DLOps 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

Project Links

- **GitHub Pages (Results and Visualizations):**
  `https://sps1001.github.io/MLOps-Sahilpreet_Singh-B23CS1061/`
- **GitHub Repository (Assignment-1 Branch):**
  `https://github.com/sps1001/MLOps-Sahilpreet_Singh-B23CS1061/tree/Assignment-1`
- **Google Colab Notebook (Executed Experiments):**
  `https://colab.research.google.com/drive/102lihgBAIuxX8q2DdsztyvbfVg_2fyIv`

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`

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

2.1 MNIST Results

	Dataset	Batch Size	Optimizer	LR	Model	Test Accuracy (%)
0	MNIST	16	SGD	0.0010	ResNet18	99.258333
1	MNIST	16	SGD	0.0001	ResNet18	98.583333
2	MNIST	16	Adam	0.0010	ResNet18	99.108333
3	MNIST	16	Adam	0.0001	ResNet18	99.333333
4	MNIST	32	SGD	0.0010	ResNet18	99.250000
5	MNIST	32	SGD	0.0001	ResNet18	97.958333
6	MNIST	32	Adam	0.0010	ResNet18	99.183333
7	MNIST	32	Adam	0.0001	ResNet18	99.216667

Figure 1: MNIST Test Accuracy Results for ResNet18

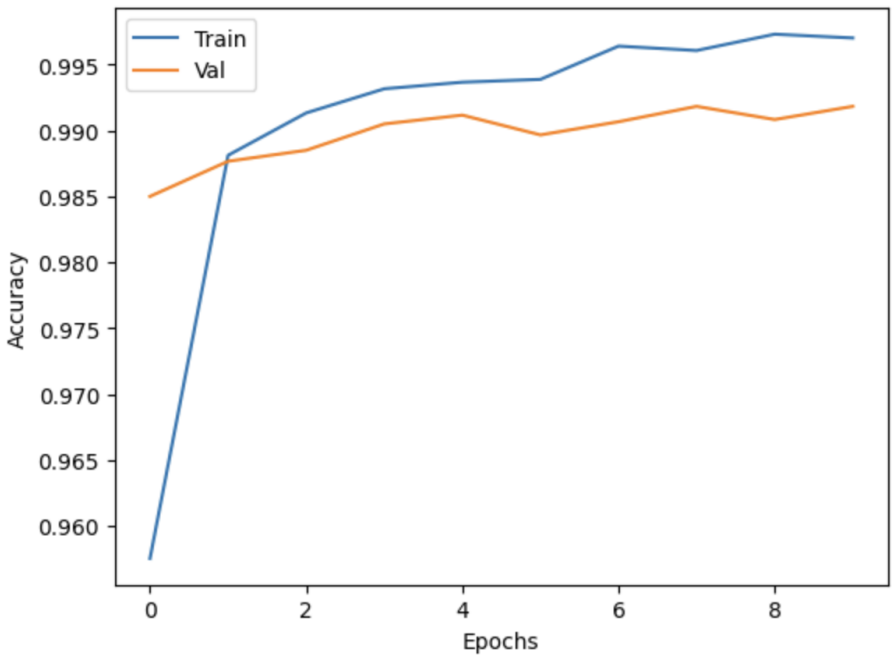

Figure 2: Training and Validation Accuracy — MNIST (ResNet-18)

	Dataset	Batch Size	Optimizer	LR	Model	Test Accuracy (%)
0	MNIST	16	Adam	0.0010	ResNet50	99.258333
1	MNIST	16	Adam	0.0001	ResNet50	99.058333
2	MNIST	16	SGD	0.0010	ResNet50	99.116667
3	MNIST	16	SGD	0.0001	ResNet50	98.116667
4	MNIST	32	Adam	0.0010	ResNet50	98.975000
5	MNIST	32	Adam	0.0001	ResNet50	99.316667
6	MNIST	32	SGD	0.0010	ResNet50	98.850000
7	MNIST	32	SGD	0.0001	ResNet50	96.900000

Figure 3: MNIST Test Accuracy Results for ResNet18

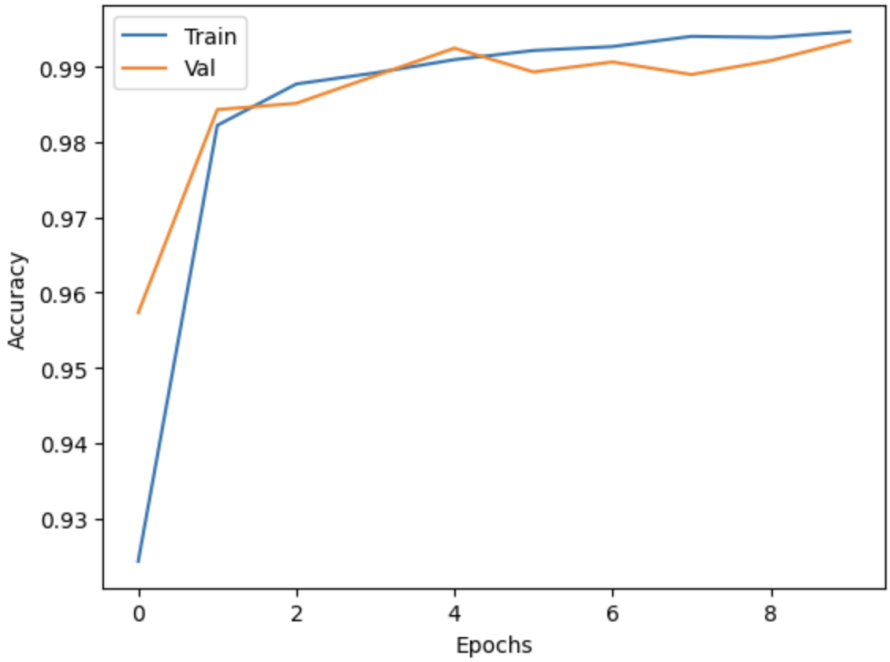

Figure 4: Training and Validation Accuracy — MNIST (ResNet-50)

2.2 FashionMNIST Results

	Dataset	Batch Size	Optimizer	LR	Model	Test Accuracy (%)
0	FashionMNIST	16	SGD	0.0010	ResNet18	92.225000
1	FashionMNIST	16	SGD	0.0001	ResNet18	90.141667
2	FashionMNIST	16	Adam	0.0010	ResNet18	93.450000
3	FashionMNIST	16	Adam	0.0001	ResNet18	93.125000
4	FashionMNIST	16	Adam	0.0010	ResNet18	92.850000
5	FashionMNIST	16	Adam	0.0001	ResNet18	92.483333
6	FashionMNIST	32	Adam	0.0010	ResNet18	92.983333
7	FashionMNIST	32	Adam	0.0001	ResNet18	92.333333

Figure 5: FMNIST Test Accuracy Results for ResNet18

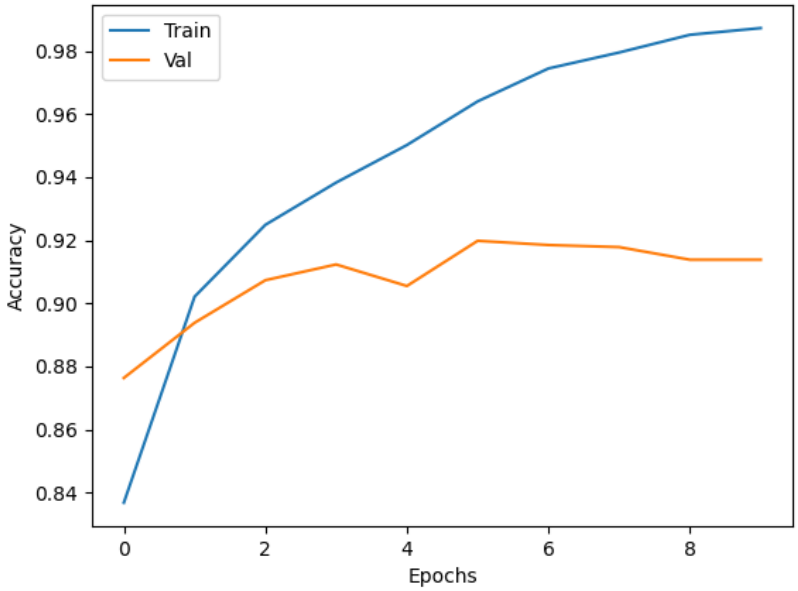

Figure 6: Training and Validation Accuracy — FashionMNIST (ResNet-18)

	Dataset	Batch Size	Optimizer	LR	Model	Test Accuracy (%)
0	FashionMNIST	16	Adam	0.0010	ResNet50	92.991667
1	FashionMNIST	32	Adam	0.0010	ResNet50	92.983333
2	FashionMNIST	16	Adam	0.0001	ResNet50	92.858333
3	FashionMNIST	32	Adam	0.0001	ResNet50	92.825000
4	FashionMNIST	16	SGD	0.0010	ResNet50	91.700000
5	FashionMNIST	32	SGD	0.0010	ResNet50	91.108333
6	FashionMNIST	16	SGD	0.0001	ResNet50	88.658333
7	FashionMNIST	32	SGD	0.0001	ResNet50	82.716667

Figure 7: FMNIST Test Accuracy Results for ResNet18

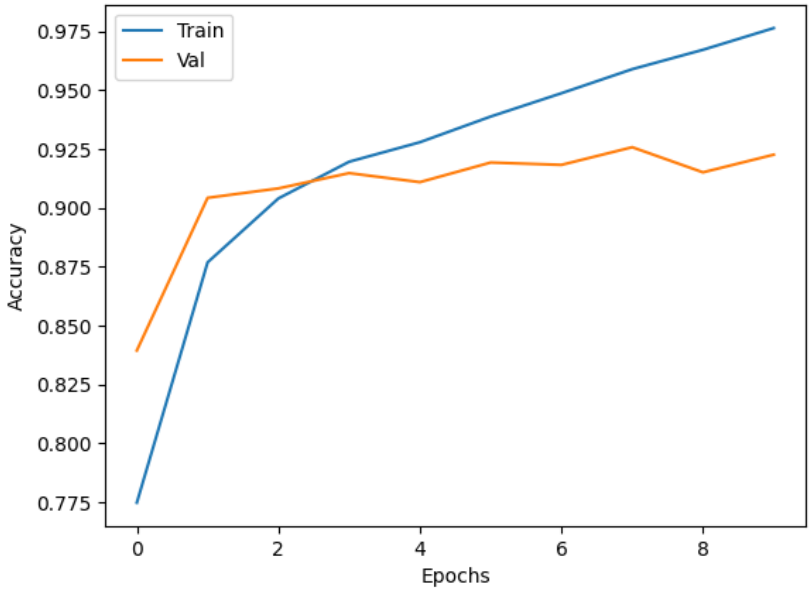

Figure 8: Training and Validation Accuracy — FashionMNIST (ResNet-50)

2.2.1 Model and Optimizer Behavior Analysis

- **ResNet18 vs ResNet50:** ResNet18 is sufficient for MNIST due to low dataset complexity, while ResNet50 provides modest gains on FashionMNIST at the cost of significantly higher FLOPs and training time.
- **SGD:** Exhibits stable but slow convergence and is highly sensitive to learning rate. Underperforms at low learning rates, especially on FashionMNIST.
- **Adam:** Consistently converges faster and achieves higher accuracy due to adaptive learning rates, benefiting deeper models and complex datasets.
- **Batch Size:** Smaller batches (16) slightly improve generalization, while larger batches (32) offer stable convergence with no clear accuracy advantage.
- **Key Takeaway:** Model depth should match dataset complexity, and optimizer choice has a greater impact on performance than batch size.

3 Hyperparameter Analysis

3.1 Effect of Pin Memory

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

Figure 9: Effect of pin_memory on Training Time and Accuracy

3.2 Effect of Number of Epochs

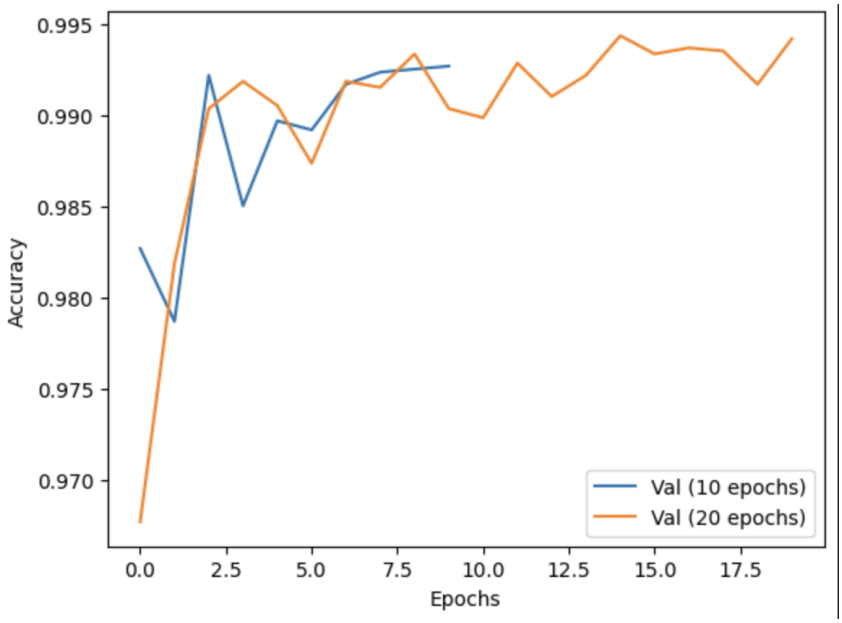

Figure 10: Validation Accuracy Comparison for 10 vs 20 Epochs

3.2.1 Hyperparameter Analysis

- **Learning Rate:** A learning rate of 0.001 consistently yields faster convergence and higher accuracy, while 0.0001 often underfits within the fixed number of epochs, particularly when using SGD.
- **Optimizer Sensitivity:** SGD is highly sensitive to learning rate selection, whereas Adam remains stable across both tested learning rates due to its adaptive update mechanism.
- **Pin Memory:** Enabling pin_memory slightly reduces training time by improving CPU–GPU data transfer efficiency, with no observable impact on classification accuracy.
- **Epoch Count:** Increasing the number of epochs improves validation accuracy initially, but performance saturates after approximately 10 epochs, indicating diminishing returns and potential overfitting beyond this point.
- **Batch Size:** Batch size has a secondary effect compared to optimizer and learning rate. Smaller batches introduce beneficial gradient noise, while larger batches offer more stable but not superior convergence.

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

Table 1: 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 11: SVM Accuracy Comparison

4.0.1 SVM Hyperparameter and Performance Analysis

- **Kernel Choice:** Non-linear kernels (RBF and polynomial) outperform linear decision boundaries, as they better capture complex class separations in image feature space.
- **MNIST Performance:** SVMs achieve competitive accuracy on MNIST due to the dataset's low complexity and well-separated classes after feature flattening.
- **FashionMNIST Performance:** Accuracy degrades on FashionMNIST because flattening destroys spatial information and increases feature dimensionality, making margin-based optimization harder.
- **Training Time:** SVM training time is significantly higher than CNNs due to quadratic or cubic complexity with respect to the number of samples, limiting scalability.
- **Key Takeaway:** While SVMs are effective for small, simple datasets, convolutional neural networks are better suited for large-scale image classification tasks.

5 Question 2: CPU vs GPU Performance (FashionMNIST)

Table 2: CPU vs GPU Performance Comparison

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

	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 12: CPU vs GPU Performance Comparison

5.0.1 CPU vs GPU Performance Analysis

- **Training Time:** GPU execution provides a speedup of approximately 15–20× compared to CPU due to massive parallelism and optimized tensor operations.

- **Accuracy Consistency:** Classification accuracy remains nearly identical across CPU and GPU executions, confirming hardware-independent model convergence.
- **Model Complexity:** ResNet50 incurs significantly higher training time and FLOPs than ResNet18, reflecting the computational cost of deeper architectures.
- **Optimizer Interaction:** Adam benefits more from GPU acceleration than SGD due to increased parallelism in adaptive moment computations.
- **Pin Memory Impact:** Enabling pin_memory reduces host-to-device data transfer overhead, leading to marginal but consistent improvements in training time.
- **Key Takeaway:** GPU acceleration is essential for efficient deep learning training, while CPUs are better suited for small-scale experimentation and debugging.

6 Overall Summary

The experiments confirm that shallow models are sufficient for simple datasets, while deeper architectures and adaptive optimizers are necessary for complex data. GPU acceleration is essential for efficient training, reinforcing best practices in modern deep learning and MLOps workflows.