A Study on Image Classification using RecurNets

Prepared by:

Md Rubaut Reshed Chowdhury

ID: 1721088

Department of Computer Science and Engineering

Independent University, Bangladesh

Supervised by:

Amin Ahsan Ali, PhD

Associate Professor

Department of Computer Science and Engineering

Independent University, Bangladesh

Attestation

We are aware of the fact that plagiarism is strictly prohibited and it conflicts with our university's rules and regulations. We guarantee the authenticity of our work. We have used some copyrighted material and models of others in our project which have been properly cited following the international standards proper guidelines.

Author Name:

. . . . . .

Signature:

Evaluation Committee

Supervisor

Name: _____ Signature: _____

Internal Examiner 1

Name: _____ Signature: _____

Internal Examiner 2

Name: _____ Signature: _____

External Examiner

Name: _____ Signature: _____

Acknowledgement

We would like to sincerely thank our esteemed professor Amin Ahsan Ali Sir of Independent University, Bangladesh. His committed leadership, perceptive advice, extraordinary endurance, and kind donation of his precious time were essential to our project's triumph. The depth and calibre of our work were significantly enhanced by our professor's knowledge and guidance, and we appreciate his steadfast support over our academic career.

We also extend our sincere gratitude to the Center for Computational and Data Sciences (CCDS Lab) for their outstanding assistance and unwavering support. Our project's conception and implementation were greatly aided by the cooperative and creative atmosphere that CCDS Lab offered. Moreover, we would like to express sincere gratitude to Mir Sazzat, who guided us throughout the project as a mentor. Furthermore, the support and encouragement have played a significant role in forming our field of study and academia. We are appreciative of the chances and materials provided to us for our study, which improved our educational experience.

Additionally, the workshops CCDS continually host on different topics helped our work immensely. To be specific, "Insider's Guide to Systemic Review and Research Paper Writing – A Hands-on Workshop" hosted by Ashraful Islam Sir, and "Workshop on Hands-on Deep Learning Coding and Code Management" hosted by AKM Mahbubur Sir. The invaluable and prolific efforts given by CCDS to aid the students are of immense contribution to our enhancement of knowledge.

We acknowledge and value all of the efforts made by everyone who assisted us with this project, both directly and indirectly.

Abstract

In an era of abundance of evolution in deep learning models due to the value it provides and the massive increase in computation power due to GPUs and TPUs, the by-product is the massive environmental damage it is doing, the training time, the increasing demand for computation power, therefore, the increasing demand of very powerful devices. More often than not, researchers emphasize more on the accuracy of models rather than efficiency, resulting in carbon emissions of prolific proportions. We came up with recurrent models that save massive amounts of computation units with a bit of sacrifice in the accuracy so that it is much more efficient, can be used across lightweight devices such as IOT, robots or mobile phones and require less time. Recurrence was traditionally not used in image classification until recently.

Contents

Chapter 1

Introduction

Popular consensus is that adding depth to neural network layers can increase and enhance performances on hard problems. A lot of work has been done in image classification to prove this statement. Some of the state-of-the-art models include CoCa, Vit-e and ModelSoups (Basic-L) [1][2][3]. All of the above require parameters near 2000 million parameters. However, computation power and speed are a luxury for a lot of people. Along with the time it requires to train. Not everyone has access to expensive GPU or computational devices.

A neural architecture with 213 million parameters can lead to up to 626,155 carbon emissions [4]. Commensurate to that, a state-of-the-art model with 2000 parameters can lead to up to 10 times more carbon emission. This has furthermore led to a prolific increase in energy consumption [5].

1.1 Problem Statement and Objectives

In this project, we, therefore, explore recurrent neural networks (REcurNets) [6]. RecurNets are organised to learn sequential or time-varying patterns. A recurrent network is a neural network with feedback loops. [7]. RecurNets do not add any new parameters to the neural network. This makes it not so apparent that performance can be enhanced using recurrent iterations. In Figure 1.1, an example of a recurrent neural network is given where h is the recurrent block that maintains and updates the sequence as it is processed. We discuss RecurNets in detail in the model description.

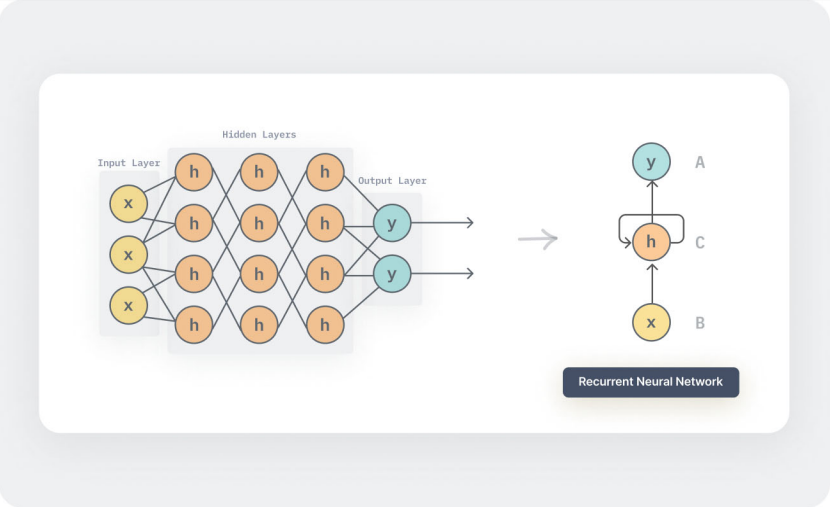

Figure 1.1: An example of a recurrent neural network

We will conduct the experiments with primarily two models, both of which are a combination of convolutional neural networks, recurrent block and a linear layer. The datasets we will run the experiments on are CIFAR-10, SVHN and MNIST. The objectives we aim to achieve are as follows:

- Reduce parameters and ultimately computation units significantly with slightly reduced accuracy for training to be faster, require low specification devices and in less time.
- Test different recurrent neural network models by experimenting with different iterations to classify images of the datasets mentioned.
- Evaluate the performance in terms of train and test accuracy, f-score, precision and recall.
- Compare the performance of our models to state-of-the-art models and the number of parameters.

1.2 Contributions

Throughout our work, the term depth will refer to the number of sequential layers. We will demonstrate that recurrent neural networks can show similar performance without any additional parameters, ultimately reducing computation power. With recurrent neural networks, we notice a substantial reduction in parameters. The drawback to this is that the accuracy is slightly sacrificed as demonstrated in our work. However, if one is willing to sacrifice the accuracy slightly, then the benefit of the massive reduction in computation power and speed can be accessed. This opens up the possibility to train models in lower specification devices such as a cheap PC and perhaps, on phones. We demonstrate two models combined with convolutional neural networks (CNN), fully connected layers and recurrent blocks that give us the answers to the problems stated. We also discuss the environmental impact our project can contribute to. Code to our project can be found at https://github.com/thinkGrow/recurrent-neural-network-for-image-classificatiom/tree/newBranch.

1.3 Outline of the Report

The report is broken down into several chapters. Firstly, we will look at related work where we look at relevant literature work in convolutional neural networks, state-of-the-art models for our datasets, recurrent neural networks and their application in several fields and image classification. In the dataset section, we look at the description of the datasets we use. Apart from that, in the model description section, we look more in-depth at our models and recurrent neural networks. Added to that, we look at the results and analyse them. In social and environmental impact, including how modern neural networks are harming the environment and how our work can reduce the damage to the environment and society in general. We then conclude our work and include future scope for research.

Chapter 2

Related Work

In this chapter, we describe the existing state-of-the-art image classification models along with relevant work in recurrent neural networks. These include VGG, Resnet, ViT-H/14, WRN28-10 (SAM), Branching/Merging CNN + Homogeneous Vector Capsules and some models for our specific dataset and the origin of Recurrent Neural Networks. We further discuss recurrent neural networks in image classification.

2.1 VGG-16

VGG has had great success in image classification. With the increasing power of GPU worldwide, the huge computation that VGG comes with is handled relatively well. So performances are not sacrificed. The VGG-16, VGG with 16 layers consist of approximately 125 million parameters, making it comparatively slower than recurrent. The architecture is fairly simple with 3*3 CNNS, stride=1, same padding and 2*2 same maxpool [8].

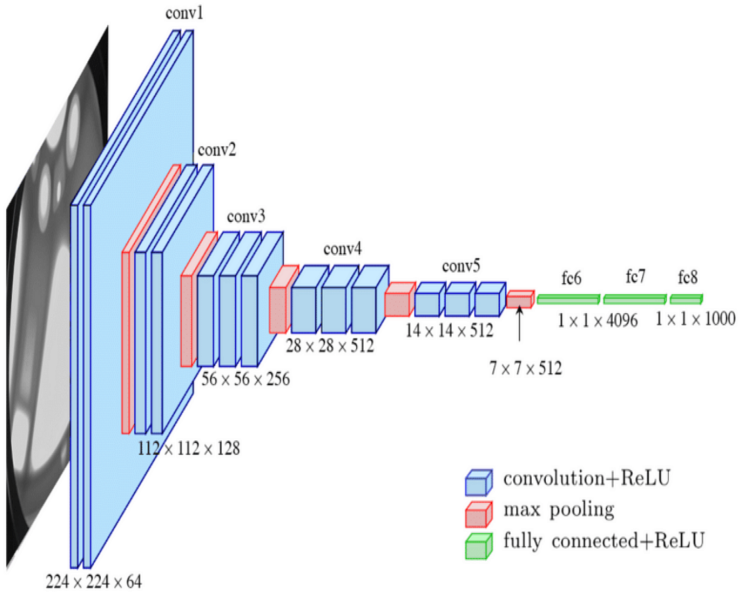

Figure 2.1: A typical VGG-16 architecture

2.2 ResNet

Deep Neural Networks are difficult to train because of vanishing gradients. ResNets help us deal with the problem and help us work with very deep layers. Originally, ResNets were trained with 152 layers which is significantly more than VGGs. [9] This model won the 1st place on the ILSVRC 2015 classification task. This model takes advantage of "shortcut connections". When a gated shortcut is "closed" (approaching zero), the layers in highway networks represent non-residual functions. [10]

2.3 ViT-H/14

The usage of transformers in image recognition is explored here. Vision Transformer (ViT) attains excellent results compared to state-of-the-art convolutional networks while requiring substantially fewer computational resources to train. This is currently one of the state-of-the-art architectures for CIFAR-10 and other datasets. However, it requires 632m parameters. [11]

2.4 WRN28-10 (SAM)

This model aims to not only optimise training loss but also minimise loss sharpness using a technique called Sharpness-Awareness Minimization (SAM). Many neural networks in this day and age can memorize any training data, and readily overfit. Only minimizing training loss is not enough to achieve satisfactory generalization. Sharpness-based measure has the highest correlation with generalization, which motivates penalizing sharpness. For image classification from scratch, this model achieved SOTA performance across several datasets. Particularly CIFAR-10 and SVHN, amongst which, lie our interest [12].

2.5 Branching/Merging CNN + Homogeneous Vector Capsules

To handle the capsule dimensional entanglement that the matrix multiplication creates, the majority of capsule network designs rely on conventional matrix multiplication between capsule layers and computationally costly routing techniques. The dimensions of the capsules are kept unentangled by employing Homogeneous Vector Capsules (HVCs), which multiply elements rather than matrices. In this study, it is demonstrated that a simple convolutional neural network extended with HVC instead of a fully connected layer gives us state-of-the-art performance for MNIST. HVCs are less computationally demanding and simpler. Thus, it requires less power and time. Zero-padding, weight decay and dropout regularization, max-pooling were avoided due to the structure and overall straightforward simplicity of the dataset of MNIST. Furthermore, standard categorical cross-entropy was used over reconstruction loss, cutting off an additional 2.1m parameters [13].

In this study, a network architecture on a basic convolutional neural network and defined design guidelines were built. Next, we proposed a system that employed homogeneous vector capsules instead of flattening to individual scalar neurons, branching out of the sequence of stacked convolutions at different places to represent various degrees of conceptualization with efficient fields of reception.

A lot of efforts were made to reduce parameters and provide efficient results. However, the parameters required were 1.5m, which is still a significant amount considering our recurrent neural network model as seen in Table 5.3.

2.6 Recurrent Neural Network and its inclusion in image classification

Because of the recent mass volume of data and resolution of the images, a new demand was required to be met which meant the need for very high computation units. Recently, very complex models have developed with good performance due to the recent breakthrough in computation power due to the evolution of very powerful GPUs and TPUs [14]. The designs of models in the 1990s had very few layers and were not so deep. Modern ConvNets come with much deeper and wider layers with improved accuracy [15] [16].

Recurrent neural networks have proven to be successful in several fields of applications. Some of them include natural language processing (NLP) [17] [18], machine translation [19], speech recognition [20] [21], weather forecasting [22], human action recognition [23], drug discovery [24], and so on. However, recurrent neural networks used for image classification are still a fairly new concept and have much room for improvement.

Although the performance is not as strong as the state-of-the-art for the datasets we have chosen, the upside is we require much fewer parameters as seen in Table 5.1, 5.3, 5.2 starting from page 27. This means we do not require a computationally powerful device or as much time to train. This is a significant advantage and an economical choice given the drawbacks, that is sacrificing accuracy, does not affect the requirements much.

Recent work regarding RecurNets being utilized in image classification shows that even better performance can be gathered compared to our work. However, the parameters are 12 million, whereas our models are less than 1 million. The same paper also worked on solving the maze problem using recurrence [25]. The paper "Rethinking Recurrent Neural Networks and Other Improvements for Image Classification" [26] also experiments with RecurNets in image classification. In their work, they include RecurNets as an additional layer when designing their models. Furthermore, they progress with end-to-end multimodel ensembles that give predictions using several models. Their work has achieved close performances to state-of-the-art models. However, these experiments were done using a lot of resources, although less than traditional models, much more than what we have worked with. RecurNets is also seen to be used in hyperspectral [27], hierarchical [28] and multi-label [29] image classification. It is evident, however, that a lot can be improved with the help of RecurNets in the field of image classification.

Chapter 3

Datasets

We have chosen 3 datasets to work with and we discussed and elaborated on the datasets in this chapter. This is because the datasets are not too large and easier to train, and are structured and clean data. Furthermore, these are extensively studied have benchmarks available and are easier to compare due its familiarity.

The datasets we have used are all image datasets. They are as follows:

1. CIFAR-10
2. SVHN
3. MNIST

3.1 CIFAR-10

CIFAR-10 consists of a total of 60,000 colourful images. The pixel size is 32x32 with 10 variety of classes. The classes are as follows: aeroplane, automobile, bird, cat, deer, frog, horse, ship and truck). Per class consists of 6000 images. The dataset is divided into 50,000 train images and 10,000 test images. Some instances of the CIFAR-10 dataset are shown in Fig. 3.1

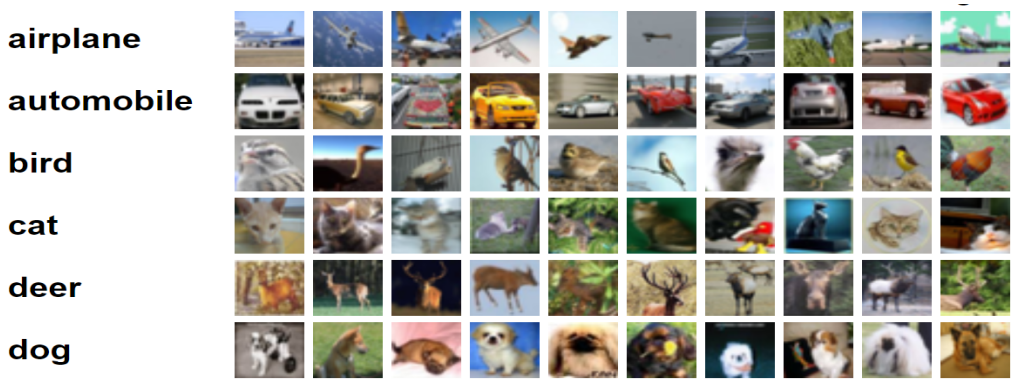

Figure 3.1: Sample of CIFAR-10 dataset

3.2 SVHN

SVHN is an image-based dataset based on real-life images with low requirements for data processing and formatting. It incorporates a high level of labelled data (over 600,000 digit images). Images are taken from Google Street for this particular dataset.

Figure 3.2: Sample of SVHN dataset

3.3 MNIST

The MNIST database (Modified National Institute of Standards and Technology) database is a large database of handwritten digits that is commonly used for training

various image processing systems. The MNIST database contains 60,000 training images and 10,000 testing images.

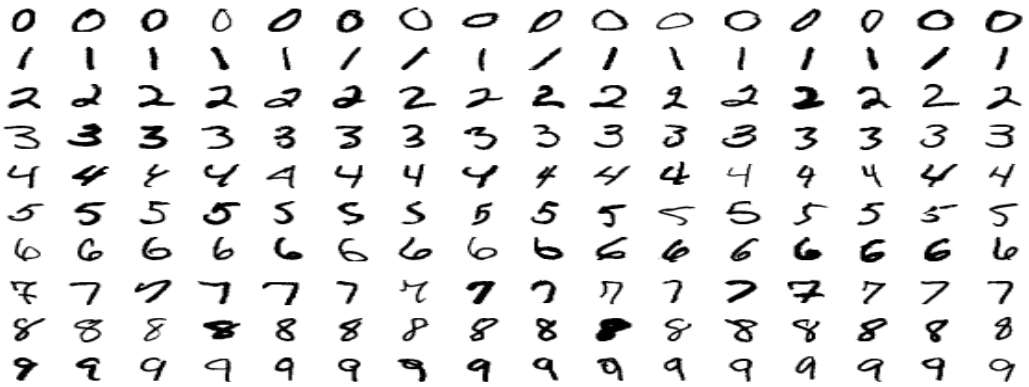

Figure 3.3: Sample of MNIST dataset

Chapter 4

RecurNet Model & Experiment Design

In this chapter, We discuss the various RecurNet models used for our project along with the parameters we used during this experimentation. We look at how we applied recurrent blocks to a traditional convolutional neural network and how the inputs and outputs occur.

4.1 Performance Metrics

The performance metrics used in this experiment are training accuracy, testing accuracy, loss, precision, recall, and f-1 score.

4.2 RecurNet

A recurrent network is a neural network with feedback loops that connect sequential data. In figure 4.1, A is denoted as the RecurNets block which maintains and updates the temporal steps as the sequence is processed while x represents the input and h, the output. A typical RecurNet block is mentioned in equation 4.2.

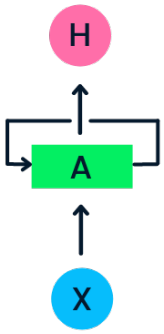

Figure 4.1: A recurrent block

h_t is the hidden state at the time step.
x_t is the input at time step
W_{ih} are the weights and biases of the input-to-hidden layer, respectively.
b_{ih}, W_{hh}, b_{hh} are the weights and biases of the hidden-to-hidden layer, respectively.
W_{hy} and b_{hy} are the weights and biases of the hidden-to-output layer, respectively.
y_t is the output at time step t.

This represents a basic one-step forward pass of a recurrent neural network, where the hidden state h_t at time step t is computed based on the current input x_t and the previous hidden state h_{t-1} and the output y_t is computed based on the hidden state h_t.

$$h_t = fw(x_t, h_{t-1}) \tag{4.1}$$

$$h_t = \tanh(W_{ih}x_t + b_{ih} + W_{hh}h_{t-1} + b_{hh}) \tag{4.2}$$

A function of a feed-forward network is defined in equation 4.3 where x is the input and y is the output. The layers of a neural network are defined as l.

$$f(x) = l_n \cdot l_n - 1 \cdot ..... \cdot l_2 \cdot l_1 \tag{4.3}$$

Each member of the set $\left\{l_i\right\}_{i=1}^{p}$ can be functions of linear or non-linear. The two models we have used are in the form of equation 4.4.

$$f(x) = l_n \cdot ... \cdot l_m + 1 \cdot z_r \cdot z_r \cdot ... \cdot l_m \cdot ... \cdot l_1(x) \tag{4.4}$$

We let $z_r(x) = l_q^z \cdot ... \cdot l_1^{(z)}(x)$ and call it a q-layer recurrent module. This is then put in the beginning and the end of the layeers. 4.3 has an effective depth of n+zq, where z is the quantity of loops of the rnn block. More detailed examples are shown in figure 4.2 and figure 4.3.

4.3 Experiments

We have primarily focused on two main experiments to proceed with our experiment. Both are a combination of convolutional neural networks with ReLu, maxpool, recurrent block and a linear layer. For all the recurrent blocks, we experiment with a variation of depth from 4 to 8. The significant difference is that we used two recurrent blocks in the second experiment and we changed the kernel for one of the blocks. We mainly want to see if adding a recurrent block can replace traditional CNNs Overall, the second experiment has given us better results.

The parameters used were the same across all experiments. The train batch size was set to 128 while the test was set to 50. SGD Optimizer was used. The learning rate was given as 0.01 with a scheduler that changed the learning rate every 50 epochs with a learning rate factor of 0.1. ReLu is used to prevent the shrinking of gradients x is greater than 0.

4.3.1 Experiment 1: One Recurrent Block

In experiment 1, we first input an image and pass it through 3 convolutional blocks. On the 3rd block, we apply a recurrent block with varying depth. We then use a maxpool to reduce the dimensions and add other convolutional and maxpool blocks. In the end, we use a linear layer and get our label.

The kernel is set to 3x3 for all blocks. The stride is set to 1x1 for all convolutional blocks. The padding is set to 1x1 for the convolutional blocks other than the last one.

An overview of the model is shown in 4.2. A 3-D image is given as input. The blue

blocks are convolutional blocks along with ReLu, while the red block is maxpool. The last block is the linear block with the final output. The block with an arrow pointing towards itself is the recurrent block looping back to the block.

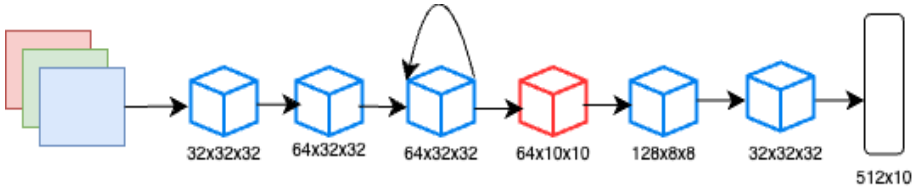

Figure 4.2: Experiment 1: One recurrent block

4.3.2 Experiment 2: Two Recurrent Blocks

In experiment 2, we first input an image and pass it through 3 convolutional blocks. On the 3rd block, we apply a recurrent block with varying depth. We then use another convolutional block and a max pool to reduce the dimensions and add another recurrent block and maxpool block. Afterwards, the last convolutional and maxpool blocks are used. At the end, we use a linear layer and get our label.

The kernel is set to 3x3 for all blocks except the second recurrent block with a kernel of 5x5 and the last convolutional block with a kernel of 2x2. The stride is set to 1x1 for all convolutional blocks. The padding is set to 1x1 for the convolutional blocks before the second recurrent block, which has a padding of 2x2.

An overview of the model is shown in 4.3. A 3-D image is given as input. The blue blocks are convolutional blocks along with ReLu, while the red block is maxpool. The last block is the linear block with the final output. The blocks with an arrow pointing towards themselves are the recurrent blocks looping back to the respective block.

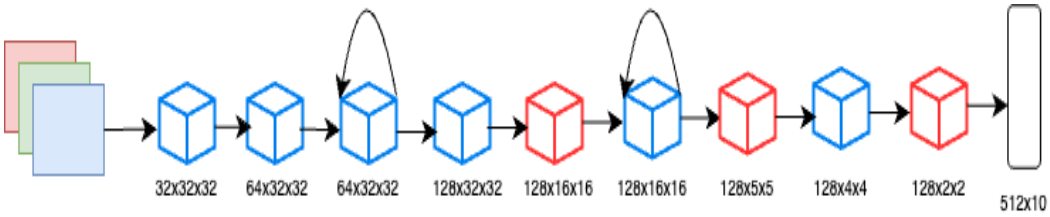

Figure 4.3: Experiment Two : Two recurrent blocks

.

Chapter 5

Results

In this chapter, we discuss the results we achieved from the experiments we covered as described in the model description. Both experiments were done on all three selected datasets: CIFAR-10, SVHN and MNIST. We also discuss various trends we observed among the metrics we have recorded. These are accuracy, f-1 score, precision, recall, depth and number of parameters used. Furthermore, we talk about the runtime for training the model.

If we observe the data for test accuracy as shown in Figure 5.1 and 5.2, for the CIFAR-10 dataset in experiment 1, depth 7 had the best accuracy while the second best was depth 4. For experiment 2, however, depth 8 gave the best performance.

For the SVHN dataset, the differences between each depth are rather marginal, for both experiments. Except for depth-4 in experiment-1 which underperformed, as shown in Figure 5.3 and 5.4.

For the MNIST dataset, the differences between each depth are slimmer than the ones observed in SVHN, for both experiments. And the accuracy achieved is above 98%, mostly being above 99%, which is close to achieving 100%, as shown in 5.5 and 5.6.

The trend amongst the other metrics are similar to the accuracy test as displayed. For charts of all the metrics, refer to the Appendix chapter in page 39.

The runtime for most of the training was done in around 2 hours for Experiment-1. Experiment-2 requires around 3-4 hours. The training was done in Google Colab

which has a limitation of 12 hours of training time in its free version, which is sufficient to train our model since the parameters are very low.

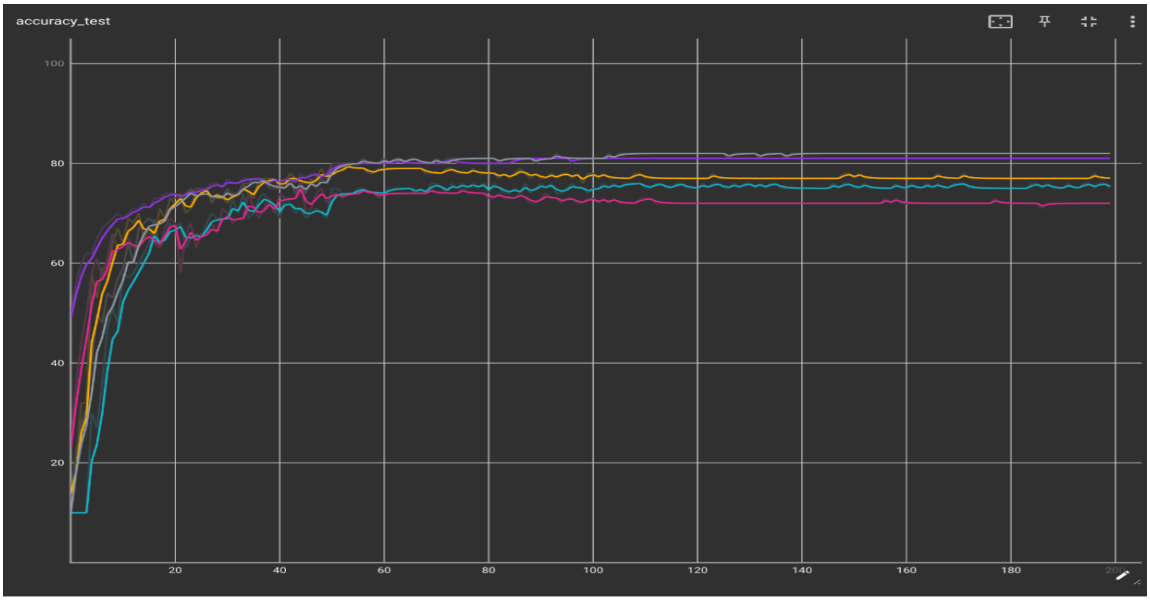

Figure 5.1: CIFAR-10: Test Accuracy (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

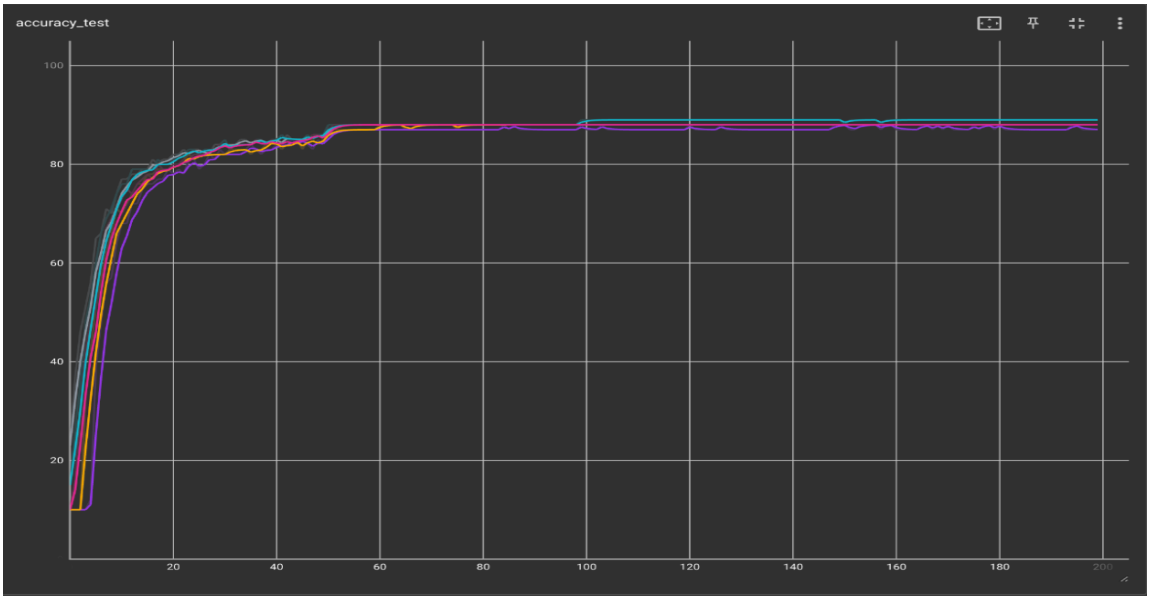

Figure 5.2: CIFAR-10: Test Accuracy (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

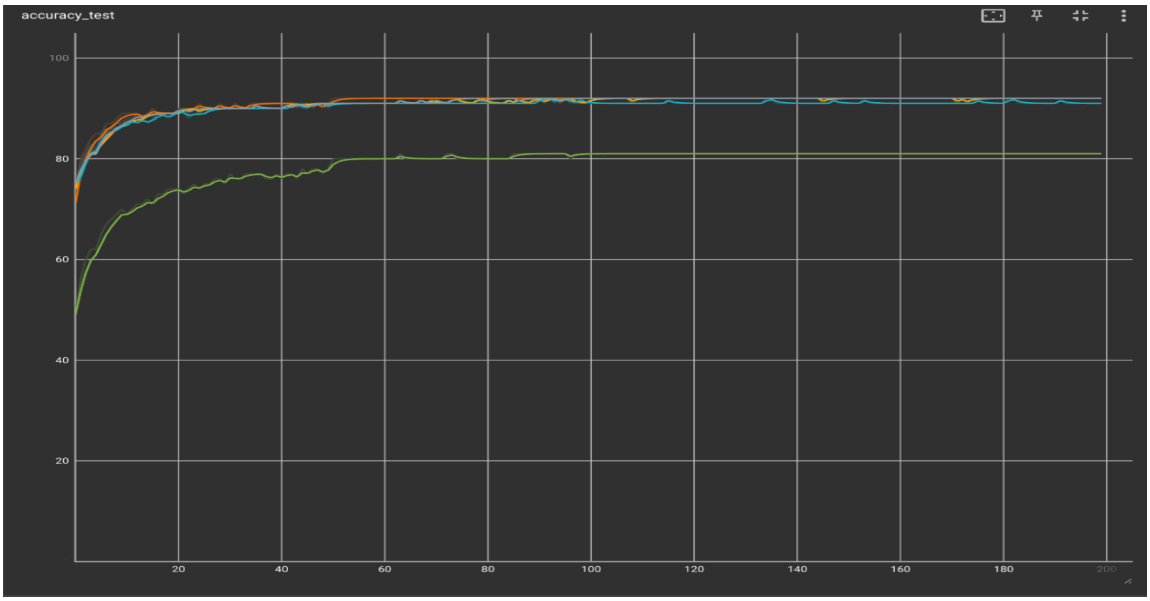

Figure 5.3: SVHN: Test Accuracy (Architecture One)
Legend: Depth-4 Depth-5, Depth-6, Depth-7, Depth-8

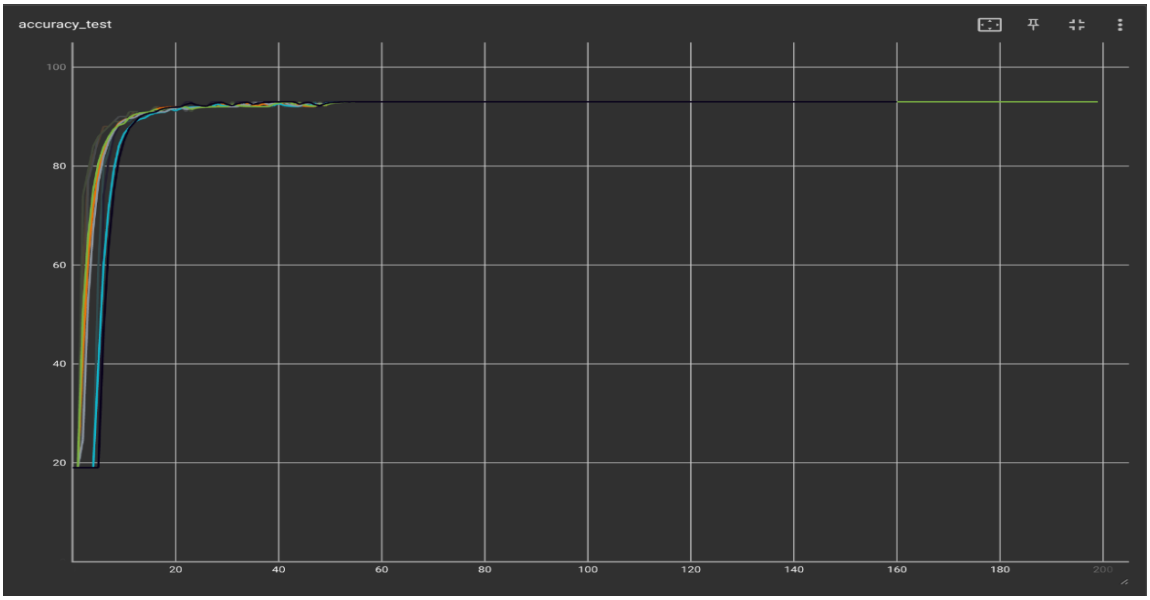

Figure 5.4: SVHN: Test Accuracy (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

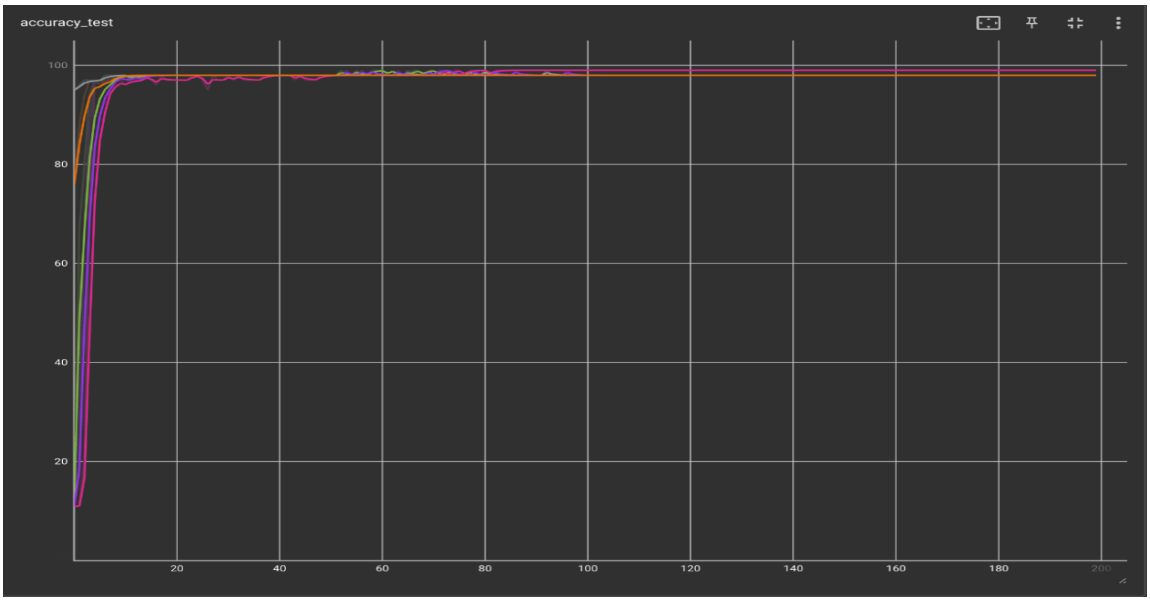

Figure 5.5: MNIST: Test Accuracy (Architecture One)
Legend: Depth-4 Depth-5 Depth-6 Depth-7 Depth-8

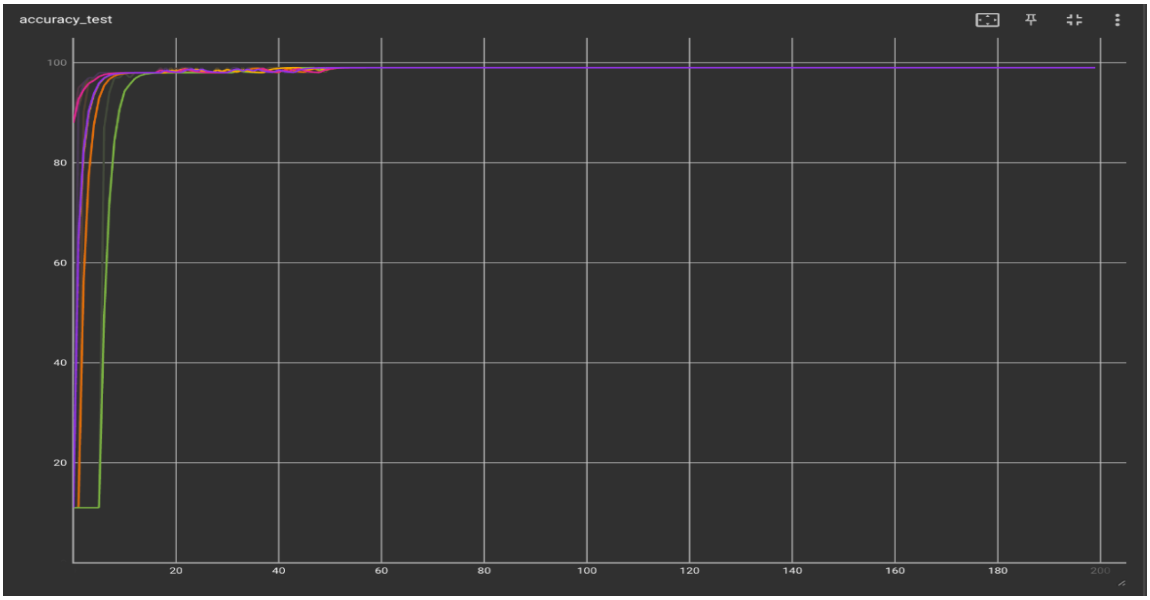

Figure 5.6: MNIST: Test Accuracy (Architecture Two)
Depth-4 Depth-5 Depth-6 Depth-7 Depth-8

All of the metrics we observed are recorded and presented in the form of tables below. From observing all the data, it is apparent from the choice of depth that did not change the result much. It seemed to be rather arbitrary. However, sometimes a random depth can underperform severely, as seen in SVHN for experiment-1 with a depth of 4. As for experiment-1 of CIFAR-10, depths 4 and 7 outperformed the other depths by a significant amount. This means that it is helpful to experiment with different depths to make sure that we are getting the best results. See Table 5.1, 5.2, 5.3 for reference.

Experiment 2 overall outperformed Experiment 1 by a good margin. This to a large extent seems to be due to the addition of another recurrent block. However, this also means that the number of parameters has increased by approximately 5 times. The number of parameters for Experiment 2 was mostly around 0.611m while for Experiment 1, was 0.136m. While this may seem like a consequential boost in parameters, it is nothing compared to the parameters being utilized in the current state-of-the-art for the CIFAR-10 (632m) and MNIST (1.5m).

The results show that adding an extra recurrent block boosted performance. This could potentially open room for further experiments with enhanced results and still

consequently use fewer parameters than what the current state-of-the-art models are using.

Model	Depth	Parameter	Accuracy	F-1 Score	Precision	Recall
Experiment-1	4	0.136m	0.81	0.81	0.81	0.81
	5	0.136m	0.72	0.73	0.79	0.73
	6	0.136m	0.77	0.78	0.81	0.78
	7	0.136m	0.82	0.82	0.83	0.82
	8	0.136m	0.75	0.75	0.80	0.76
Experiment-2	4	0.611m	0.88	**0.89**	**0.89**	**0.89**
	5	0.611m	**0.89**	**0.89**	**0.89**	**0.89**
	6	0.611m	0.88	**0.89**	**0.89**	**0.89**
	7	0.611m	0.82	0.82	0.83	0.82
	8	0.611m	0.87	0.88	0.88	0.88
ViT-H/14		632m	0.95			

Table 5.1: CIFAR-10 depth 4-8 for both experiments

Model	Depth	Parameter	Accuracy	F-1 Score	Precision	Recall
Experiment-1	4	0.136m	0.81	0.81	0.81	0.81
	5	0.136m	0.92	0.92	0.93	0.92
	6	0.136m	0.92	0.92	0.92	0.92
	7	0.136m	0.92	0.92	0.92	0.92
	8	0.136m	0.92	0.92	0.92	0.92
Experiment-2	4	0.611m	**0.93**	**0.94**	**0.94**	**0.94**
	5	0.611m	**0.93**	**0.94**	**0.94**	**0.94**
	6	0.611m	**0.93**	0.93	0.93	0.93
	7	0.611m	**0.93**	0.93	**0.94**	0.93
	8	0.611m	**0.93**	0.93	0.93	0.93
WRN28-10 (SAM)			0.99			

Table 5.2: SVHN depth 4-8 for both experiments

Model	Depth	Parameter	Accuracy	F-1 Score	Precision	Recall
Experiment-1	4	0.135m	0.98	0.99	0.99	0.99
	5	0.135m	0.98	0.99	0.99	0.99
	6	0.135m	0.98	0.99	0.99	0.99
	7	0.135m	0.99	0.99	0.99	0.99
	8	0.135m	0.99	0.99	0.99	0.99
Experiment-2	4	0.611m	0.99	0.99	0.99	0.99
	5	0.611m	0.99	0.99	0.99	0.99
	6	0.611m	0.99	0.99	0.99	0.99
	7	0.611m	0.99	0.99	0.99	0.99
	8	0.611m	0.99	0.99	0.99	0.99
Branching/Merging CNN + Homogeneous Vector Capsules		1.5m	0.99			

Table 5.3: MNIST depth 4-8 for both experiments

Chapter 6

Social and Environmental Impact

In this chapter, we discuss the disadvantages of models with large parameters and their impact on the environment and society. We looked at how this can be reduced using our models.

Artificial intelligence (AI) showed promise in 2016 to overtake human intellect, six decades after it was first introduced at the 1956 Dartmouth Conference [30]. This was demonstrated when AlphaGo [31], a computer program, defeated a human Go player with its highest score for the first time. This was a historic milestone in AI.

Deep neural networks (DNNs), the abundance of large data, and improvements in computer technology have all contributed to the AI revolution. Because of this, artificial intelligence (AI) technologies have advanced a wide range of fields, such as computer vision [32], natural language processing (NLP) [33], healthcare [34], manufacturing [35], and finance technology (FinTech) [36]. As a result, major global economies like China, the US, and the EU have acknowledged AI as a national strategic priority and it is quickly emerging as a vital engine of future productivity.

The bulk of academics focus on achieving new state-of-the-art (SOTA) outcomes in the era of deep learning, particularly with the emergence of Large Language Models. This has led to an increase in both model size and computational complexity. High processing power requirements increase carbon emissions and compromise research equity by excluding small and medium-sized research organizations and businesses with tight budgets. Green computing is a popular area of study as a response to the problems with AI's impact on the environment and computer resources. We urge more scientists to focus on this area and develop environmentally friendly AI.

This tendency is best shown by large language models (LLMs), such as ChatGPT [37], but their use presents serious ethical and environmental [38, 39] issues. Due to the high processing power requirements of these systems, energy consumption and greenhouse gas emissions are increasing [40].

Gebreuetal's 2020 seminal paper on the risks of LLMs already discussed the careful collection of datasets taking in the environmental and the economical aspect [39].

Accuracy is valued more highly by researchers than efficiency. Approximately 75% of the publications report accuracy-related indicators rather than efficiency-related ones. This finding suggests that efficiency-based performance, such as execution time, model size, etc., is disregarded in favour of performance metrics like accuracy, which are the focus of the research community's shared interests [41].

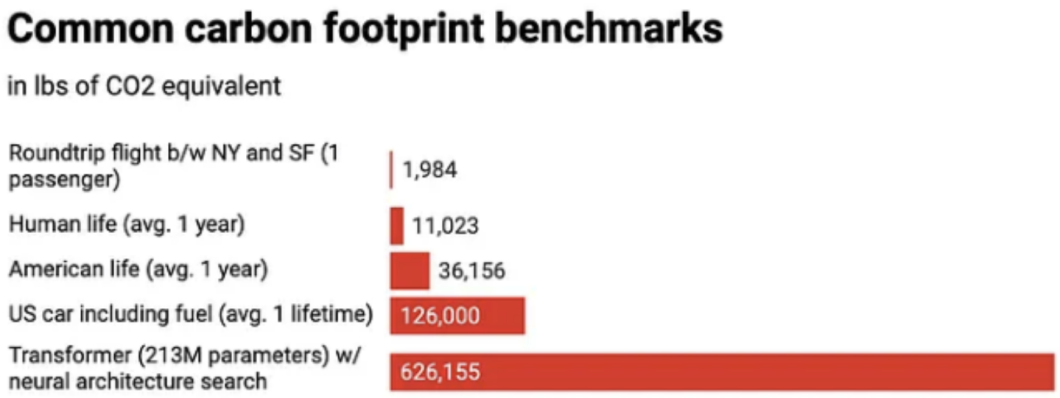

Figure 6.1: Carbon footprint benchmarks. Chart MIT Technology Review and source [4].

"Greenness" is significantly influenced by model size. Therefore, while model size might partially reflect the "greenness" of the algorithm, it is unable to capture the influence of variables like data volume, training iterations, and other facets of the model's training and inference process.

Some classical neural network models of deep learning, like LeNet, VGG, GooogleNet, etc., have shown to be fairly good at image classification, target detection, and other AI tasks. However, these models have several obvious limitations, involve a lot of arguments, and necessitate redundant computation. Certain real-world scenarios using lightweight devices—such as autonomous cars, robotics, identification tasks,

etc.—are challenging to implement and must be completed on schedule.

If we look at one of the state-of-the-art models for CIFAR-10 [11], the parameters are 632m, while our recurrent models do not even exceed a million. Considering the green revolution, our model is exponentially better at reducing extremely significant carbon emissions while sacrificing a bit of accuracy.

Chapter 7

Conclusion

This chapter concludes the study by summarising key findings concerning the problem stated, as well as the value it will provide. It will also review the study's limitations and the scope for future research.

We find that recurrent neural networks can be used as an alternative to traditional convolutional depth with its feedback loop which we can alternatively use as depth. Not only that, this reduces the parameters significantly. However, using recurrent neural networks in our models reduces the accuracy. On a positive note, however, this means we can train our data in significantly less time, with less powerful devices, and more cost-effectively. Due to the heavy reduction in parameters, carbon emissions have decreased exponentially. We also found out that changing depth between 4 to 8 did not significantly affect the results. Adding a second recurrent block, however, improved the performances by a good margin.

With recurrent neural networks being rather uncommon in the field of image classification, we are exploring a relatively new venture that has room for massive improvements in future.

Due to time and money constraints, experimentation was limited. Experimenting with parameters such as optimizers, learning rate, loss function, and a higher number of epochs would have improved the performance. Because the author of this project is relatively new to research and in the field of AI, some key findings were missed, such as rounding off the accuracy values during training and testing. This means we missed out on a small portion of precision due to not having decimal places. F1 scores were averaged for all the 10 classes. In the future, there is room for exper-

imentation in how the model is designed. Experiments can be done in design choices and parameters. Hypertuning the parameters could give more optimal results. Early stop algorithms can be used to automatically stop the training when the learning converges for more efficient use of time and energy.

Overall, the ultimate goal of this project was to create more environment-friendly, less energy-efficient and time-consuming AI models with significantly lower number of parameters but still give relatively competitive accuracy in the field of image classification with the use of recurrent neural networks. Our work shows that it is indeed possible and there is room for massive improvements in the performance in the future with further experimentation.

Bibliography

[1] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

[2] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.

[3] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR, 2022.

[4] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

[5] David A. Patterson, Joseph Gonzalez, Quoc V. Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David R. So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *CoRR*, abs/2104.10350, 2021.

[6] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5(64-67):2, 2001.

[7] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5(64-67):2, 2001.

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[13] Adam Byerly, Tatiana Kalganova, and Ian Dear. No routing needed between capsules. *Neurocomputing*, 463:545–553, 2021.

[14] Nguyen Huu Phong and Bernardete Ribeiro. Rethinking recurrent neural networks and other improvements for image classification. *arXiv preprint arXiv:2007.15161*, 2020.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[17] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016.

[18] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

[19] Sumit Chopra, Michael Auli, and Alexander M Rush. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 conference of the North American chapter of the association for computational*

linguistics: human language technologies, pages 93–98, 2016.

[20] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4774–4778. IEEE, 2018.

[21] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4960–4964. IEEE, 2016.

[22] Xiangyun Qing and Yugang Niu. Hourly day-ahead solar irradiance prediction using weather forecasts by lstm. *Energy*, 148:461–468, 2018.

[23] Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In *Proceedings of the IEEE international conference on computer vision*, pages 2117–2126, 2017.

[24] Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1):120–131, 2018.

[25] Avi Schwarzschild, Arjun Gupta, Amin Ghiasi, Micah Goldblum, and Tom Goldstein. The uncanny similarity of recurrence and depth. *arXiv preprint arXiv:2102.11011*, 2021.

[26] Nguyen Huu Phong and Bernardete Ribeiro. Rethinking recurrent neural networks and other improvements for image classification. *CoRR*, abs/2007.15161, 2020.

[27] Yan Ju, Lingling Li, Licheng Jiao, Zhongle Ren, Biao Hou, and Shuyuan Yang. Modified diversity of class probability estimation co-training for hyperspectral image classification. *CoRR*, abs/1809.01436, 2018.

[28] Jaehoon Koo, Diego Klabjan, and Jean Utke. Combined convolutional and recurrent neural networks for hierarchical classification of images. *CoRR*, abs/1809.09574, 2018.

[29] Junjie Zhang, Qi Wu, Chunhua Shen, Jian Zhang, and Jianfeng Lu. Multi-

label image classification with regional latent semantic dependencies. *CoRR*, abs/1612.01082, 2016.

[30] James Moor. The dartmouth college artificial intelligence conference: The next fifty years. *Ai Magazine*, 27(4):87–87, 2006.

[31] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[32] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48, 2016.

[33] Daniel W Otter, Julian R Medina, and Jugal K Kalita. A survey of the usages of deep learning for natural language processing. *IEEE transactions on neural networks and learning systems*, 32(2):604–624, 2020.

[34] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Katherine Chou, Claire Cui, Greg Corrado, Sebastian Thrun, and Jeff Dean. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29, 2019.

[35] Ruhul Amin Khalil, Nasir Saeed, Mudassir Masood, Yasaman Moradi Fard, Mohamed-Slim Alouini, and Tareq Y Al-Naffouri. Deep learning in the industrial internet of things: Potentials, challenges, and emerging applications. *IEEE Internet of Things Journal*, 8(14):11016–11040, 2021.

[36] Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer. Deep learning for financial applications: A survey. *Applied Soft Computing*, 93:106384, 2020.

[37] Atoosa Kasirzadeh. Chatgpt, large language technologies, and the bumpy road of benefiting humanity. *arXiv preprint arXiv:2304.11163*, 2023.

[38] Le Nguyen Hoang and El Mahdi El Mhamdi. *Le fabuleux chantier: Rendre l'intelligence artificielle robustement bénéfique*. Number BOOK. EDP Sciences, 2019.

[39] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too

big?. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

[40] Alexandra Sasha Luccioni and Alex Hernandez-Garcia. Counting carbon: A survey of factors influencing the emissions of machine learning. *arXiv preprint arXiv:2302.08476*, 2023.

[41] You Zhou, Xiujing Lin, Xiang Zhang, Maolin Wang, Gangwei Jiang, Huakang Lu, Yupeng Wu, Kai Zhang, Zhe Yang, Kehang Wang, Yongduo Sui, Fengwei Jia, Zuoli Tang, Yao Zhao, Hongxuan Zhang, Tiannuo Yang, Weibo Chen, Yunong Mao, Yi Li, De Bao, Yu Li, Hongrui Liao, Ting Liu, Jingwen Liu, Jinchi Guo, Xiangyu Zhao, Ying WEI, Hong Qian, Qi Liu, Xiang Wang, Wai Kin, Chan, Chenliang Li, Yusen Li, Shiyu Yang, Jining Yan, Chao Mou, Shuai Han, Wuxia Jin, Guannan Zhang, and Xiaodong Zeng. On the opportunities of green computing: A survey, 2023.

Appendix

CIFAR-10 - Accuracy Test

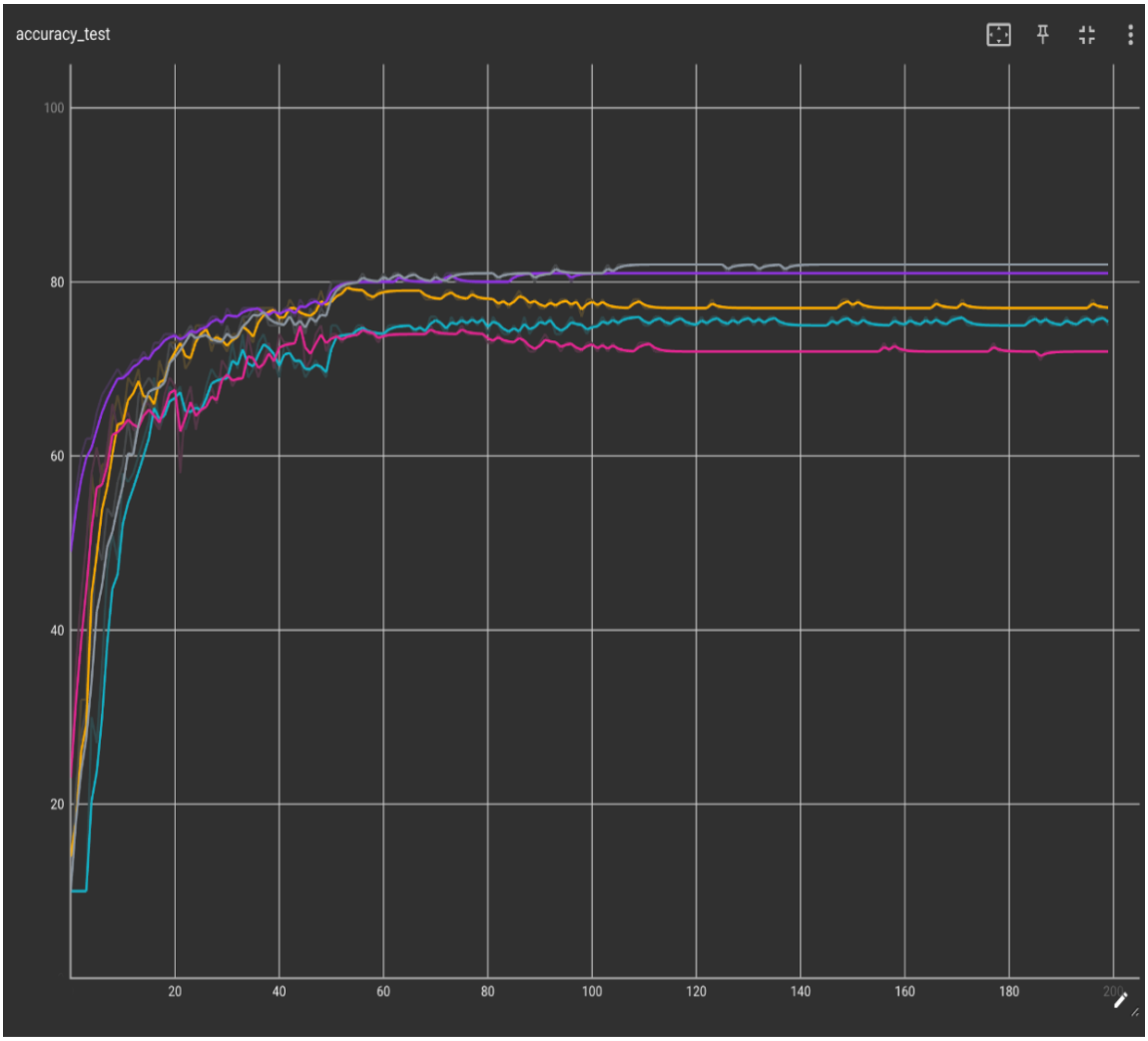

Figure 7.1: CIFAR-10: Test Accuracy (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

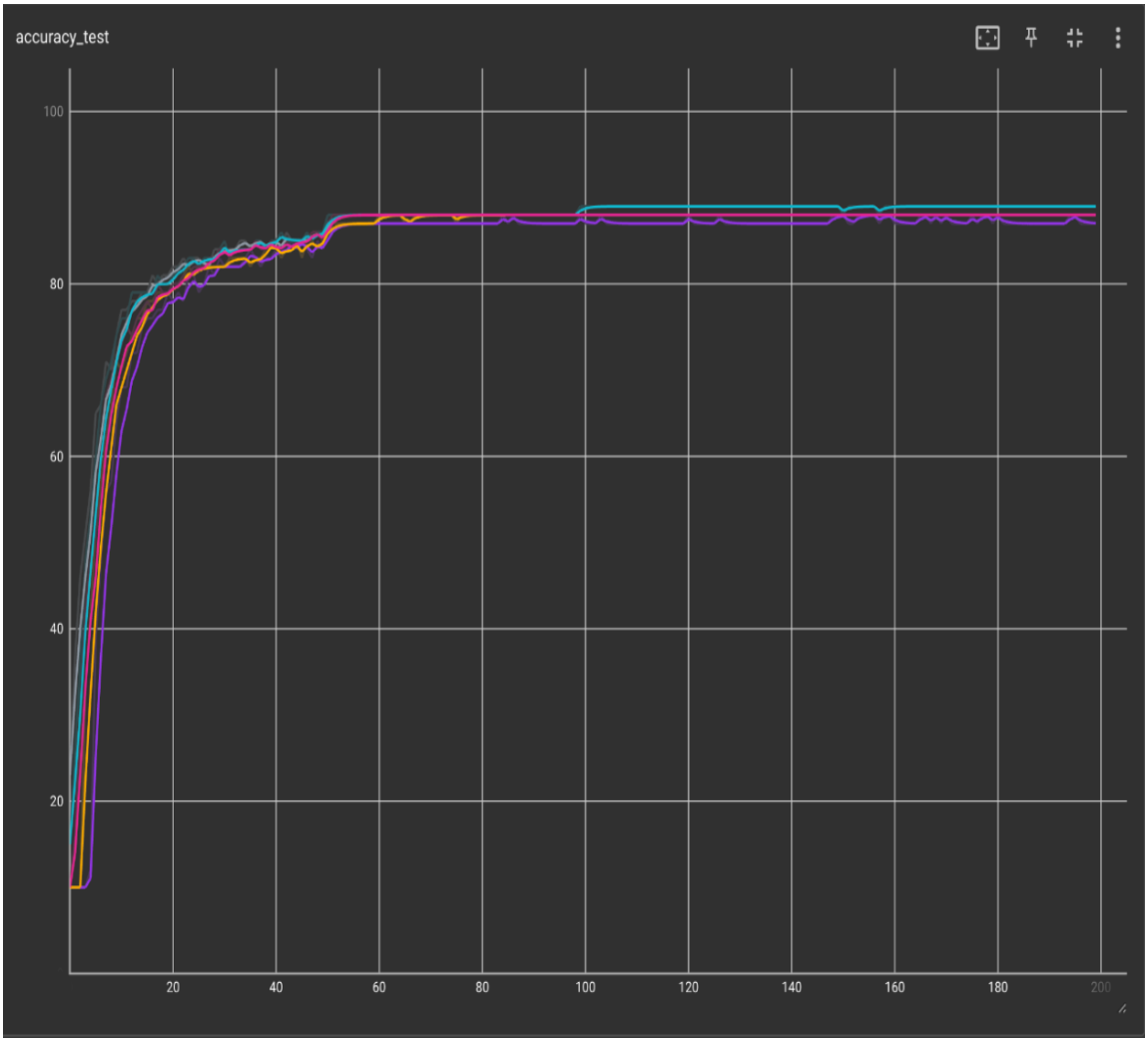

Figure 7.2: CIFAR-10: Test Accuracy (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

CIFAR-10 - Accuracy Train

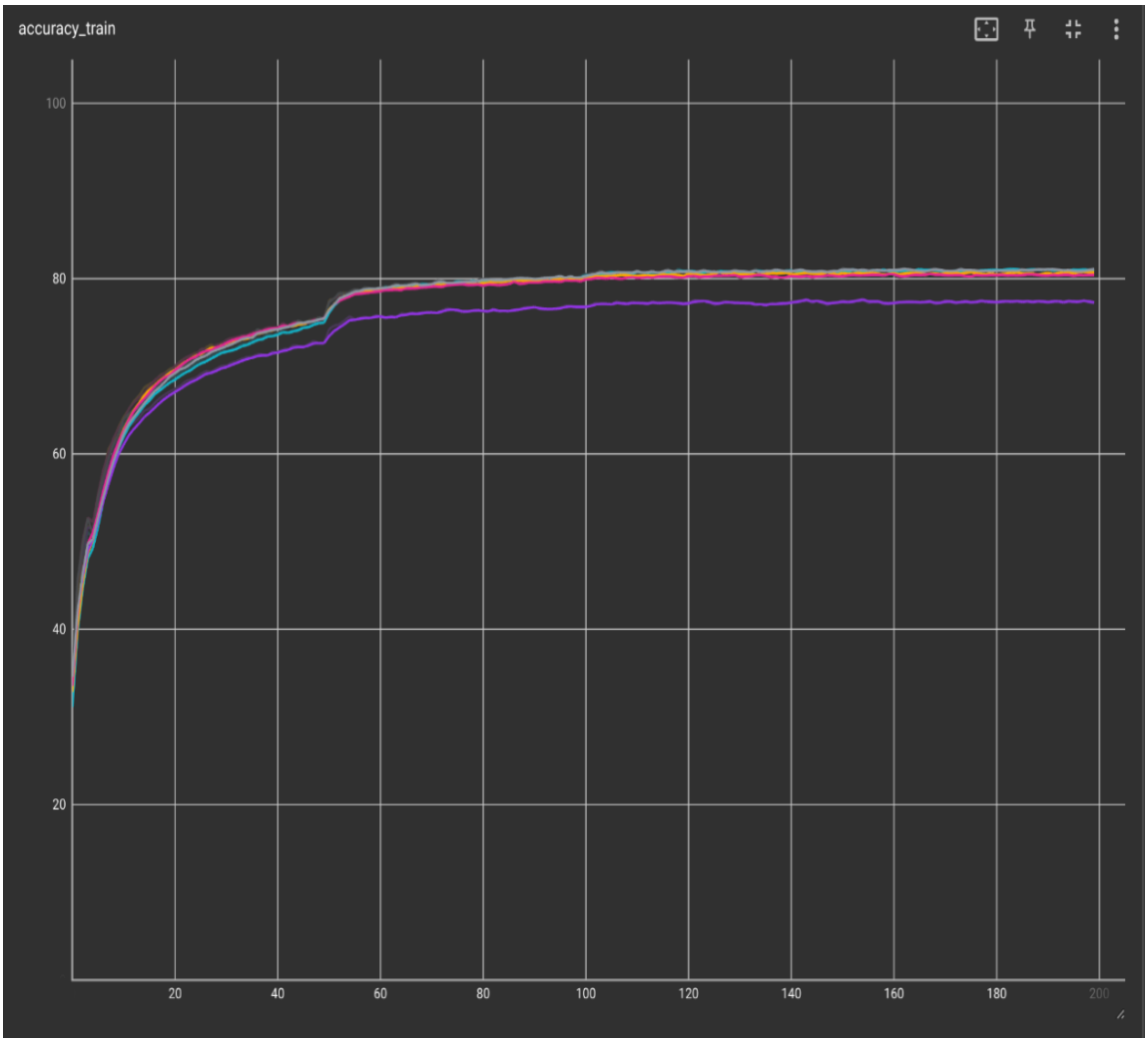

Figure 7.3: CIFAR-10: Train Accuracy (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

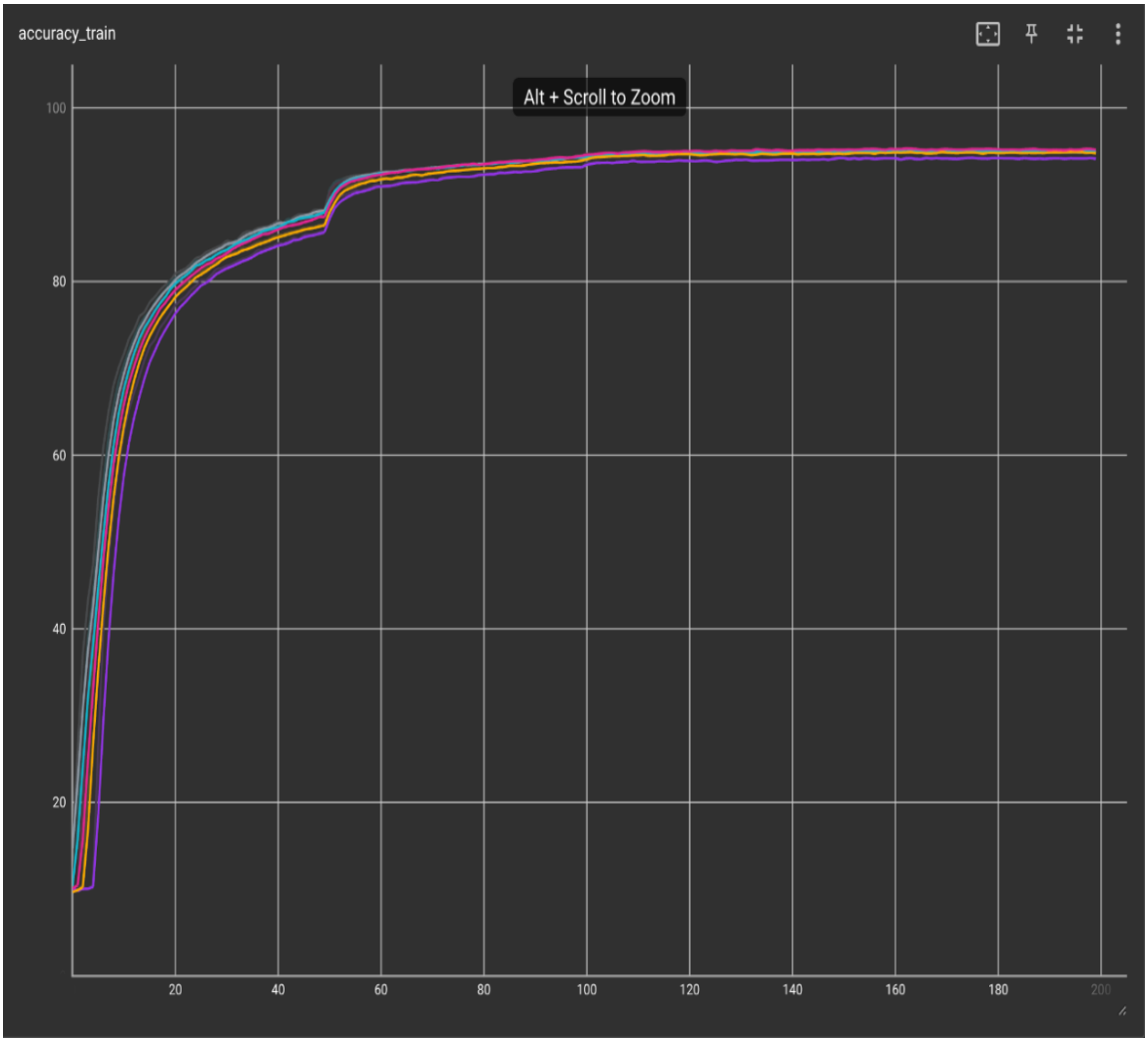

Figure 7.4: CIFAR-10: Train Accuracy (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

CIFAR-10 - Loss

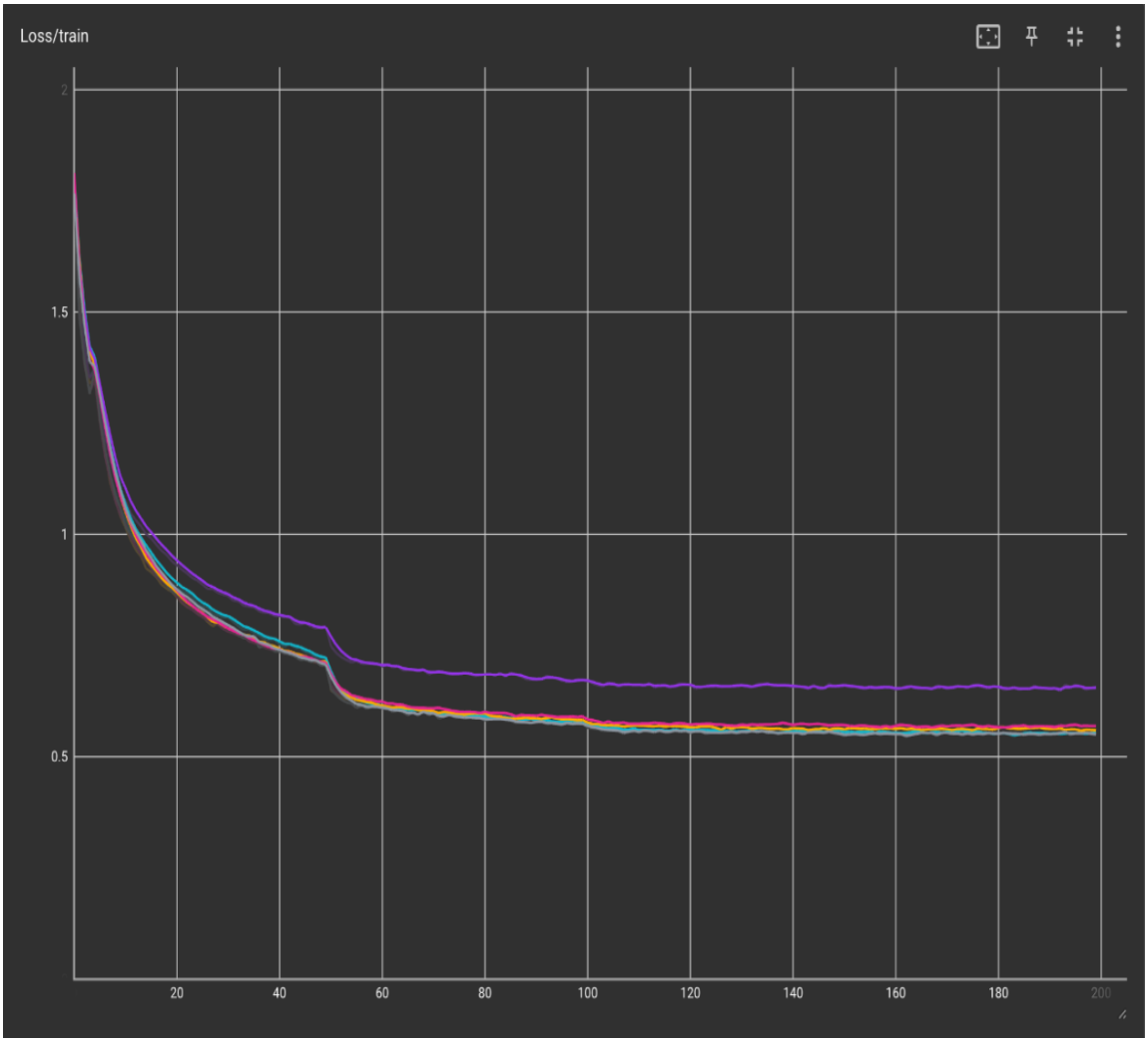

Figure 7.5: CIFAR-10: Loss (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

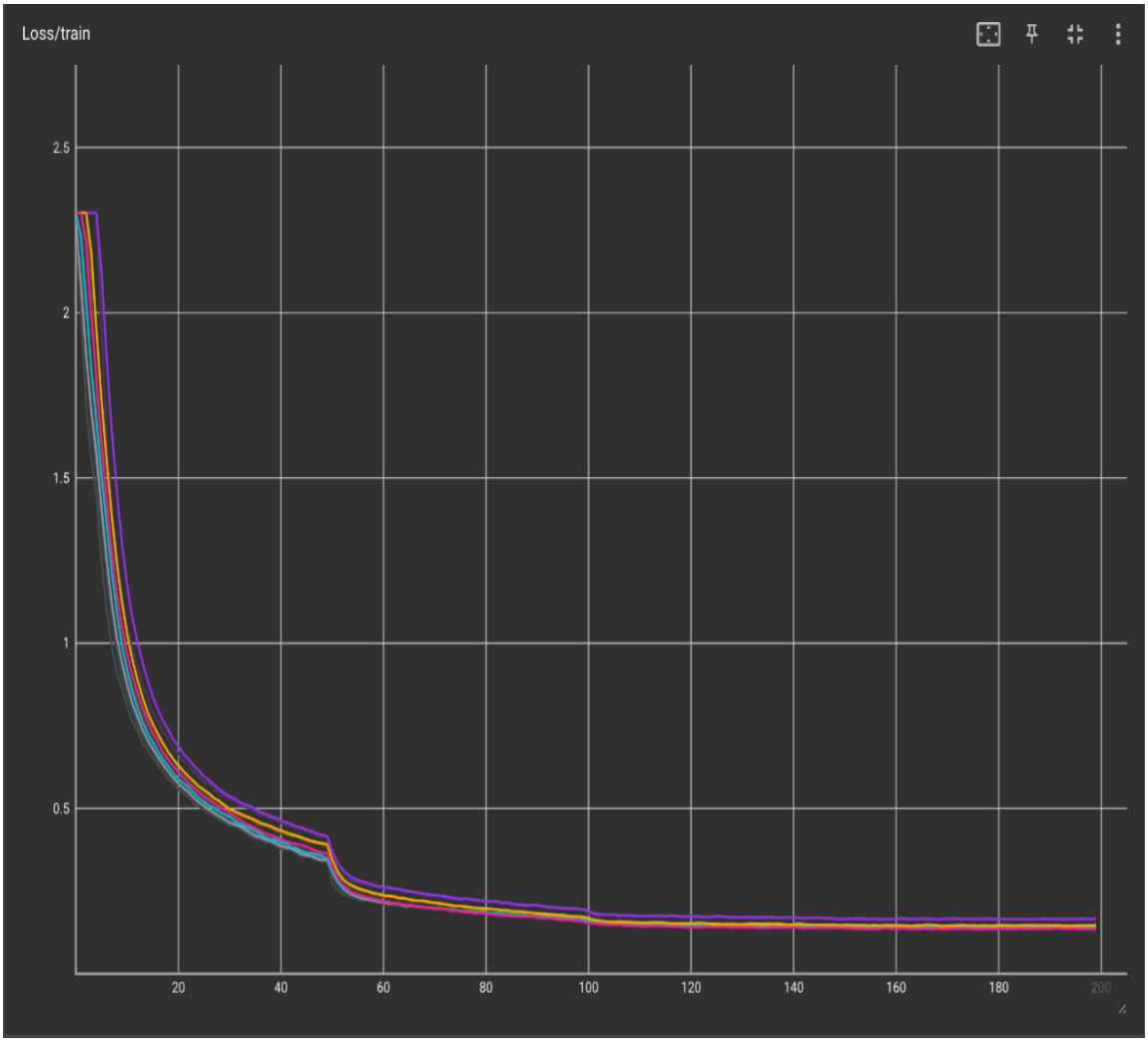

Figure 7.6: CIFAR-10: Loss (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

CIFAR-10 - F1

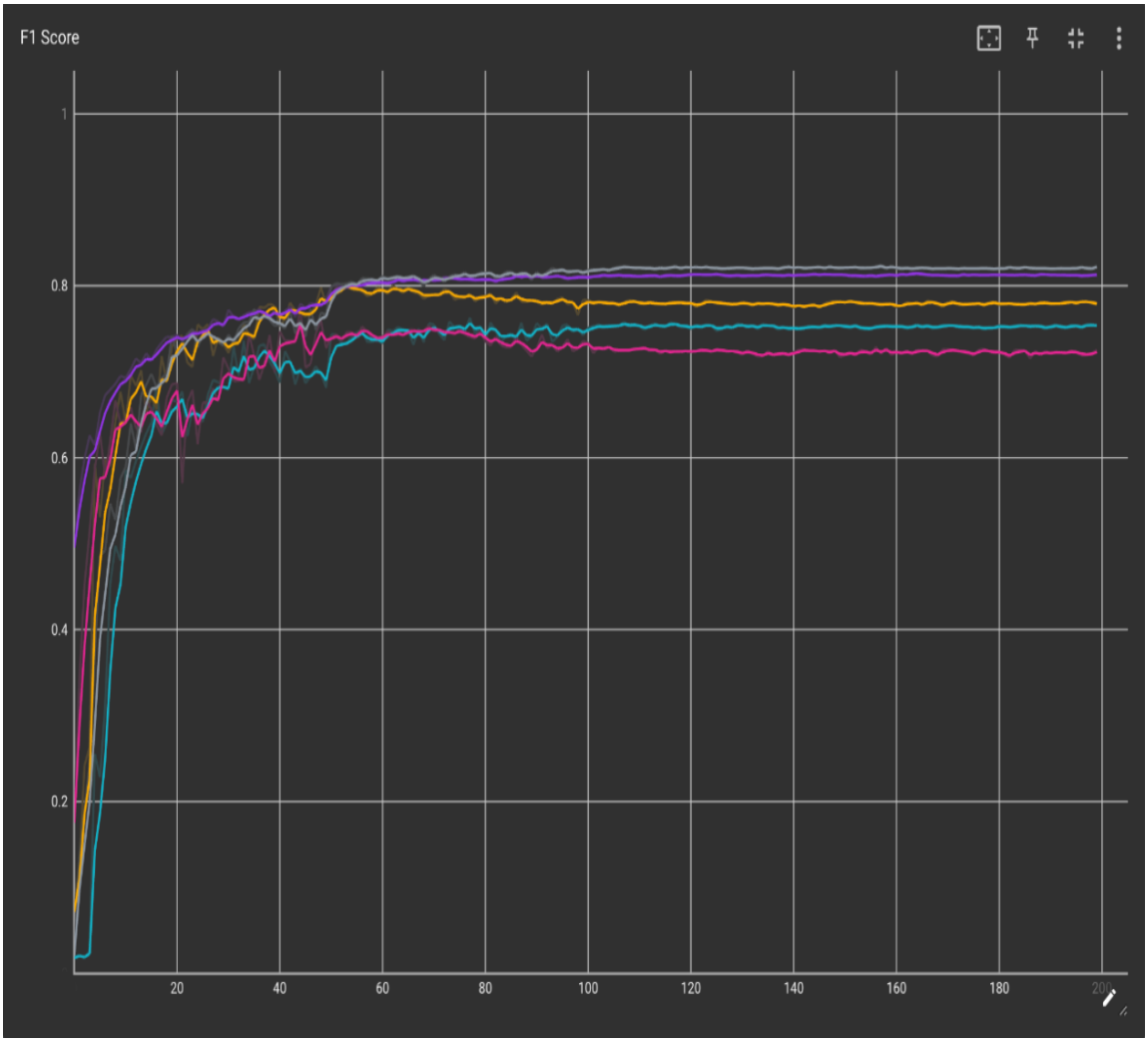

Figure 7.7: CIFAR-10: F1 (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

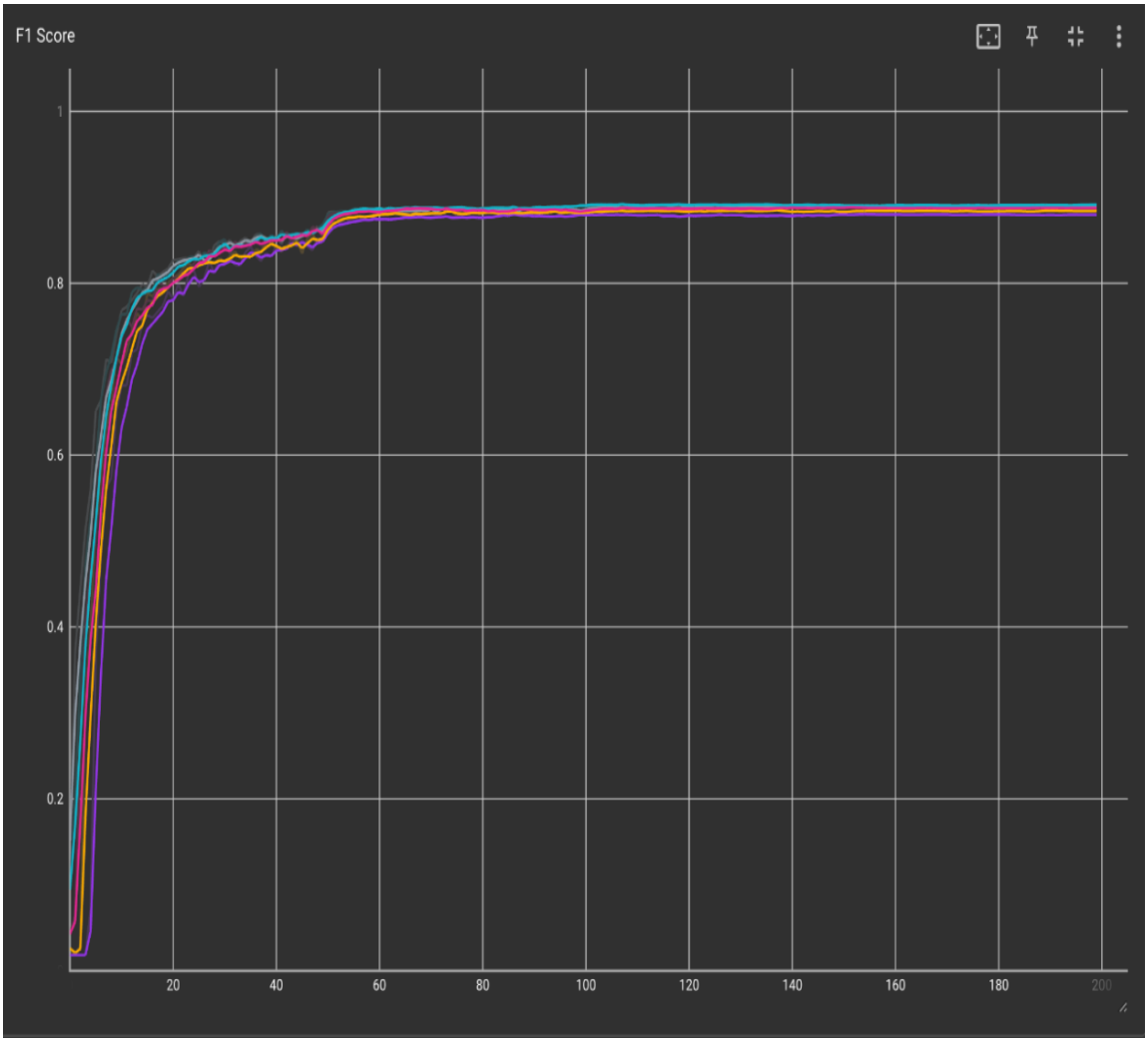

Figure 7.8: CIFAR-10: F1 (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

CIFAR-10 - Precision

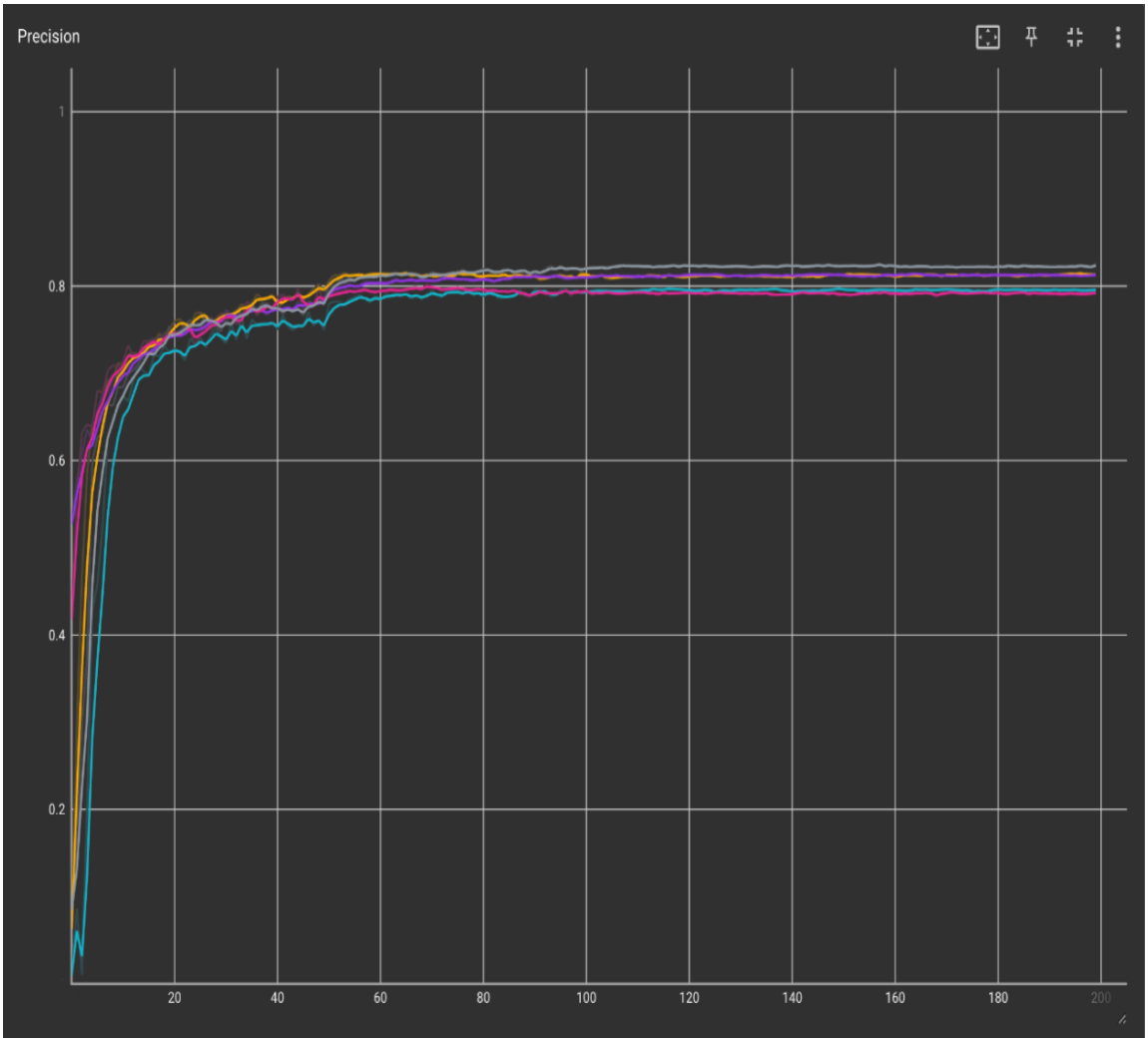

Figure 7.9: CIFAR-10: Precision (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

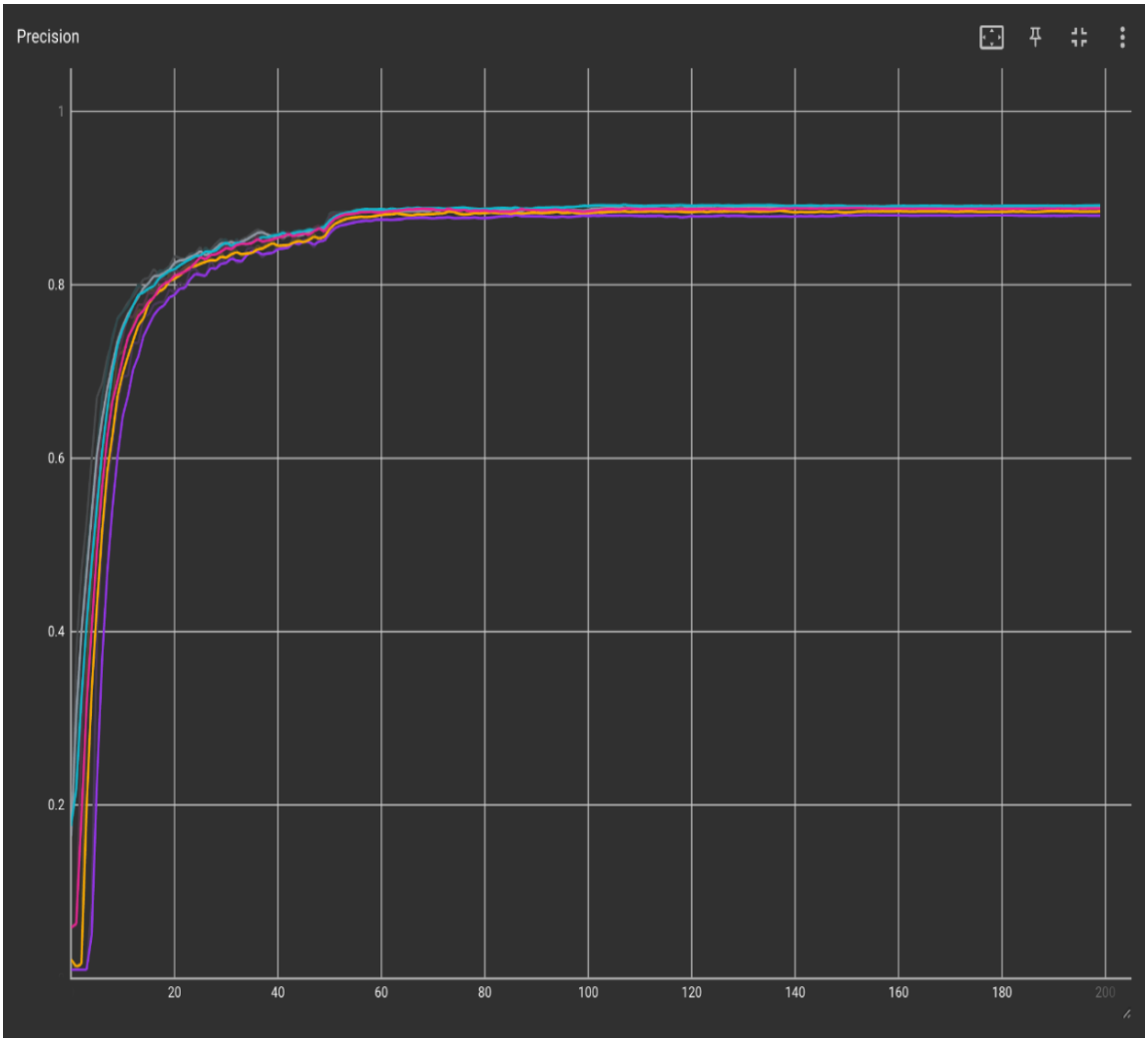

Figure 7.10: CIFAR-10: Precision (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

CIFAR-10 - Recall

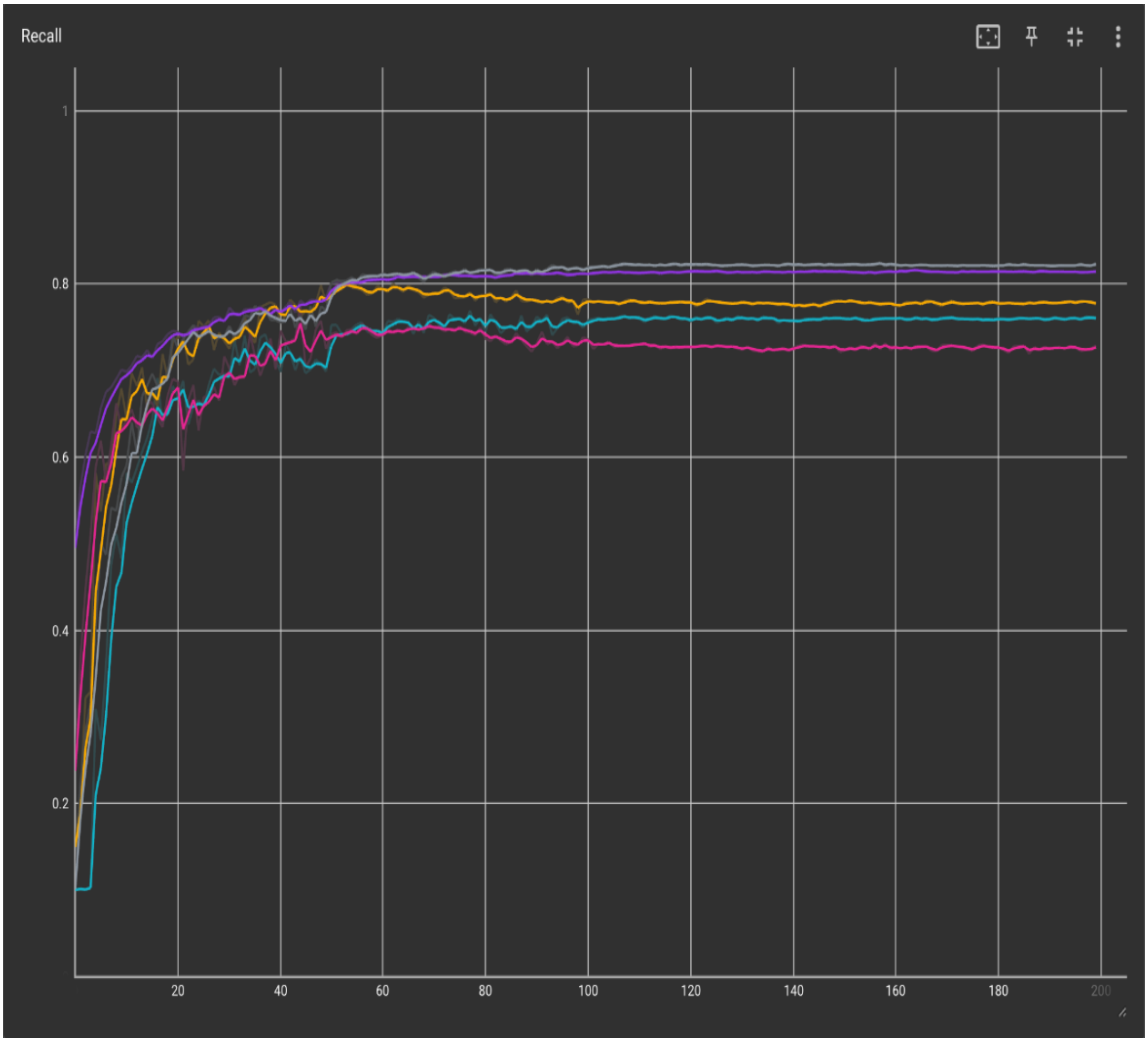

Figure 7.11: CIFAR-10: Recall (Architecture One)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

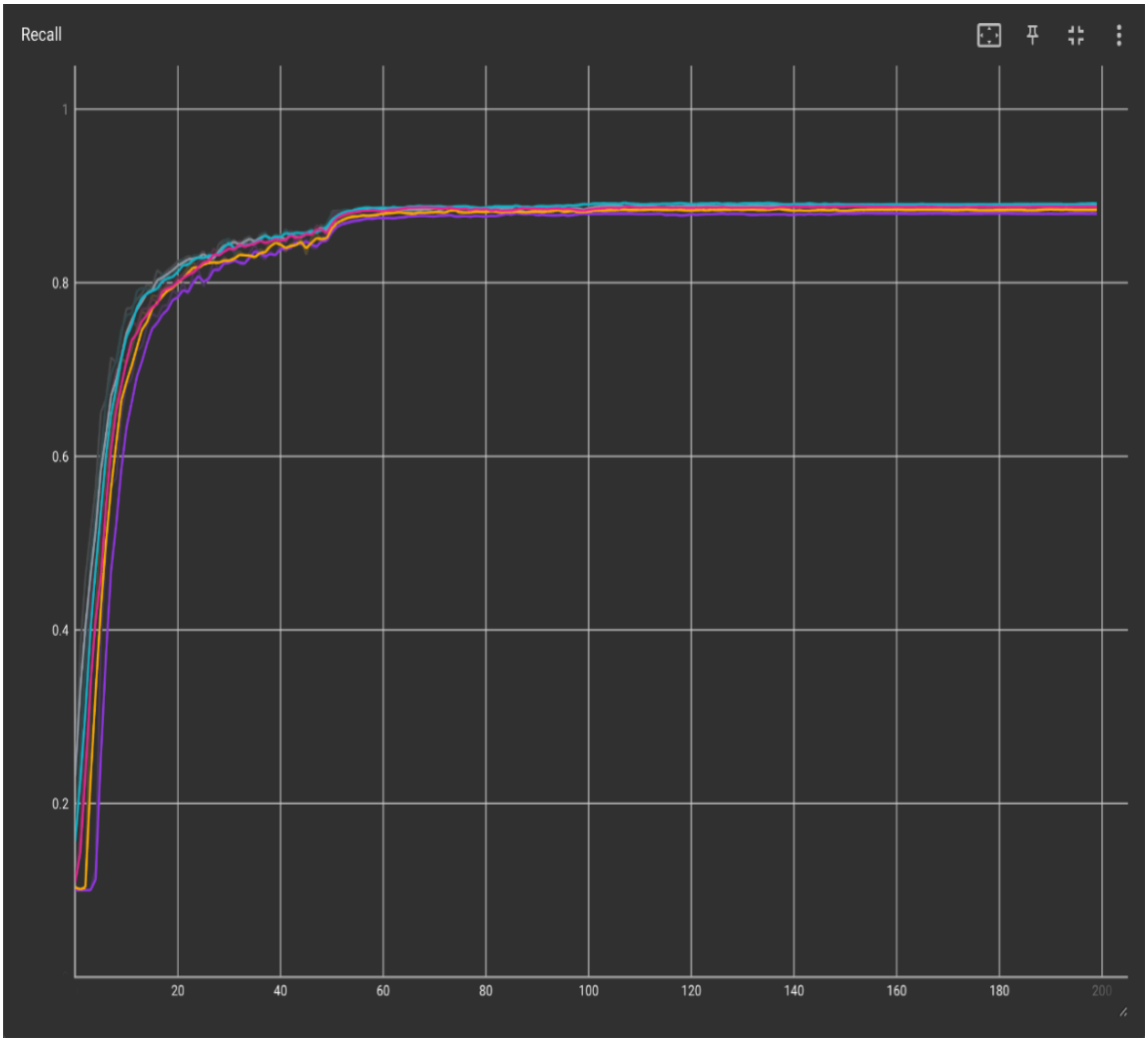

Figure 7.12: CIFAR-10: Recall (Architecture Two)
Legend : Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

SVHN - Accuracy Test

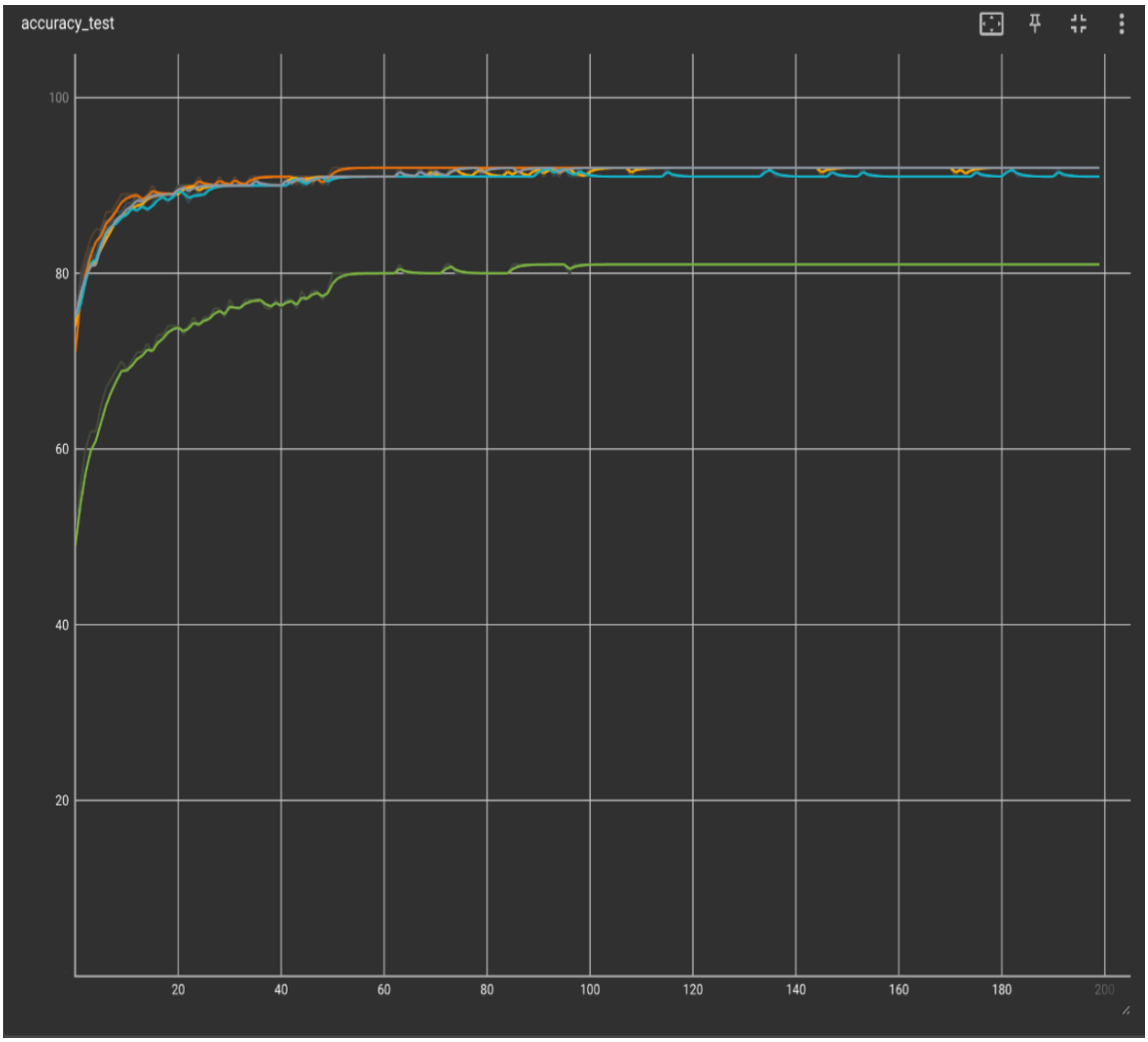

Figure 7.13: SVHN: Test Accuracy (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

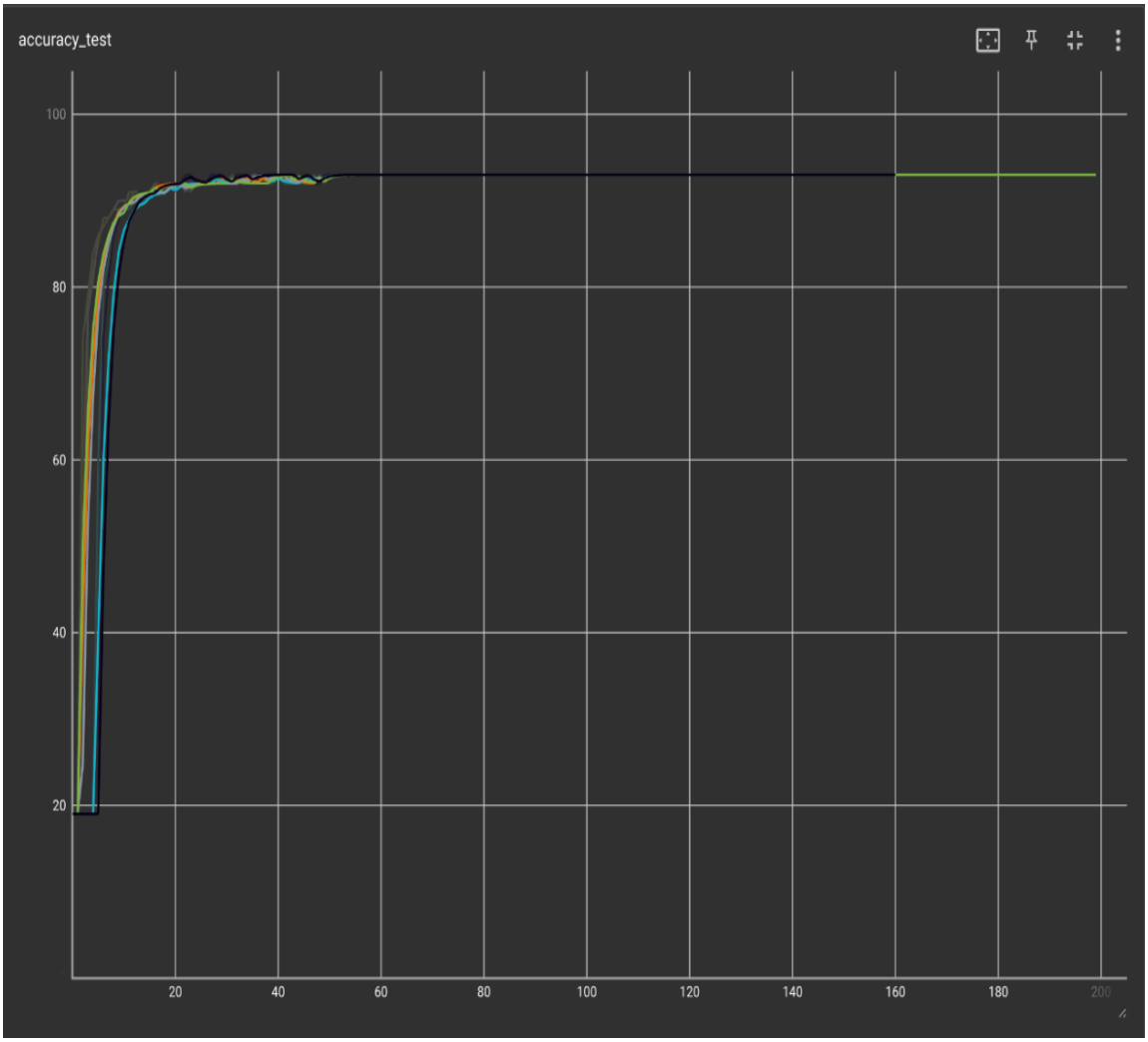

Figure 7.14: SVHN: Test Accuracy (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

SVHN - Accuracy Train

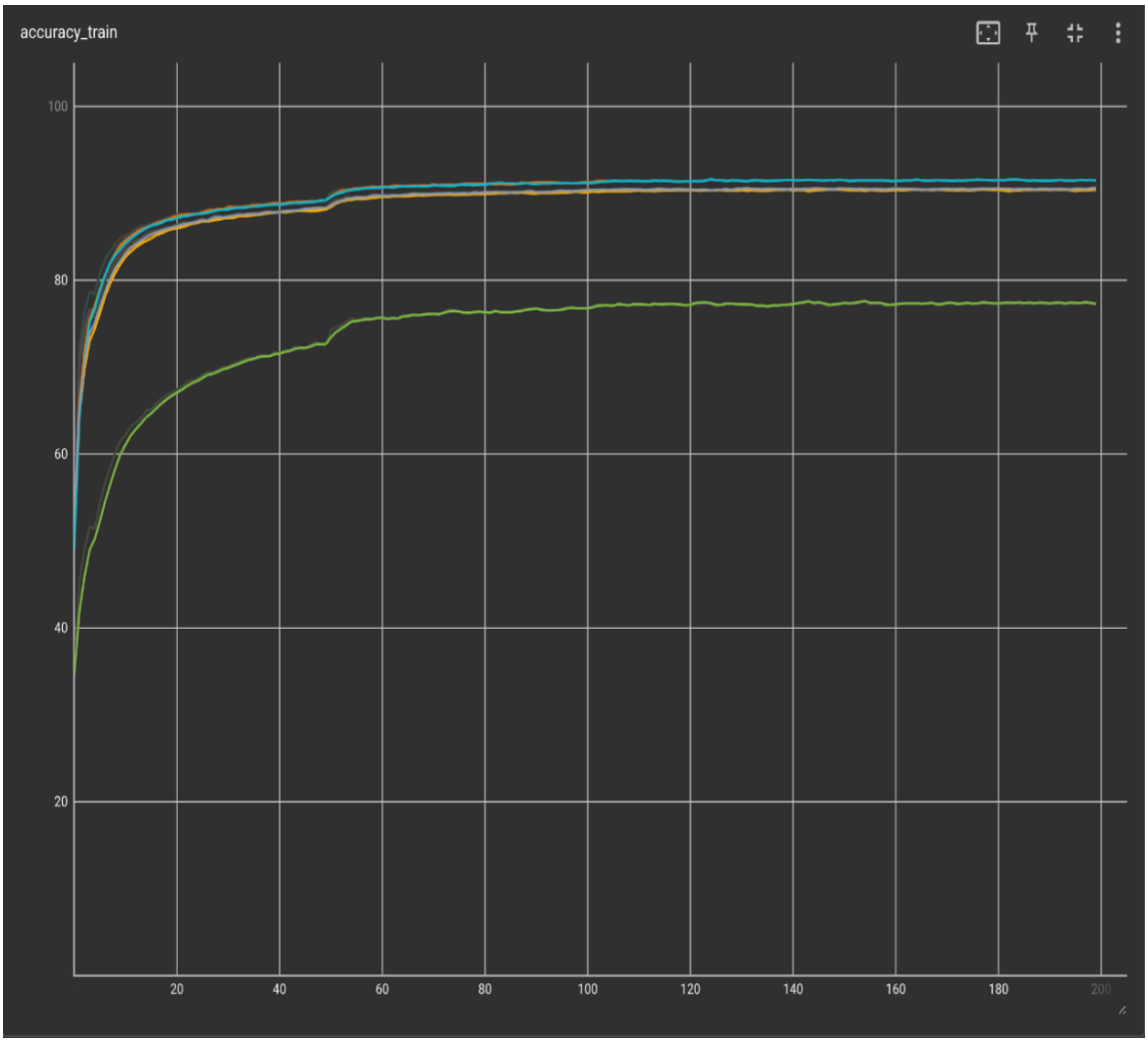

Figure 7.15: SVHN: Train Accuracy (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

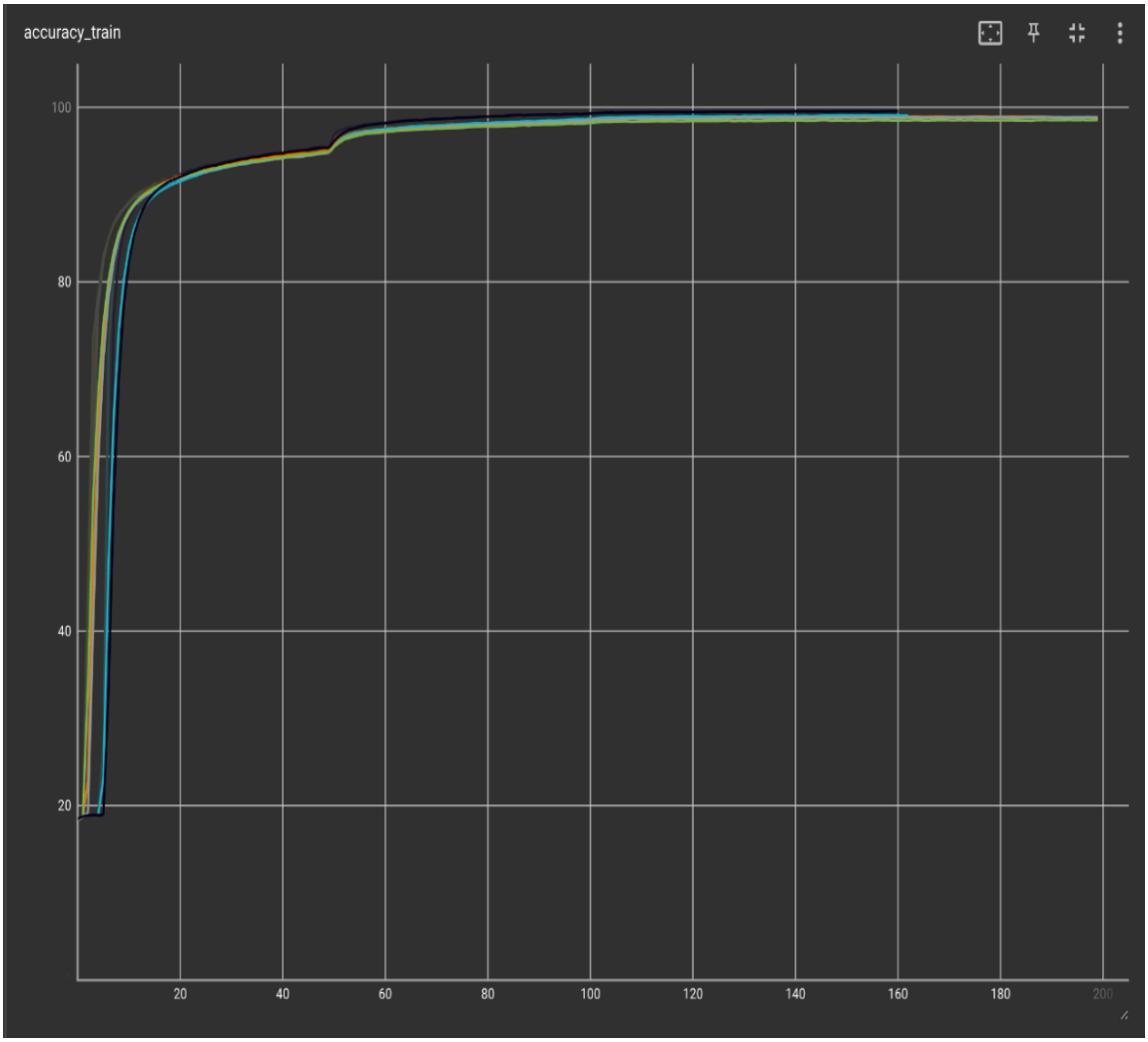

Figure 7.16: SVHN: Train Accuracy (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

SVHN - Loss

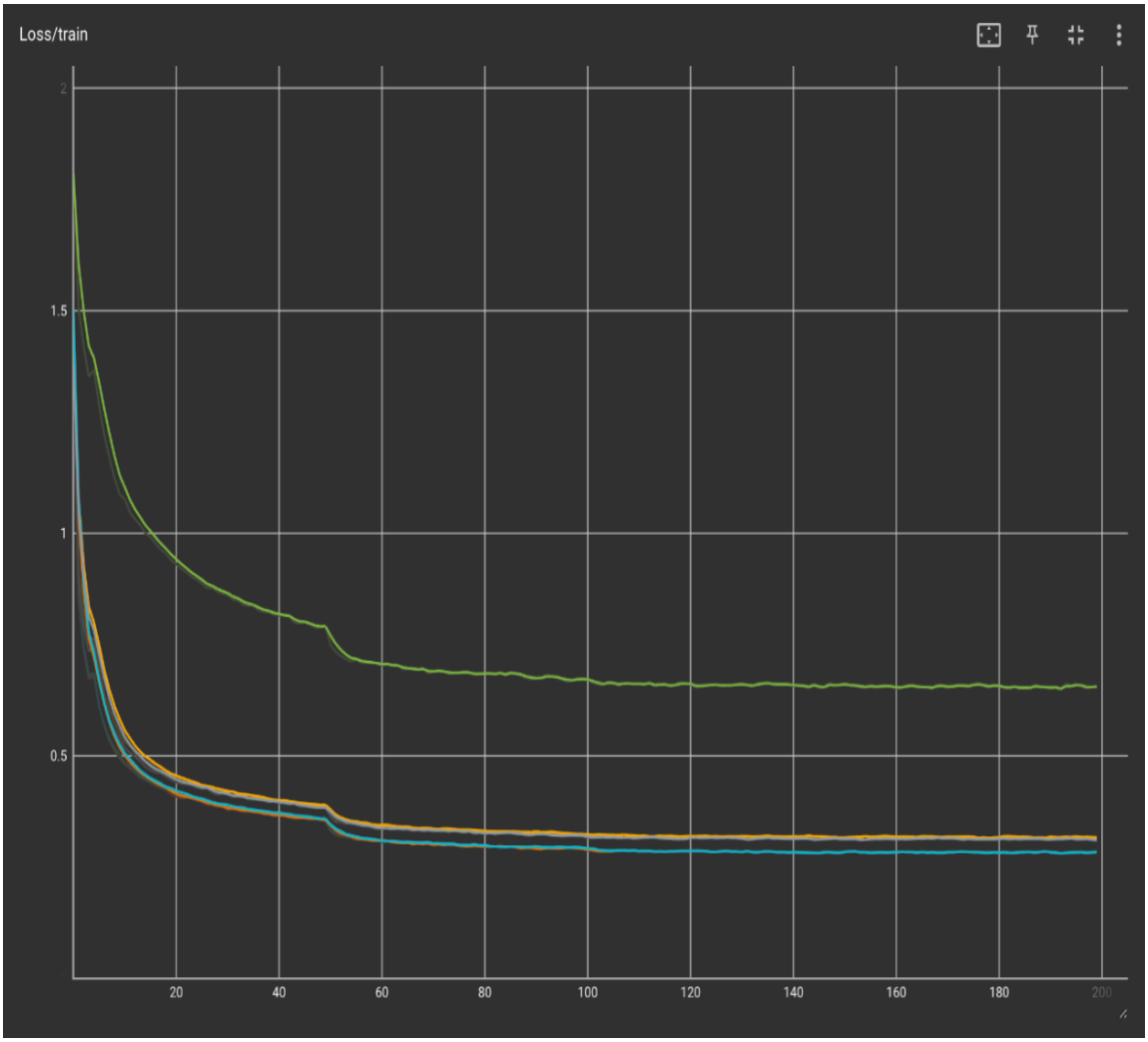

Figure 7.17: SVHN: Loss (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

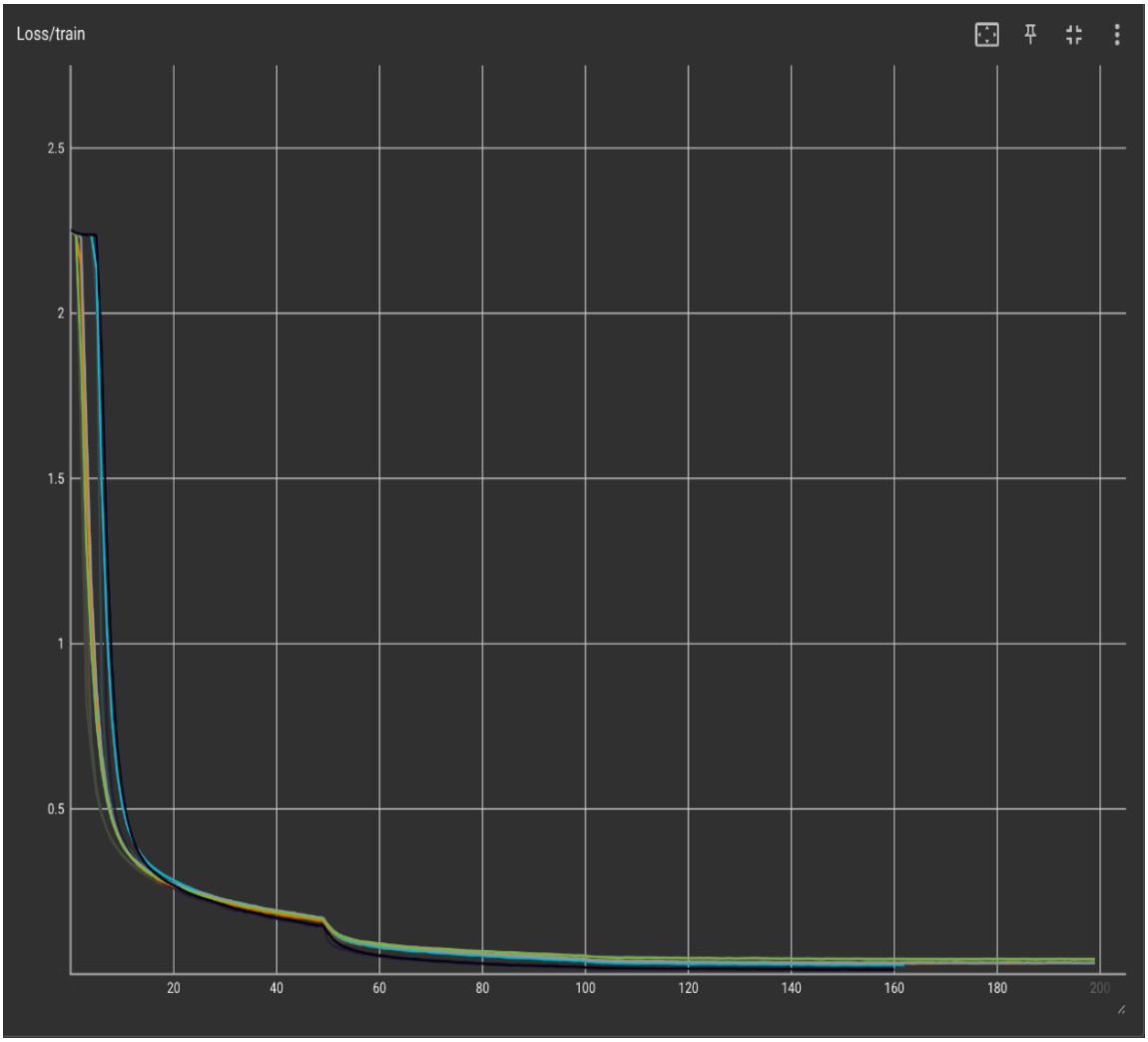

Figure 7.18: SVHN: Loss (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

SVHN - F1

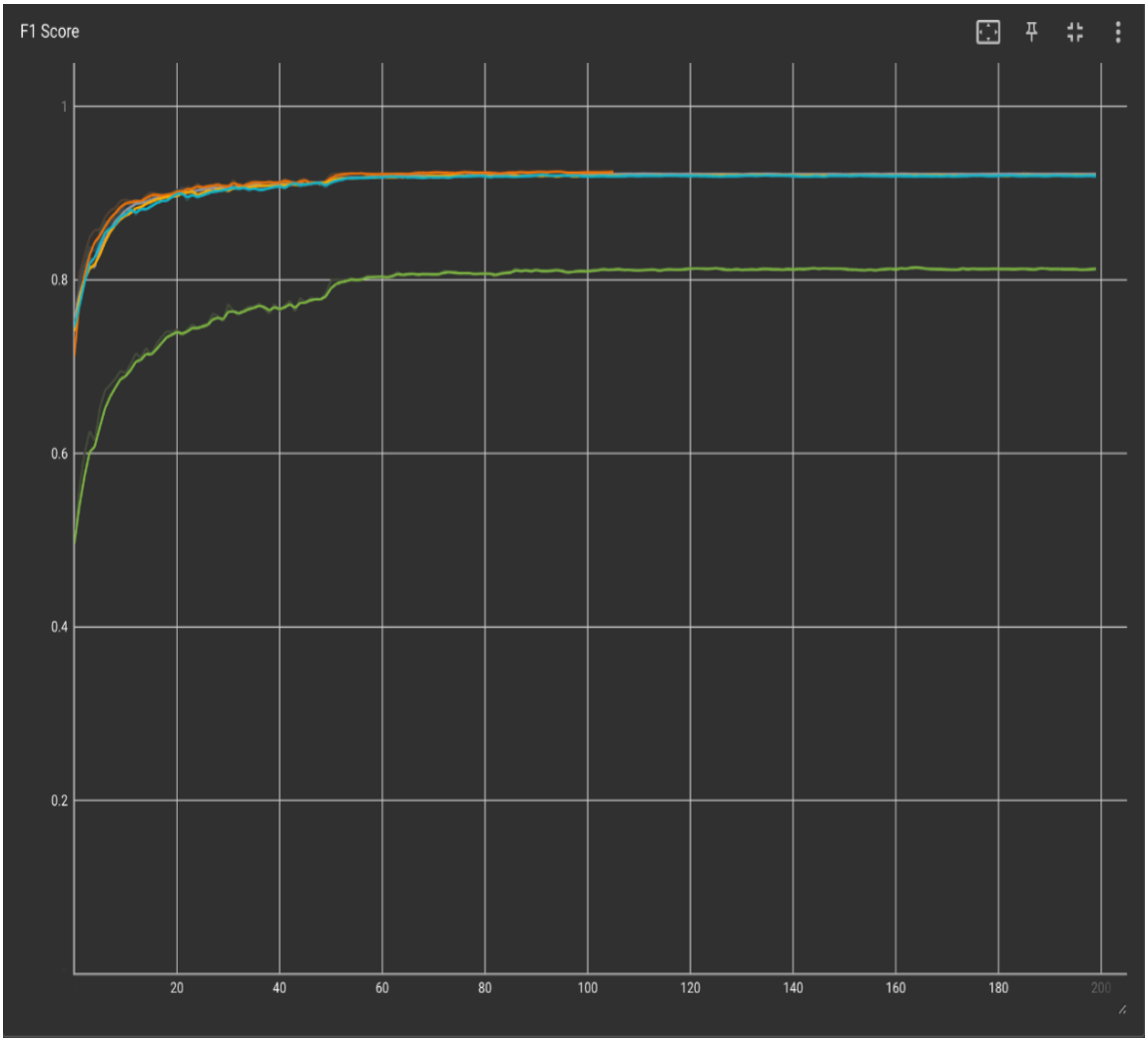

Figure 7.19: SVHN: F1 (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

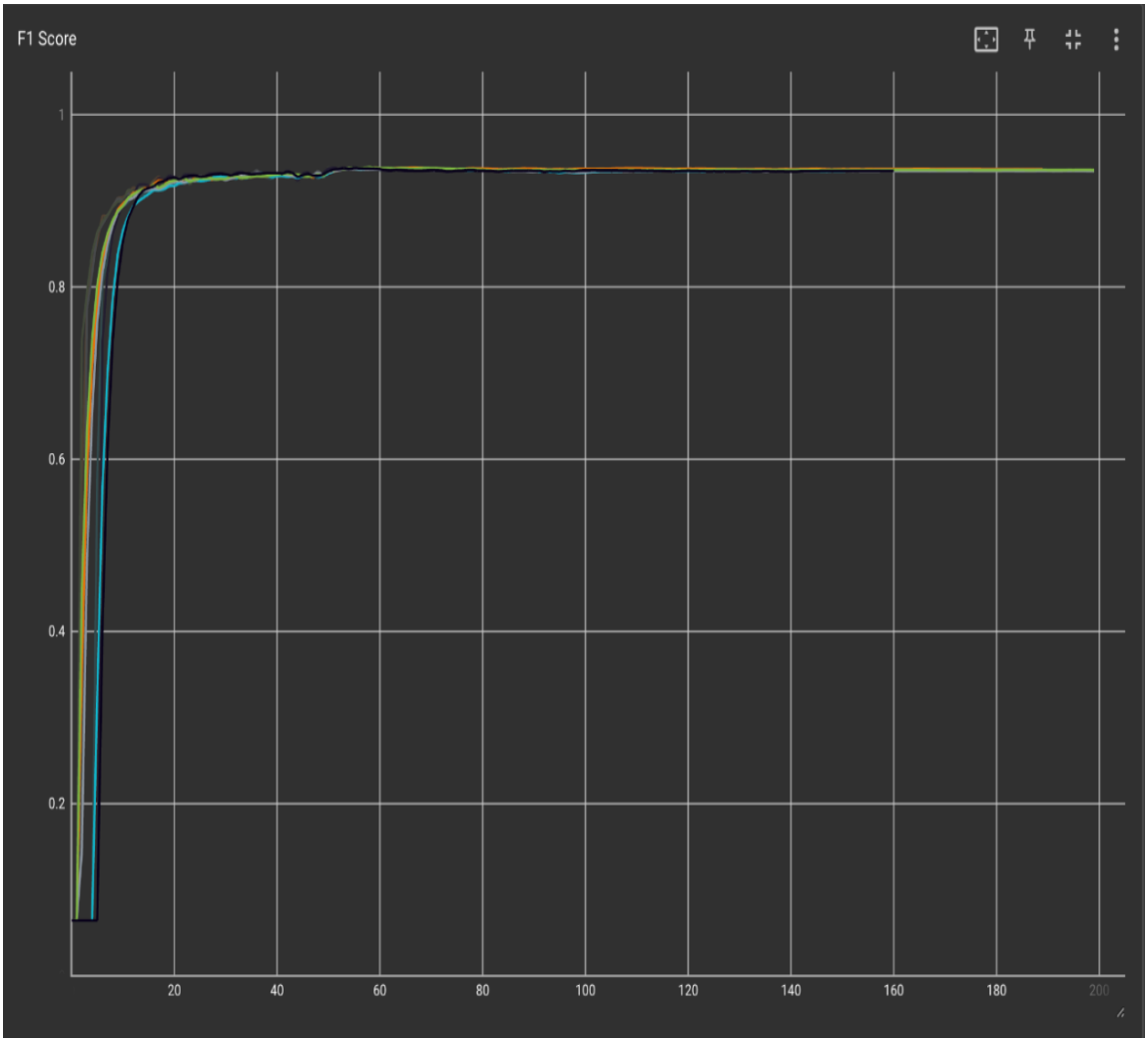

Figure 7.20: SVHN: F1 (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

SVHN - Precision

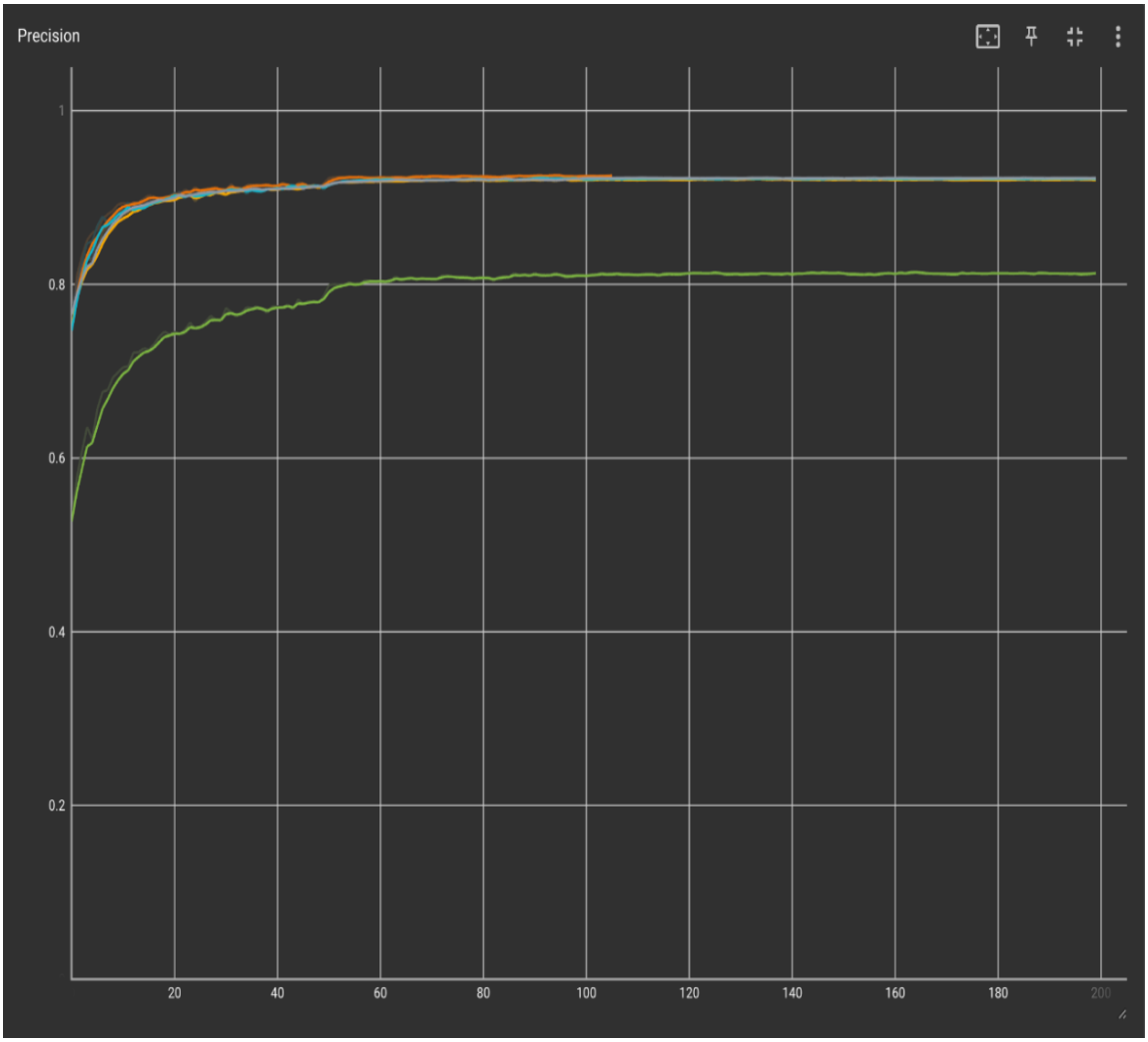

Figure 7.21: SVHN: Precision (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

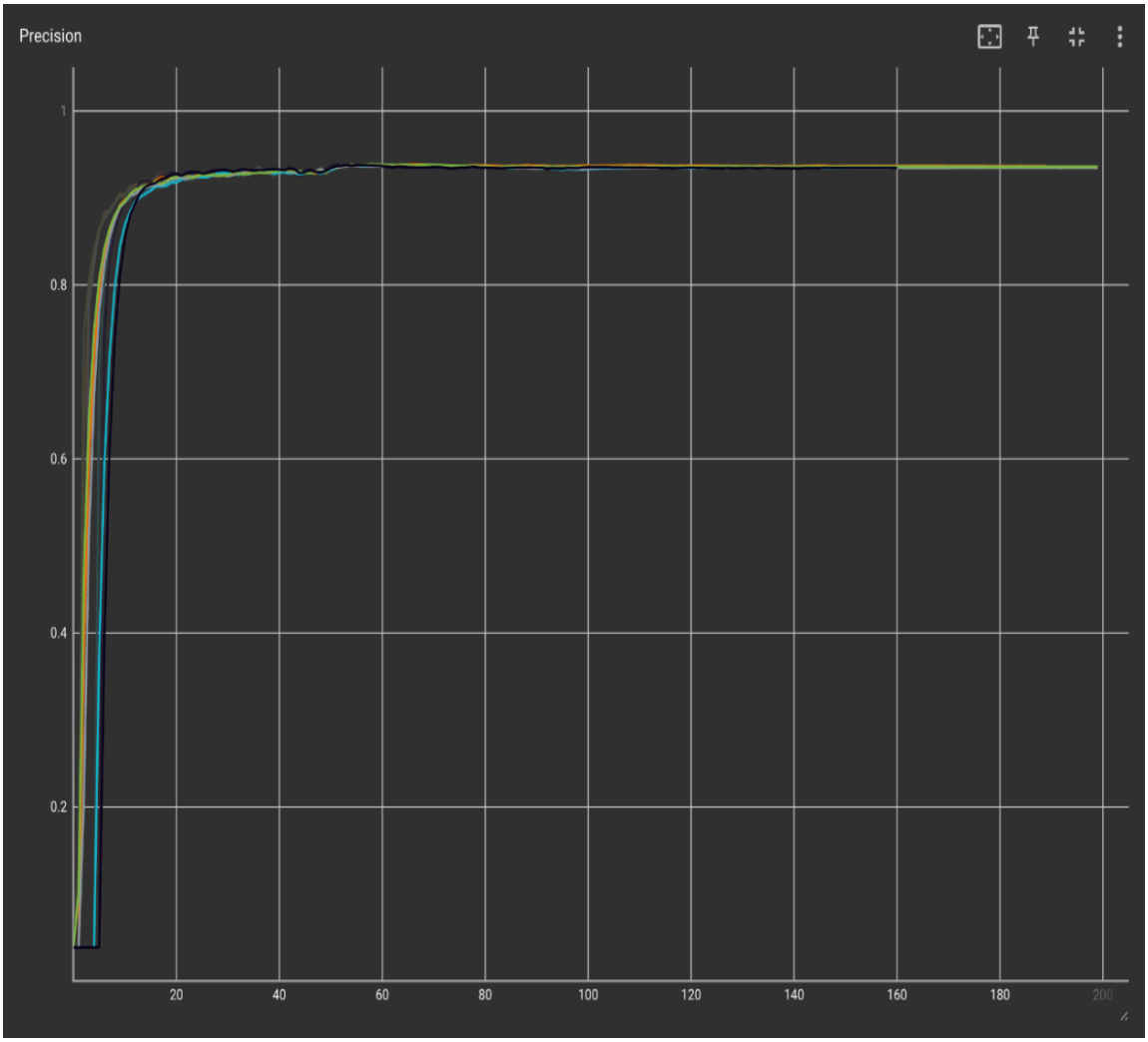

Figure 7.22: SVHN: Precision (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

SVHN - Recall

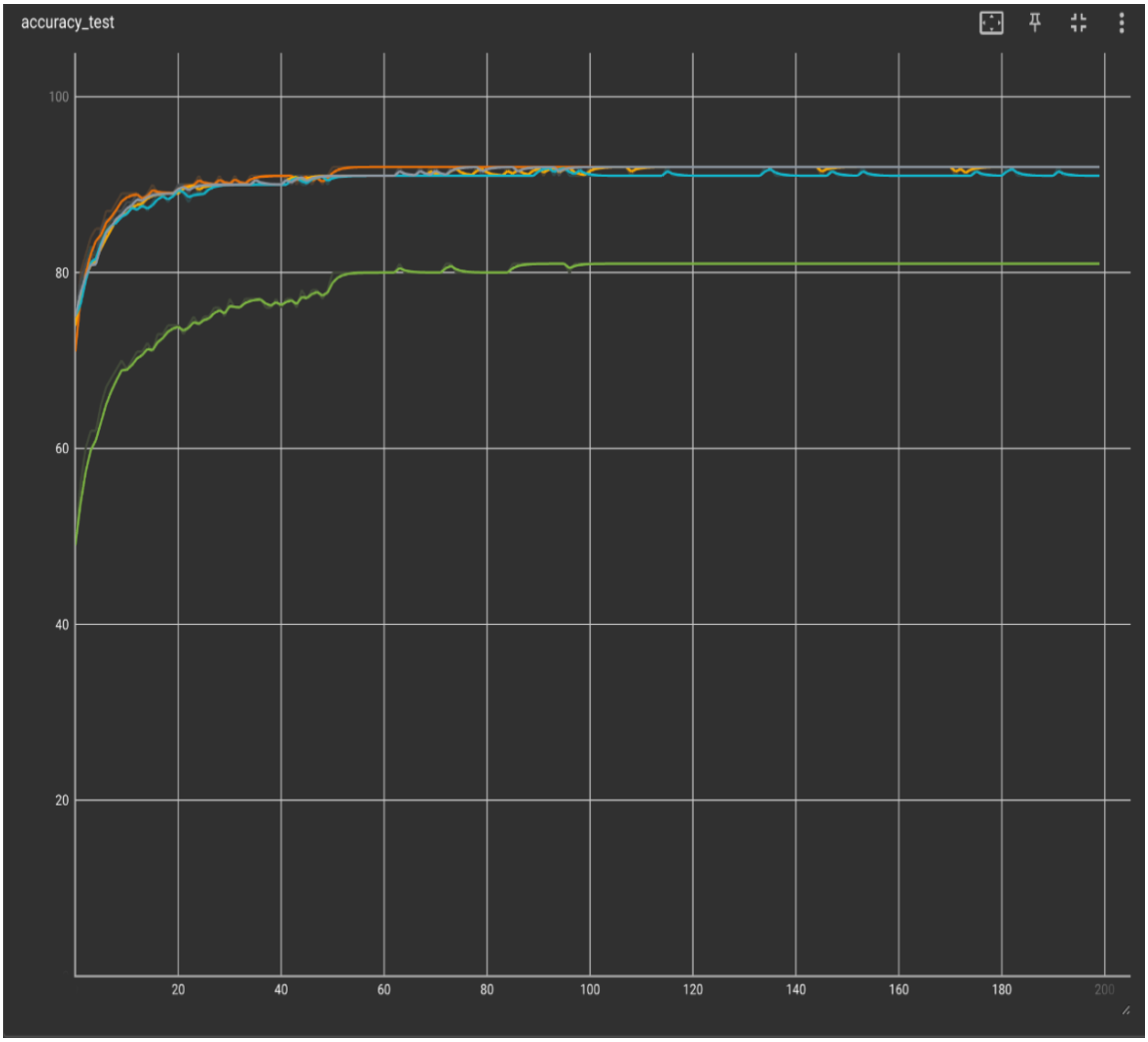

Figure 7.23: SVHN: Recall (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

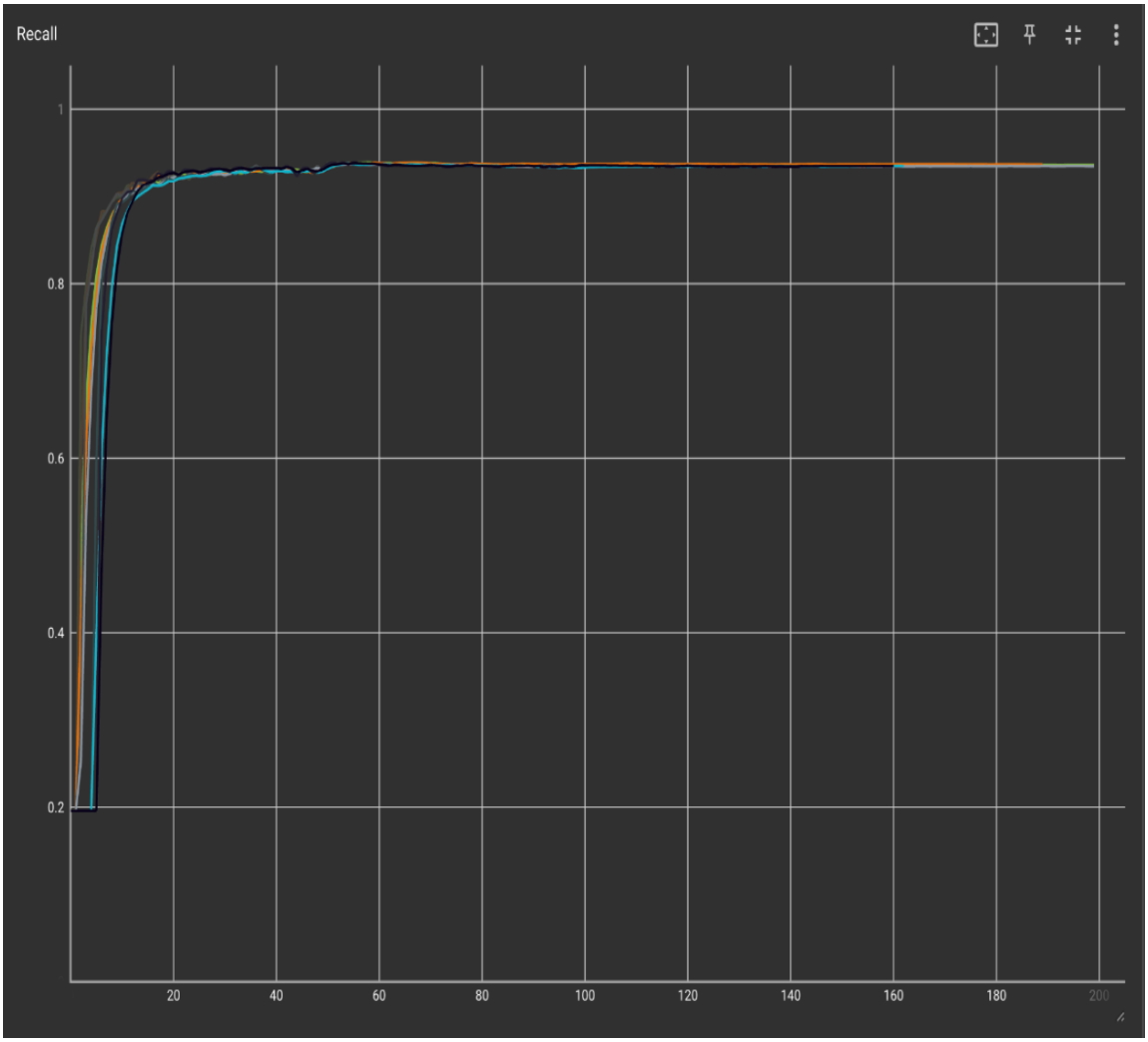

Figure 7.24: SVHN: Recall (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8,

MNIST - Accuracy Test

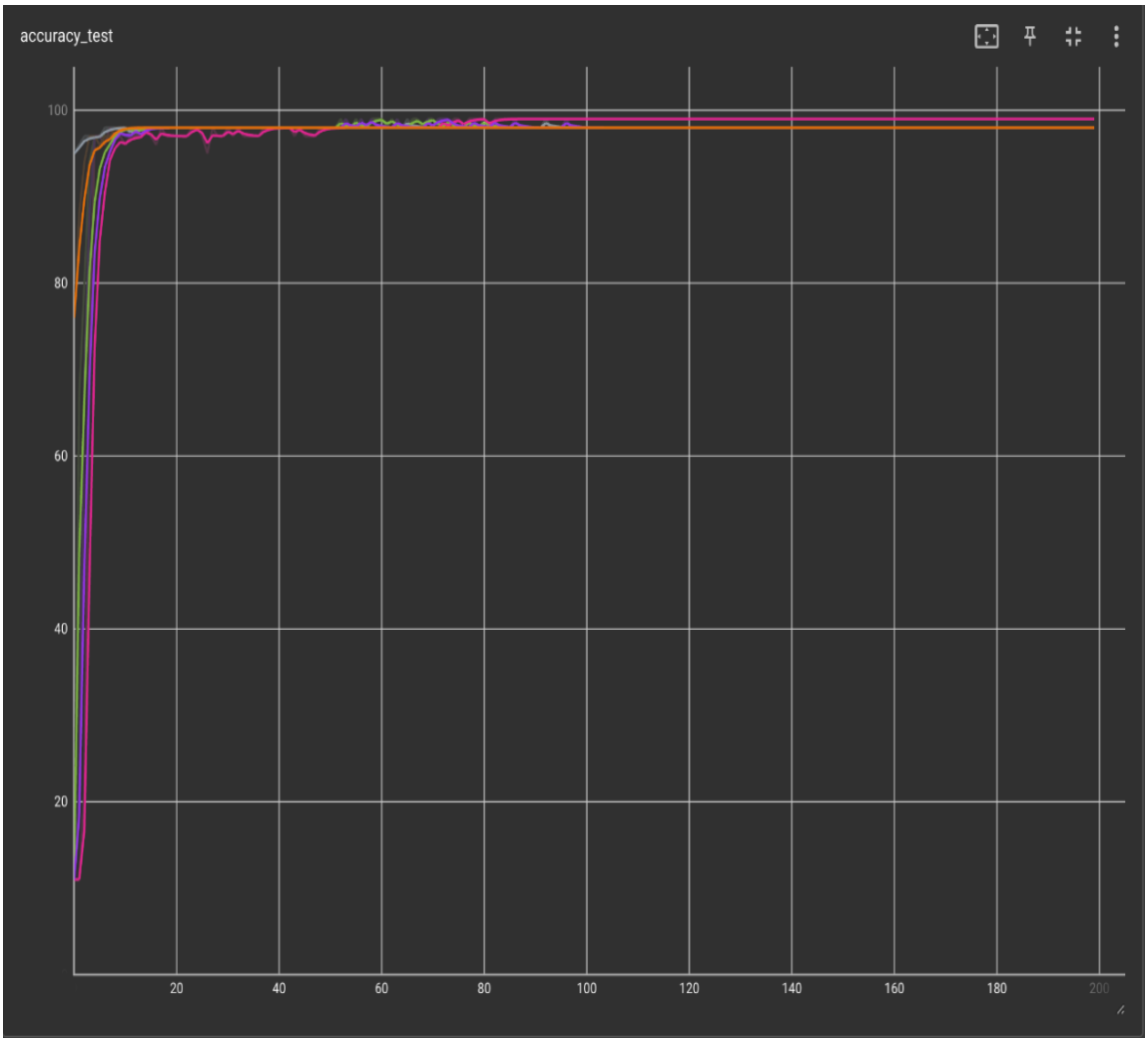

Figure 7.25: MNIST: Test Accuracy (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

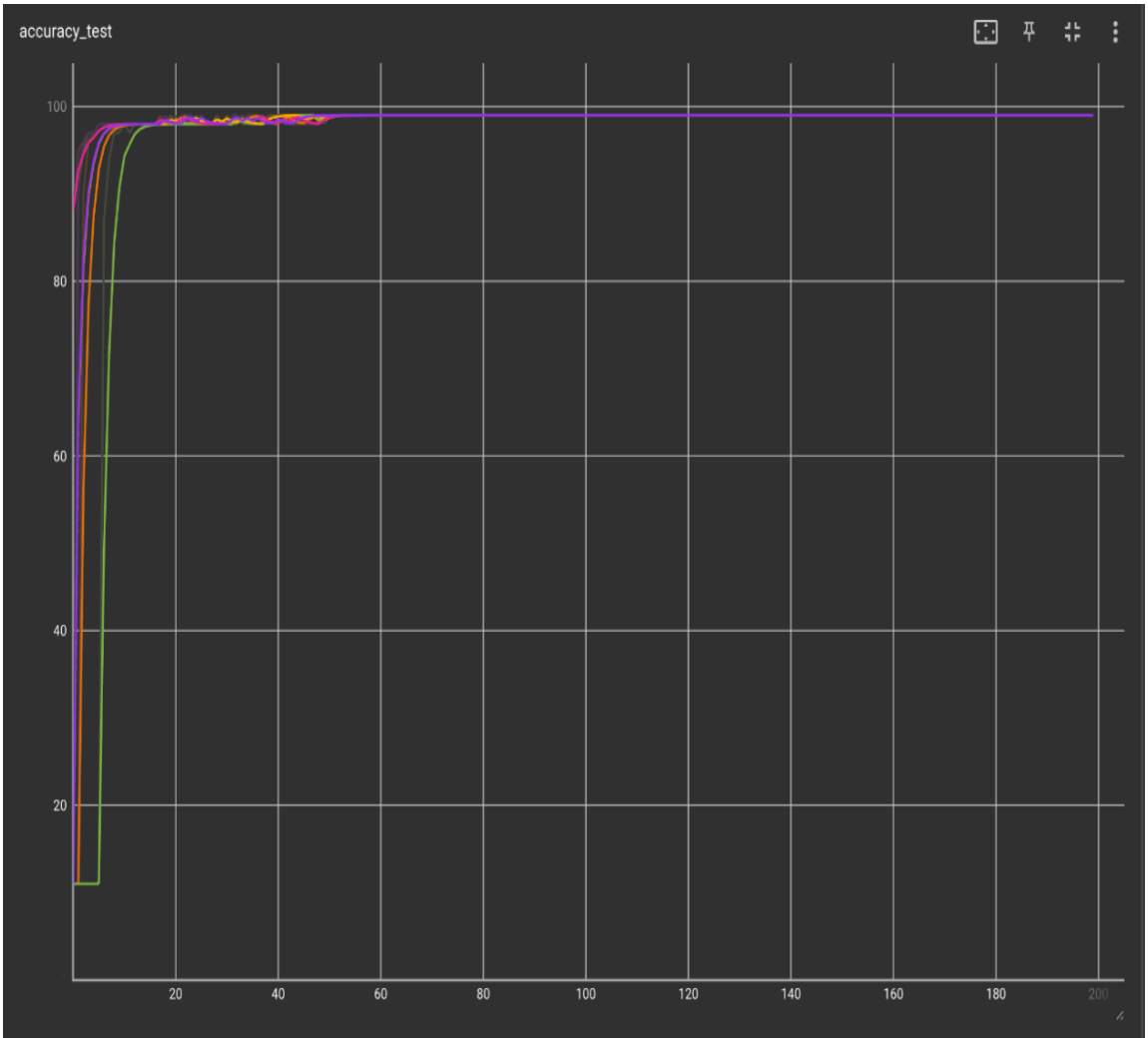

Figure 7.26: MNIST: Test Accuracy (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

MNIST - Accuracy Train

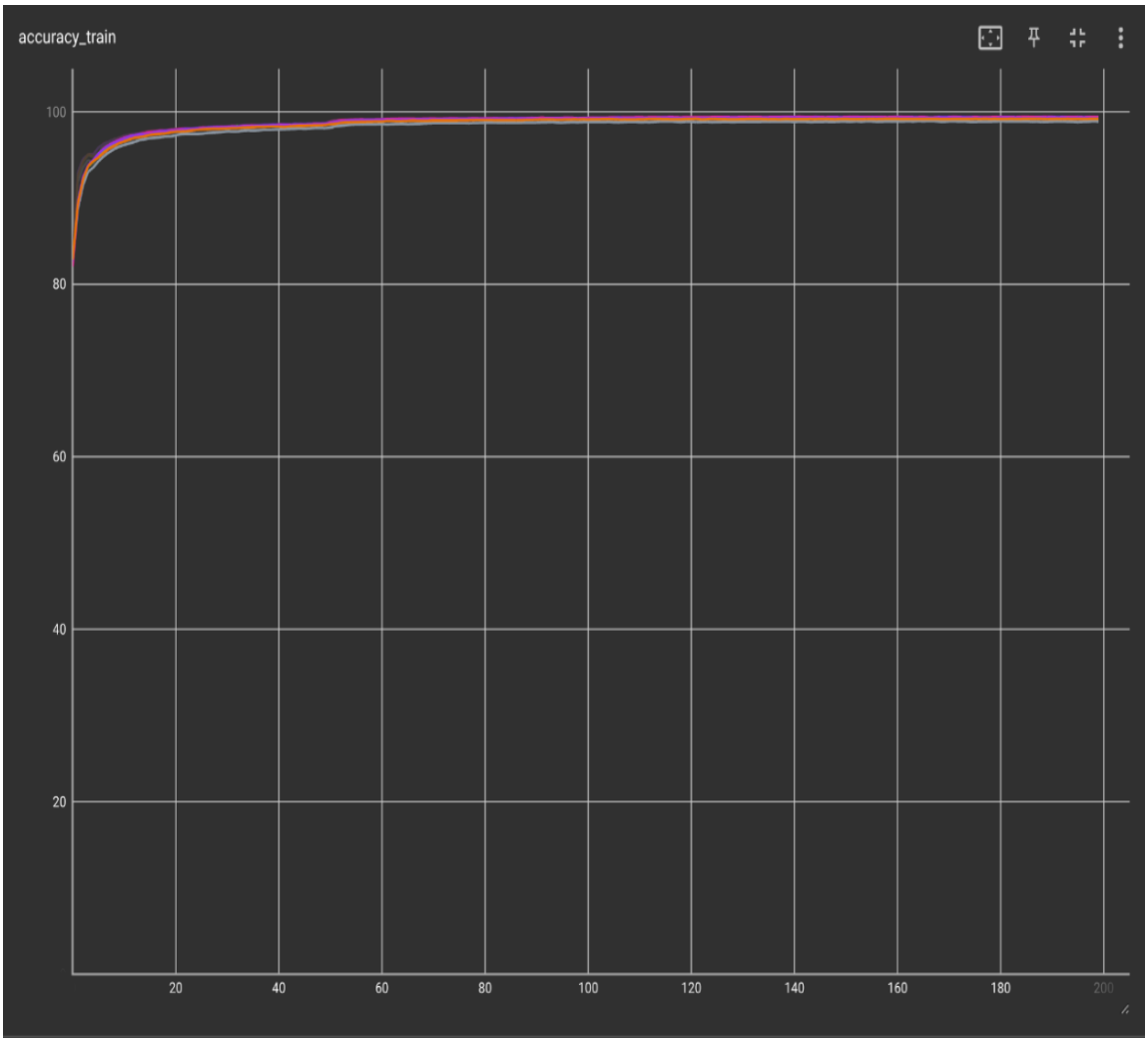

Figure 7.27: MNIST: Train Accuracy (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

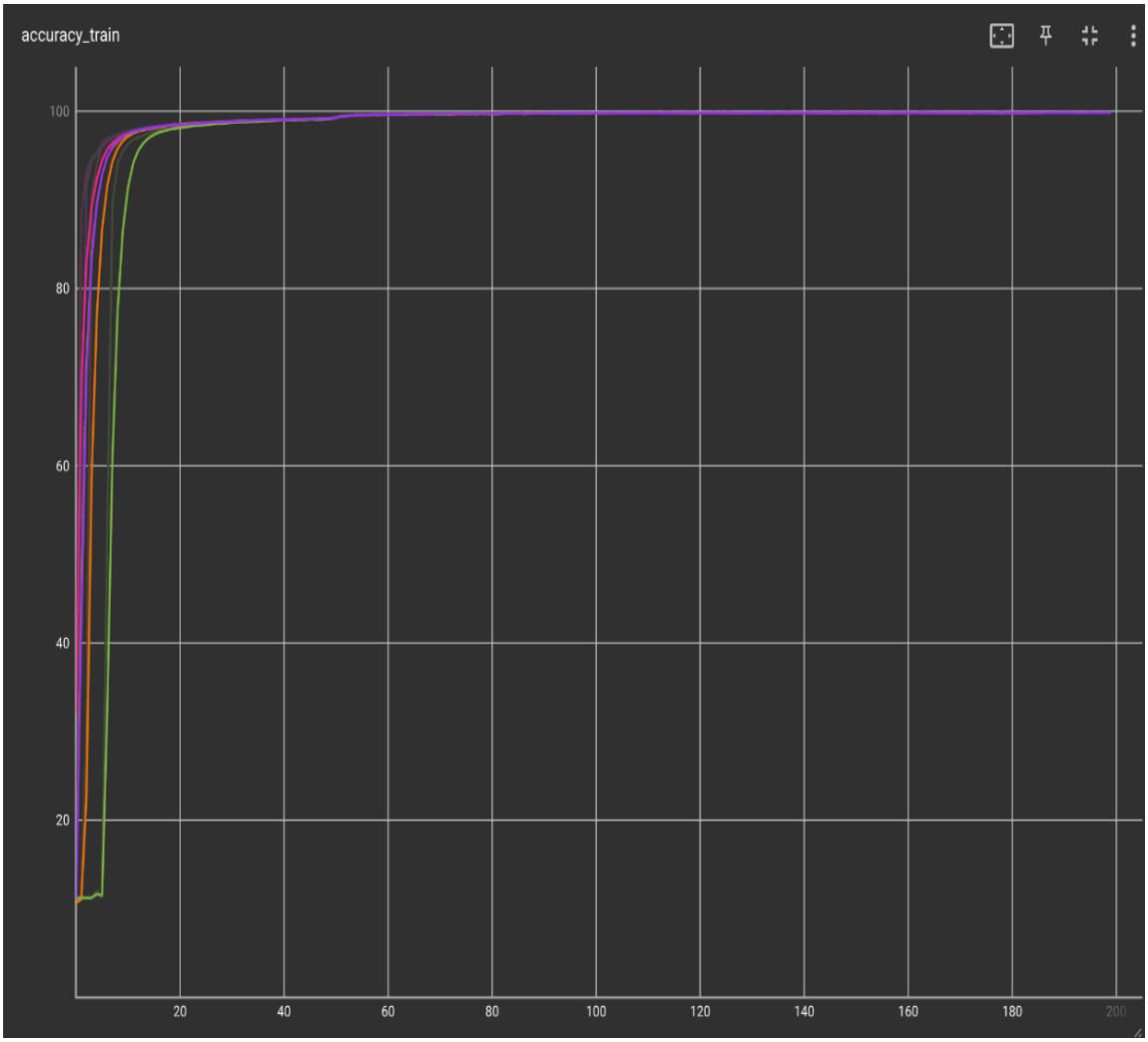

Figure 7.28: MNIST: Train Accuracy (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

MNIST - Loss

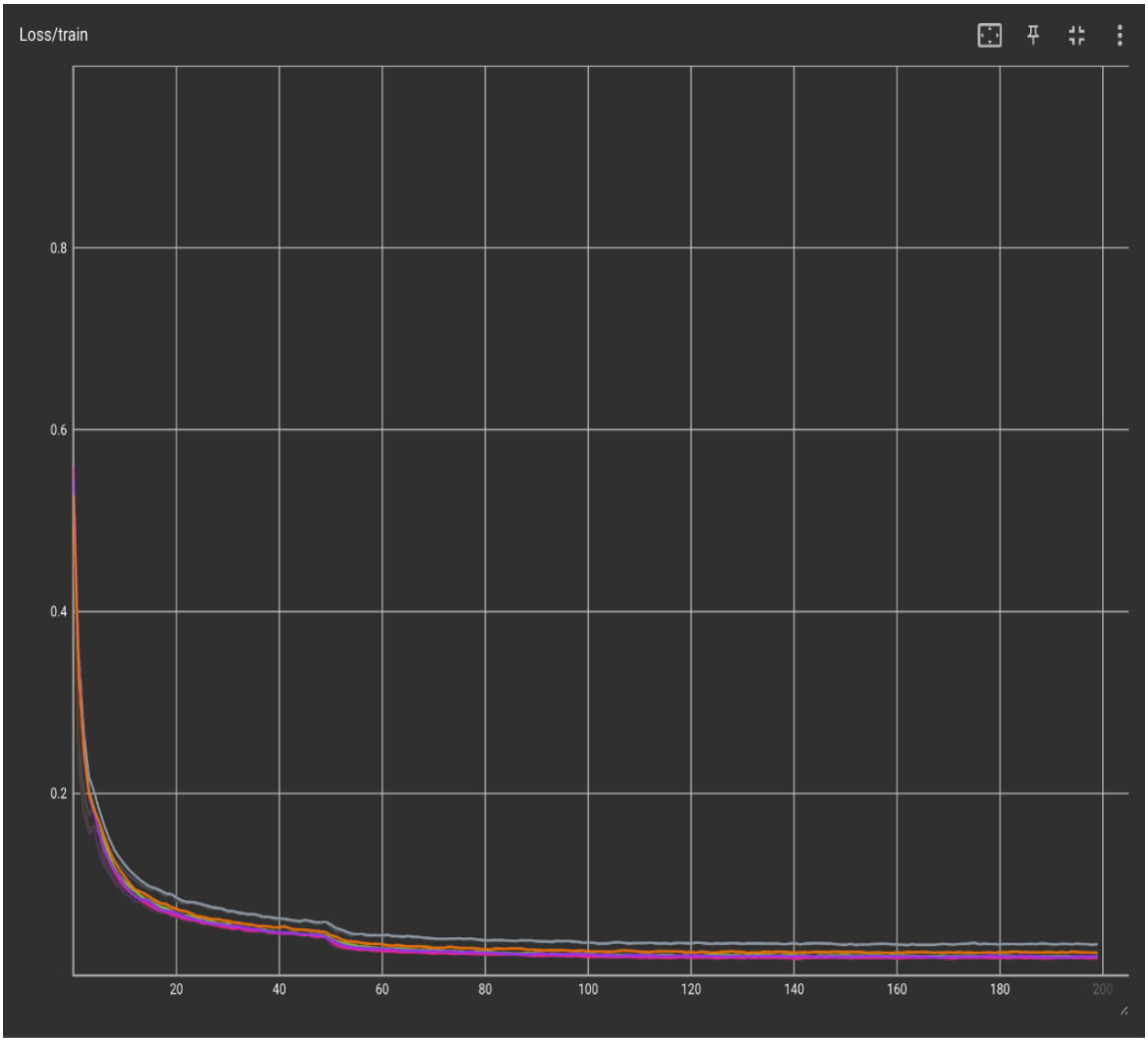

Figure 7.29: MNIST: Loss (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

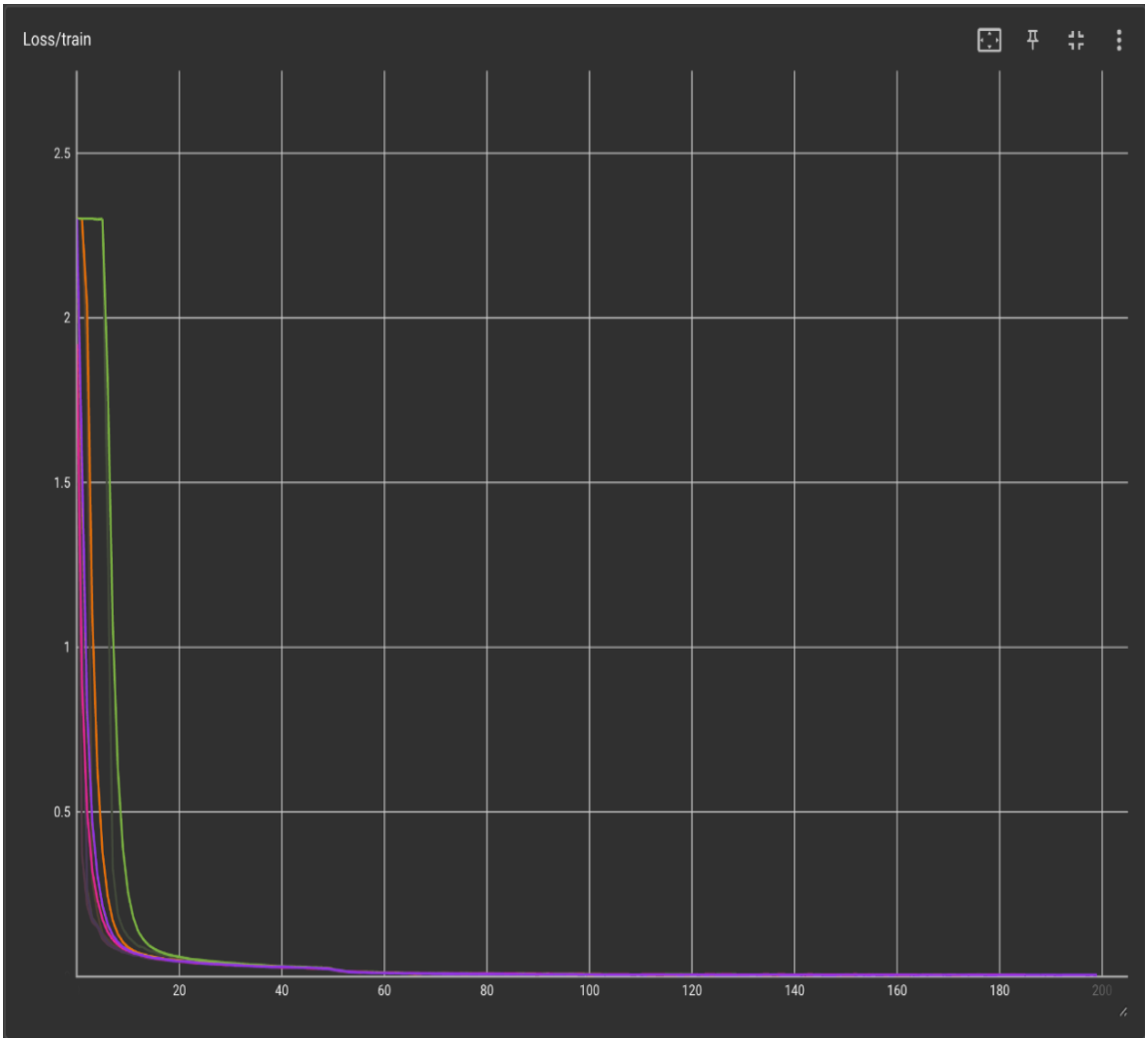

Figure 7.30: MNIST: Loss (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

MNIST - F1

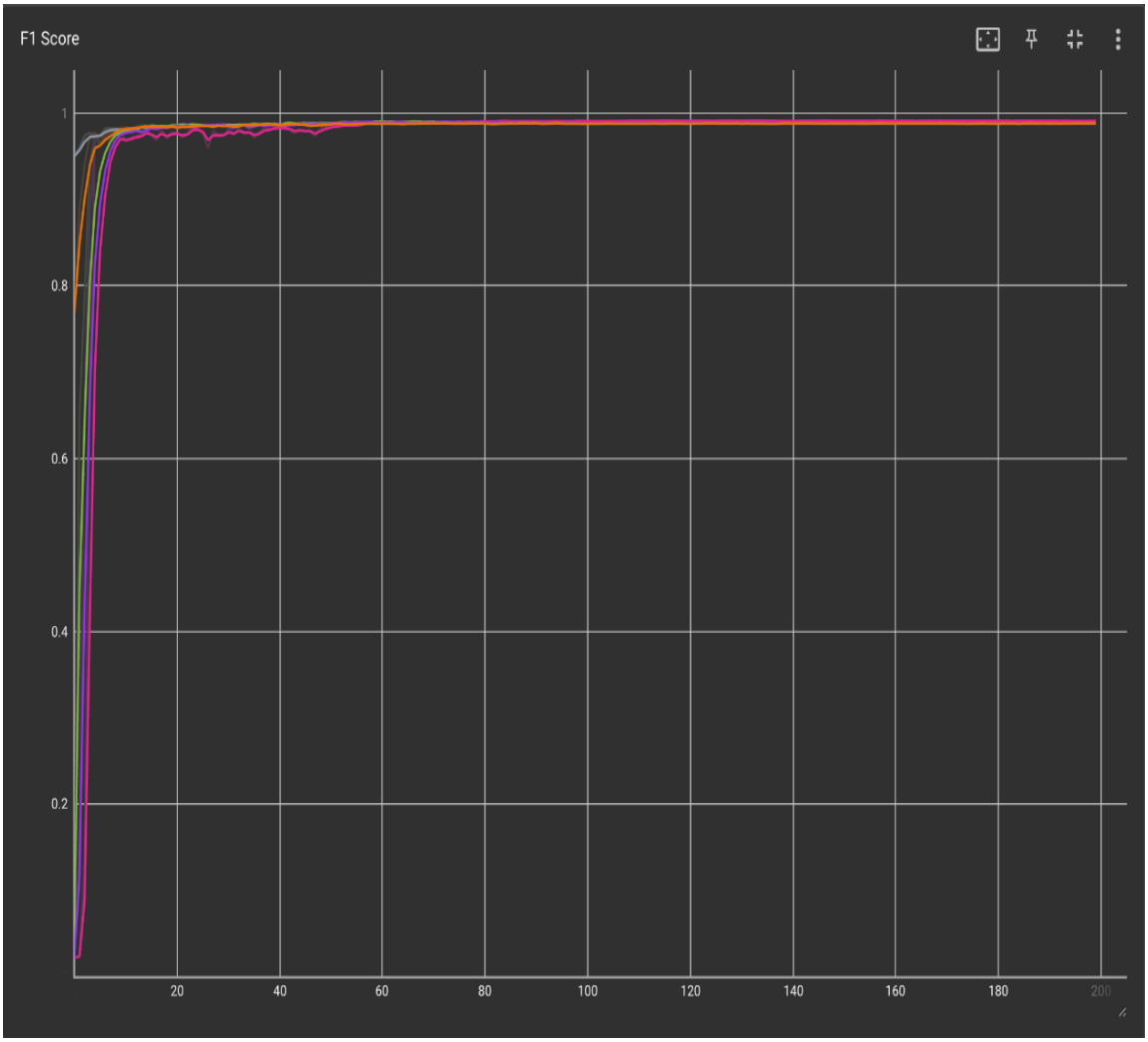

Figure 7.31: MNIST: F1 (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

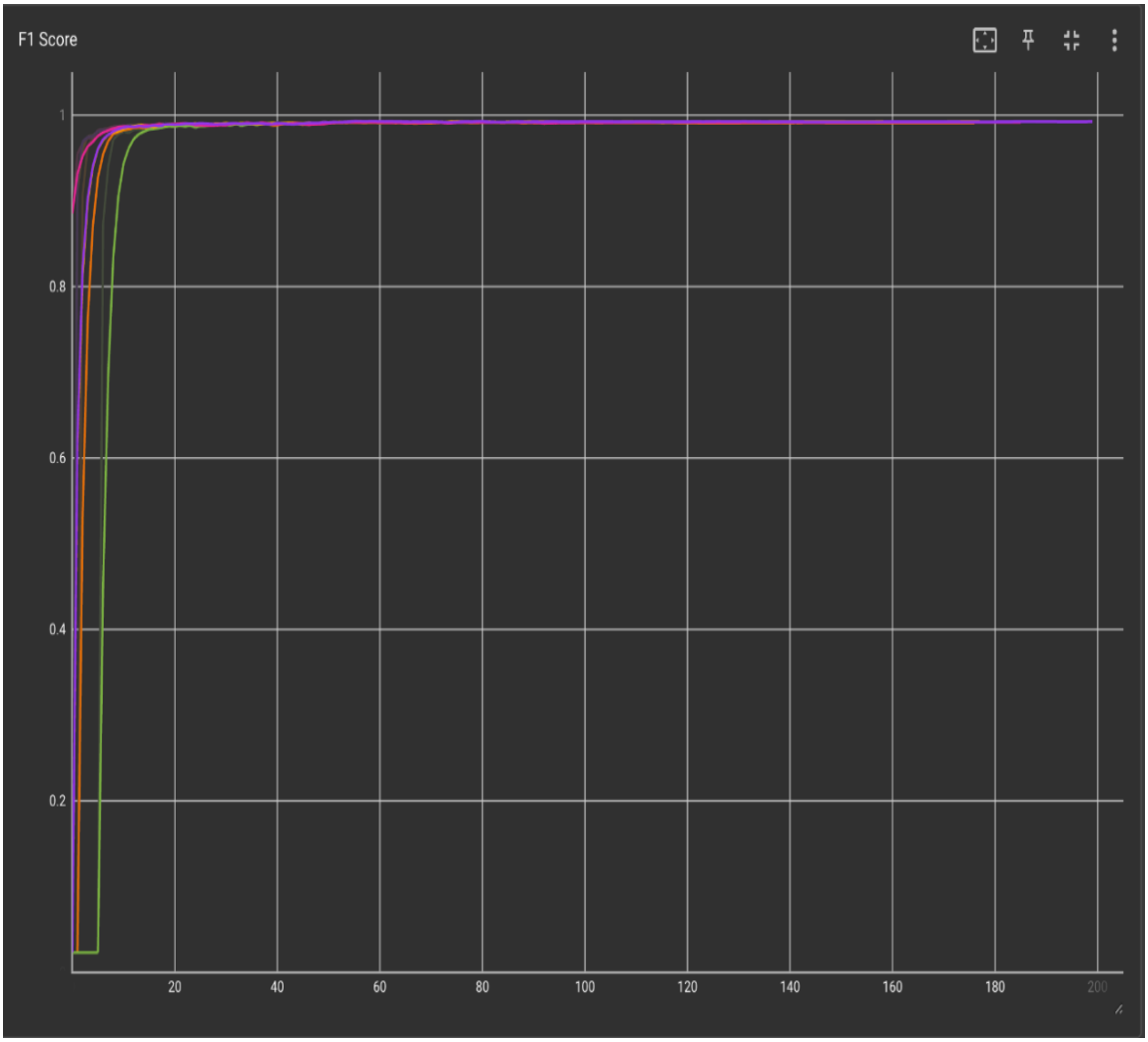

Figure 7.32: MNIST: F1 (Architecture Two)
Legend: Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

MNIST - Precision

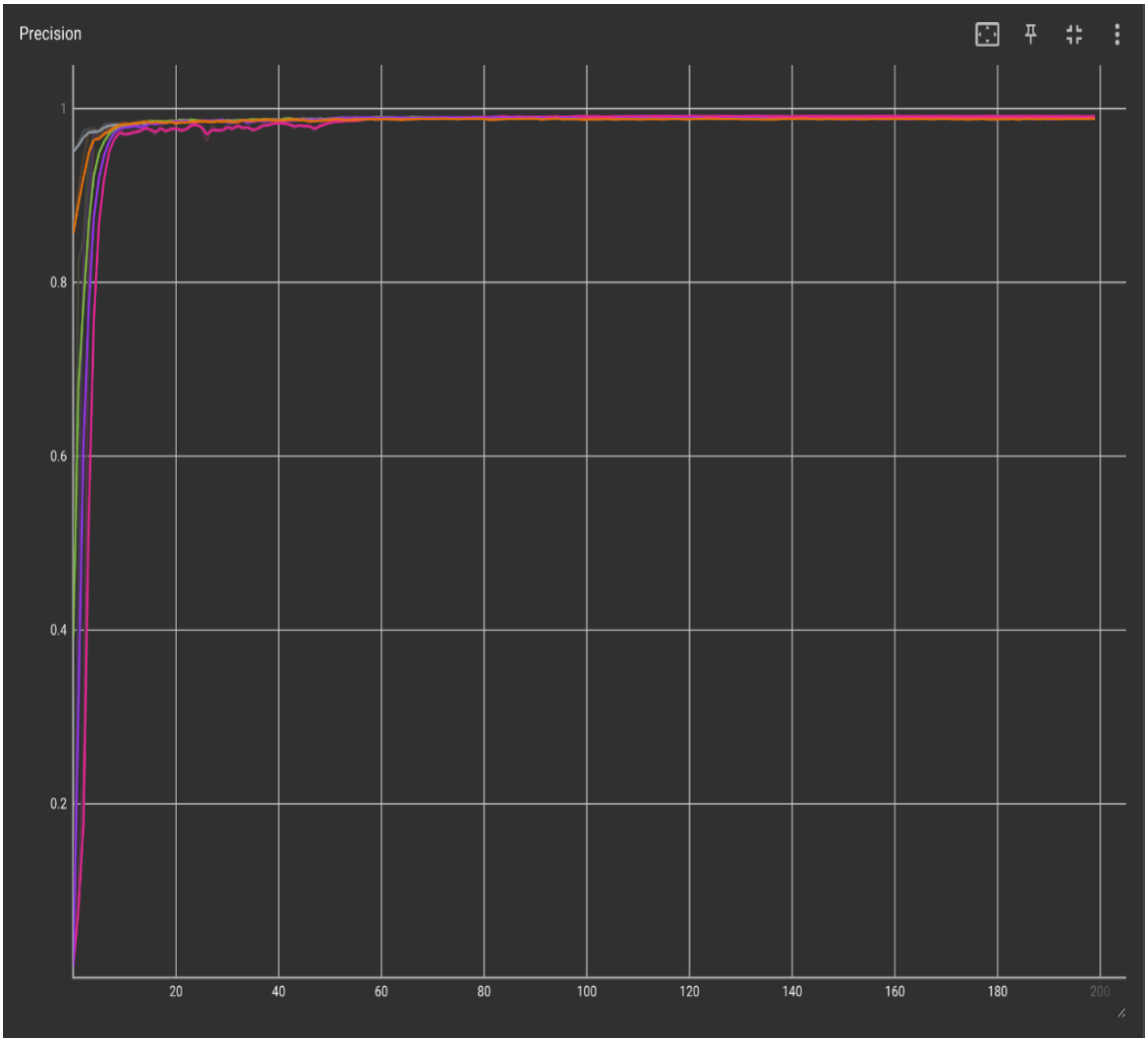

Figure 7.33: MNIST: Precision (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

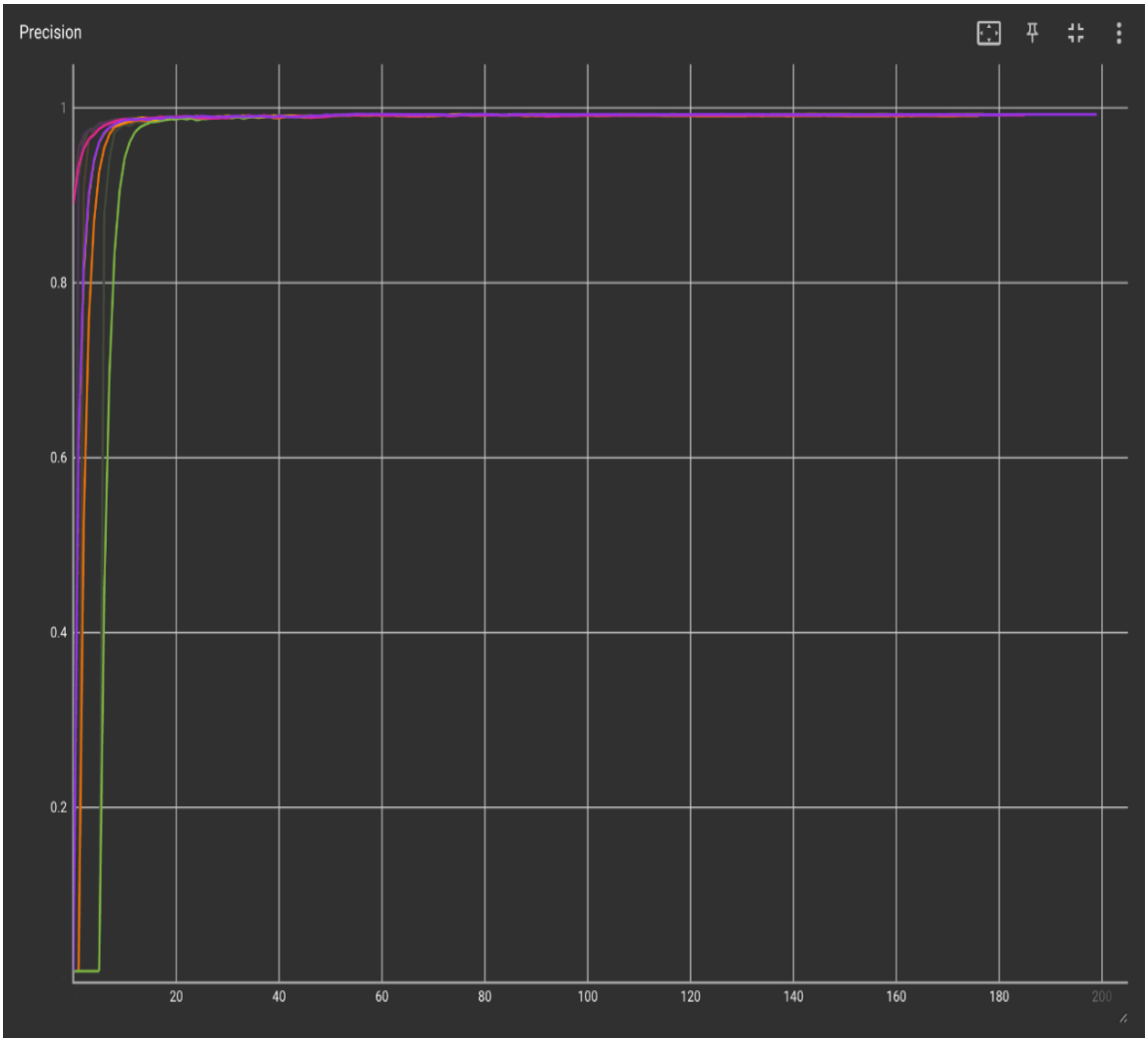

Figure 7.34: MNIST: Precision (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

MNIST - Recall

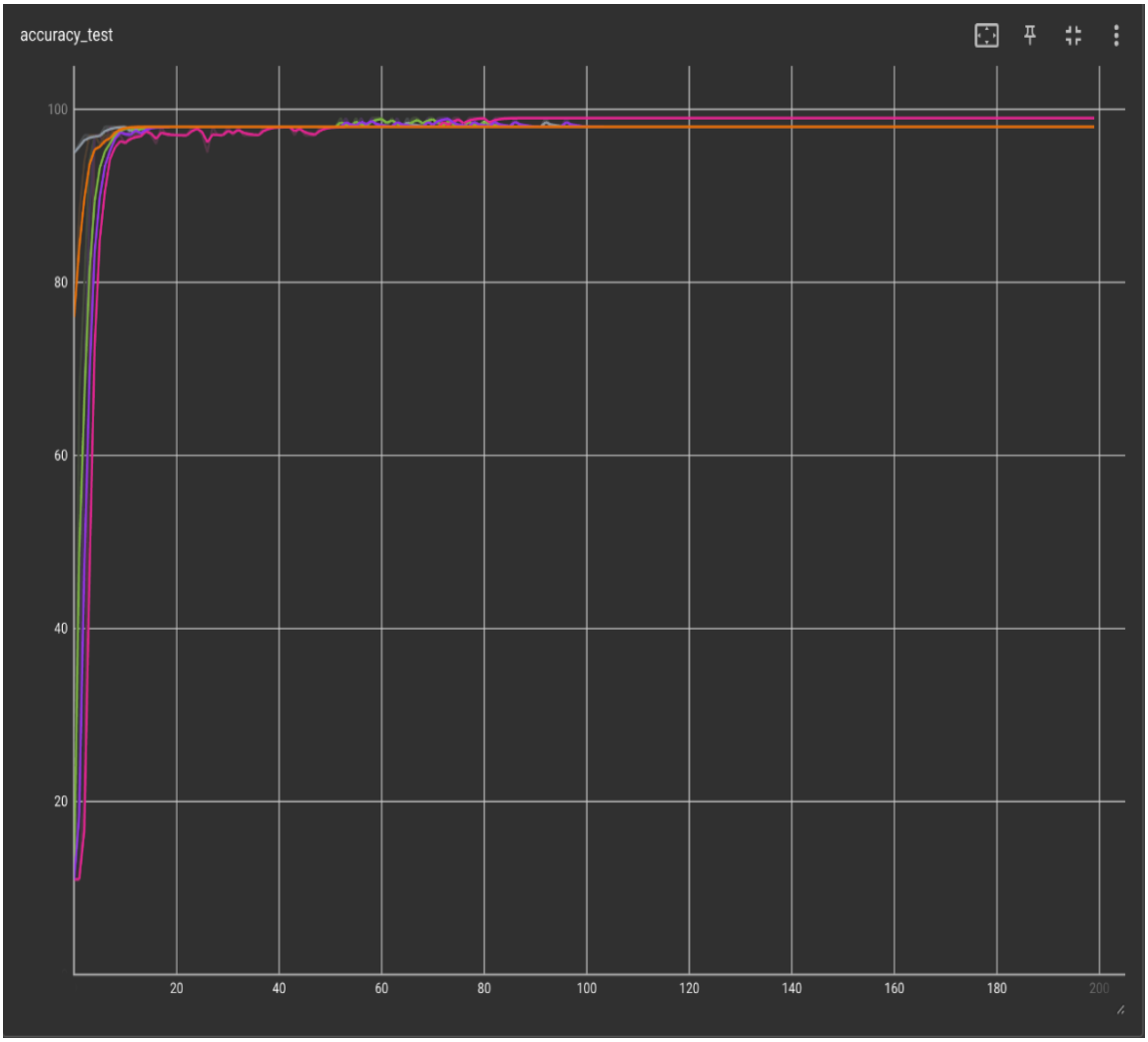

Figure 7.35: MNIST: Recall (Architecture One)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8

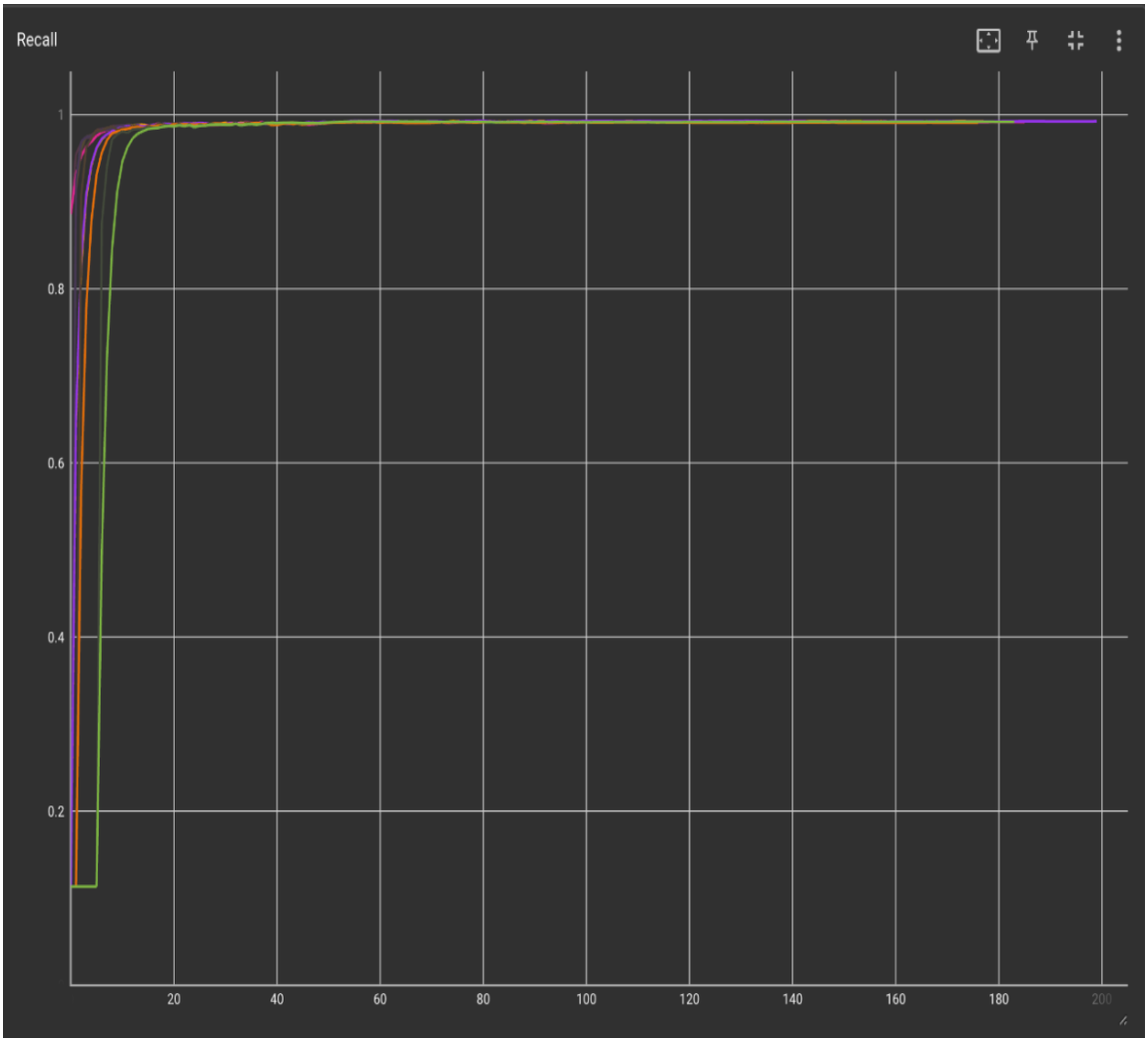

Figure 7.36: MNIST: Recall (Architecture Two)
Legend: Depth-4, Depth-5, Depth-6, Depth-7, Depth-8