Self-Configuring Convolutional Neural Network using Evolutionary Algorithm

BY

Md. Hafizur Rahman Arfin
ID: 161-15-7144
AND

Md. Majedul Islam
ID: 161-15-6784

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Syeda Tanjila Atik
Lecturer
Department of CSE
Daffodil International University

Co-Supervised By

Md. Ferdouse Ahmed Foysal
Lecturer
Department of CSE
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

DECEMBER 2019

APPROVAL

This research titled **"Self-configuring Convolutional Neural Network using Evolutionary Algorithm"**, submitted by **Md. Hafizur Rahman Arfin, ID NO: 161-15-7144** & **Md. Majedul Islam, ID NO: 161-15-6784** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc in Computer Science & Engineering and approved as to its style and contents. The presentation has been held on 5 Dec, 2019.

BOARD OF EXAMINERS

[illegible]

Dr. Syed Akhter Hossain **Chairman**
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Abdus Sattar **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Farah Sharmin **Internal Examiner**
Senior Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

[illegible]

Dr. Md. Saddam Hossain **External Examiner**
Assistant Professor
Department of Computer Science and Engineering
United International University

DECLARATION

We hereby declare that this research has been done by us under the supervision of **Syeda Tanjila Atik, Lecturer, Department of Computer Science and Engineering** and co-supervision of **Md. Ferdouse Ahmed Foysal, Lecturer, Department of Computer Science and Engineering, Faculty of Science and Information Technology, Daffodil International University.** We also declare that neither this research nor any part of this research has been submitted elsewhere for the award of any degree.

Supervised by:

[illegible] 10/12/19

Syeda Tanjila Atik
Lecturer
Department of CSE
Daffodil International University

Co-Supervised by:

[illegible] 10.12.19

Md. Ferdouse Ahmed Foysal
Lecturer
Department of CSE
Daffodil International University

Submitted by:

[illegible]

(Md. Hafizur Rahman Arfin)
ID: -161-15-7144
Department of CSE
Daffodil International University

majedulislam

(Md. Majedul Islam)
ID: -161-15-6784
Department of CSE
Daffodil International University

ACKNOWLEDGEMENT

At First, we would like to express our gratitude to the Almighty Allah for His divine blessing made it possible to complete our final year thesis successfully.

We would like to express our gratitude to our supervisors **Syeda Tanjila Atik, Lecturer, Department of CSE** and **Md. Ferdouse Ahmed Foysal, Lecturer, Department of CSE, Daffodil International University, Dhaka.** Their Deep Knowledge & keen interest with supportive instructions helped us in the field of deep learning-based research, finally we completed our work on *"Self-configuring Convolutional Neural Network using Evolutionary Algorithm"*.

We really want to thank our previous supervisor **Mr. Nafis Neehal, Lecturer, Department of CSE, Daffodil International University, Dhaka.** Without his encouragement, this work would not see the light. He guided us with his deep knowledge in the field of "*Metaheuristics"* and *"Deep Learning"* that helped us to complete this work.

We would like to express our heartiest gratitude to **Dr. Syed Akhter Hossain, Professor and Head, Department of CSE, Daffodil International University, Dhaka.** For his valuable support and advice to finish our project and also heartiest thanks to other faculty members and the staff of the department of CSE, Daffodil International University.

At last, we want to thank all the good wishers, friends, family, seniors for all the help and inspiration. This research is a result of hard work and all those inspirations and assistance.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

One of the key drivers of today's AI revolution is an improvement in the field of deep learning. The evolution of deep learning along with big data and the improvement in the hardware industry impacted the Artificial Intelligence industry like never before. As time is going on this field of deep learning is getting improved and we are finding different problems that can be solved to improve it even further. One of them is the structure of a Neural Network. The structure of the neural net must be predefined. Which is done by humans where human error is more likely to occur. Also, it takes a lot of time to form a perfect structure using trial and error method. In this work, we tried to solve this problem using Evolutionary methods. We applied the Genetic Algorithm to find the best architecture of neural network automatically. We used the Genetic Algorithm in Convolutional Neural Networks to create structures. We applied them in two datasets, first the Fashion Mist dataset and then PataNet dataset. The results were interesting. The model provided by the Genetic algorithm performed better than the predefined human structure. Also, the genetic algorithm gave some unusual structure which performed better than any human models.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

Artificial Intelligence has been around for more than 50 years. There has been a lot of ups and downs in this field of computer science. But in this decade we can surely say that the AI trend is going upward. And not only going upward but also the rate of improvement seems exponential. The thanks goes to a subfield of Artificial Intelligence called Machine Learning. The algorithms of machine learning like Logistic Regression, KNN, SVM, Random Forest, etc. helped the field of AI to grow [1]. But there was a plateau hit. These traditional Machine Learning algorithms initially performed better but as the data increased and the hardware performance increased, the performance of these algorithms didn't increased. There was a fear of another AI Winter [2].

Another algorithm in the field of Machine Learning is Deep Learning which uses Artificial Neural Network structure for model building. This algorithm saved us from getting into another AI winter. The difference between the general Machine Learning algorithm and Deep Learning is that in deep learning, data is the food. The more we feed data the stronger the model becomes, thus the accuracy increases [3]. The sudden boom in data collection from the internet and this criteria of Deep Learning algorithm consistently increased the improvement of Artificial Intelligence. Hardware is a big factor here, but recently the hardware development and the improvement in Graphics Processing Unit can process a thousand amounts of data in seconds. Because of the massive improvement of the field of deep learning, some scientists consider this as a separate part of AI along with ML [4].

As much as Deep Learning is impressive, there are a few drawbacks. The primary one is that the structure of the Artificial Neural Network has to be defined by the developer before training the net. The weights and biases of the network are learned through forwarding propagation and backward propagation. But the number of layers, number of nodes in each layer, the parameters in the layers are defined by the developer. The structure of the neural net is very important. Because of it as a huge effect in the final accuracy of the model.

Furthermore, training the wrong net for hours and then changing a single variable and then training the net again is really time-consuming and not as efficient as it should be.

The problem with structure is not easy to solve as the search space is huge. There are neural nets layers to choose, their number can be from one to thousands. The combination of the changing variable is very large. To implement a brute force solution to solve it is not feasible. Using mathematical methods like Regression is not efficient as they can be stuck in local minima.

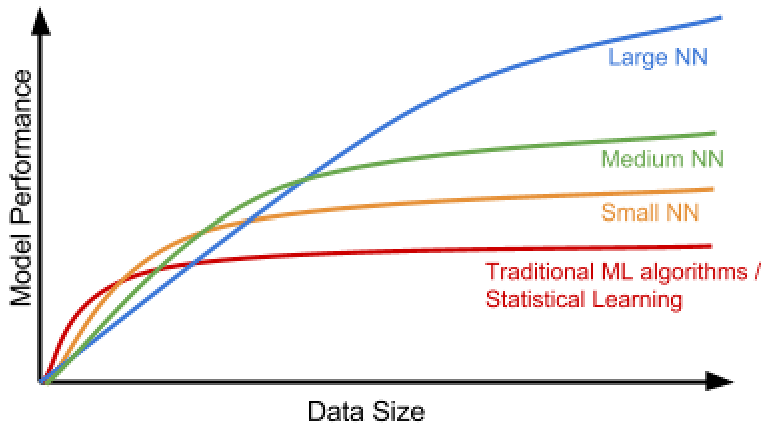


Figure 1.1: Data size vs Model Performance Graph

So to solve this problem we used Metaheuristics Algorithms. Metaheuristic Algorithm imposes certain type randomness to the search criteria but follows a little tweak to find the best solution in a huge search space. One of the subfields of Metaheuristics is Evolutionary Algorithm, Genetic Algorithm is one of the algorithms that is used here. We used a genetic algorithm because it is relatively common in the field of Artificial Intelligence but the use of the Genetic Algorithm in Convolutional Neural Network is not very frequent.

We applied the Genetic Algorithm to find the proper structure of a convolutional neural network. We designed vectors that represent the architecture of a single neural net. We generated a few populations and trained them. The best candidate from all the generations was trained for the maximum number of epochs to reach the final result. We propose a complete algorithm that removes the difficulties for human intervention in making a Convolutional Neural Network structure.

1.2 Objective

One of the key driving forces of AI is Deep Neural Networks. But in current neural networks, a lot of things need to be pre-determined by the programmer and a lot of things are solely depend on trial and error method. If our brain needs to learn something, it doesn't need a lot of variables to be predefined and it doesn't randomly search for the right answer. It can start as a small "seed" and through each step it evolves on its own, making an intelligent guess and tweaking them along the way as needed. The hidden layer node counts are randomly assumed and programmed. Which is exhausting, slow and less resembling true intelligence. We want to improve this process by:

1. Use genetic algorithm to search best neural net structure and avoid getting stuck in the local minima of the search space.
2. Using custom variant of Genetic Algorithm operators we will optimize the process of hidden layer node count initialization and the activation function used in the hidden layer.
3. By optimizing the neural network we can evolve and set the best nodes count for hidden layer by the type of data-set, instead of assuming.
4. The network will be more accurate due to optimized weights for nodes and optimal node number for hidden layer.

1.3 Motivation

There are thousands of neural networks are already implemented and thousands are developing every day! The existing neural nets framework doesn't resemble true intelligence. That is, starting a neural network takes a lot of assumptions and random guessing. Our brain takes careful assumptions and doesn't need to start with a lot of initial data. This is what "Evolutionary Neural Network" does, it tries to take the neural network close to the human brain structure. As the human brain evolves and optimizes intelligently. A neural network can also evolve and optimize itself without any human intervention. We found out that this research field is booming right now. Along with the University of Texas Austin [5], a lot of prestigious universities are working on this topic. Also, our supervisor has already worked on topics like that. So we thoughts “Evolutionary Convolutional Neural Network” is a very good topic to contribute in.

1.4 Rational of the study

There is a lot of work going on in the field of deep learning. But there is a gap in implementing the Evolutionary Strategies to this field. So our work is will be a new approach to the problem described. We want to develop the existing development process of the neural network using our best effort.

Deep learning by itself is a fascinating field. Merging it with another amazing field of

Evolutionary Algorithms is a work of great excitement.

1.5 Research Questions

This work will solve some puzzling problems surrounding the Neural Network structure. An efficient but effective algorithm establishing is the primary goal. To guide a research work one must have a set of good questions that will be answered at the end of the work. We have some questions that we want to find answers through our work. They are,

1. Is it possible to find and generate neural net structure automatically?
2. Can genetic algorithm be merged with Convolutional Neural Network?
3. How can we choose best operators and hyper parameters to optimize and design the network for it?
4. Will the evolutionary methods result better than traditional method or not?

1.6 Expected Outcome

Every step of our digital life is surrounded by AI. Computer vision is another field of AI that is seeing massive improvements. After this work, we will be able to just create a little neural net called "seed" which will configure its structure by analyzing the dataset using Genetic Algorithms. We can describe our outcomes as follows.

1. A Self Configuring neural network with improved speed and accuracy can be built from very little code.
2. Optimized Neural Networks will significantly improve performance.
3. The current existing trial and error method for training a neural network is slow and exhausting. By finding the best network before training the neural network will significantly reduce the speed of training of neural network.
4. Accuracy of Neural Networks will improve.
5. Instead of assuming the number nodes in hidden layers, the layers node will be assigned for the best outcome.
6. This will increase the accuracy of neural network significantly.

1.7 Layout of the Report

We described how AI is becoming an important thing with the help of Deep Learning. It is also described where the problem with deep learning is. How we formulated our research question. And how we approached the solution in the first chapter.

In the second chapter, we described some background work done in this field. Some previous works done in this field are old but were very useful to get insight into the work we are doing. Also, we put some background information for people who are not experts in this field. So that they can also understand our work.

The whole workflow is described in detail in the research methodology section of this report. From data collection to designing algorithms. Everything is described in detail in this section.

Chapter four talks about the results of experiment. We also talked about how we ran the experiment and what are the results. Also explained results in detail.

Finally, we concluded our report with chapter five describing what we did, what we observed and where is the future scope of work.

CHAPTER 2
BACKGROUND STUDY

2.1 Introduction

The structure definition problem with Artificial Neural Networks is not new. There has been a long string of research done in the first era of AI. But recently this field is getting traction again. And the field of Evolutionary Algorithms is getting priority. In this section of Background, we will discuss the previous work done in this area in the related work section, some background information that is described in the later section that will help in the understanding of the research work, lastly, some challenges were discussed.

2.2 Related Works

David J. Montana et al. [6] in their paper on "Training Feedforward Neural Networks Using Genetic Algorithms" showed that genetic algorithms outperform the traditional backpropagation algorithm. Backpropagation algorithm can get stuck in local optima. But genetic algorithms application can overcome this problem. The dataset was of sonar images of underwater acoustic receivers with 1200 images. Weights and Biases were used for chromosome encoding. For the fitness functions, the sum of the square of errors was used. Initially, for the chromosomes, the weights and biases were selected randomly with probability t. For the operators, they tested with different sets of operators and chosen that performed best.

For example UNBIASED-MUTATE-WEIGHTS, BIAS ED-MUTATE-WEIGHTS, MUTATE-NODES, MUTATE-WEAKEST-NODES, CROSSOVER-WEIGHTS, CROSSOVER-NODES, HILL Climb. Parameters can greatly influence the performance, the parameters optimized in this work are PARENT-SCALAR, OPERATOR-PROBABILITIES, POPULATION-SIZE. Set of 10 experiments with different operators and parameter values was done during the experiment. Mutate nodes operator performed far better than Biased and Unbiased mutate weights. Crossover nodes, crossover features, and crossover weights performed similarly, Plain GA performed better than Mutate

Weakest node. In the last experiment, the genetic algorithm was compared with the traditional backpropagation, The genetic algorithm converged to minima quickly and gave better accuracy. The runs consisted of 10000 iterations of the genetic algorithm and 5000 iterations of backpropagation. The accuracy proved that the genetic algorithm can find global minima in a large dataset, unlike backpropagation. There are a lot of other operators can be used, this paper only showed some of them. Also, this algorithm cannot perform better in continually changing training data, it'd be difficult for GA in this domain. This was a GA application on only the feed forward neural network, there are other neural networks where GA can be applied and improved.

Geoffrey F. Miller et al. [7] In their paper on "Designing Neural Networks Using Genetic Algorithms" proposed a method of automatically designing architecture of neural networks using genetic algorithms. Network architecture design is intuitive and no methodical way of devising an architecture is present. Evolutionary genetic algorithms are used here to solve this problem. First genetic algorithm generated a population of architectures and then backpropagation is used to train them. After the training the less error networks are chosen for the next generation. The best was selected using a threshold probability.

A strong specification scheme of representation was used in this work. Which allowed more robust and out of the ordinary structures to evolve. The network was represented as a matrix first which then converted into a bit string. Working with the bit string allowed the operators to easily operate on them.

The crossover operator was row swapping. The mutation operator was to change a random entry in the matrix to one or zero. Fitness evaluation method Grefenstette's (1987) was used. Error metric Total Sum Squared used to evaluate architecture.

The perfect structure to solve the XOR problem with 90% accuracy was found in 2nd generation. Within 10 generations the network architecture is discovered. The striking discovery was that the architecture that performed best showed a connection from input to direct output which is not generally shown in typical network design. In Four-Quadrant Problem, in 10 generations a satisfying solution was found. But the tree was also very convoluted. This might be showing that the network is human bias-free. In Pattern Copying

Problem, 10 inputs and 10 output setting was used, after 6 generations a successful architecture was found, after 20th generation, the best architecture was found, this problem is used to evaluate if the system could search architecture that is in very large search space, which can take a lot more generations. And results prove that genetic search is capable of achieving that.

A strong specification can result in highly specific, unexpected designs. This is a very early stage and basic work on GA in network design. Alternate representation scheme, genetic operators, fitness evaluation method, genetic algorithm parameters can yield better performance.

H. K. Lam, S.H.Ling et al [8] paper "Tuning of the Structure and Parameters of Neural Network using an Improved Genetic Algorithm" showed that the tuning of the structure and parameters of a neural network(NN) using an improved genetic algorithm (GA). The Processing time is much faster the tradition GA in binary form on some benchmark test functions. They use sunspot forecasting data set. In this paper, there is the contribution of six-fold:

1. New genetic operators in improved GA.
2. Improved GA implemented in floating point numbers.
3. Improved GA has one parameter instead of three parameters.
4. Three-layer Neural Network.
5. It can be used to tune structure and parameters as well.

Here improved GA is similar to traditional GA but there are some changes like we select the chromosomes that have higher fitness value should have a higher chance to select. Then they use new genetic operation in floating point numbers and we replace the parent with the best child. Then they use benchmark test functions to examine the efficiency of the improved GA. They use five test functions and improved GA goes throw this function and they compare it with traditional GA performance. An improved GA has been proposed in this paper. The improved GA is implemented by floating-point numbers for learning using the improved GA is faster. New genetic operators have been used to the improved GA. It has been shown that the improved GA performs more efficiently than the traditional GA, by using the benchmark De Jong's test functions. Besides, they introducing a switch to each link in the NN that facilitates the tuning of its structure. Using the improved GA, the proposed neural network is able to learn both the input-output relationship of an application and the network structure. As a result, a given fully connected neural network reduced to a piece of a connected network. This implies that the cost of implementation of the NN can be reduced. As a result in training error and application example forecasting error that shows that improved GA is better than traditional GA.

Hiroaki Kitano [9] in his paper on “Empirical Studies on the Speed of Convergence of Neural Network training using Genetic Algorithms” took a systematic approach to compare the performance of Genetic Algorithms and Backpropagation in weight training of the neural network. The genetic algorithm as suggested in the paper has a problem with local fine-tuning. So to properly compare, a variation of Genetic algorithm with backpropagation and a faster variation of backpropagation named Quickprop was tested. To benchmark results the tests were done on XOR problem, various types of Encoder/Decoder problems and Two-Spirals problems.
In Genetic Algorithm, an array of floating point number was used which corresponds to weights and biases represented a neural network. The fitness function to evaluate chromosomes Total Sum Square(TTS) error was used. The fitness of a chromosome is = 1/TTS^2. measure. Reproduction strategy is a proportional reproduction which normalizes fit-ness and assigns higher reproduction probability for higher fitness chromosomes. In addition, they used an elitist re-production which always chooses the two best chromosomes and simply copies them, without crossover or mutation, to the population of the next generation.

2.3 Background Information

2.3.1. Convolutional Neural Nets

Recently the improvement in Convolutional Neural Network (CNN) [10] and advancement in computer vision technology has made things easier. From object recognition from images [11] to speech recognition from audio [12] CNN had showed remarkable results. Back-propagation algorithm was integrated in 1988 [13]. Now CNN has revolutionized unsupervised learning by identifying patterns in images and classifying them without the need for a lot of feature definition and pre-processing. Though it was not specifically made for images it has achieved very good results in problems like recognizing images and classifying them.

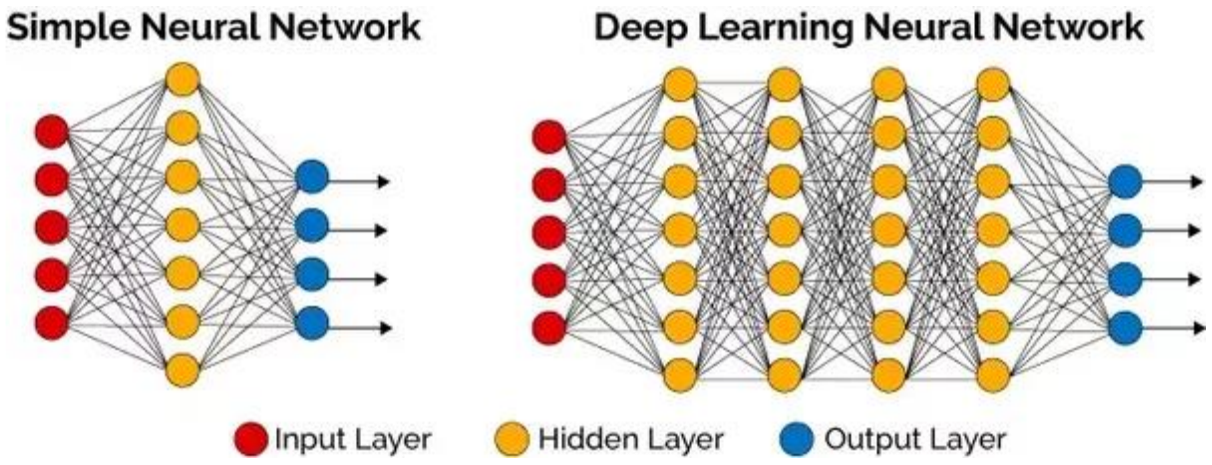


Figure 2.3.1.1: Simple Neural Network and Deep Learning Neural Network

Convolutional Neural Network (CNN) are getting very popular and showing promising results in the field of computer vision [14], [15]. The main reason behind its widespread use is its efficiency in pattern recognition in data. It can give better performance than other long-established methods [16].

CNN layers are made of two components. The convolutional layers and max-pooling layers. Convolutional layers have filters that take a small local part of the image and processes it and it is replicated on the whole image. Max-pooling layers take maximum filter activations from the filtered image and a convolutional layer of lower resolution is created. This process adds tolerance and translation invariance to little differences of positions in objects parts.

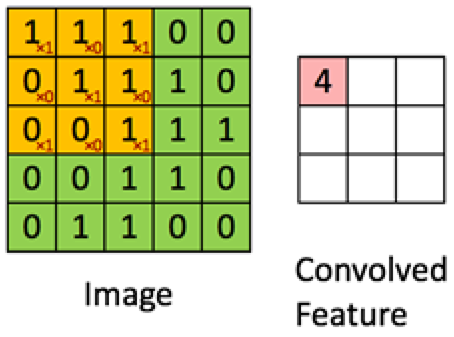


Figure 2.3.1.2: Convolution of a filter over 2D image

2.3.2. Genetic Algorithm

Genetic algorithm is a part of Evolutionary strategies. Evolutionary Strategies are set of algorithms that are inspired by the natural processes. Like particle swarm algorithm, ant colony algorithm or genetic algorithm. Genetic Algorithm is inspired by the evolution of animals.

There are 6 primary steps of Genetic algorithm. This algorithm is used to search for solution in a very large search space. The initial population is made, then a fitness function is evaluated. The fitness function is used to check the strength of solution.

1. Generate initial population
2. Design a fitness function
3. Evaluate the population using fitness function
4. Crossover between them to generate child
5. Mutate the generated child
6. Select the best child

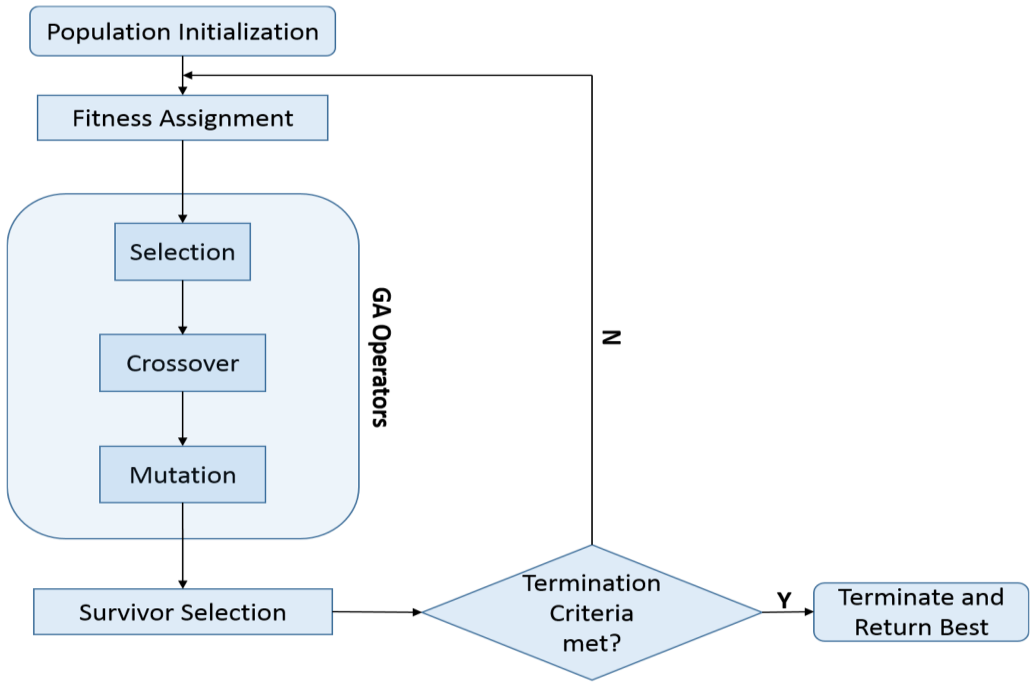


Figure 2.3.3.1: Genetic Algorithm Steps

2.4 Challenges

The primary challenge of this work was to design the gene and fitness functions. The selection of operators to mutate was also a problem. Without careful experiments, this will yield to wrong results. Another challenge was to collect and edit dataset. We collected our dataset to experiment along with MINST. Fashion MINST and PataNet dataset were used in this work. The Fashion MINST is already pre-processed. But the PataNet Dataset was not pre-processed.

CHAPTER 3
RESEARCH METHODOLOGY

3.1 Introduction

Many well-known research groups have started developing optimized evolutionary neural networks and incorporating them in their research works where the hyper-parameter space is huge. It recently has become very popular for fusing with Artificial Neural Networks. Also, among typical Evolutionary Algorithms. Genetic Algorithm is the most popular one to use with Neural Networks.

3.2 Research Subject and Instrumentation

Research subject is a research area which was reviewed and studied for clearing concepts. For our work it was Deep Learning, Metaheuristics and CNN. We learned not only for implementation but also to design model, collect data, process data and train the model. The other section is Instrumentation that is the technology and the methods we used. We used windows platform, python language with many packages like Numpy, Pandas, Scikit learn, Matplotlib, etc. Anaconda application was used for all the training and testing process, Anaconda is a free and open-source distribution of the Python and R programming languages for data science and machine learning applications.

3.3 Workflow

Primary 3 Steps:

- Algorithm Design
- Implement algorithm for Self Configuring Neural Networks
- Performance Evaluation(Accuracy Percentage)

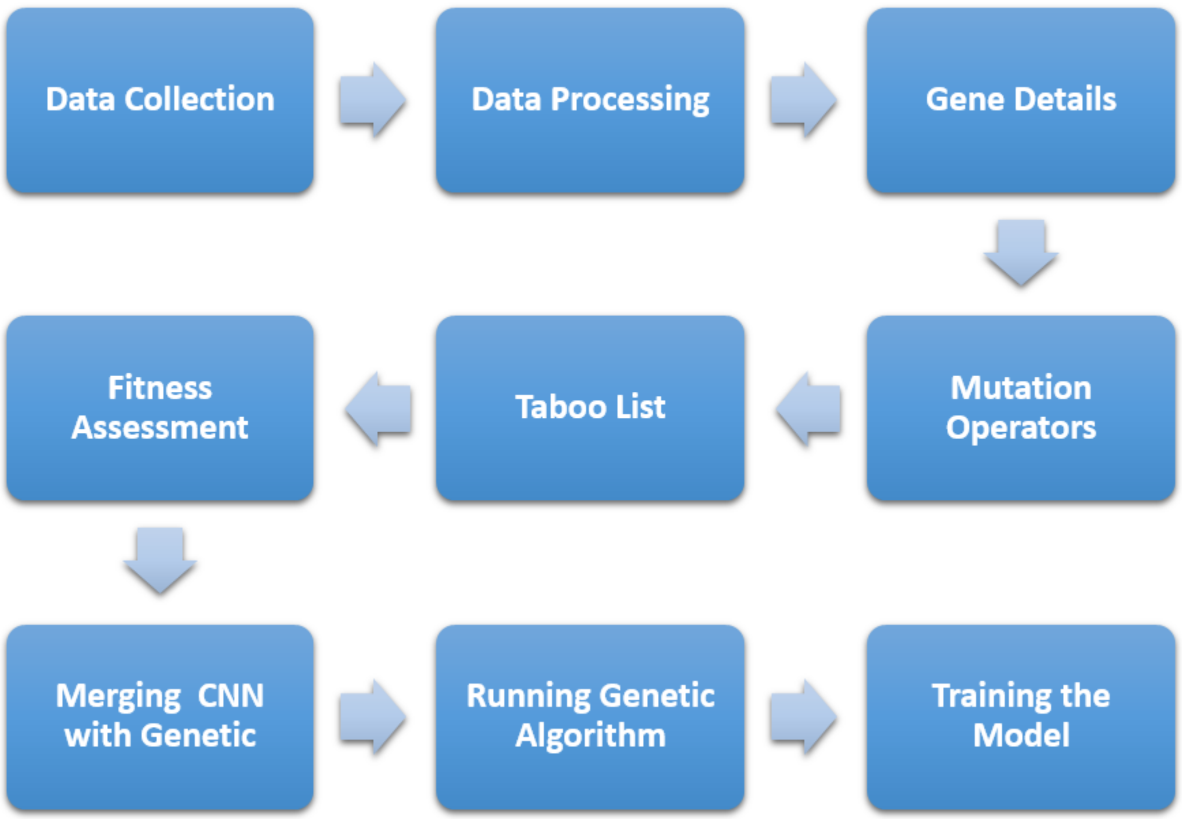


Figure 3.3.1: Workflow Breakdown.

3.4 Data Collection Procedure

The original MNIST dataset contains a lot of handwritten digits images. Members of the AI/ML/Data Science community love this dataset and use it as a benchmark to test their algorithms.

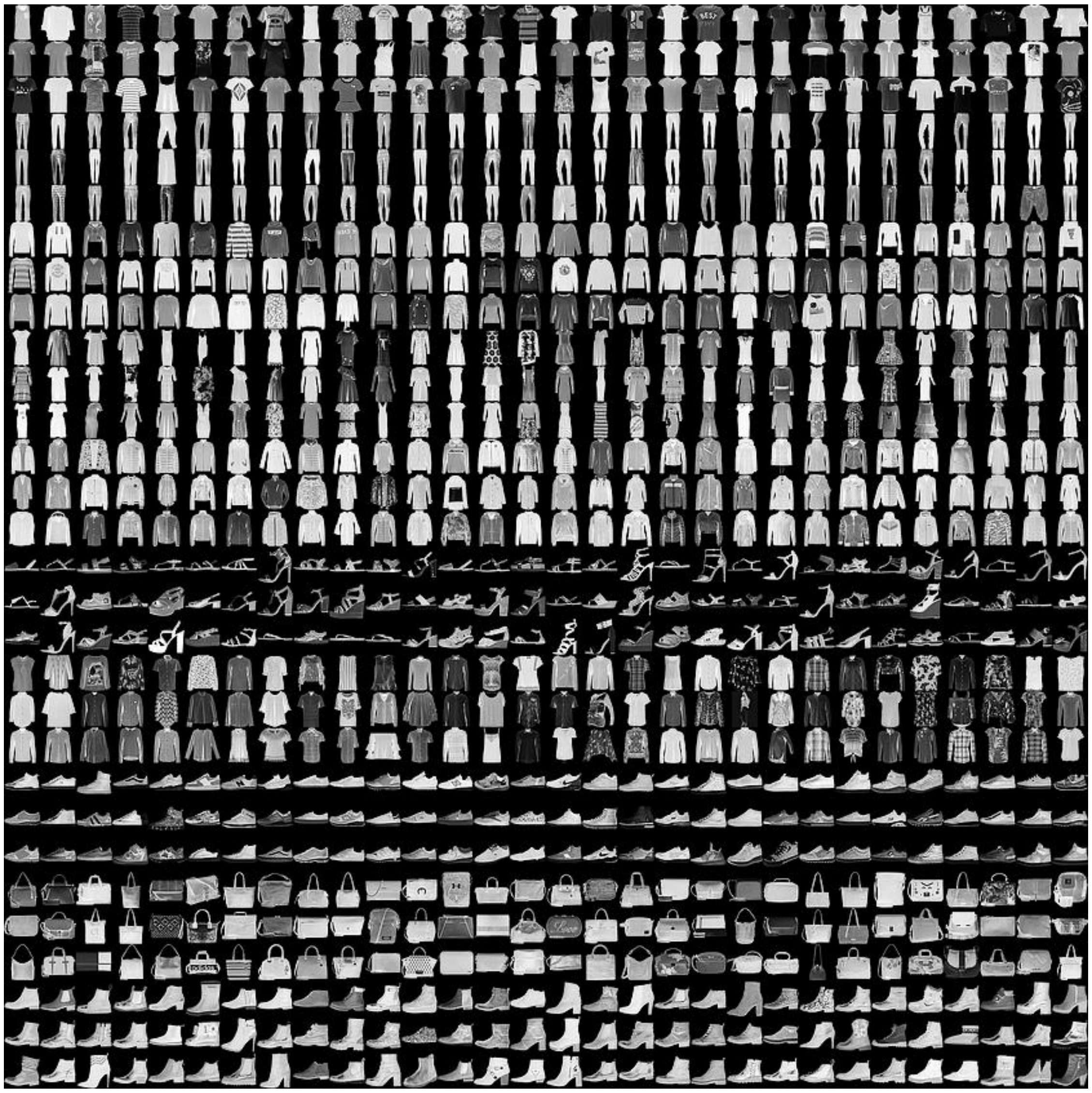


Figure 3.4.1: Fashion-MNIST dataset

The PataNet dataset has 3600 images of 6 plants species. They are divided into two categories. Category-1 is single leaf image. Category-2 is direct tree image.

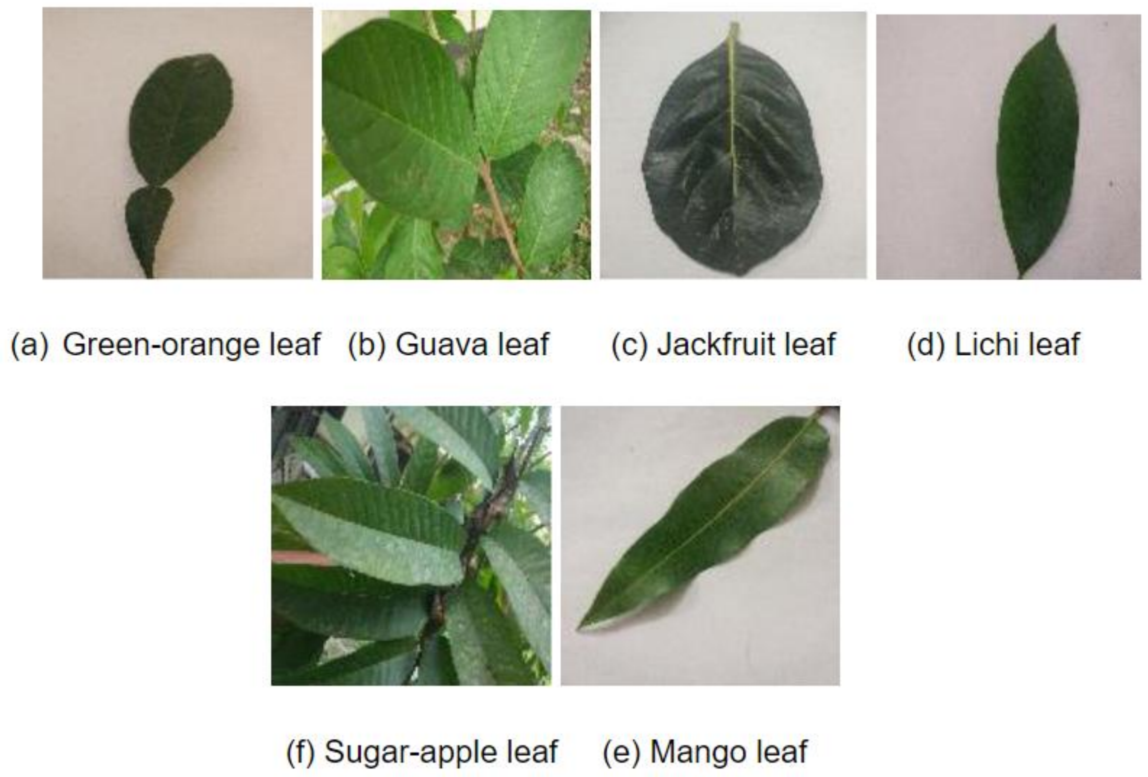


Figure 3.4.2: PataNet Dataset Classes

3.5 Data Processing

Fashion MNIST has 60,000 images. There are 10,000 images in test set. Images are of resolution 28 by 28. The intentions of making fashion MNIST is to test CNN networks performance. It replaced the traditional MNIST dataset of handwritten digits. This is used to benchmark CNN algorithms. The images are of 10 categories and grayscale.

PataNet dataset Category-1 images did not have a fully white background. So they were manually whitened to fit the dataset. It is easier for the CNN model to work with square images. So the initial images were cropped from wide to square. Images of the dataset were Full HD, meaning they have resolution over 1920 x 1080 pixel. Training the model with high-resolution image is computationally expensive, so the images were resized to 128 x 128 pixels. Also, to reduce the illumination differences, Minmax (1) normalization was used on the dataset.

$$Z_i = \frac{X_i - minimum(X)}{maximum(X) - minimum(X)} \quad (1)$$

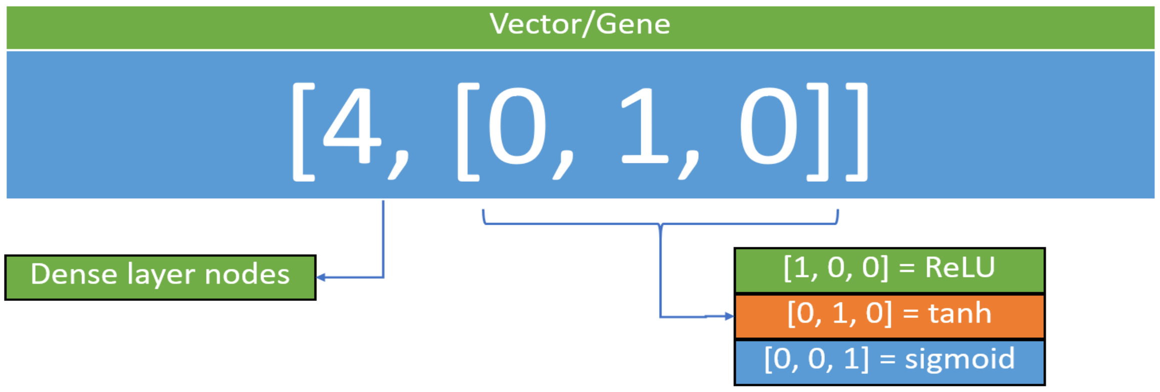
3.6 Proposed Methodology

At first, We imported the data and all other necessary modules and packages into our program we implemented Neural Network with structure selected by Evolution Strategies and randomly initialized weights which will be optimized using Backpropagation.

We applied each of those nets on the datasets and plotted training and validation loss and also calculated precision, recall, f-measure and accuracy acquired by using those nets on test dataset.

3.6.1. Chromosome and Gene Details

In (5,20) ES, each chromosome will be a vector like this [4, [1, 0, 0]]. Here, the first element will be an integer between 4-32 inclusive. This will indicate node count of each of the dense layer. Second element is a boolean vector which can be mapped - [1, 0, 0] = ReLU, [0, 1, 0] = tanh and [0, 0, 1] = Sigmoid.



Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool -> Fully Connected

Fig 3.6.1.1: Vector mapping

Equation of ReLU Activation Function:

$$\text{ReLU}(X) = \text{MAX}(0,X) \quad (2)$$

Equation of tanh Activation Function:

$$\tanh x = \frac{2}{1+e^{-2x}} - 1 \qquad (3)$$

Equation of Softmax Activation Function:

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_{j=1}^{n} e^{x_i}} \qquad (4)$$

Equation of Sigmoid Activation Function:

$$f(X) = \frac{1}{1+e^{-x}} \qquad (5)$$

3.6.2. Selection for Mutation

A coin toss will be done for selecting whether any gene of the chromosome will be mutated or not. For gene A (node count in the hidden layer), selection threshold was set to 30% and for Gene B (Boolean vector for activation function selection in hidden layer) was set to 50 Mutation Operators: Mutation operator for A is selecting a random number within [4, 32]. Used this instead of the random walk to enforce exploration a bit to avoid possible local minima. The mutation operator for Gene B is a simple bitwise right shift. For instance, [1,0,0] -> [0,1,0]; [0,1,0]-> [0,0,1] and [0,0,1]->[1,0,0]

3.6.3. Taboo List

We've also maintained a Taboo list. Whenever a new child is created, it is first checked whether it is in the taboo list. If yes, then another new child is created. If not, then the first element is popped from the taboo list and this new child is appended at the end of the taboo list besides being added as the new member of the main population. So, each new candidate will get a fair lifetime of 19 new not-present-in-taboo birth.

3.6.4. Fitness Assessment

Fitness Assessment was done using a simple forward pass without any backward pass. For each candidate structure, a neural network with random weight is created. Then a single forward pass is made and then cost was calculated. This is the measurement of fitness. Lower the cost means fitter the candidate.

3.6.5. Merging CNN with Genetic

We merged the model we created (Described Below) with genetic algorithm. We checked each chromosome/vector for one forward pause. So we inserted the CNN model inside the loop of chromosome generation. The loss calculated from the running of forward pause is then saved for future use of fitness assessment.

3.6.6 Convolutional Layer

Convolutional Layers are used mainly in processing images. Images has thousands of pixels associated with them. In order to process these huge data through neural net we use a technique called convolution. It uses different types of filters to detect patterns in images and reduce the image information to important only. The layers that does this tasks are known as convolutional layers. In our work we are using CNN that is Convolutional Neural Networks instead of traditional ANNs.

3.6.7 Max pooling

Max pooling layer is used in CNN networks. It is used for computation reduction. It reduces the data coming from the previous layer to small size and feeds to next network.

3.6.8 Dense Layer

Dense layers are the ANNs attached at the end of Convolutional Neural Network. It takes all the output of CNN and feeds to a basic ANN to generate the final output.

3.7 Training the Model

We ran our whole algorithm for 10 generation with population count of 20 in each generation. For each generation we took 5 best candidate and used them to create other children. The created children then assessed again. We saved the best candidate in a global variable.

Then we trained the best model given by the genetic algorithm.

3.8 Implementation Requirements

To run this model in the whole algorithm we had to use a good system. The system requirements are given below:

- Hardware:
    - Processor: Core i5
    - Ram: 12GB
    - GPU: Google Colab Service Provided.
- Software:
    - Jupyter Notebook
    - Language: Python 3.7
    - Libraries: Tensorflow, Pandas, Numpy

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Introduction

In this section, we are going to discuss the results that we got from the overall experiment. We will see the evaluation of performance and discuss results based on them at the end, using different types of measurements.

4.2 Performance Evaluation

The performance of this whole model can be determined by the ultimate result of the final model. This model was given after the final simulation of the genetic algorithm. The final trained model was applied to both of the datasets. Each dataset is divided into two categories the training and testing category. The accuracy of the test dataset is known as validation accuracy. Loss is the measurement of how the prediction is missing the actual data. The training loss determines the loss of the training dataset. The validation set determines the loss on the test dataset.

4.2.1 Performance without genetic algorithm

4.2.1.1 Fashion MINST Base Model

Here we created a plain model artibituraly and trained it on both of the data sets. From the first 2 graph below, we can see the model performance in MINST dataset. We can see that the model has very high rate of over-fitting. The validation accuracy is 90.09% but the training accuracy is 84.39%. So there is overfitting occurring in this model.

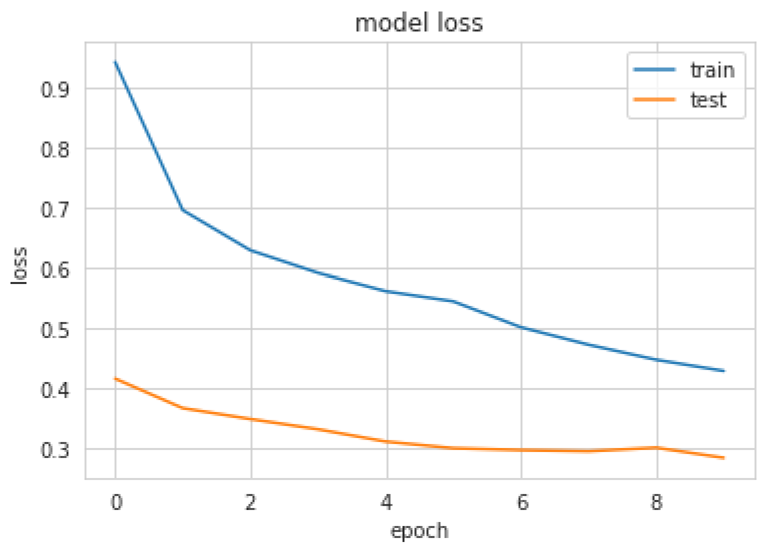


Figure 4.2.1: Training and validation loss

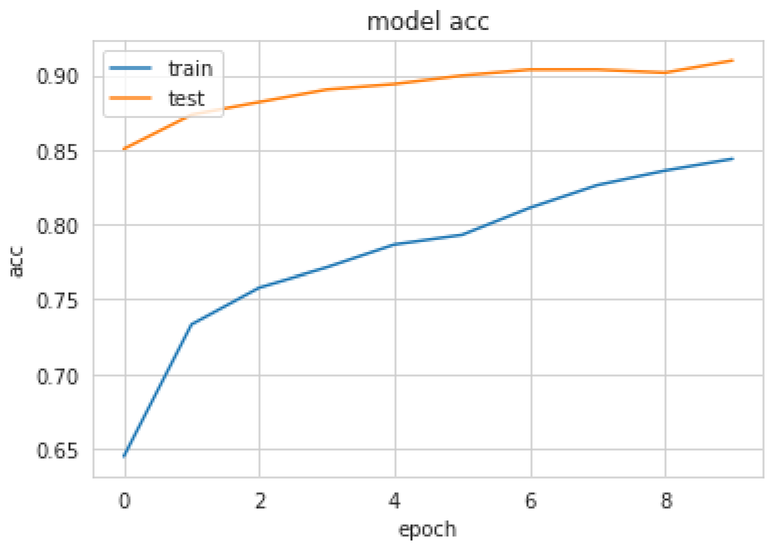


Figure 4.2.2: Training and validation accuracy

4.2.1.2 PataNet Base Model

We applied a second model in PataNet Dataset to test its accuracy. The model ran for 50 epoch resulted training accuracy 96.54% and validation accuracy 95.86%. At epoch 29 the learning rate was changed from 0.0005 to 0.0025. We can see improvement in accuracy

3% due to changing in learning rate. The learning rate is further reduced in iterations and the best resulted 96.54% training accuracy and 95.86% validation accuracy. We can see the results in below.

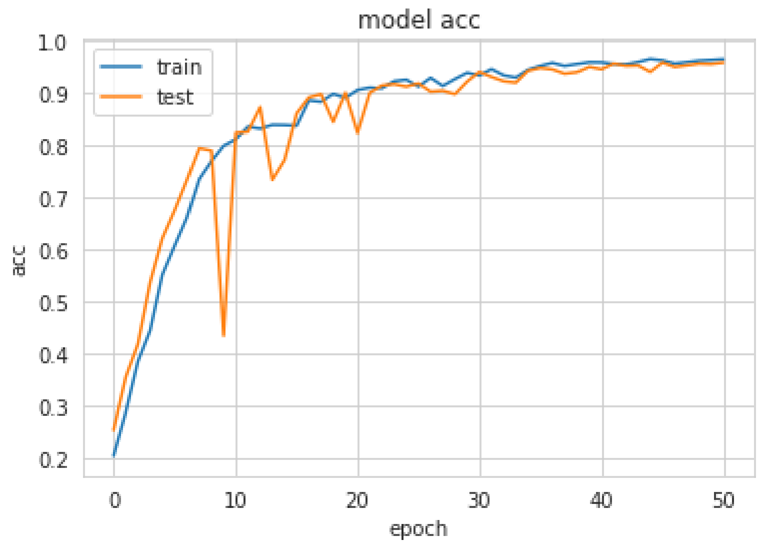


Figure 4.2.2: Training and validation accuracy

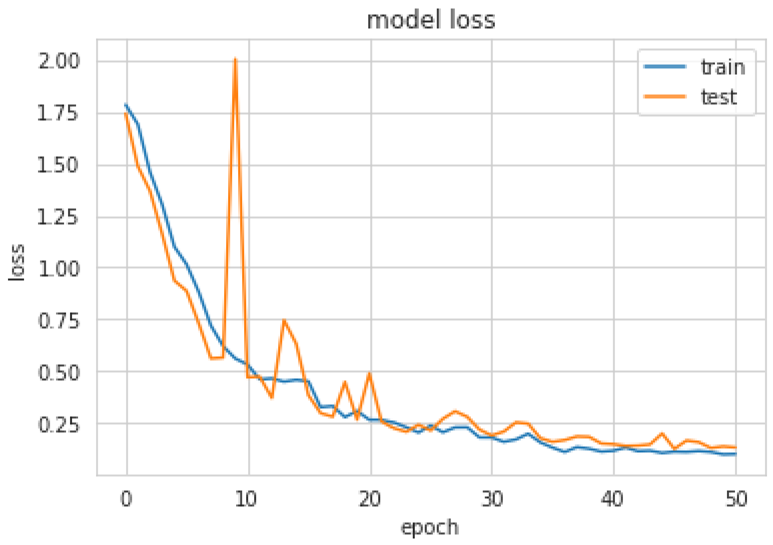


Figure 4.2.1: Training and validation loss

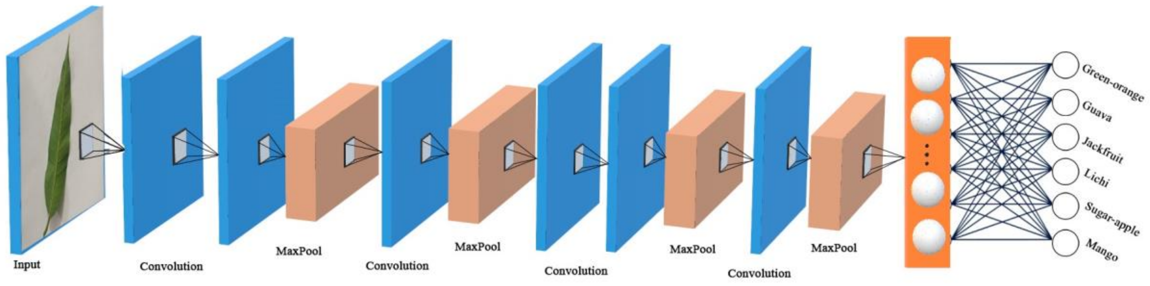


Figure 4.2.3: PataNet Model

The learning rate is further reduced in iterations and the best resulted 96.54% training accuracy and 95.86% validation accuracy.

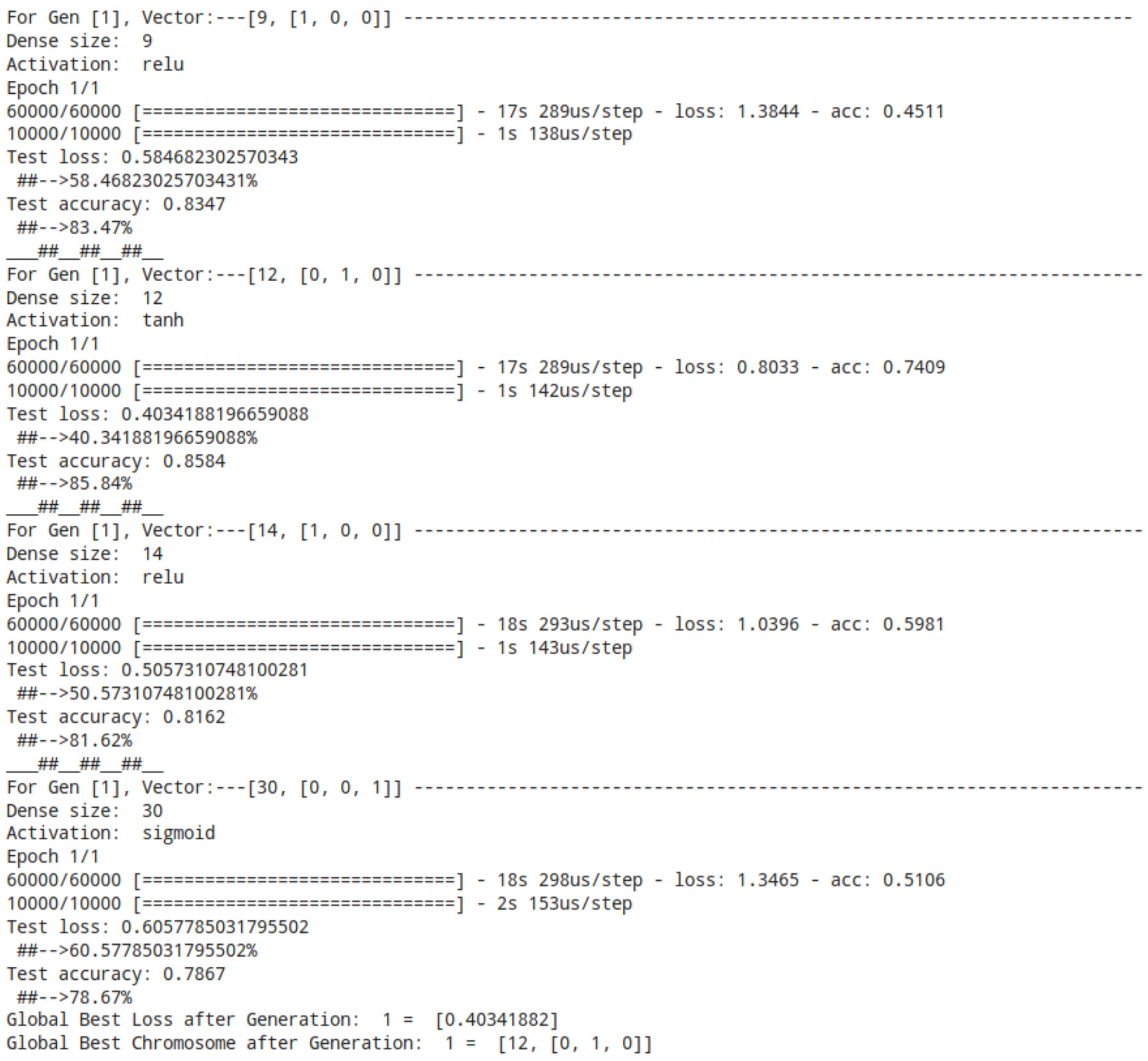
4.2.2 Performance with Genetic Algorithm

4.2.2.1 Fashion MINST with Genetic Algorithm

We applied genetic algorithm in Fashion MNIST dataset. We set the parameters

- Mutation = 5
- Population = 20
- Generation = 10

Mutation = 5, means 5 best parents will crossover to get new generation. There was population count of 10 that means population in each generation is 10. We ran the genetic simulation for 10 generations. This is the generation = 10.



```
For Gen [1], Vector:---[9, [1, 0, 0]] ----------------------------------------------------------------------
Dense size:  9
Activation:  relu
Epoch 1/1
60000/60000 [==============================] - 17s 289us/step - loss: 1.3844 - acc: 0.4511
10000/10000 [==============================] - 1s 138us/step
Test loss: 0.584682302570343
 ##-->58.46823025703431%
Test accuracy: 0.8347
 ##-->83.47%
___##__##__##__
For Gen [1], Vector:---[12, [0, 1, 0]] ----------------------------------------------------------------------
Dense size:  12
Activation:  tanh
Epoch 1/1
60000/60000 [==============================] - 17s 289us/step - loss: 0.8033 - acc: 0.7409
10000/10000 [==============================] - 1s 142us/step
Test loss: 0.4034188196659088
 ##-->40.34188196659088%
Test accuracy: 0.8584
 ##-->85.84%
___##__##__##__
For Gen [1], Vector:---[14, [1, 0, 0]] ----------------------------------------------------------------------
Dense size:  14
Activation:  relu
Epoch 1/1
60000/60000 [==============================] - 18s 293us/step - loss: 1.0396 - acc: 0.5981
10000/10000 [==============================] - 1s 143us/step
Test loss: 0.5057310748100281
 ##-->50.57310748100281%
Test accuracy: 0.8162
 ##-->81.62%
___##__##__##__
For Gen [1], Vector:---[30, [0, 0, 1]] ----------------------------------------------------------------------
Dense size:  30
Activation:  sigmoid
Epoch 1/1
60000/60000 [==============================] - 18s 298us/step - loss: 1.3465 - acc: 0.5106
10000/10000 [==============================] - 2s 153us/step
Test loss: 0.6057785031795502
 ##-->60.57785031795502%
Test accuracy: 0.7867
 ##-->78.67%
Global Best Loss after Generation:  1 =  [0.40341882]
Global Best Chromosome after Generation:  1 =  [12, [0, 1, 0]]
```

Figure 4.2.4: Genetic Algorithm Simulation

After running the simulation the best output of the simulation was the model with dense layer size of 99 nodes and activation function of tanh. We then ran the model again for 15 epochs.

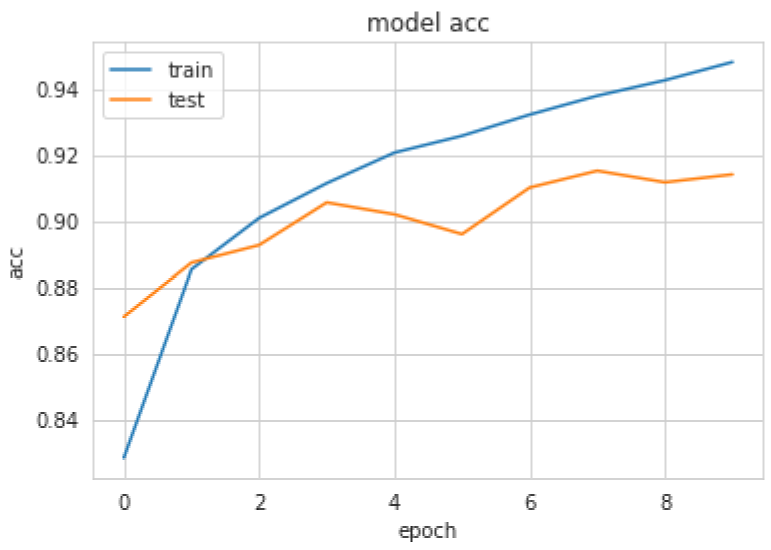


Figure 4.2.2: Training and validation accuracy

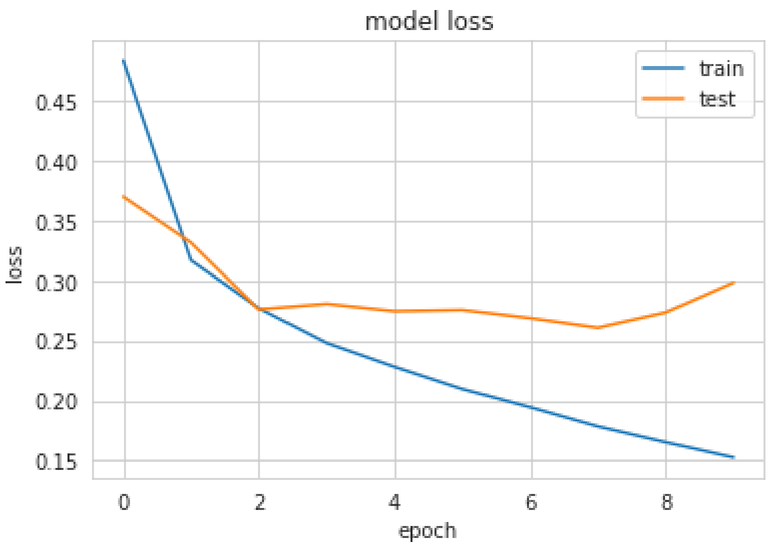
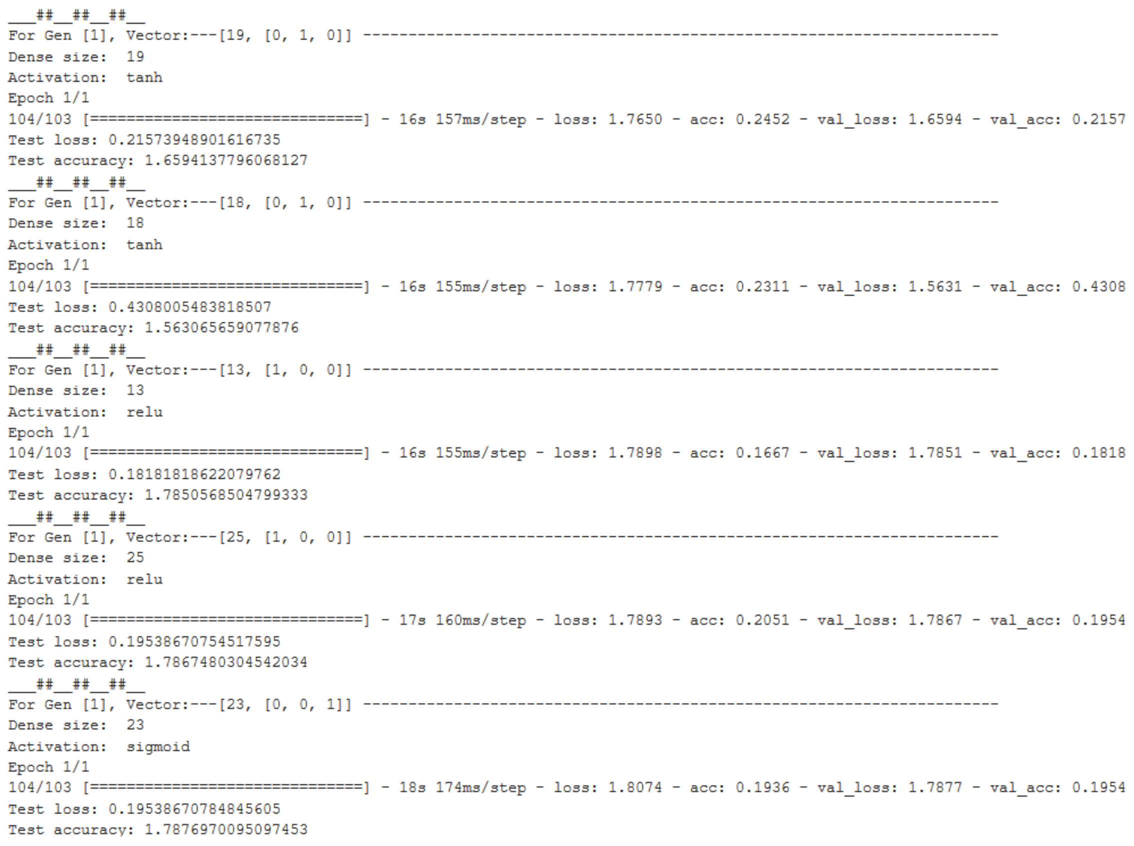


Figure 4.2.1: Training and validation loss

4.2.2.2 PataNet With Genetic Algorithm

We applied genetic algorithm in PataNet dataset. There was 20 population in each generation. We ran the genetic simulation for 10 generations. Same as the we set the parameters mutation = 5, that means 5 best parents will crossover to get new generation.



```
___##__##__##__
For Gen [1], Vector:---[19, [0, 1, 0]] ------------------------------------------------------------------------
Dense size:  19
Activation:  tanh
Epoch 1/1
104/103 [==============================] - 16s 157ms/step - loss: 1.7650 - acc: 0.2452 - val_loss: 1.6594 - val_acc: 0.2157
Test loss: 0.21573948901616735
Test accuracy: 1.6594137796068127
___##__##__##__
For Gen [1], Vector:---[18, [0, 1, 0]] ------------------------------------------------------------------------
Dense size:  18
Activation:  tanh
Epoch 1/1
104/103 [==============================] - 16s 155ms/step - loss: 1.7779 - acc: 0.2311 - val_loss: 1.5631 - val_acc: 0.4308
Test loss: 0.4308005483818507
Test accuracy: 1.563065659077876
___##__##__##__
For Gen [1], Vector:---[13, [1, 0, 0]] ------------------------------------------------------------------------
Dense size:  13
Activation:  relu
Epoch 1/1
104/103 [==============================] - 16s 155ms/step - loss: 1.7898 - acc: 0.1667 - val_loss: 1.7851 - val_acc: 0.1818
Test loss: 0.18181818622079762
Test accuracy: 1.7850568504799333
___##__##__##__
For Gen [1], Vector:---[25, [1, 0, 0]] ------------------------------------------------------------------------
Dense size:  25
Activation:  relu
Epoch 1/1
104/103 [==============================] - 17s 160ms/step - loss: 1.7893 - acc: 0.2051 - val_loss: 1.7867 - val_acc: 0.1954
Test loss: 0.19538670754517595
Test accuracy: 1.7867480304542034
___##__##__##__
For Gen [1], Vector:---[23, [0, 0, 1]] ------------------------------------------------------------------------
Dense size:  23
Activation:  sigmoid
Epoch 1/1
104/103 [==============================] - 18s 174ms/step - loss: 1.8074 - acc: 0.1936 - val_loss: 1.7877 - val_acc: 0.1954
Test loss: 0.19538670784845605
Test accuracy: 1.7876970095097453
```

Figure 4.2.2.2.1: Genetic Algorithm Simulation For Patanet

The followings are the result of final model. The dense layer resulted in 245 and it gave as the best activation function output.

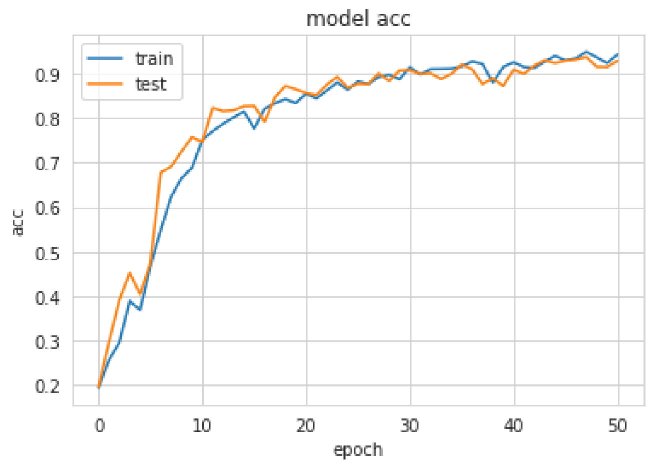


Figure 4.2.2: Training and validation accuracy

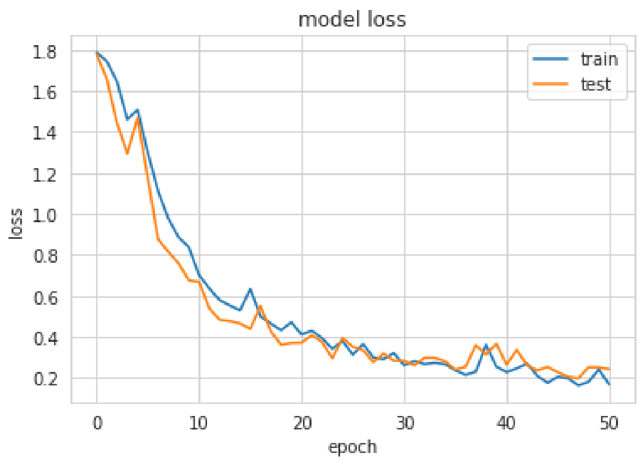


Figure 4.2.1: Training and validation loss

TABLE 1: DATASET INFORMATION

Name	Train Data	Test Data	Input shape	Labeled
Patanet	3091	784	48*48	YES
MINST Fashion	60000	10000	28*28	YES

4.3 Result Discussion

From the first experiment, we saw how a seed model performed on the MINST dataset. We just implement a model and measured its accuracy. The models showed a high rate of overfitting. But the accuracy was good at the first go.

We then applied our Genetic Algorithm on the seed model. We generated the population and the algorithm selected the best candidate. After running for 10 generations. We got the best candidate. We ran a single epoch and measured the loss to determine the best candidate. We then trained that candidate for a long period. We saw a slight increase in accuracy, close to 2% but the increase overfitting reduced.

In the second experiment, we applied our well-established model of PataNet and got the results. But then we used the same model in our genetic algorithm. Again this model ran for 10 generations to give the best structure. We saw a slight increase in inaccuracy. As the model had no overfitting previously. The effect of the genetic algorithm on the overfitting criteria was negligible.

The below-given table shows a summary of the results of both experiments.

TABLE 2: SUMMARY OF THE RESULTS OF BOTH EXPERIMENTS

Model Name	Dese	Activation	Train Acc	Validation Acc
Minst (Base)	1024	relu, softmax	90.09%	84.39%
Patanet (Base)	128	relu, softmax	96.54%	95.86%
Minst (GA)	124	tanh, softmax	94.80%	91.45%
Patanet (GA)	127	relu, softmax	95.89%	95.39%

CHAPTER 5

CONCLUSION AND FUTURE WORKS

5.1 Conclusion

From the above discussion, we can say that a genetic algorithm can be used to evolve a basic network. It will take the network and create generations. At each generation, we will get a better network. It can also reduce overfitting. But from the second experiment, we can see that established model like PataNet models is already at its best. The genetic algorithm showed little improvement there. So if we apply the genetic algorithm at the very initial stage of model creation it will help a lot to get out the best model without any human intervention. But for a model that reached its peak. Running genetic algorithm there has little impact.

5.2 Future Works

In this work we established a genetic algorithm that is used to find the best structure for a model. But there were some drawbacks.

Firstly we used only established the genetic algorithm for two hyper parameters. It can be extended to other parameters like the hidden layer count, hidden layer node count, stride and filter size and so on.

But as we saw some improvements in optimizing only 2, we think adding more parameters will result in more robust models.

Also there is a huge choice on how to choose mutation, different types of mutations will result in different types of results. There is a scope of work here. Also this hybrid structure of Genetic Algorithm Generated CNN model can be applied to other varieties of models to find if they can be improved as well.

REFERENCES

[1] Medium. (2019). Notes on Artificial Intelligence, Machine Learning and Deep Learning for curious people. [online] Available at: https://towardsdatascience.com/notes-on-artificial-intelligence-ai-machine-learning-ml-and-deep-learning-dl-for-56e51a2071c2 [Accessed 29 Oct. 2019]

[2] Cognilytica. (2019). Will There Be Another AI Winter? | Cognilytica. [online] Available at: https://www.cognilytica.com/2018/02/22/will-another-ai-winter/ [Accessed 29 Oct. 2019].

[3] Quora.com. (2019). Why do deep learning algorithms need more data than machine learning algorithms? - Quora. [online] Available at: https://www.quora.com/Why-do-deep-learning-algorithms-need-more-data-than-machine-learning-algorithms [Accessed 29 Oct. 2019].

[4] Medium. (2019). Artificial Intelligence Hardware. [online] Available at: https://becominghuman.ai/artificial-intelligence-hardware-76fa88581e53 [Accessed 29 Oct. 2019].

[5] Nn.cs.utexas.edu. (2019). NNRG Projects - NEAT: Evolving Increasingly Complex Neural Network Topologies. [online] Available at: http://nn.cs.utexas.edu/?neat [Accessed 29 Oct. 2019].

[6] D. Montana, L. Davis, "Training feedforward neural networks using genetic algorithms", *Proc. 11th Int. Joint Conf. Artificial Intelligence*, pp. 762-767, 1989.

[7] Miller, Geoffrey F., Peter M. Todd, and Shailesh U. Hegde. "Designing Neural Networks using Genetic Algorithms." ICGA. Vol. 89. 1989.

[8] F. Leung, H. Lam, S. Ling and P. Tam, "Tuning of the structure and parameters of a neural network using an improved genetic algorithm", IEEE Transactions on Neural Networks, vol. 14, no. 1, pp. 79-88, 2003.

[9] Kitano, Hiroaki. "Empirical Studies on the Speed of Convergence of Neural Network Training Using Genetic Algorithms." In AAAI, pp. 789-795. 1990.

[10] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” science, vol. 313, no. 5786, pp. 504–507,2006.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only lookonce: Unified, real-time object detection,” inProceedings of the IEEEconference on computer vision and pattern recognition, 2016, pp. 779–788.

[12] Y. LeCun, Y. Bengioet al., “Convolutional networks for images, speech,and time series,”The handbook of brain theory and neural networks,vol. 3361, no. 10, p. 1995, 1995.

[13] D. E. Rumelhart, G. E. Hinton, R. J. Williamset al., “Learningrepresentations by back-propagating errors,”Cognitive modeling, vol. 5,no. 3, p. 1, 1988.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classificationwith deep convolutional neural networks,”Advances in neural informa-tion processing systems, pp. 1097–1105, 2012.

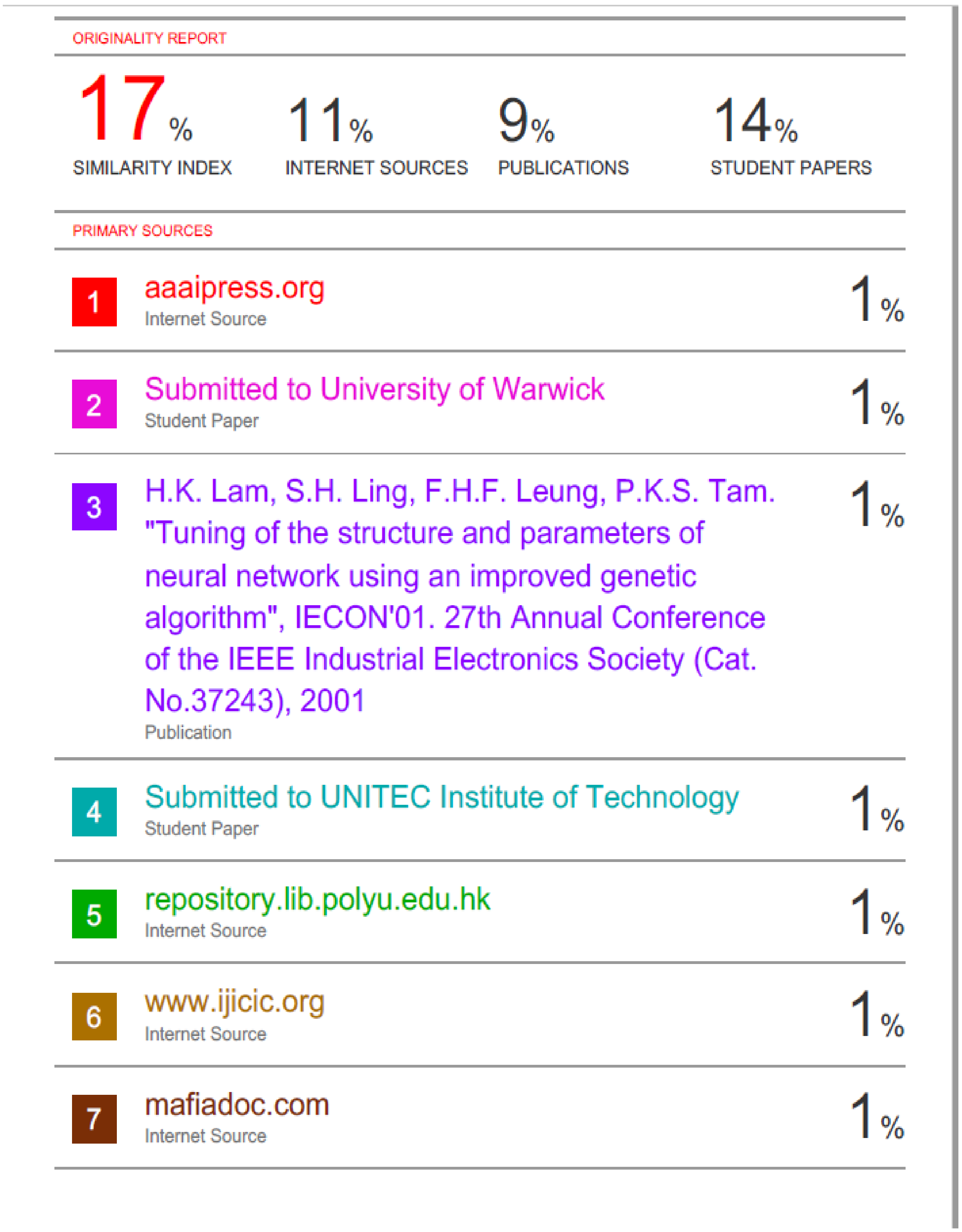
[15] K. Simonyan and A. Zisserman, “Very deep convolutional networks forlarge-scale image recognition,”arXiv preprint arXiv:1409.1556, 2014.

[16] K. Chatfield, V. S. Lempitsky, A. Vedaldi, and A. Zisserman, “The devilis in the details: an evaluation of recent feature encoding methods.”BMVC, vol. 2, no. 4, p. 8, 2011

APPENDIX

Appendix A: Related Issues

We faced some complicated issues doing this project. We had to collect data for a long time, especially for PataNet dataset. Also we had to study and combine the two algorithms which were not designed to combine. Specially making the genetic algorithm we faced difficulties to implement the mutation functions. We had to find the operators to properly configure. Also we had to study CNN and find the hyper parameters to optimize.



ORIGINALITY REPORT

17%	11%	9%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	aaaipress.org Internet Source	1%
2	Submitted to University of Warwick Student Paper	1%
3	H.K. Lam, S.H. Ling, F.H.F. Leung, P.K.S. Tam. "Tuning of the structure and parameters of neural network using an improved genetic algorithm", IECON'01. 27th Annual Conference of the IEEE Industrial Electronics Society (Cat. No.37243), 2001 Publication	1%
4	Submitted to UNITEC Institute of Technology Student Paper	1%
5	repository.lib.polyu.edu.hk Internet Source	1%
6	www.ijicic.org Internet Source	1%
7	mafiadoc.com Internet Source	1%