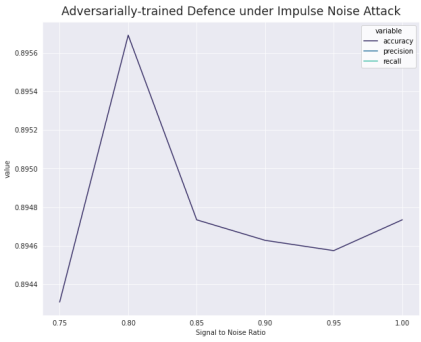

Figure 17: Impulse Noise Defence (Adversarial Training)

The Randomisation Method

The randomisation method applies random zooms and shifts to the test input, and applied as the defence to the FGSM adversarial examples. While this method ranked no.2 in the 2017 NIPS adversarial examples defence challenge, in our implementation, it was not found to be an effective defense, and at most only gave a 10% increase in accuracy over no defence.

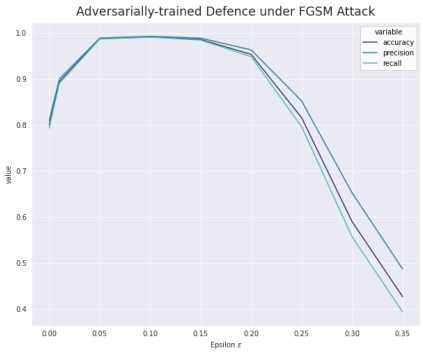

Figure 18: FGSM Defense (adversarial training

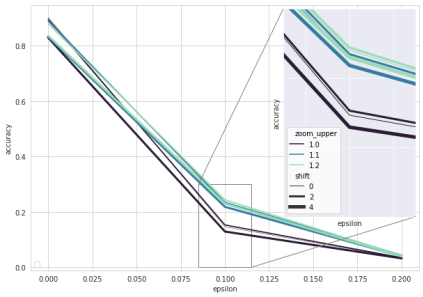

Figure 19: FGSM Defense (randomisation)

Interpreting the Fig 19, various levels of zoom and shift were tried. It was found that a small shifts could improve the accuracy, but too large a shift would cause the accuracy to decrease again. There are signs that increasing zoom also increases accuracy, but it is hypothesised that there is a limit where it will decrease again. Unfortunately, computational limitations meant it was difficult to achieve more granularity than the 9 shift and zoom combinations shown to test this hypothesis.

This discrepancy between our result and original authors could be due to the smaller image size, as this method was originally suggested for larger ImageNet examples. This means the random transformations have a proportionately larger effect on our inputs. Furthermore, the original authors applied a comprehensive image augmentation regime at training, which we did not implement as it was found to negatively impact our classifier performance.

5 Conclusion

Through the experimentation, it is evident that even the slight manipulation of the data can drastically affect the performance of the model. Overall, the methods deployed to improve the robustness of the model work well. The model trained adversarially on the FGSM examples provide the best defence strategy. For future, the work can be extended to word recognition with the aid of an english corpus. Afterwhich, words can be combined together to perform sentence recognition with the assistance of Hidden Markov Chains which can be further expanded into Natural Language Processing. For example, we can create an integrated functioning system that can take in scanned handwritten documents as input data and feed it through our model to derive useful results. Enabling the ability to truncate manual surveyed data collection and ultimately speed up analysis. By being noise resistant, slight noise added in the manual scanning can be effectively countered. This can also be extended to self driving cars to decipher and process street traffic signs.

References

[1] A. F. Mollah, N. Majumder, S. Basu, and M. J. a. p. a. Nasipuri, "Design of an optical character recognition system for camera-based handheld devices," 2011

[2] Cohen, G., Afshar, S., Tapson, J., van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters. Retrieved from http://arxiv.org/abs/1702.05373

[3] N. Islam, Z. Islam, and N. J. a. p. a. Noor, "A survey on optical character recognition system," 2017.

[4] A. Osareh, M. Mirmehdi, B. Thomas, and R. Markham, "Comparative exudate classification using support vector machines and neural networks," in International Conference on Medical Image Computing and Computer-Assisted Intervention, 2002, pp. 413-420: Springer.

[5] J. Bergstra and Y. J. T. J. o. M. L. R. Bengio, "Random search for hyper-parameter optimization," vol. 13, no. 1, pp. 281-305, 2012.

[6] A. K. Dubey and V. Jain, "Comparative Study of Convolution Neural Network's ReLu and Leaky-ReLu Activation Functions," in Applications of Computing, Automation and Wireless Systems in Electrical Engineering: Springer, 2019, pp. 873-880.

[7] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, "Adam optimization algorithm for wide and deep neural network," vol. 2, no. 1, pp. 41-46, 2019.

[8] K. Eykholt et al., "Robust physical-world attacks on deep learning visual classification," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1625-1634.

[9] I. J. Goodfellow, J. Shlens, and C. J. a. p. a. Szegedy, "Explaining and harnessing adversarial examples," 2014.

[10] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. J. a. p. a. Yuille, "Mitigating adversarial effects through randomization," 2017.

[11] L. Perez and J. J. a. p. a. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017.