

A Phase - I Project Report on

Multi-Objective Optimization for the Design of Neural Networks

Submitted in partial fulfillment of the requirement for the degree of

Master of Technology In

Computer Science and Engineering

Submitted by:

Pratibha M Goudar [01FE20MCS010]

Under the guidance of

Dr. Meena S M

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

KLE TECHNOLOGICAL UNIVERSITY, HUBBALLI – 580031

2021-22

2021 - 2022



SCHOOL OF COMPUTER SCIENCE &ENGINEERING



CERTIFICATE

This is to certify that a Phase-I Project titled **“Multi-Objective Optimization for the Design of Neural Network”** is a bonafide work carried out by the student comprising of **PRATIBHA M GOUDAR 01FE20MCS010** for partial fulfillment of completion of third-semester M.Tech in Computer Science and Engineering during the academic year 2021-22.

Guide P. G. Coordinator Head, SoCSE

Dr. Meena S M. Dr. Vishwanath. P. Baligar Dr. Meena S. M

External Viva -Voce:

Name of the Examiners Signature with date 1.

2.

**ACKNOWLEDGEMENT**

The successful completion of any task is incomplete without complementing those who made it possible and whose guidance and encouragement made my effort successful. Directions were mainly responsible for my successful completion of the project.

I also take this opportunity to express my deep gratitude and sincere thanks to the HOD of the School of Computer Science & Engineering, my beloved Guide Dr. Meena S. M. for her tremendous source of inspiration and valuable help in challenging my effort in the right direction. Who was a constant source of enthusiasm and whose sincere guidance, valuable suggestions, and benevolent direction were mainly responsible for my successful completion of the project.

I am indebted to our beloved P. G. Coordinator Dr. Vishwanath. P. Baligar is a constant source of enthusiasm and whose sincere guidance, valuable suggestions, and benevolent direction were mainly responsible for my successful completion of the project.

I take this opportunity to thank our Registrar Dr.N. H. Ayachit and our Vice-Chancellor Dr. Ashok.S.Shettar, for providing a healthy environment in the college, which helped in concentrating on the task.

Finally yet importantly I will take the pleasure to thank all the teaching and non-teaching staff of our college and also my student friends.

I also thank K.L.E Technological University for providing computational support.

PRATIBHA M GOUDAR: [01FE20MCS010]

ABSTRACT

Neural Architecture Search is one of the prominent emerging sub-domains in Automatic Machine Learning (AutoML) which makes it easy to build architectures automatically for a given data set, objectives, and input parameters. This makes its application more robust to build any kind of deep learning model. In this project, we build a deep learning model for image classification tasks specifically for use on resource-constrained embedded devices. We use a Multi-objective optimization-based Evolutionary algorithm (NSGA-NET)to search the architecture for accuracy, maximize performance on computing devices that are often constrained by hardware resources in terms of power consumption, available memory, available FLOPs, and latency constraints, to name a few. NSGA-Net is explicitly designed to optimize such competing objectives. Moreover, we fine-tune the searched architecture for a specified value of accuracy. We deliver the final model based on searched architecture, trained weights.

**Keywords:** Machine Learning, Deep Learning, Evolutionary Algorithm, Multi-objective

CONTENTS

|  |  |  |
| --- | --- | --- |
| Chapter No | TABLE OF CONTENTS | Page No |
|  | **ACKNOWLEDGEMENT** | i |
|  | **ABSTRACT** | ii |
|  | **CONTENTS** | iii |
|  | **LIST OF FIGURES** | v |
| 1 | **INTRODUCTION** | 8 |
|  | 1.1 Preamble | 8 |
|  | 1.2 Motivation | 8 |
|  | 1.3 Problem Statement | 8 |
|  | 1.4 Objectives | 8 |
|  | 1.5 Evolutionary Computation | 9 |
|  | 1.5.1 chromosomes | 9 |
|  | 1.5.2 Crossover Techniques | 9 |
|  | 1.5.3 Exploration and Exploitation | 10 |
|  | 1.5.4 Selection | 10 |
|  | 1.5.5 Types of Evolutionary Computation | 11 |
| 2 | **LITERATURE SURVEY** | 12 |
| 3 | **PROPOSED SYSTEM** | 14 |
|  | 3.1 Description of the proposed system with a block diagram | 14 |
|  | 3.2 Advantages of the proposed system | 15 |
|  | 3.3 Scope and Boundary of the proposed system | 15 |
| 4 | **SOFTWARE REQUIREMENT SPECIFICATION** | 16 |
|  | 4.1 Overview of SRS | 16 |
|  | 4.2 Requirement Specification | 16 |
|  | 4.2.1 Functional requirements | 16 |
|  | 4.2.2 Non-Functional requirements | 16 |
|  | 4.3 Software and Hardware Requirements | 17 |
| 5 | **SYSTEM DESIGN** | 18 |
|  | 5.1 Architecture of the system | 18 |

iii

|  |  |  |
| --- | --- | --- |
| 6 | **IMPLEMENTATION** | 20 |
|  | 6.1 Proposed Methodology | 20 |
|  | 6.2 Dataset Description: Cifar10 | 20 |
|  | 6.3 Description of modules | 21 |
| 7 | **RESULTS AND DISCUSSIONS** | 22 |
| 8 | **CONCLUSION AND FUTURE SCOPE** | 25 |
|  | **REFERENCES** | 26 |

iv

LIST OF FIGURES

|  |  |  |  |
| --- | --- | --- | --- |
| Sl.No | Figure | Figure Name | Page No |
| 1 | 3.1 | Workflow of system | 14 |
| 2 | 5.1 | High-level design | 18 |
| 3 | 5.2 | Detailed design | 19 |
| 4 | 6.1 | The Cifar10 dataset | 20 |
| 5 | 7.1 | Macro search space | 22 |
| 6 | 7.2 | Macro visualization NSGA-Net | 23 |
| 7 | 7.3 | Micro search space | 24 |
| 8 | 7.4 | Micro visualization NSGA-Net | 24 |

v

**Chapter1**

**Introduction**

Neural networks are a series of algorithms that mimic the operations of a human brain to recognize the relationship between vast amounts of data. Multiobjective optimization has been applied to many fields of science, including engineering, where optimal decisions need to be taken in the presence of trade-offs between two or more objectives that may conflict. Multi-objective algorithms have become popular in the last years to solve the problem of the simultaneous training and topology optimization of neural networks. Optimizers are algorithms or methods used to change the attributes of neural networks such as weights and learning rate to reduce the losses.

* 1. **Preamble**

Binary Neural Network [4] is a quantized neural network where weights activations are binarized, neural architecture search is a subdomain of AutoML, where the search for the best architecture for the neural network is automated.[2]Neural Architecture Search has 3 major components: 1. Search Space 2. Search Strategy 3. Performance Estimation Strategy We apply Neural Architecture Search methods on Binary Neural Networks for image classification.

* 1. **Motivation**

Currently, building efficient neural networks is time-consuming, error-prone computationally expensive; there is a need to democratize the development of neural networks for its wider use [8]. NAS allows for the automated search of the best architecture. We also need to ensure that such networks work on embedded devices, which are resource-constrained [1].

* 1. **Problem Statement**

Design and development of neural networks using multiobjective optimization by considering the latency, accuracy, and power usage to classify images using the Neural Architecture Search method.

* 1. Objectives
* The major goal in our implementation is to improve the accuracy of the model.
* To reduce latency in performing operations.
* To have minimum FLOPS and power usage.
* To implement a multi-objective optimization technique that is based on the NSGA-II algorithm [13].
  1. **Evolutionary Computation**

Evolutionary computation is the method of applying the concept of evolution to optimize a solution or solve problems. Evolutionary computation is a method of machine learning that is proven effective to many applications and highly scalable. If a problem solution can be reduced to a series of values then the problem is a potential candidate for evolutionary computation.

**1.5.1Chromosomes**

Every potential solution to a problem is referred to as a chromosome. How each chromosome is defined dramatically affects how well the evolutionary computation performs. The chromosome definition could dramatically reduce the search space removing either redundant or incompatible solutions. Two problems that are widely used, as examples for evolutionary computation are the traveling salesman and the knapsack problem.

The traveling salesman problem consists of a salesperson that has to go to many cities and end back home. While there are, many possible solutions to the problem the salesperson wants to know what path will get him home the fastest. The solution to the problem would be a list of cities in the order in which he would visit. Therefore, to improve the speed of optimization, the number of potential solutions needs to be minimized. An example of this minimization for the traveling salesman problem can be, not allowing solutions with repeats of the same city to be generated. The knapsack problem is another problem, which is common. The problem is defined as; if the knapsack has a limited amount of space, fill it with the greatest value. There will be many items of different sizes and values. The task is to fit the most items with the highest value in the bag without overfilling it. The chromosome would then be the items, which go into the knapsack after each item is assigned a unique identifier.

**1.5.2 Crossover Techniques**

In some evolutionary computation techniques, there is a process called crossover. This is the combination of chromosomes to produce new chromosomes. These new chromosomes are referred to as children and the chromosomes that generated the child are referred to as a parent. Most crossover techniques have two parents producing one or two children, but this varies dramatically on each application. A few popular techniques are n-point crossover, arithmetic crossover, and uniform crossover.

The first discussed technique is the n-point crossover. The value n can range too many values, but the most common of which is a single point crossover. There will be two parents to produce two children. First, it picks a point at random then each parent is broken in half from that point. Once the parents are split each half from the parents is swapped with the other half from the other parent. This produces two unique children. The value n for the n-point crossover points signifies the number of breakpoints for each chromosome. The simplest variation of this crossover is a single-point crossover.

A single point crossover is when there is only a single point of separation from each parent. Another popular one is arithmetic crossover. The most common of which is taking two parents to produce a single child. This crossover adds the two parents together then dividing by two. The parents create the child by averaging themselves together. This method prompts much faster convergence, thus lacks exploitation. The uniform crossover technique takes two parents and generates two children like the n-point crossover. However, instead of declaring a certain number of breakpoints half of each parent will be picked at random. Yet, any part that does not make it to the first child is the part of the second child. This allows for exactly half of both parents to exist with each child.

**1.5.3 Exploration and Exploitation**

Exploration and exploitation is the largest problem any search optimization technique encounters. A specific scenario is with infinite search spaces. This search space could then only contain a singular global minimum. This infinite search space could also have many local minimums so it becomes difficult to determine when the algorithm has found the global minimum. This is where the concept of exploration vs. exploitation is important. Exploration is the process of trying samples scattered through a search space. Exploitation is the process of trying samples near well-performing solutions. The best method is to explore the search space and once enough information is gathered the algorithm should start to exploit it. When performing the crossover techniques this usually promotes more exploration than exploitation. Another method exists to control the concept of exploration vs. exploitation.

The common technique for exploring the search space is by randomizing the children created from the crossover. This method is referred to as a mutation. The amount of mutation can be controlled by limiting how often a gene mutates. This probability of mutation is referred to as a mutation rate. In most applications, it is best to start with a higher mutation rate and lower it slowly after each generation. This is the primary technique to ensure adequate exploration vs. exploitation.

**1.5.4 Selection**

The selection process is another key aspect of evolutionary computation. This step chooses which members of the populace to keep with each generation. Common selection processes are elitist selection, roulette wheel selection, and tourney selection. The selection process is important since it is an additional method in controlling exploration vs. exploitation. Elitist selection is the most common and the simplest selection technique. The method is to only keep the strongest chromosomes. This makes sure that no valuable chromosomes are lost. However, this highly promotes exploration overexploitation. This could to very fast convergence, but to a poor solution.

Roulette wheel selection creates a probability distribution when selecting which chromosomes stay in the populace. The chromosomes with higher scores are weighted so they are selected more often. This is a compromise between explorations vs. exploitation. There are some instances where it is hard to quantitatively, evaluate how a chromosome is performing. An example of this would be if evolutionary computation were being used to optimize a robot to play soccer. A robot by itself is hard to objectively assign a performance value. Instead of comparing a single value, each chromosome will compete against the other. The winners are chosen by hosting a tournament against the other chromosomes with each generation. The top winners of the tournament are then kept in the populace.

1.5.5 Types of Evolutionary Computation

Evolutionary computation contains many subdivisions such as genetic algorithms, evolutionary strategies, and genetic programming. These methods are chosen based on the application. In each of these sub-divisions, there can also be a drastic variation in the implementation. Genetic algorithms are the most common and basic variation of evolutionary computation [1]. Genetic algorithms require the problem to have a solution defined as a set of parameters, which defines the chromosome. Genetic algorithms also need the problem to have a way to evaluate the solution. Genetic algorithms then use the techniques described above such as crossover, mutation, and selection to find an optimized solution through the process of evolution.

Evolutionary strategies attempt to simplify and increase the adaptability of genetic algorithms. Genetic algorithms require a significant amount of fine-tuning when using them on an application. Most evolutionary strategies attain this by simplifying the mutation rate by adding the mutation rate into the chromosome. Evolutionary strategies also remove the crossover technique using only mutation to generate new chromosomes. Other techniques even have a mutation rate for each gene. Evolutionary strategies assign a mutation rate to each gene. This evolves the amount of exploration and exploitation for each gene. Another type of evolutionary computation is genetic programming. This is unique in how it evaluates chromosomes. The chromosome passes through a grammar engine, which generates code. This code is then compiled and evaluated to determine the performance. This makes development easier, on the other hand, has difficulties in converging to a solution. The complexity of convergence is due to the intricacy of programming languages.

**Chapter 2**

Literature Survey

Researchers have carried out work on the deep convolutional neural network (CNN) is the state-of-the-art (level of development) solution for large-scale visual recognition.[1] Researchers Lingxi Xie, AlanYuille follows some of the basic principles such as increasing network depth and constructing highway connections, researchers have manually designed a lot of fixed network architectures and verified their effectiveness. They have published this paper in the year 2017, IEEE International Conference on Computer vision. In this paper, researchers discuss the possibility of learning deep network structures automatically. Researchers' main idea is to use an encoding scheme to represent each network structure as a fixed-length binary string, and evaluate each generated individual via a standalone training process on a reference dataset. They used CIFAR 10, ILSVRC2012 Dataset. This paper applies the genetic algorithm to automatically learn the structure of deep convolutional neural networks.

Further researchers Zhichao Lu, Ian Whalen, Vishnu Boddeti, et al, have used for NSGA-Net for population-based search algorithm that explores a space of potential neural network architectures in three steps Initialization step, Exploration step, Exploitation step and they used CIFAR-10. Optimizing both prediction performance and computational complexity NSGA-Net finds networks that are significantly better than handcrafted networks [2].

Zhichao Lu, Ian Whalen, Yashesh Dhebar, et al worked on They used genetic operators and a Bayesian-model-based learning procedure. Used the LEMONADE method, which is formulated to develop networks with high predictive performance and lower resource constraints. And they used CIFAR-10, and CIFAR-100, ImageNet. Built NSGANetV1 for search efficiency .and provides an application to common thorax disease classification on human chest X-rays [3].

Further Kalyanmoy Deb, Erik Goodman et al worked on NSGANetV2: Evolutionary Multi-objective Surrogate-Assisted Neural Architecture Search. The Proposed idea for an efficient NAS algorithm for generating task-specific models that are competitive under multiple competing objectives. It comprises of two surrogates, one at the architecture level to improve sample efficiency and one at the weights level, through a supernet, to improve gradient descent training efficiency. They used datasets (C10, C100, ImageNet), the resulting models like NSGANetV2 [4].

Authors Kaoutar, Ettaouil Mohamed worked on Multi-criteria optimization of neural networks using multi-objective genetic algorithm [5]. They propose a new multi-objective model optimization that allows training the multi-layer perceptron neural network (MLPNN) and optimizing its architecture. This model aims to satisfy two objectives: the first one is minimizing the perceptron error (training objective) and the second one is minimizing the sum of the absolute weights (optimizing architecture objective).To solve the proposed model, They have chosen the NSGA II algorithm (Non- Dominated Sorting Genetic Algorithm II).

Further researcher Kalyanmoy Deb [6] suggests a nondominated sorting-based multiobjective EA (MOEA), called nondominated sorting genetic algorithm II (NSGA- II). This alleviates 1) computational complexity, 2) nonelitism approach 3) the need for specifying a sharing parameter. the proposed NSGA-II was able to maintain a better spread of solutions and converge better in the obtained non-dominated front compared to two other elitist MOEAs—PAES and SPEA.

Again, researcher N.Srinivas, Kalyanmoy Deb [7] carried out, even though there exist several classical multiobjective optimization techniques; they require some a priori problem information. Because genetic algorithms use a population of points, they may be able to find multiple Pareto-optimal solutions simultaneously. In this paper, a nondominated sorting genetic algorithm, suggested by Goldberg, is described and used to solve three multiobjective optimization problems. The proof-of-principle simulation results have shown that this algorithm (called NSGA) can maintain stable and uniform reproductive potential across nondominated individuals, which is a serious drawback of VEGA. The results suggest that NSGA can be successfully used to find multiple Pareto- optimal solutions, the knowledge of which could be very useful to designers or decision-makers.

The researchers Franklin Johnson, Alvaro Valderrama et al [8], propose a new genetic algorithm for the optimization of the CNN architecture for a given image classification problem. This algorithm extends and refines existing research in the field, by allowing depth exploration, introducing a novel sequential crossover operator, using an incremental selective pressure schedule over evolution (favoring higher diversity in early generations), and evaluating individual performances over the validation set with early stopping. The technique is validated in three image classification datasets, namely, CIFAR10, MNIST, and Caltech256 datasets. This paper was published in August 2020. Authors have implemented different approaches for the crossover operation and the environmental selection over the generations as well, to allow for greater diversity in the earlier generations and intensify the search towards the latter stages.

Chapter 3

Proposed system

* 1. Description of the proposed system with a block diagram

Search for the architecture for a given dataset using Evolutionary algorithm

Using models (one for the architecture and another for the weights)-Neural

Architecture Search

Perform a search optimization for obtaining the non-dominating solutions

On the pareto-front, (with respect to accuracy and additional objectives)-

Multiobjective optimization

Fine tune the obtained model and re-train it to optimize it for a specific dataset and a specific device using device look-up table parameters

Figure 3.1: Workflow of system

Figure 3.1: Depict the workflow of the system including the 3 major steps. First, search for the optimal architecture for the given dataset by using an evolutionary algorithm using models for the architecture and weights (neural architecture search). In the second step, it will perform the evolution search optimization for obtaining the non –dominating solutions on the Pareto-front concerning accuracy and latency, flops for multi-objective optimization. In the third step fine-tune the obtained model and re-train it to optimize it for a specific device using the device look-up table parameters.

* 1. Advantages of the proposed system
* Neural Architecture Search-based Binary Neural Network automates the process of generation of the best architecture, which is computationally less expensive [15], is not prone to errors, and has higher accuracy.
  1. Scope and Boundary of the proposed system
* Neural Architecture Search is a naive field community support is limited hence; it is challenging to design it as there is a lot of progress to be made.
* Binarization of Neural Networks may lead to loss of information, though this can be minimized [11] [18] [18] [5].
* The training time required to search the cells is an overhead in addition to the time required to train the network [14].
* Although NAS can be used in any application, the scope of the project is to design a research-based prototype for image classification [6].

Chapter 4

Software requirement specification

* 1. Overview of SRS

Software Requirement Specification has two major components:

* + - Functional Requirements: Requirements that our optimization model for BNN should have.
    - Non-Functional Requirements: System, performance, and reliability requirements that our optimization model for BNN should have.
  1. Requirement Specification

4.2.1Functional requirements

* + - The system shall be able to identify the state-of-the-art architecture for the image classification task.
    - The system shall be able to quantize the floating-point weights to binary.
    - The Binary neural architecture is generated by the Binary architecture search.
    - The system shall train the model optimizing it for just 1-bit operations.
    - Able to get the architecture out of a set of architectures in an automated manner without manually engineering the architecture.
    - The model shall be able to accurately classify images given as input.

4.2.2 Non -functional requirements

* + - The model shall be able to accurately classify images given as input.
    - The generated model's accuracy should be higher than that of the existing state-of-the-art models (it is around 60 for cifar10).
    - The model should be able to classify the objects with latency lesser than the existing state-of-the-art models.
    - The model should be reliable enough to work in real-time classify real-life images using the devices
  1. Software and Hardware Requirements
     + PyTorch
     + Nvidia-CUDA (for GPU parallelization).
     + CuDNN (Cuda Library for Deep Neural Networks).
     + Intel Xeon Processor - 3.2 GHz.
     + 32 GB Main Memory.
     + Nvidia Quadro GPU - 8GB.

Chapter 5

System Design

5.1 Architecture of the system

Apply NAS on Search Space

Save Optimized Architecture

Binarized the model

Multi-Objective Optimization of The

Model

Compare our model with other state of art

Models

Figure 5.1: High-level design

Figure 5.1 depicts the high-level view of the entire process in which neural architecture search is applied on a predefined search space and the optimal architecture is generated for the requirements, the obtained model is binarized (weights are binarized) and optimized for multi-objective optimization to optimize flops, latency or power usage, after the final results are obtained the model is compared with the existing state of the art models.

Building Architecture –Evolutionary

Algorithms

Choose efficient models

Binarized the weights and train the model

Fine-tune the model for latency, power

Usage; or particular device

Port the model onto the embedded device

And test for the objectives

Figure 5.2: Detailed design

Figure 5.2: depicts the internal process of neural architecture search and multi-objective optimization By using the genetic algorithm taking sequential model as the base model or parent model, further the weights are binarized and the model obtained is trained for accuracy further the model is fine-tuned for secondary parameters such as latency, power usage or flops to get the desired output. The obtained model is port on the embedded device and tested for its performance on that device for image classification and compared with other state-of-the-art models.

Chapter 6

* 1. Proposed methodology

**Implementation**

First, we use Evolutionary Algorithm NSGANET [2] to search for the appropriate architecture which is optimized for multiple objectives and a particular dataset (FLOPS plus secondary objectives next, we fine-tune the model architecture for a particular value of accuracy and the secondary objective as per requirements. This stage also involves optimizing the model for a particular device according to the device parameters. Next, we train the final model for a given dataset to get the trained weights [2]. We deliver this model with Searched Architecture and trained weights.

**6.2 Dataset description: Cifar10**

The CIFAR-10 dataset [13] is a collection of images that are commonly used to train Machine learning and Computer Vision algorithms. It has 60000 32\*32 color images in 10 different classes. 6000 images of each class. i.e. Airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Images in CIFAR-10 are low-resolution (32x32), this dataset can allow researchers to quickly try different algorithms to see what works. Images are taken in varying lighting conditions and at different angles, and since these are colored images, there are many variations in the color itself of similar objects. This leads to good accuracy.

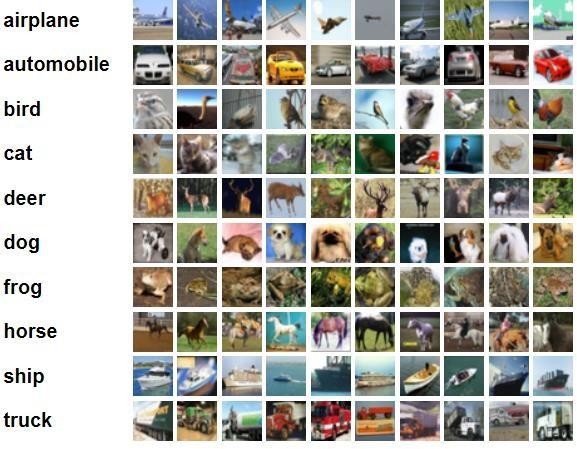


Figure 6.1: The Cifar10 dataset

6.3 Description of modules

**Multi-objective Neural Architecture Search Module**: This module searches the architecture for multiple objectives (accuracy plus secondary objective) on the Pareto front using NSGA NET evolutionary algorithm [2].

**Search Module:** This is the module that fine-tunes the architecture for given values of accuracy, latency, etc.

**Training Module:** This module trains the searched architecture to get trained weights for the specified dataset. Here, we select the weights from a weight matrix, avoiding the traditional Gradient Descent Methods, which are time-consuming and resource inefficient.

Chapter 7

Results and discussions

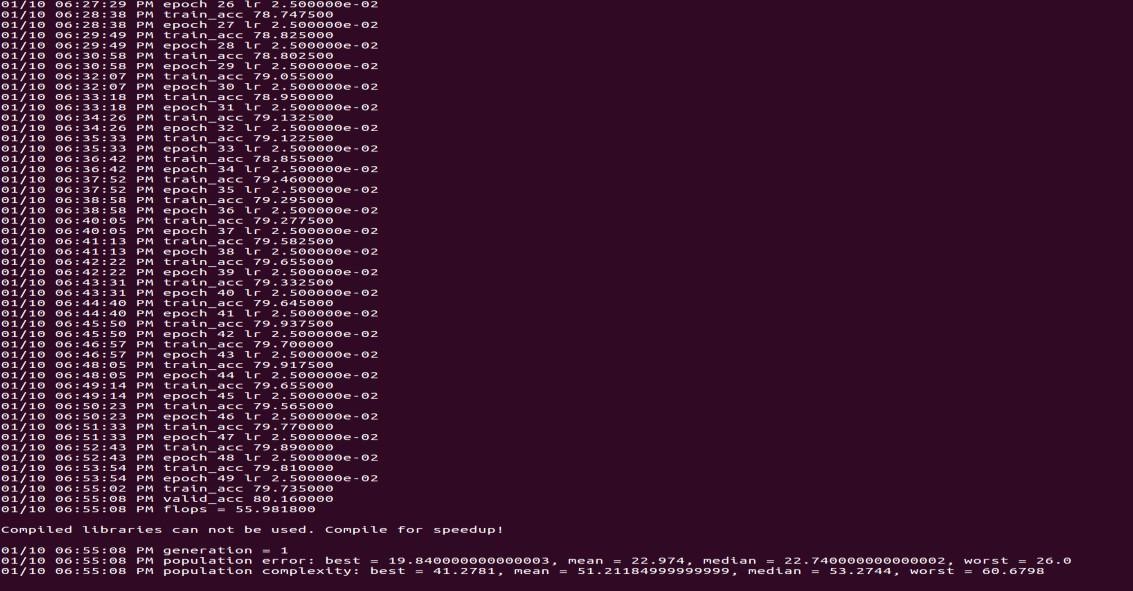
Model search for multiple objectives has been completed. It used genetic algorithm-based NSGANET.

Figure 7.1: Macro search space

Figure 7.1: depicts the macro search space for accuracy with flops; the program was run for 50 epochs with 2 generations. And NSGANET is the evolutionary algorithm used the time taken to search the architecture is 24 hours.

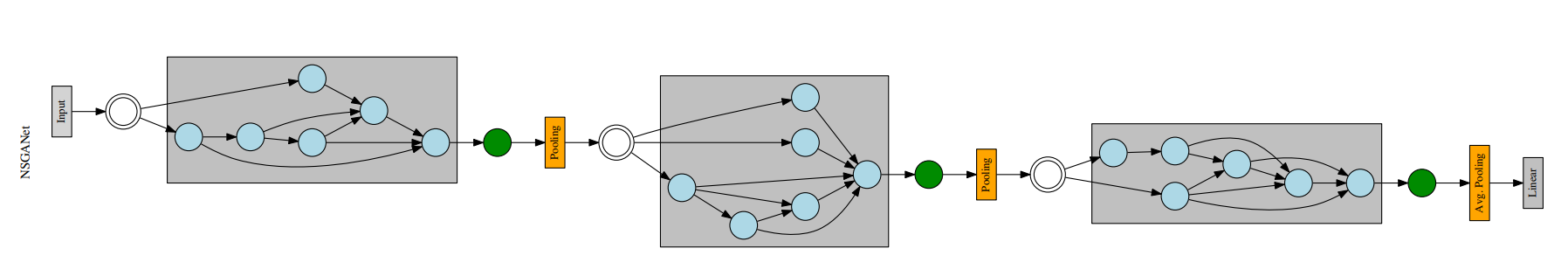


Figure 7.2: Macro visualization NSGANet.

Figure 7.2: Depicts the networks architectures on the trade-off frontier discovered by NSGA-Net.

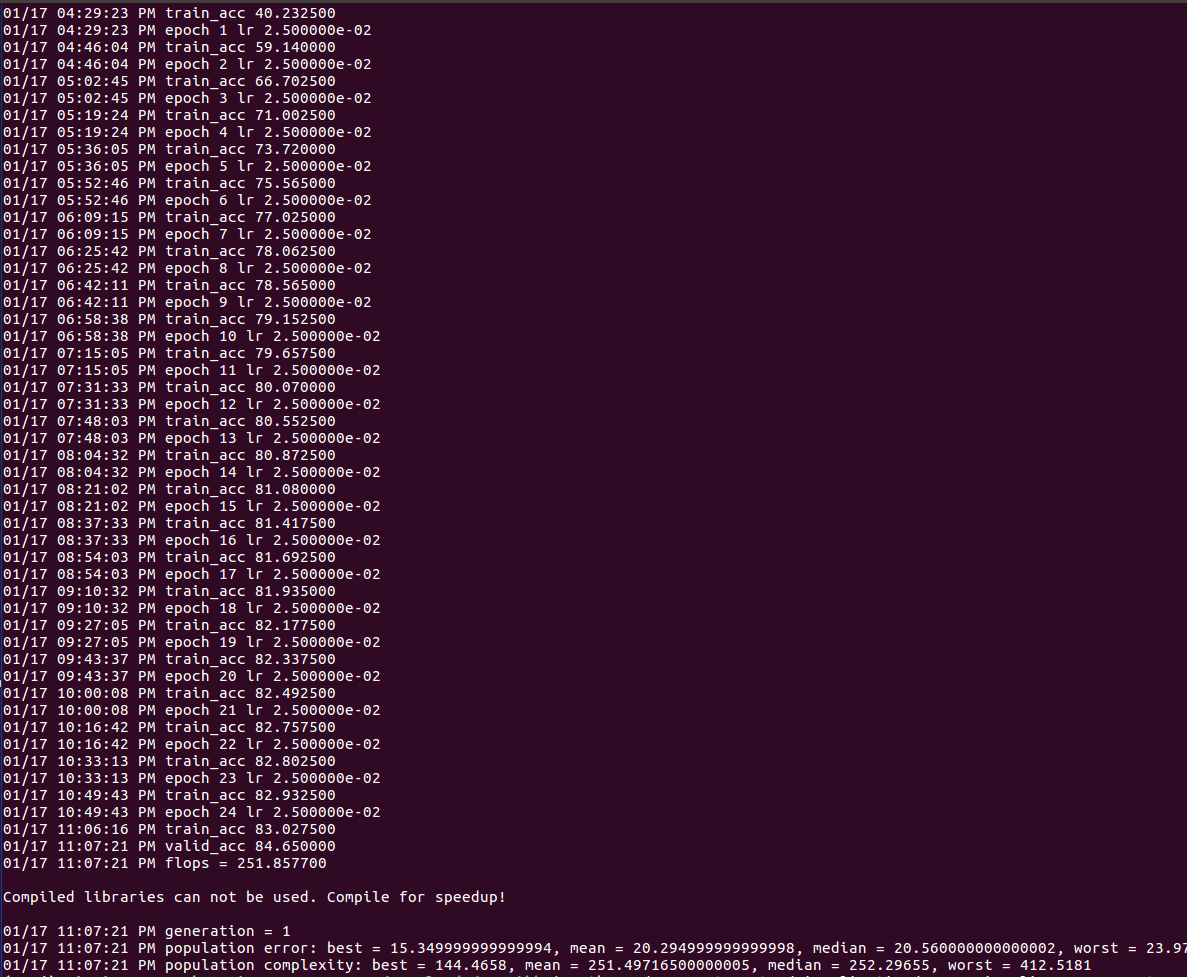


Figure 7.3: Micro search space

Figure 7.3: depicts the micro search space for accuracy with flops; the program was run for 25 epochs with 1 generation. And NSGANET is the evolutionary algorithm used the time taken to search the architecture is 24 hours.

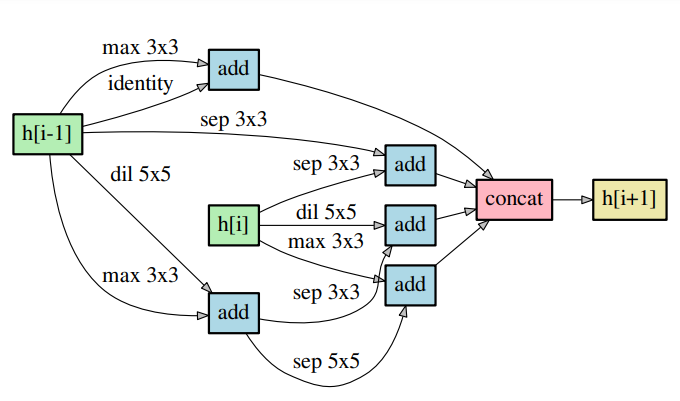


Figure 7.4: Micro visualization NSGA-Net.

Figure 7.4: Depicts the convolutional cell architectures found by NSGA-Net applied to NASNet micro search space

Chapter 8

Conclusion and Future Scope

Our model can be used to build automated model architecture generation for the cifar10 dataset. The specific accuracy and secondary objective value as specified. Moreover, the model weights are binarized to make the model light. And the structure of our approach can be followed to build automated architecture generation for multiple objectives. In the future scope, we can perform with some other datasets and compare our model with other state-of-art approaches.

REFERENCES

[1]Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten,and Kilian Q Weinberger. Multi-scale dense networks for resource-efficient image classification. *arXiv preprint arXiv:1703.09844*, 2017.

[2]Dahyun Kim, Kunal Pratap Singh, and Jonghyun Choi. Learning architectures for binary networks. In *ECCV*, 2020.

[3]Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. *arXiv preprint arXiv:1711.11294*, 2017.

[4]Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei- Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018.

[5]Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[6]Zechun Liu, Wenhan Luo, Baoyuan Wu, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Binarizing deep network towards real-network performance. *International Journal of Computer Vision*, 128(1):202–219, 2020

[7]Zhichao Lu, Kalyanmoy Deb, Erik Goodman, Wolfgang Banzhaf, and Vishnu Naresh Boddeti. NSGANetV2: Evolutionary multi-objective surrogate-assisted neural architecture search. In *European Conference on Computer Vision (ECCV)*, 2020.

[13]Zhichao Lu, Ian Whalen, Vishnu Boddeti, Yashesh Dhebar, Kalyanmoy Deb, Erik Good- man, and Wolfgang Banzhaf. Nsga-net: a multi-objective genetic algorithm for neural architecture search. 2018.

[8]Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. *arXiv preprint arXiv:2003.11535*, 2020.

[9]Haotong Qin, Ruihao Gong, Xianglong Liu, Xiao Bai, Jingkuan Song, and Nicu Sebe.

Binary neural networks: A survey. *Pattern Recognition*, 105:107281, 2020.

[10]Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

[11]Lingxi Xie and Alan Yuille. Genetic CNN. In*2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1388–1397, 2017.

[12]Zhe Xu and Ray CC Cheung. Accurate and compact convolutional neural networks with trained binarization. *arXiv preprint arXiv:1909.11366*, 2019.

[13]Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa- net: Training low bandwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[14]Baozhou Zhu, Zaid Al-Ars, and H Peter Hofstee. Nasb: Neural architecture search for binary convolutional neural networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[15]Xie, Lingxi, and Alan Yuille. "Genetic cnn." In *Proceedings of the IEEE international conference on computer vision*, pp. 1379-1388. 2017.

[16] Kaoutar, Sinai, and Ettaouil Mohamed. "Multi-criteria optimization of neural networks using multi-objective genetic algorithm." In *2017 Intelligent Systems and Computer Vision (ISCV)*, pp. 1-4. IEEE, 2017.

[17] Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and T. A. M. T. Meyarivan. "A fast and elitist multiobjective genetic algorithm: NSGA-II." *IEEE transactions on evolutionary computation* 6, no. 2 (2002): 182-197.

[18] Srinivas, Nidamarthi, and Kalyanmoy Deb. "Multiobjective optimization using nondominated sorting in genetic algorithms." *Evolutionary computation* 2, no. 3 (1994): 221-248.

[19] Johnson, Franklin, Alvaro Valderrama, Carlos Valle, Broderick Crawford, Ricardo Soto, and Ricardo Ñanculef. "Automating configuration of convolutional neural network hyperparameters using genetic algorithm." *IEEE Access* 8 (2020): 156139-156152.