Ege Konya
CS 6375.001

Project 3 Report

Datasets and Preprocessing

Transformations:

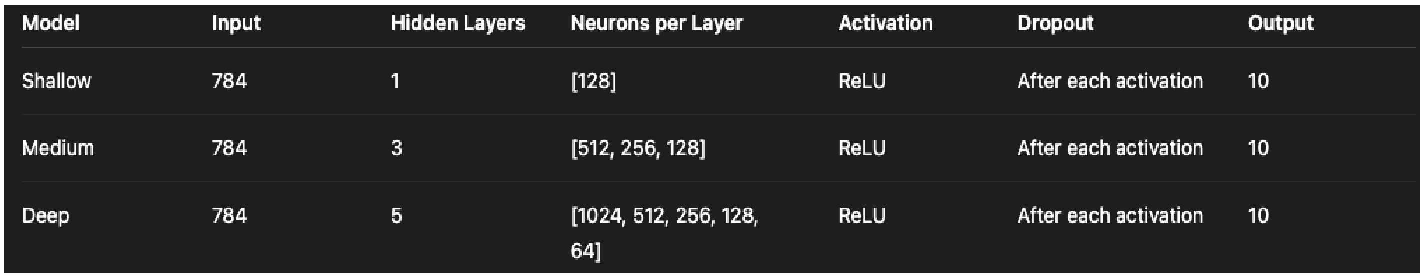
- To preprocess the datasets for MLP's we normalize pixel values to [0,1] and we flattened the image by using " transforms.Compose([transforms.ToTensor(), transforms. Lambda(lambda x: x.view(-1))])". Here the transforms.ToTensor() is used to normalize pixel values to [0,1] while transforms.Lambda(lambda x: x.view(-1))] flattens the image.
- To preprocess the dataset for CNN's we only normalize pixel values to [0,1] using the transforms.Compose([transforms.ToTensor()]). We do not flatten the images in CNN because CNN relies on the spatial relationships between pixels.

For the MNIST dataset:
After we set the transformations, we then downloaded the training dataset by applying the transformation. After downloading it, we divide the training dataset as requested to be 50,000 training samples and 10,000 validation samples for MNIST. We also applied the same transformation when we were downloading the test dataset.

For the CIFAR-10 dataset:
After we set the transformations, we then downloaded the training dataset by applying the transformation. After downloading it, we divide the training dataset as requested to be 45,000 training samples and 5,000 validation samples for CIFAR-10. We also applied the same transformation when we were downloading the test dataset.

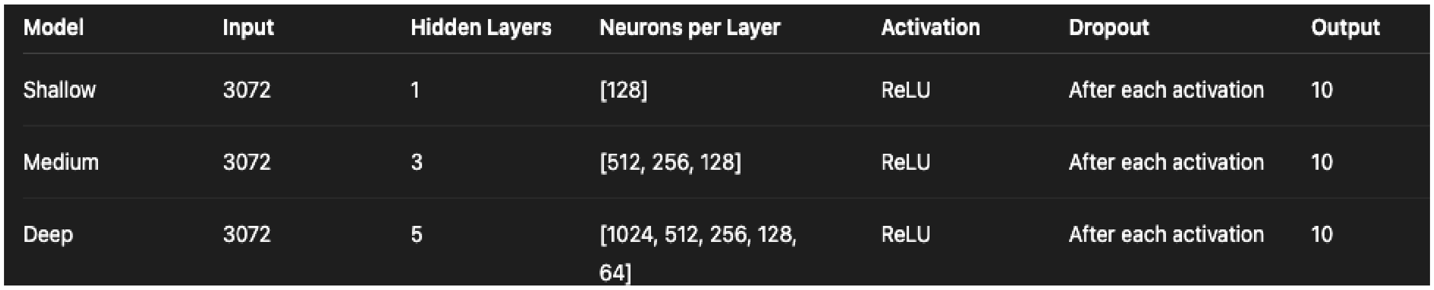
Detailed architectures tested

MLP MNIST ARCHITECTURE:



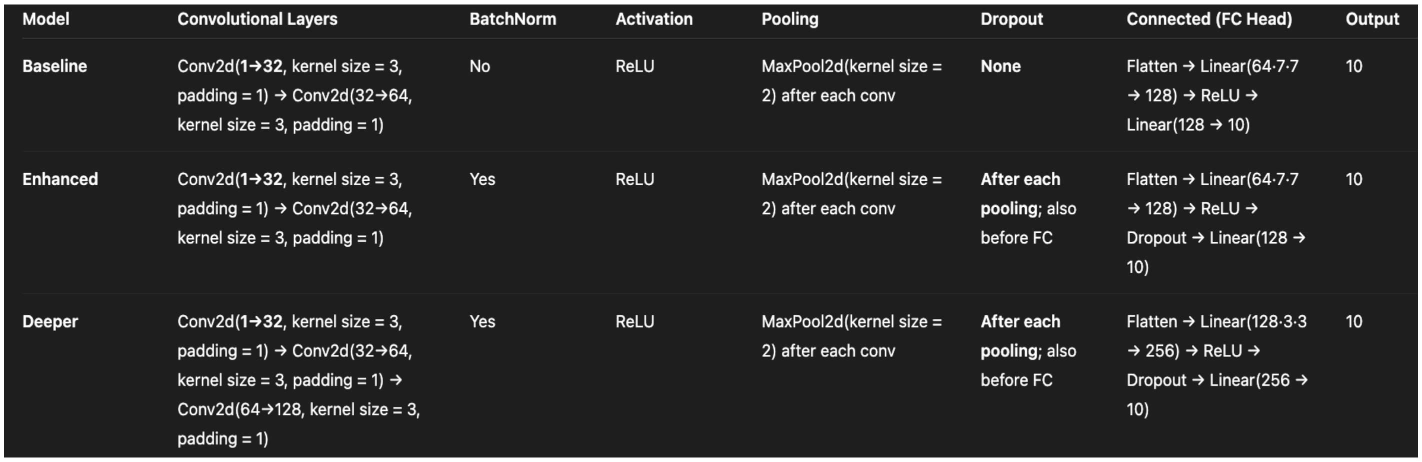
Model	Input	Hidden Layers	Neurons per Layer	Activation	Dropout	Output
Shallow	784	1	[128]	ReLU	After each activation	10
Medium	784	3	[512, 256, 128]	ReLU	After each activation	10
Deep	784	5	[1024, 512, 256, 128, 64]	ReLU	After each activation	10

MLP CIFAR-10 ARCHITECTURE:



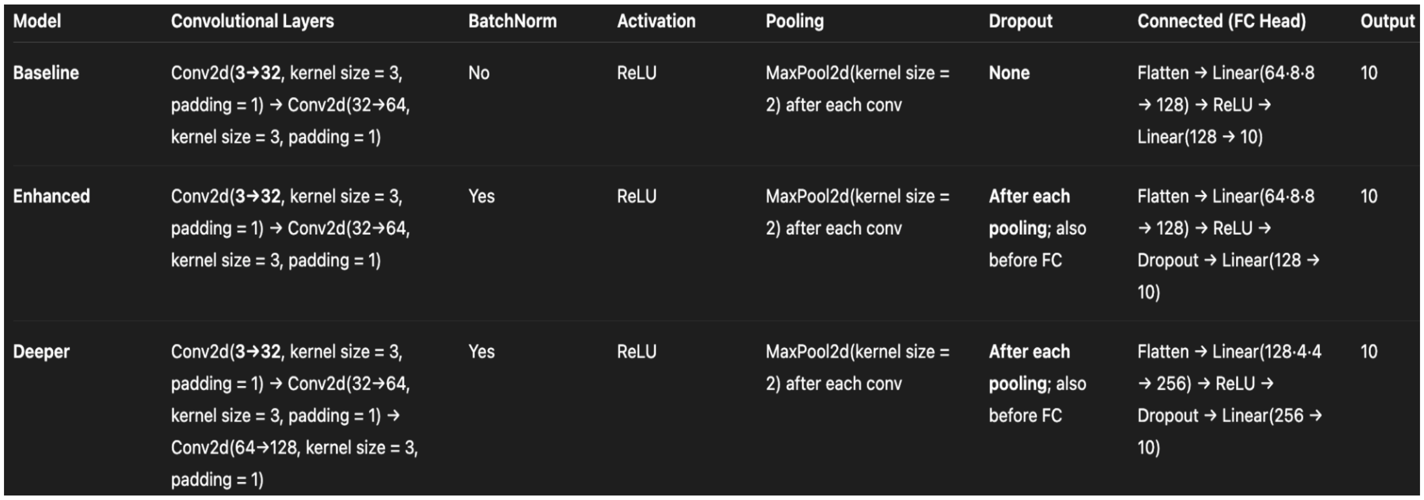
Model	Input	Hidden Layers	Neurons per Layer	Activation	Dropout	Output
Shallow	3072	1	[128]	ReLU	After each activation	10
Medium	3072	3	[512, 256, 128]	ReLU	After each activation	10
Deep	3072	5	[1024, 512, 256, 128, 64]	ReLU	After each activation	10

CNN MNIST ARCHITECTURE:



Model	Convolutional Layers	BatchNorm	Activation	Pooling	Dropout	Connected (FC Head)	Output
Baseline	Conv2d(**1→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1)	No	ReLU	MaxPool2d(kernel size = 2) after each conv	**None**	Flatten → Linear(64·7·7 → 128) → ReLU → Linear(128 → 10)	10
Enhanced	Conv2d(**1→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1)	Yes	ReLU	MaxPool2d(kernel size = 2) after each conv	**After each pooling**; also before FC	Flatten → Linear(64·7·7 → 128) → ReLU → Dropout → Linear(128 → 10)	10
Deeper	Conv2d(**1→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1) → Conv2d(64→128, kernel size = 3, padding = 1)	Yes	ReLU	MaxPool2d(kernel size = 2) after each conv	**After each pooling**; also before FC	Flatten → Linear(128·3·3 → 256) → ReLU → Dropout → Linear(256 → 10)	10

CNN CIFAR-10 ARCHITECTURE:

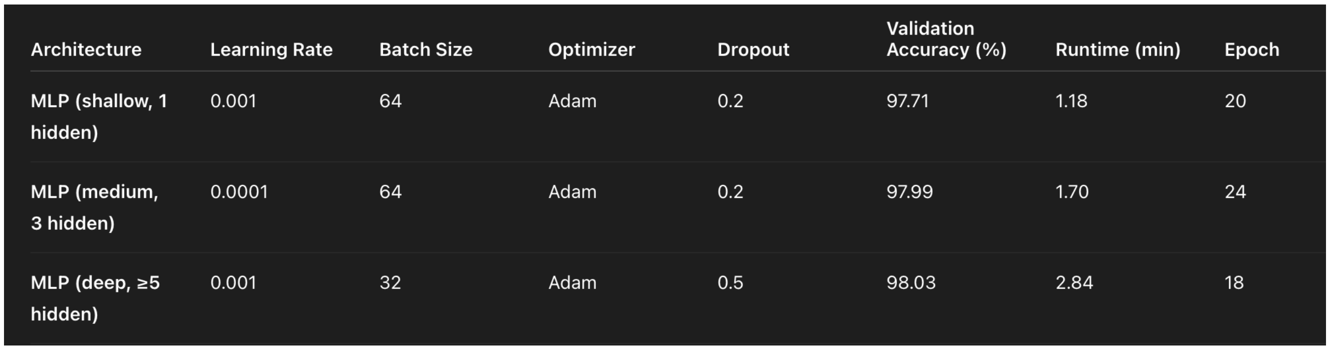


Model	Convolutional Layers	BatchNorm	Activation	Pooling	Dropout	Connected (FC Head)	Output
Baseline	Conv2d(**3→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1)	No	ReLU	MaxPool2d(kernel size = 2) after each conv	**None**	Flatten → Linear(64·8·8 → 128) → ReLU → Linear(128 → 10)	10
Enhanced	Conv2d(**3→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1)	Yes	ReLU	MaxPool2d(kernel size = 2) after each conv	**After each pooling**; also before FC	Flatten → Linear(64·8·8 → 128) → ReLU → Dropout → Linear(128 → 10)	10
Deeper	Conv2d(**3→32**, kernel size = 3, padding = 1) → Conv2d(32→64, kernel size = 3, padding = 1) → Conv2d(64→128, kernel size = 3, padding = 1)	Yes	ReLU	MaxPool2d(kernel size = 2) after each conv	**After each pooling**; also before FC	Flatten → Linear(128·4·4 → 256) → ReLU → Dropout → Linear(256 → 10)	10

RESULTS

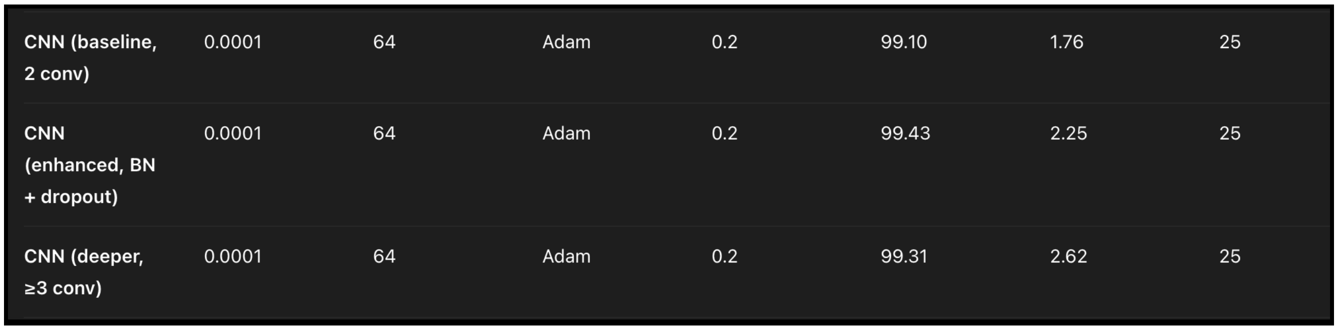
Note: I set the random seed to 40, so if you run the code, you should also get the same results.

Table 1: MIST Results



Architecture	Learning Rate	Batch Size	Optimizer	Dropout	Validation Accuracy (%)	Runtime (min)	Epoch
MLP (shallow, 1 hidden)	0.001	64	Adam	0.2	97.71	1.18	20
MLP (medium, 3 hidden)	0.0001	64	Adam	0.2	97.99	1.70	24
MLP (deep, ≥5 hidden)	0.001	32	Adam	0.5	98.03	2.84	18

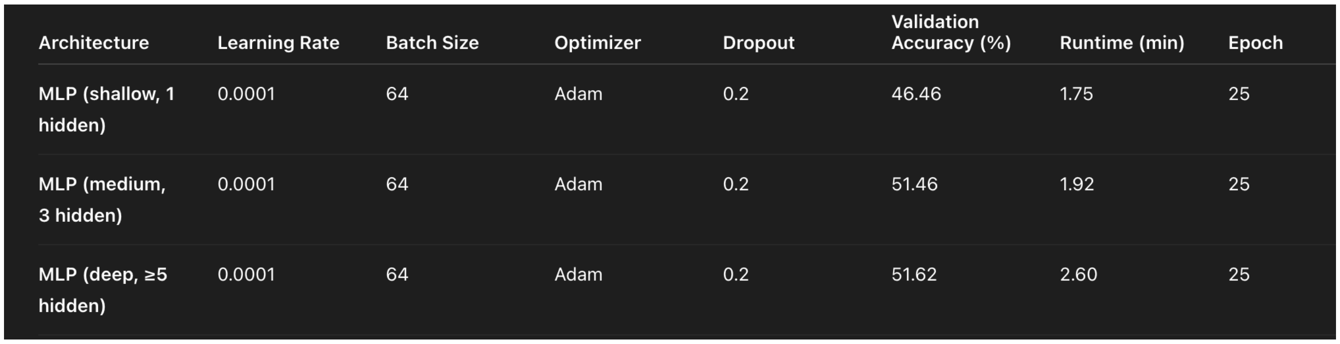
Test accuracy on final MLP model: 98.17%



CNN (baseline, 2 conv)	0.0001	64	Adam	0.2	99.10	1.76	25
CNN (enhanced, BN + dropout)	0.0001	64	Adam	0.2	99.43	2.25	25
CNN (deeper, ≥3 conv)	0.0001	64	Adam	0.2	99.31	2.62	25

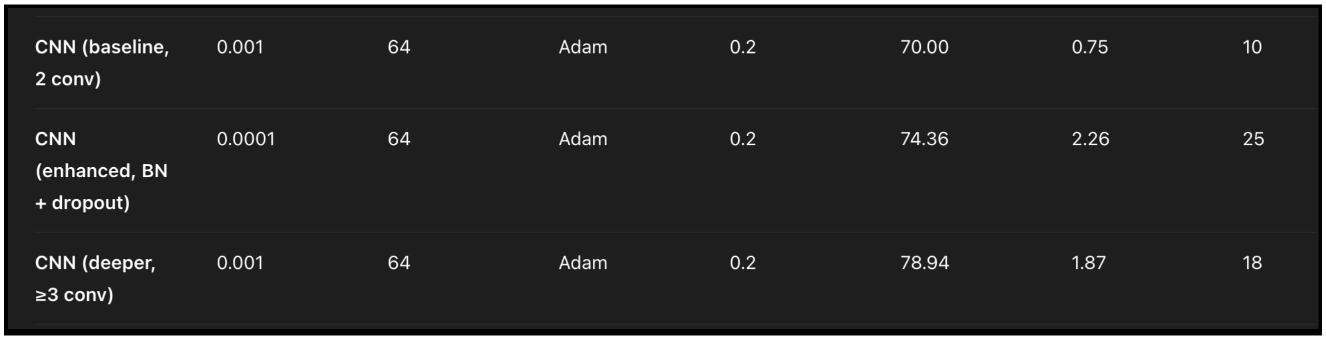
Test accuracy on final CNN model: 99.34%

Table 2: CIFAR-10 Results



Architecture	Learning Rate	Batch Size	Optimizer	Dropout	Validation Accuracy (%)	Runtime (min)	Epoch
MLP (shallow, 1 hidden)	0.0001	64	Adam	0.2	46.46	1.75	25
MLP (medium, 3 hidden)	0.0001	64	Adam	0.2	51.46	1.92	25
MLP (deep, ≥5 hidden)	0.0001	64	Adam	0.2	51.62	2.60	25

Test accuracy on final MLP model: 53.00%



CNN (baseline, 2 conv)	0.001	64	Adam	0.2	70.00	0.75	10
CNN (enhanced, BN + dropout)	0.0001	64	Adam	0.2	74.36	2.26	25
CNN (deeper, ≥3 conv)	0.001	64	Adam	0.2	78.94	1.87	18

Test accuracy on final CNN model: 81.10%

- To find the test accuracies for the final models, we looked at the validation accuracies each model generated with their best hyperparameters and chose the model that produced the best validation accuracy as our final model, and got the Test results for that model as our final test model.

Why certain hyperparameters/architectures performed better (supported by table results)

MLPs:
In both datasets for MLPs, the deeper architectures achieved the highest validation accuracies.
- MNIST: shallow(97.71%) -> medium(97.99%) -> deep(98.03)
- CIFAR-10: shallow(46.46%) -> medium(51.46%) -> deep(51.62)

This is due to, more depth actually allowed the model to learn more abstract and non-linear feature representations. The dropout helped deeper architectures to retain generalization and avoid overfitting. These resulted in deeper architectures to get better validation accuracies in MLPs.

CNNs:
From the results we obtained, it is clear that CNNs outperform MLPs.
- MNIST: Final model accuracy improves from 98.17%(MLP) to 99.34%(CNN)
- CIFAR-10: Final model accuracy improves from 53%(MLP) to 81.10%(CNN)

MNIST (CNN):
We can see that the final model improvement on the MNIST dataset is not that much(+1.3%), but this is due to the dataset is near the saturation-accuracy.

- On baseline architecture, we can see that the accuracy(99.10%) is already higher than what we were getting in any of the MLP models, which highlights the benefit of even shallow convolutions.

- The enhanced architecture gives the best overall accuracy(99.43%), which means that adding batch normalization stabilized the training process and made it converge faster, and adding dropout helped reduce overfitting.
- The deeper architecture accuracy(99.31%) was slightly lower than the enhanced architecture accuracy. This is due to the fact that the MNIST dataset was not complex enough to benefit the deeper CNN model.

CIFAR-10 (CNN):
The model showed a substantial improvement (+28.1%) on CIFAR-10. This is because CIFAR-10 is a much more complex dataset than MNIST, containing color images with diverse textures and objects. MLPs flatten the input images, which causes them to lose spatial information and prevents them from effectively capturing patterns and relationships between neighboring pixels. In contrast, CNNs preserve spatial structure through convolutional layers, allowing them to learn local patterns and spatial hierarchies—resulting in much better performance on the CIFAR-10 dataset.

- On the baseline architecture, the accuracy was 70%, which is way more than any of the MLP model accuracies. This clearly indicates that even the basic CNN model was doing a way better job than MLPs at capturing spatial relationships between pixels.
- On the enhanced architecture, the accuracy increased to 74.36% which shows that the addition of batch normalization and dropout was again helpful to increase the accuracy for the same reasons we mentioned on the MNIST dataset
- On the deeper architecture, we can see that we get the best accuracy out of all the CNNs, 78.94. This is because increasing the depth helped the model to learn more shapes, lines, and edges etc. In the CIFAR-10 dataset, we can clearly see that the model is definitely complex enough to take advantage of a deeper model, unlike the MNIST dataset, which was not complex enough

Hyperparameter Selections:
In this project, 12 random hyperparameter combinations were tested from the predefined grid. The consistently optimal set across most models was:
{Learning rate = 0.0001, Batch size = 64, Optimizer = Adam, Dropout = 0.2}

Note: You can see the hyperparameters chosen for this specific seed by looking at the pictures I attached at the very end of the report

- Learning rate(0.0001): Small learning rates result in smoother convergence, and they avoid divergence. Although a smaller learning rate causes longer training time, it usually gives better accuracy.
- Batch size(64): It is the sweet spot between gradient stability and the training speed. A small batch size, like 32, introduces more noise, and a large batch size, like 128, introduces poor generalization and slow convergence rates. So it is understandable why batch size 64 was the best overall choice
- Optimizer(ADAM): This result is expected, as Adam often performs better in practice due to its ability to automatically adjust the learning rate for each parameter during training.

By computing individual adaptive learning rates, Adam typically converges faster and achieves better overall performance compared to SGD, which uses a single global learning rate for all parameters

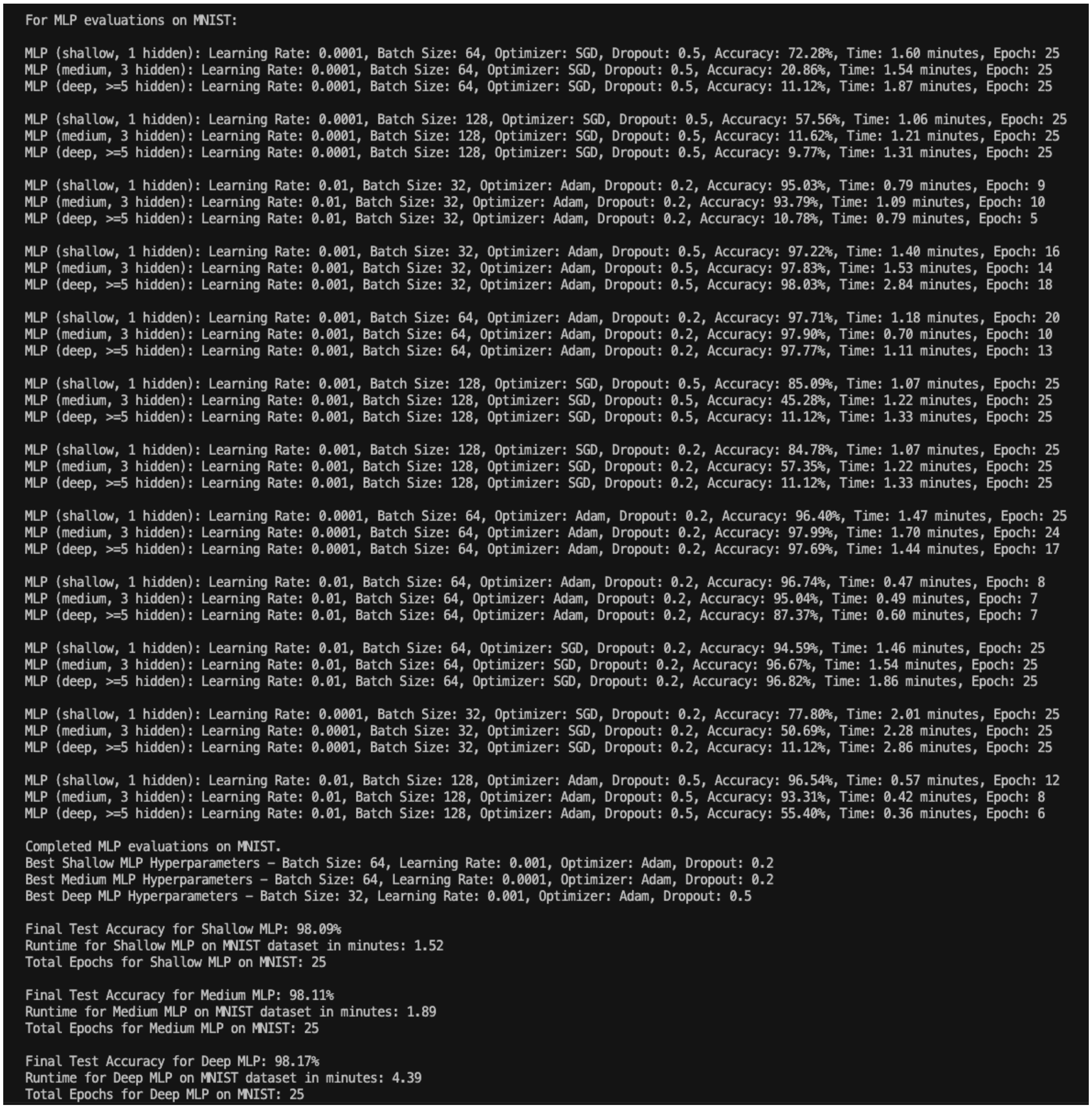
- Dropout(0.2): This is a good dropout rate because it introduces regularization without suppressing learning. Dropout: 0.5, on the other hand, is a more aggressive regularization, which clearly interrupts the learning phase. Dropout: 0.2 was a good enough regularization to prevent overfitting in the end

While there are some best hyperparameters for models that do not fit perfectly with hyperparameters {Learning rate: 0.0001, batch_size: 64, optimizer: ADAM, dropout: 0.2}, for example, baseline CNN CIFAR-10 learning rate is 0.001 instead of 0.0001, etc. The general trend shows that smaller learning rates, balanced batch size, moderate dropout, and Adam optimization consistently yielded stable and high-performing models.

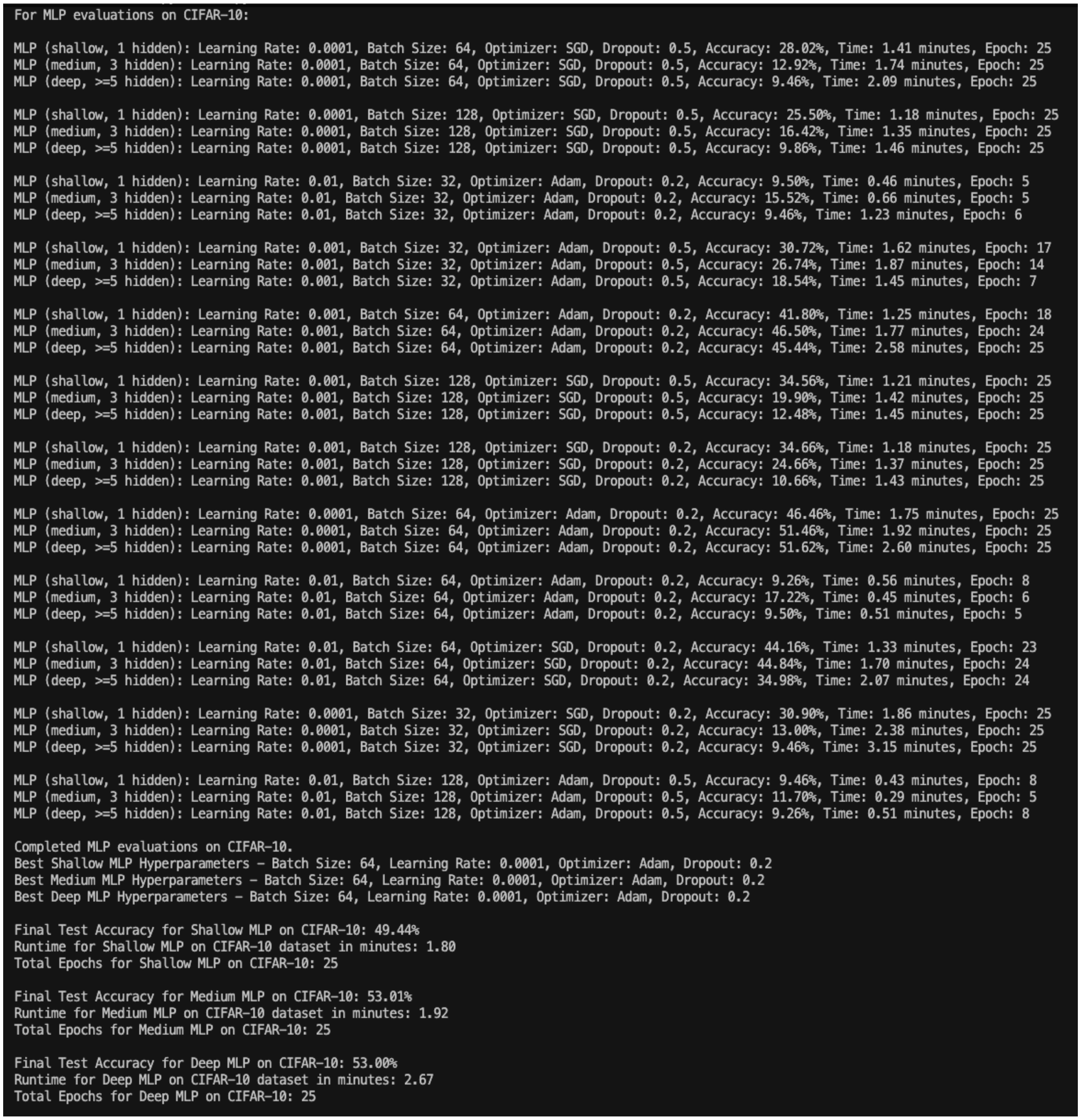
Describe one key challenge you faced and how you resolved it

In this project, I have encountered many challenges, but one of the most important challenges I encountered was the long computational runtime. I started the project using my own laptop's CPU (M1 Pro) and finished the whole project while using the CPU, but I realized it was taking too much time, so I tried to make it faster by following the instructions that were given to us, so I changed my environment to Google Colab GPU, and I change the code so it will put every computation in the GPU memory so it would compile faster however upon making those changes I realized, it was not compiling faster but it actually got slower. First, I thought my implementation of the code to use the GPU memory was not correct, but upon debugging, I realized I was not making a mistake in the implementation phase, so I was not sure why it was taking a longer time to compile on Colab GPU. As a last resort, I also wanted to try on my personal machine's GPU to see if it would compile faster, so I implemented it there, and it worked without an issue. I also made sure it was using the GPU by going into the "Activity Monitor" in my MacBook, and it was utilizing the GPU, so the problem was fixed. I talked about this problem with the professor during office hours, and she told me that many other students were also facing the same issue with Google Colab GPU, and the reason was unclear. After I was done with the GPU implementation, I then wanted to make this run even faster, so I added early stopping in the tuning phase. I set my patience as 4, and I checked the validation loss of that particular epoch run, and if the validation loss increased 4 times in a row, then I stopped, since that meant validation accuracy was no longer improving. By making these changes, my computation time reduced dramatically, so the challenge was resolved.

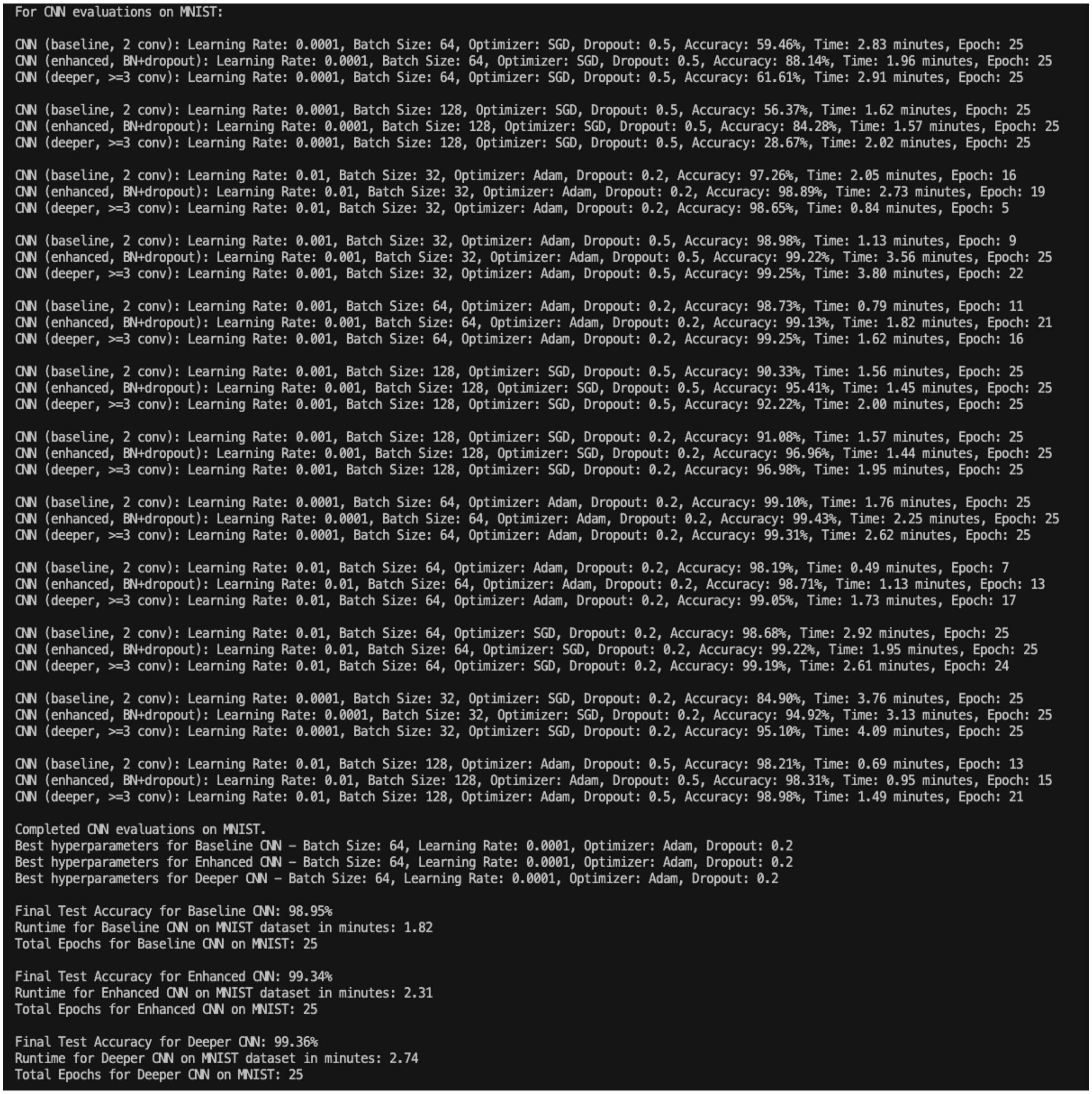
All the Results after running the code(includes 12 hyperparameter selection/tuning and the test accuracies for each architecture using the best hyperparameter):



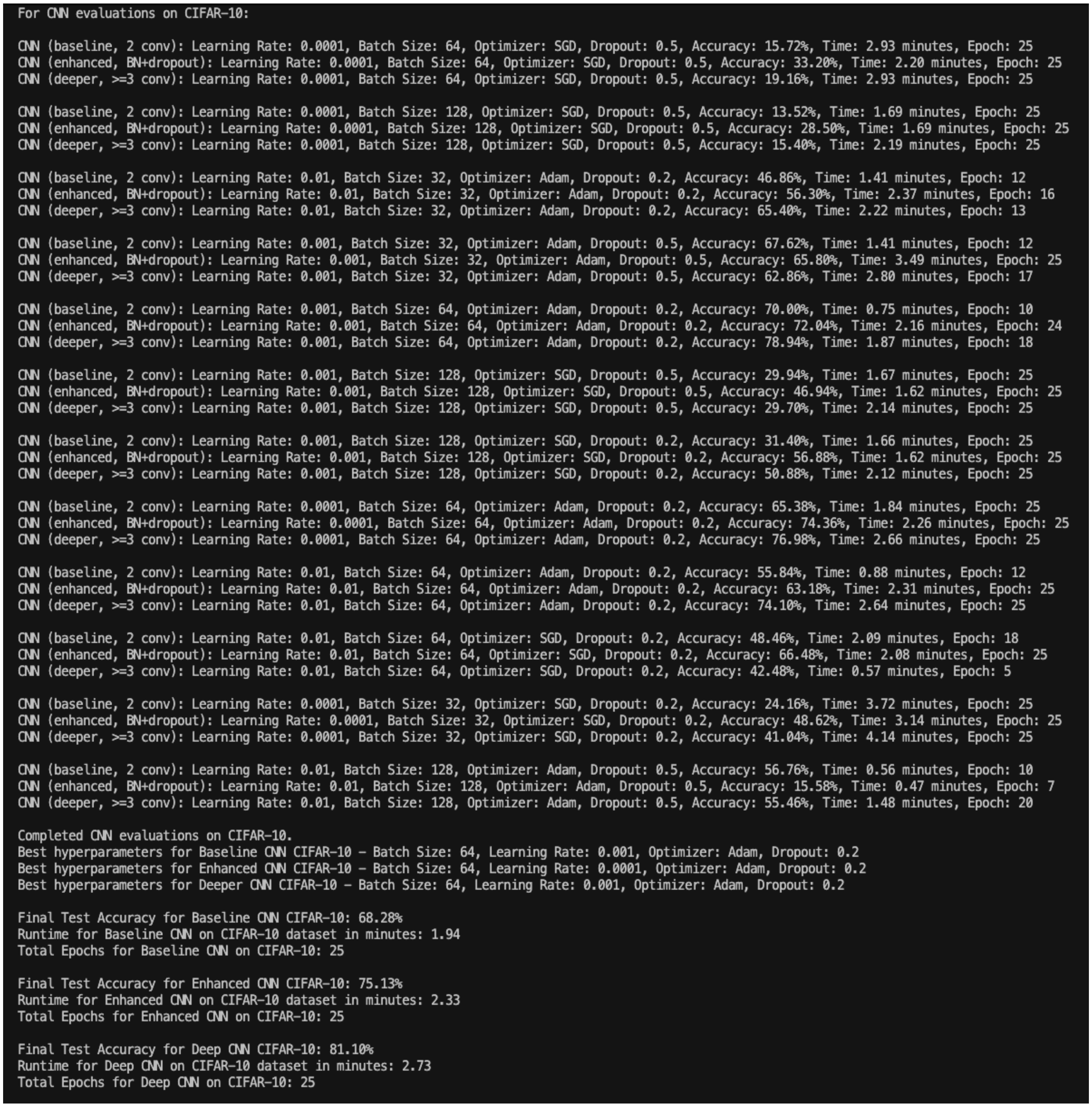
```
For MLP evaluations on MNIST:

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 72.28%, Time: 1.60 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 20.86%, Time: 1.54 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 11.12%, Time: 1.87 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 57.56%, Time: 1.06 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 11.62%, Time: 1.21 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 9.77%, Time: 1.31 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 95.03%, Time: 0.79 minutes, Epoch: 9
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 93.79%, Time: 1.09 minutes, Epoch: 10
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 10.78%, Time: 0.79 minutes, Epoch: 5

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 97.22%, Time: 1.40 minutes, Epoch: 16
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 97.83%, Time: 1.53 minutes, Epoch: 14
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 98.03%, Time: 2.84 minutes, Epoch: 18

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.71%, Time: 1.18 minutes, Epoch: 20
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.90%, Time: 0.70 minutes, Epoch: 10
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.77%, Time: 1.11 minutes, Epoch: 13

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 85.09%, Time: 1.07 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 45.28%, Time: 1.22 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 11.12%, Time: 1.33 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 84.78%, Time: 1.07 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 57.35%, Time: 1.22 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 11.12%, Time: 1.33 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 96.40%, Time: 1.47 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.99%, Time: 1.70 minutes, Epoch: 24
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.69%, Time: 1.44 minutes, Epoch: 17

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 96.74%, Time: 0.47 minutes, Epoch: 8
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 95.04%, Time: 0.49 minutes, Epoch: 7
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 87.37%, Time: 0.60 minutes, Epoch: 7

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 94.59%, Time: 1.46 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 96.67%, Time: 1.54 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 96.82%, Time: 1.86 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 77.80%, Time: 2.01 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 50.69%, Time: 2.28 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 11.12%, Time: 2.86 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 96.54%, Time: 0.57 minutes, Epoch: 12
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 93.31%, Time: 0.42 minutes, Epoch: 8
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 55.40%, Time: 0.36 minutes, Epoch: 6

Completed MLP evaluations on MNIST.
Best Shallow MLP Hyperparameters - Batch Size: 64, Learning Rate: 0.001, Optimizer: Adam, Dropout: 0.2
Best Medium MLP Hyperparameters - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best Deep MLP Hyperparameters - Batch Size: 32, Learning Rate: 0.001, Optimizer: Adam, Dropout: 0.5

Final Test Accuracy for Shallow MLP: 98.09%
Runtime for Shallow MLP on MNIST dataset in minutes: 1.52
Total Epochs for Shallow MLP on MNIST: 25

Final Test Accuracy for Medium MLP: 98.11%
Runtime for Medium MLP on MNIST dataset in minutes: 1.89
Total Epochs for Medium MLP on MNIST: 25

Final Test Accuracy for Deep MLP: 98.17%
Runtime for Deep MLP on MNIST dataset in minutes: 4.39
Total Epochs for Deep MLP on MNIST: 25
```



```
For MLP evaluations on CIFAR-10:

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 28.02%, Time: 1.41 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 12.92%, Time: 1.74 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 9.46%, Time: 2.09 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 25.50%, Time: 1.18 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 16.42%, Time: 1.35 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 9.86%, Time: 1.46 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 9.50%, Time: 0.46 minutes, Epoch: 5
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 15.52%, Time: 0.66 minutes, Epoch: 5
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 9.46%, Time: 1.23 minutes, Epoch: 6

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 30.72%, Time: 1.62 minutes, Epoch: 17
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 26.74%, Time: 1.87 minutes, Epoch: 14
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 18.54%, Time: 1.45 minutes, Epoch: 7

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 41.80%, Time: 1.25 minutes, Epoch: 18
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 46.50%, Time: 1.77 minutes, Epoch: 24
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 45.44%, Time: 2.58 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 34.56%, Time: 1.21 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 19.90%, Time: 1.42 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 12.48%, Time: 1.45 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 34.66%, Time: 1.18 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 24.66%, Time: 1.37 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 10.66%, Time: 1.43 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 46.46%, Time: 1.75 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 51.46%, Time: 1.92 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 51.62%, Time: 2.60 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 9.26%, Time: 0.56 minutes, Epoch: 8
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 17.22%, Time: 0.45 minutes, Epoch: 6
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 9.50%, Time: 0.51 minutes, Epoch: 5

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 44.16%, Time: 1.33 minutes, Epoch: 23
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 44.84%, Time: 1.70 minutes, Epoch: 24
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 34.98%, Time: 2.07 minutes, Epoch: 24

MLP (shallow, 1 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 30.90%, Time: 1.86 minutes, Epoch: 25
MLP (medium, 3 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 13.00%, Time: 2.38 minutes, Epoch: 25
MLP (deep, >=5 hidden): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 9.46%, Time: 3.15 minutes, Epoch: 25

MLP (shallow, 1 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 9.46%, Time: 0.43 minutes, Epoch: 8
MLP (medium, 3 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 11.70%, Time: 0.29 minutes, Epoch: 5
MLP (deep, >=5 hidden): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 9.26%, Time: 0.51 minutes, Epoch: 8

Completed MLP evaluations on CIFAR-10.
Best Shallow MLP Hyperparameters - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best Medium MLP Hyperparameters - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best Deep MLP Hyperparameters - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2

Final Test Accuracy for Shallow MLP on CIFAR-10: 49.44%
Runtime for Shallow MLP on CIFAR-10 dataset in minutes: 1.80
Total Epochs for Shallow MLP on CIFAR-10: 25

Final Test Accuracy for Medium MLP on CIFAR-10: 53.01%
Runtime for Medium MLP on CIFAR-10 dataset in minutes: 1.92
Total Epochs for Medium MLP on CIFAR-10: 25

Final Test Accuracy for Deep MLP on CIFAR-10: 53.00%
Runtime for Deep MLP on CIFAR-10 dataset in minutes: 2.67
Total Epochs for Deep MLP on CIFAR-10: 25
```



```
For CNN evaluations on MNIST:

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 59.46%, Time: 2.83 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 88.14%, Time: 1.96 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 61.61%, Time: 2.91 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 56.37%, Time: 1.62 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 84.28%, Time: 1.57 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 28.67%, Time: 2.02 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 97.26%, Time: 2.05 minutes, Epoch: 16
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 98.89%, Time: 2.73 minutes, Epoch: 19
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 98.65%, Time: 0.84 minutes, Epoch: 5

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 98.98%, Time: 1.13 minutes, Epoch: 9
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 99.22%, Time: 3.56 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 99.25%, Time: 3.80 minutes, Epoch: 22

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 98.73%, Time: 0.79 minutes, Epoch: 11
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.13%, Time: 1.82 minutes, Epoch: 21
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.25%, Time: 1.62 minutes, Epoch: 16

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 90.33%, Time: 1.56 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 95.41%, Time: 1.45 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 92.22%, Time: 2.00 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 91.08%, Time: 1.57 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 96.96%, Time: 1.44 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 96.98%, Time: 1.95 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.10%, Time: 1.76 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.43%, Time: 2.25 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.31%, Time: 2.62 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 98.19%, Time: 0.49 minutes, Epoch: 7
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 98.71%, Time: 1.13 minutes, Epoch: 13
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 99.05%, Time: 1.73 minutes, Epoch: 17

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 98.68%, Time: 2.92 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 99.22%, Time: 1.95 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 99.19%, Time: 2.61 minutes, Epoch: 24

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 84.90%, Time: 3.76 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 94.92%, Time: 3.13 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 95.10%, Time: 4.09 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 98.21%, Time: 0.69 minutes, Epoch: 13
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 98.31%, Time: 0.95 minutes, Epoch: 15
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 98.98%, Time: 1.49 minutes, Epoch: 21

Completed CNN evaluations on MNIST.
Best hyperparameters for Baseline CNN - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best hyperparameters for Enhanced CNN - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best hyperparameters for Deeper CNN - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2

Final Test Accuracy for Baseline CNN: 98.95%
Runtime for Baseline CNN on MNIST dataset in minutes: 1.82
Total Epochs for Baseline CNN on MNIST: 25

Final Test Accuracy for Enhanced CNN: 99.34%
Runtime for Enhanced CNN on MNIST dataset in minutes: 2.31
Total Epochs for Enhanced CNN on MNIST: 25

Final Test Accuracy for Deeper CNN: 99.36%
Runtime for Deeper CNN on MNIST dataset in minutes: 2.74
Total Epochs for Deeper CNN on MNIST: 25
```



```
For CNN evaluations on CIFAR-10:

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 15.72%, Time: 2.93 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 33.20%, Time: 2.20 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: SGD, Dropout: 0.5, Accuracy: 19.16%, Time: 2.93 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 13.52%, Time: 1.69 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 28.50%, Time: 1.69 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 15.40%, Time: 2.19 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 46.86%, Time: 1.41 minutes, Epoch: 12
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 56.30%, Time: 2.37 minutes, Epoch: 16
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 32, Optimizer: Adam, Dropout: 0.2, Accuracy: 65.40%, Time: 2.22 minutes, Epoch: 13

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 67.62%, Time: 1.41 minutes, Epoch: 12
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 65.80%, Time: 3.49 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 32, Optimizer: Adam, Dropout: 0.5, Accuracy: 62.86%, Time: 2.80 minutes, Epoch: 17

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 70.00%, Time: 0.75 minutes, Epoch: 10
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 72.04%, Time: 2.16 minutes, Epoch: 24
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 78.94%, Time: 1.87 minutes, Epoch: 18

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 29.94%, Time: 1.67 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 46.94%, Time: 1.62 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.5, Accuracy: 29.70%, Time: 2.14 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 31.40%, Time: 1.66 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 56.88%, Time: 1.62 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.001, Batch Size: 128, Optimizer: SGD, Dropout: 0.2, Accuracy: 50.88%, Time: 2.12 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 65.38%, Time: 1.84 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 74.36%, Time: 2.26 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 76.98%, Time: 2.66 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 55.84%, Time: 0.88 minutes, Epoch: 12
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 63.18%, Time: 2.31 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: Adam, Dropout: 0.2, Accuracy: 74.10%, Time: 2.64 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 48.46%, Time: 2.09 minutes, Epoch: 18
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 66.48%, Time: 2.08 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 64, Optimizer: SGD, Dropout: 0.2, Accuracy: 42.48%, Time: 0.57 minutes, Epoch: 5

CNN (baseline, 2 conv): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 24.16%, Time: 3.72 minutes, Epoch: 25
CNN (enhanced, BN+dropout): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 48.62%, Time: 3.14 minutes, Epoch: 25
CNN (deeper, >=3 conv): Learning Rate: 0.0001, Batch Size: 32, Optimizer: SGD, Dropout: 0.2, Accuracy: 41.04%, Time: 4.14 minutes, Epoch: 25

CNN (baseline, 2 conv): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 56.76%, Time: 0.56 minutes, Epoch: 10
CNN (enhanced, BN+dropout): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 15.58%, Time: 0.47 minutes, Epoch: 7
CNN (deeper, >=3 conv): Learning Rate: 0.01, Batch Size: 128, Optimizer: Adam, Dropout: 0.5, Accuracy: 55.46%, Time: 1.48 minutes, Epoch: 20

Completed CNN evaluations on CIFAR-10.
Best hyperparameters for Baseline CNN CIFAR-10 - Batch Size: 64, Learning Rate: 0.001, Optimizer: Adam, Dropout: 0.2
Best hyperparameters for Enhanced CNN CIFAR-10 - Batch Size: 64, Learning Rate: 0.0001, Optimizer: Adam, Dropout: 0.2
Best hyperparameters for Deeper CNN CIFAR-10 - Batch Size: 64, Learning Rate: 0.001, Optimizer: Adam, Dropout: 0.2

Final Test Accuracy for Baseline CNN CIFAR-10: 68.28%
Runtime for Baseline CNN on CIFAR-10 dataset in minutes: 1.94
Total Epochs for Baseline CNN on CIFAR-10: 25

Final Test Accuracy for Enhanced CNN CIFAR-10: 75.13%
Runtime for Enhanced CNN on CIFAR-10 dataset in minutes: 2.33
Total Epochs for Enhanced CNN on CIFAR-10: 25

Final Test Accuracy for Deep CNN CIFAR-10: 81.10%
Runtime for Deep CNN on CIFAR-10 dataset in minutes: 2.73
Total Epochs for Deep CNN on CIFAR-10: 25
```