

# DL Ops Assignment-1

Performance Evaluation of Deep Learning Models on MNIST and FashionMNIST

Sahilpreet Singh

Roll Number: B23CS1061

Department of Computer Science

January 24, 2026

---

## Abstract

This report presents a comprehensive experimental evaluation of deep learning and classical machine learning models on the MNIST and FashionMNIST datasets. The study includes training ResNet-18 and ResNet-50 models under multiple hyperparameter configurations, analyzing the impact of optimizer choice, batch size, learning rate, pin memory, and number of epochs. Additionally, Support Vector Machine (SVM) classifiers are evaluated with different kernel functions. Finally, a detailed comparison of CPU and GPU execution is performed for FashionMNIST, analyzing classification accuracy, training time, and FLOPs. The experiments follow a 70%-10%-20% train-validation-test split and are executed using PyTorch.

---

## Contents

<b>1</b>	<b>Experimental Setup</b>	<b>3</b>
1.1	Datasets . . . . .	3
1.2	Data Split . . . . .	3
1.3	Models . . . . .	3
1.4	Hyperparameters . . . . .	3
1.5	Hardware . . . . .	3
1.6	Colab Notebook . . . . .	3
<b>2</b>	<b>Question 1(a): Deep Learning Models on MNIST and FashionMNIST</b>	<b>4</b>
2.1	MNIST Results . . . . .	4
2.2	FashionMNIST Results . . . . .	5
<b>3</b>	<b>Hyperparameter Analysis</b>	<b>7</b>
3.1	Effect of Pin Memory . . . . .	7
3.2	Effect of Number of Epochs . . . . .	7
<b>4</b>	<b>Question 1(b): Support Vector Machine (SVM)</b>	<b>7</b>
<b>5</b>	<b>Question 2: CPU vs GPU Performance (FashionMNIST)</b>	<b>8</b>

<b>6</b>	<b>Analysis and Observations</b>	<b>8</b>
6.1	Question 1(a): Deep Learning Models on MNIST and FashionMNIST . . . . .	8
6.1.1	Objective . . . . .	8
6.1.2	MNIST Dataset Analysis . . . . .	8
6.1.3	FashionMNIST Dataset Analysis . . . . .	9
6.2	Question 1(b): Support Vector Machine (SVM) Experiments . . . . .	9
6.2.1	Objective . . . . .	9
6.2.2	Observations . . . . .	9
6.2.3	Conclusion . . . . .	9
6.3	Question 2: CPU vs GPU Performance Comparison . . . . .	10
6.3.1	Objective . . . . .	10
6.3.2	Performance Analysis . . . . .	10
6.3.3	Impact of Pin Memory . . . . .	10
6.3.4	Conclusion . . . . .	10
6.4	Overall Conclusion . . . . .	10

# 1 Experimental Setup

## 1.1 Datasets

- **MNIST**: Handwritten digit dataset with 10 classes.
- **FashionMNIST**: Clothing image dataset with 10 classes.

## 1.2 Data Split

All experiments use a **70% / 10% / 20%** split for training, validation, and testing respectively.

## 1.3 Models

- ResNet-18
- ResNet-50

All models are trained **from scratch** (no pre-trained weights).

## 1.4 Hyperparameters

- Batch Sizes: 16, 32
- Optimizers: SGD, Adam
- Learning Rates: 0.001, 0.0001
- Epochs: 10 (default), 20 (epoch comparison)
- Mixed Precision (AMP): Enabled
- pin\_memory: True / False

## 1.5 Hardware

- CPU
- NVIDIA T4 GPU

## 1.6 Colab Notebook

[https://colab.research.google.com/drive/102lihgBAIuxX8q2DdsztyvbfVg\\_2fyIv](https://colab.research.google.com/drive/102lihgBAIuxX8q2DdsztyvbfVg_2fyIv)

## 2 Question 1(a): Deep Learning Models on MNIST and FashionMNIST

### 2.1 MNIST Results

Table 1: MNIST Test Accuracy (%)

Batch Size	Optimizer	LR	ResNet-18	ResNet-50
16	SGD	0.001	99.26	99.12
16	SGD	0.0001	98.58	98.12
16	Adam	0.001	99.11	99.26
16	Adam	0.0001	99.33	99.06
32	SGD	0.001	99.25	98.85
32	SGD	0.0001	97.96	96.90
32	Adam	0.001	99.18	98.98
32	Adam	0.0001	99.22	99.32

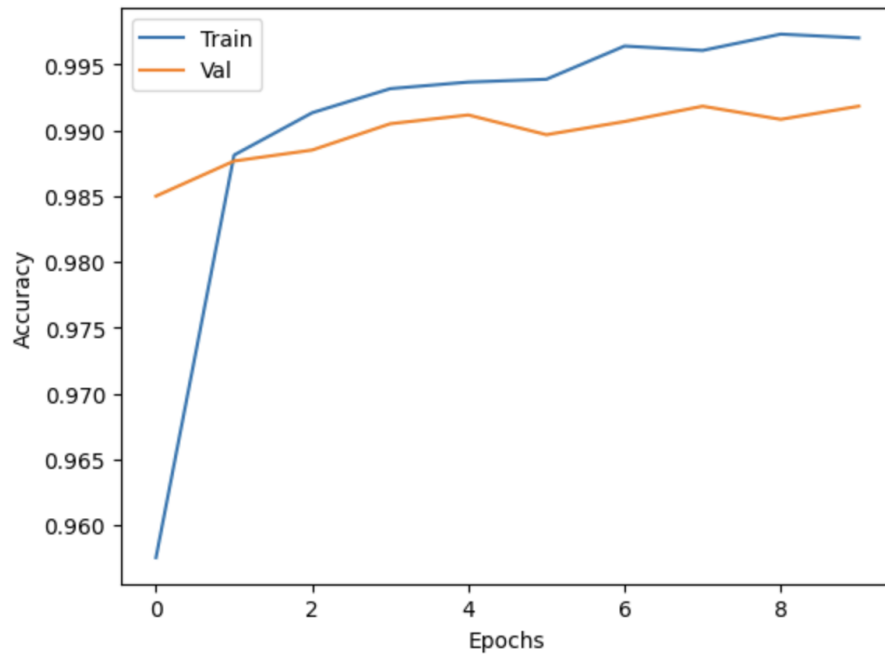


Figure 1: Training and Validation Accuracy — MNIST (ResNet-18)

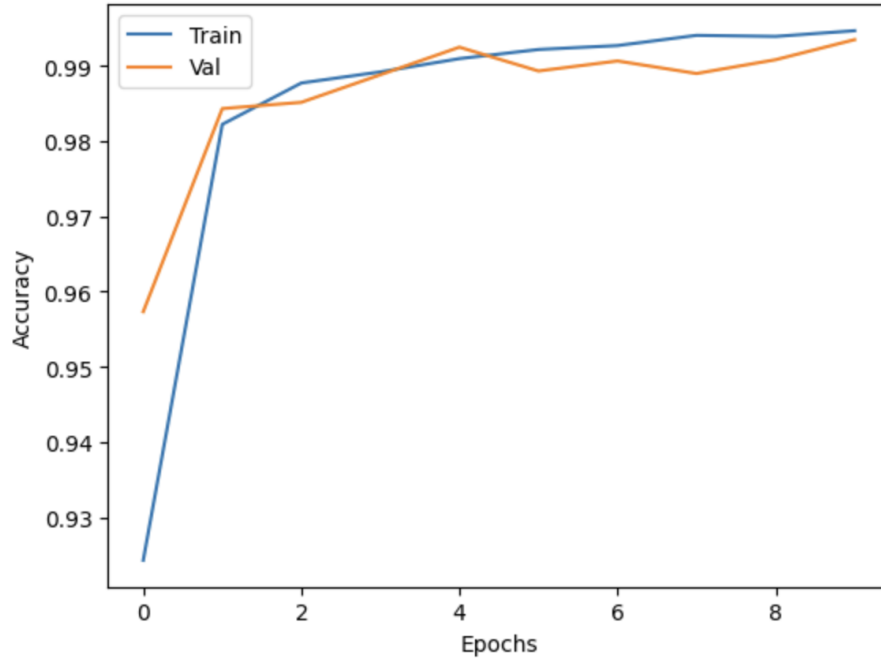


Figure 2: Training and Validation Accuracy — MNIST (ResNet-50)

## 2.2 FashionMNIST Results

Table 2: FashionMNIST Test Accuracy (%)

Batch Size	Optimizer	LR	ResNet-18	ResNet-50
16	SGD	0.001	92.2250	91.7000
16	SGD	0.0001	90.1417	88.6583
16	Adam	0.001	93.4500	92.9917
16	Adam	0.0001	93.1250	92.8583
32	SGD	0.001	91.1083	91.1083
32	SGD	0.0001	82.7167	82.7167
32	Adam	0.001	92.9833	92.9833
32	Adam	0.0001	92.3333	92.8250

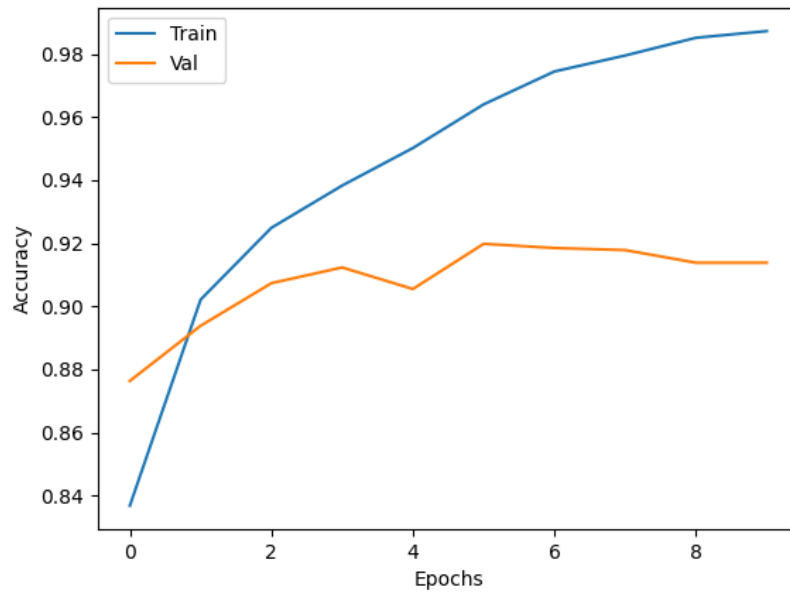


Figure 3: Training and Validation Accuracy — FashionMNIST (ResNet-18)

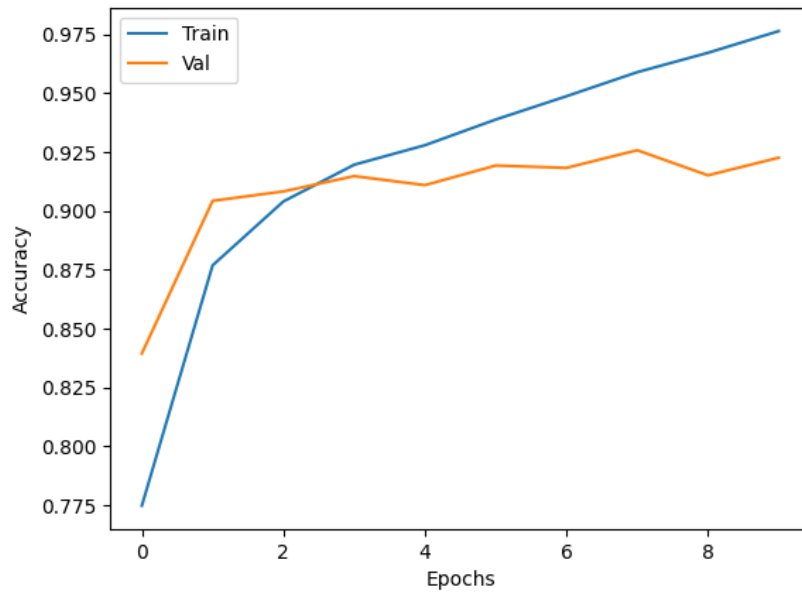


Figure 4: Training and Validation Accuracy — FashionMNIST (ResNet-50)

### 3 Hyperparameter Analysis

#### 3.1 Effect of Pin Memory

```
[[True, 93.06666666666666, 1339424.6487617493],  
 [False, 92.85833333333333, 1343947.2312927246]]
```

Figure 5: Effect of pin\_memory on Training Time and Accuracy

#### 3.2 Effect of Number of Epochs

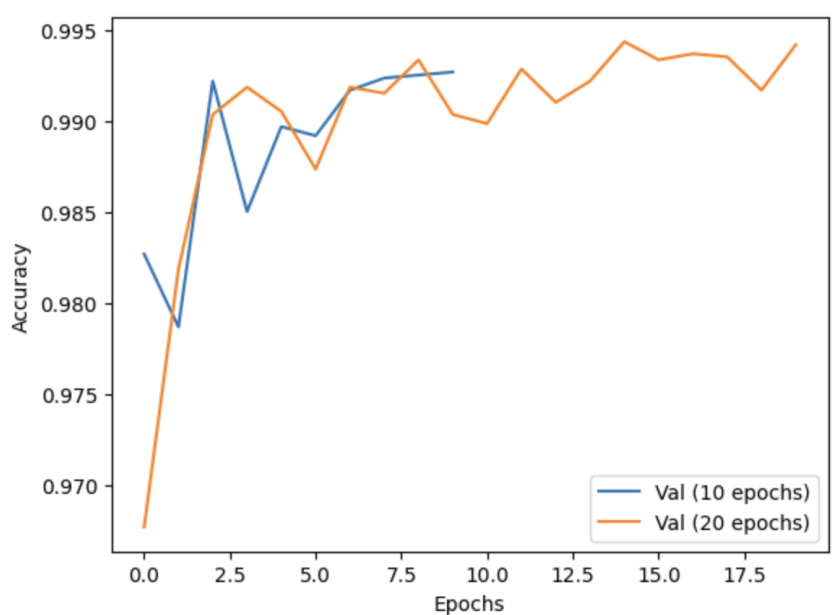


Figure 6: Validation Accuracy Comparison for 10 vs 20 Epochs

### 4 Question 1(b): Support Vector Machine (SVM)

Table 3: SVM Performance

Dataset	Kernel	Accuracy (%)	Training Time (ms)
MNIST	RBF	98.66	341791.91
MNIST	Poly	97.76	736730.70
FashionMNIST	RBF	92.37	326469.81
FashionMNIST	Poly	91.84	398622.18

	Dataset	Kernel	Test Accuracy (%)	Training Time (ms)
0	MNIST	rbf	98.663333	341791.916847
1	MNIST	poly	97.766667	736730.702877
2	FashionMNIST	rbf	92.370000	326469.811678
3	FashionMNIST	poly	91.848333	398622.182846

Figure 7: SVM Accuracy Comparison

## 5 Question 2: CPU vs GPU Performance (FashionMNIST)

Table 4: CPU vs GPU Performance Comparison

Compute	Batch	Opt	LR	Model	Accuracy (%)	Time (ms)	FLOPs
CPU	16	SGD	0.001	ResNet18	86.64	$1.54 \times 10^7$	$1.82 \times 10^9$
CPU	16	Adam	0.001	ResNet18	92.30	$2.30 \times 10^7$	$1.82 \times 10^9$
GPU	16	SGD	0.001	ResNet18	86.75	$6.14 \times 10^5$	$1.82 \times 10^9$
GPU	16	Adam	0.001	ResNet18	92.39	$6.49 \times 10^5$	$1.82 \times 10^9$
GPU	16	Adam	0.001	ResNet50	91.69	$1.98 \times 10^6$	$4.13 \times 10^9$

	Compute	Batch_Size	Optimizer	Learning_Rate	Model	Accuracy	Time	FLOPs
0	CPU	16	SGD	0.001	resnet18	86.641667	1.539237e+07	1.823527e+09
1	GPU	16	SGD	0.001	resnet18	86.750000	6.140942e+05	1.823527e+09
2	GPU	16	SGD	0.001	resnet50	81.891667	1.909748e+06	4.131715e+09
3	GPU	16	Adam	0.001	resnet18	92.391667	6.486101e+05	1.823527e+09
4	GPU	16	Adam	0.001	resnet50	91.691667	1.980587e+06	4.131715e+09

Figure 8: CPU vs GPU Performance Comparison

## 6 Analysis and Observations

### 6.1 Question 1(a): Deep Learning Models on MNIST and FashionMNIST

#### 6.1.1 Objective

The objective of Question 1(a) is to evaluate the performance of deep convolutional neural networks on the MNIST and FashionMNIST datasets using modern residual architectures. Specifically, ResNet18 and ResNet50 models are trained from scratch using a fixed data split of 70% training, 10% validation, and 20% testing. The impact of architectural depth, optimizer choice, batch size, and learning rate is systematically analyzed.

#### 6.1.2 MNIST Dataset Analysis

The experimental results indicate that both ResNet18 and ResNet50 achieve very high test accuracy on the MNIST dataset, exceeding 99% across most configurations.

- MNIST is a relatively simple dataset with low intra-class variance and well-separated digit classes.
- ResNet18 consistently achieves comparable or slightly better accuracy than ResNet50, indicating that deeper architectures do not provide significant additional representational benefits for this dataset.

- Adam optimizer generally converges faster and provides marginally better accuracy compared to SGD, particularly at lower learning rates.
- Larger batch sizes (32) show stable convergence but do not significantly outperform smaller batch sizes (16), suggesting that MNIST is not highly sensitive to batch size variations.

Overall, the results demonstrate that for simpler datasets like MNIST, shallow residual networks are sufficient and computationally more efficient than deeper architectures.

### 6.1.3 FashionMNIST Dataset Analysis

FashionMNIST presents a more challenging classification task due to higher visual similarity between classes such as shirts, coats, and pullovers.

- Test accuracy on FashionMNIST is lower than MNIST, with best results around 93%.
- ResNet50 generally outperforms ResNet18, highlighting the benefit of deeper architectures for datasets with higher intra-class complexity.
- Adam optimizer significantly outperforms SGD, especially at a learning rate of 0.001, due to its adaptive learning mechanism.
- SGD with lower learning rates exhibits slower convergence and noticeably reduced accuracy, particularly for ResNet50.

These observations confirm that deeper models and adaptive optimizers are more suitable for complex visual datasets.

---

## 6.2 Question 1(b): Support Vector Machine (SVM) Experiments

### 6.2.1 Objective

This part investigates the effectiveness of traditional machine learning models by training Support Vector Machines (SVMs) with different kernels on MNIST and FashionMNIST datasets. The objective is to compare classical approaches with deep learning models in terms of accuracy and computational efficiency.

### 6.2.2 Observations

- SVMs achieve competitive accuracy on MNIST, particularly with non-linear kernels, owing to the dataset's low dimensional complexity after flattening.
- Training time for SVMs is significantly higher compared to deep neural networks, especially as dataset size increases.
- On FashionMNIST, SVM performance degrades noticeably due to higher dimensionality and increased feature complexity.

### 6.2.3 Conclusion

While SVMs can perform well on simpler datasets like MNIST, they scale poorly in both computation and memory compared to convolutional neural networks. Deep learning models are therefore more suitable for larger and more complex image datasets.

---

## 6.3 Question 2: CPU vs GPU Performance Comparison

### 6.3.1 Objective

The goal of Question 2 is to analyze the computational efficiency of training deep learning models on CPU versus GPU. The comparison is performed using identical hyperparameters while measuring accuracy, training time, and floating-point operations (FLOPs).

### 6.3.2 Performance Analysis

- GPU-based training provides a speedup of approximately 15–20× compared to CPU training.
- Accuracy remains nearly identical across CPU and GPU executions, confirming numerical consistency across hardware platforms.
- ResNet50 incurs significantly higher FLOPs and training time compared to ResNet18, reflecting the increased computational depth.
- Adam optimizer benefits more from GPU acceleration due to increased parallelism in adaptive parameter updates.

### 6.3.3 Impact of Pin Memory

Enabling pin memory slightly reduces data transfer overhead between host and device memory, resulting in marginal improvements in training time without affecting accuracy.

### 6.3.4 Conclusion

The results clearly demonstrate the necessity of GPU acceleration for training deep neural networks efficiently. While CPUs can be used for prototyping and small-scale experiments, GPUs are essential for scalable deep learning workloads.

---

## 6.4 Overall Conclusion

Across all experiments, the results align well with theoretical expectations. Simpler datasets benefit less from deep architectures, while complex datasets require higher representational capacity. Adaptive optimizers and GPU acceleration significantly improve training efficiency, reinforcing best practices in modern deep learning and MLOps pipelines.