A Systematic Evaluation of Neural Network Architectures for Fashion-MNIST

Kazi Ashhab Rahman Nusaibah Binte Rawnak

Abstract

We explored neural network performance on Fashion-MNIST, starting with NumPy-based MLPs and extending to CNNs and transfer learning. Our MLP experiments examined how depth, activation choice, regularization, and pre-processing influence convergence and accuracy.

A PyTorch CNN clearly outperformed all MLP variants, highlighting the advantage of convolutional feature extraction. We further analyzed CNN behavior through small architecture and dropout studies. Transfer learning was also tested, but underperformed the custom CNN due to domain mismatch.

Overall, the results show how architectural choices and regularization shape model performance, offering practical insight into building image classifiers.

1 Introduction

Neural networks play a central role in modern machine learning, and understanding how they learn requires working beyond high-level abstractions. This mini-project provided hands-on experience with implementing and training core neural architectures, beginning with multilayer perceptrons (MLPs) coded from scratch and later extending to convolutional neural networks (CNNs).

We used the Fashion-MNIST dataset [4], a benchmark of 70,000 grayscale images across ten clothing classes, to compare how architectural choices affect image classification performance. Implementing MLPs in NumPy allowed us to explore forward/backward propagation, initialization, and gradient-based optimization, while the PyTorch CNN highlighted the advantages of convolutional feature extraction over fully connected models.

Beyond the required components, we examined how variations in model design and regularization influence learning behavior. Overall, the project emphasized developing intuition for how different network structures, training decisions, and pre-processing steps shape performance on image data.

2 Dataset

We used the Fashion-MNIST dataset [4], a widely used benchmark containing 70,000 grayscale images of clothing items across ten categories. Each sample is a 28×28 image labeled with one of ten classes (e.g., T-shirt/top, Trouser, Dress, Sneaker, Ankle boot). The dataset is split into 60,000 training and 10,000 test examples and requires only scaling to [0,1] for pre-processing.

To understand the class balance, we computed the distribution of training samples across all categories (Figure 1). Fashion-MNIST is nearly perfectly balanced, with each class containing roughly 5,000 examples. This uniform distribution allows for fair comparison across models without needing class weighting or resampling.

Figure 1: Class distribution in the Fashion-MNIST training set.

2.1 Ethical Considerations

While Fashion-MNIST does not contain personal or sensitive data, it still reflects general concerns relevant to image classification tasks. Some classes (e.g., Shirt vs. T-shirt/top) are visually similar, and model errors in these categories highlight the limitations of neural networks when distinguishing fine-grained patterns. More broadly, working with this dataset serves as a reminder that real-world vision systems require careful evaluation for fairness, robustness, and transparency, even when the benchmark itself poses minimal ethical risk.

3 Results

We ran a series of experiments to study how architectural choices and training decisions affect performance on the Fashion-MNIST dataset. Our analysis covers MLP depth, activation functions, regularization, pre-processing, data augmentation, convolutional models, and transfer learning. We also include two extended studies, CNN architecture search and dropout regularization, to better understand model capacity and overfitting. All results are reported using accuracy and learning curves.

3.1 Hyperparameter Search: Learning Rate and Batch Size

We ran a grid search over learning rates $\{0.001, 0.01, 0.1\}$ and batch sizes $\{32, 64, 128\}$. While the heatmap in Figure 2 shows the highest accuracy at `lr=0.01, bs=32`, the train-validation gap plot (Figure 3) reveals that this setting severely overfits due to noisy, high-variance gradients. In contrast, `lr=0.001, bs=64` produced much more stable learning dynamics with a consistently smaller gap, indicating better generalization. For this reason, we chose `lr=0.001` and `bs=64` as the default MLP hyperparameters used in all subsequent experiments.

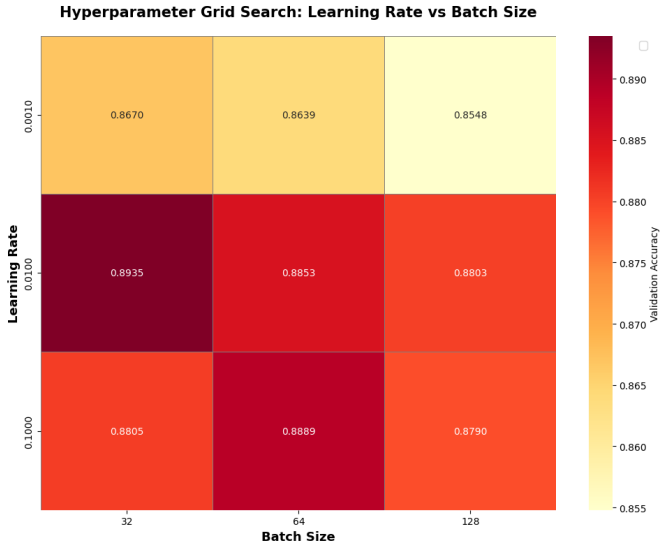

Figure 2: Validation accuracy by configuration.

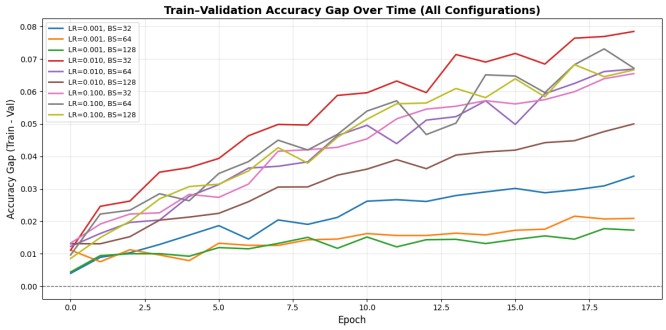

Figure 3: Train - validation accuracy gap across all configurations.

3.2 Effect of Network Depth

We trained MLPs with 0, 1, and 2 hidden layers to study how non-linearity and depth influence accuracy on Fashion-MNIST (Figure 4). The linear model performed poorly (73.80%), as expected for a dataset with non-linear structure. Introducing one hidden layer raised accuracy to 84.69% and produced much smoother learning, showing that a single non-linear transformation already captures most of the needed complexity. Adding a second hidden layer gave only a slight gain (84.90%) with almost identical curves, indicating diminishing returns.

Overall, the results match expectations: non-linearity is essential, while extra depth provides only marginal improvement.

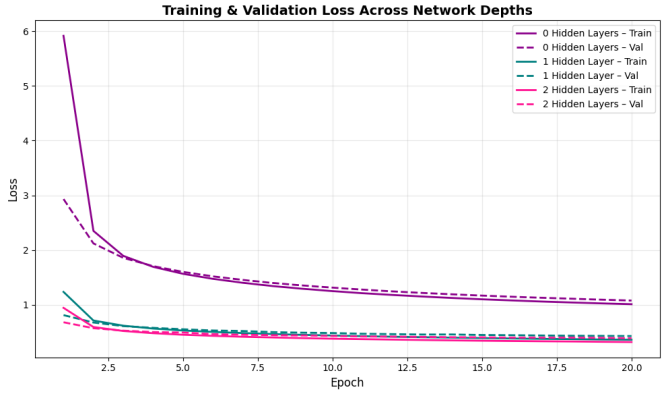

Figure 4: MLP loss across depths.

3.3 Effect of Activation Functions

We compared ReLU, tanh, and Leaky-ReLU using the same two-hidden-layer MLP. As seen in Figure 5, all three produced nearly identical results, with validation accuracies between 85.15% (ReLU) and 85.71% (Leaky-ReLU). The differences were under 1%, and their learning curves overlapped closely.

These results suggest that for a shallow MLP on Fashion-MNIST, the choice of activation is far less important than simply having a non-linear function. Because the dataset is small and low-complexity, the nuanced strengths of different activations do not meaningfully impact performance.

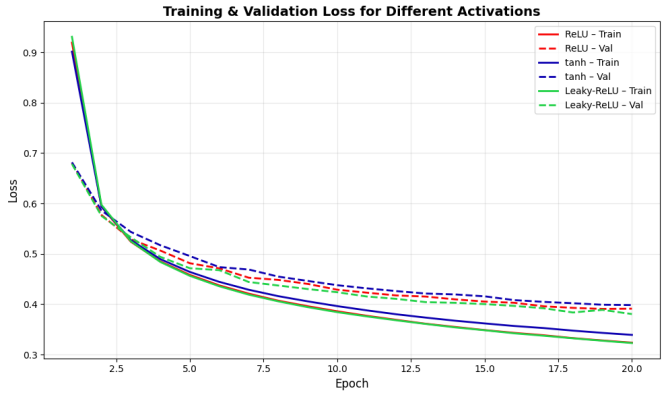

Figure 5: MLP loss across activation functions.

3.4 Effect of Regularization

We tested L1 and L2 regularization (with $\lambda = 0.001$) on the two-hidden-layer MLP. As seen in Figure 6, both penalties slightly increased training loss but helped stabilize validation loss, reducing mild overfitting present in the baseline. This translated into small accuracy gains: the unregularized model reached 84.90%, while L1 and L2 improved performance to 85.38% and 85.20%.

Overall, regularization provides a modest boost by encouraging simpler, more generalizable weights, though its impact is limited because the baseline model was already close to its optimal capacity on Fashion-MNIST.

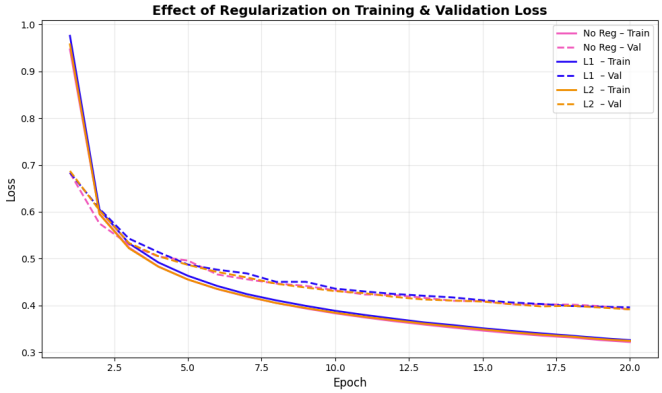

Figure 6: MLP loss with no regularization, L1, and L2.

3.5 Effect of Normalization

We compared the two-hidden-layer MLP trained on normalized inputs with the same model trained on raw pixel values. As shown in Figure 7, normalization led to faster, smoother convergence and slightly better generalization. The normalized model reached 84.9% validation accuracy, while the unnormalized model achieved 83.5%, a drop of 1.4%. Without normalization, the network began with higher loss and plateaued earlier, reflecting unstable gradients due to inconsistent input scales.

Overall, normalization provides a small but reliable improvement in both optimization stability and final accuracy.

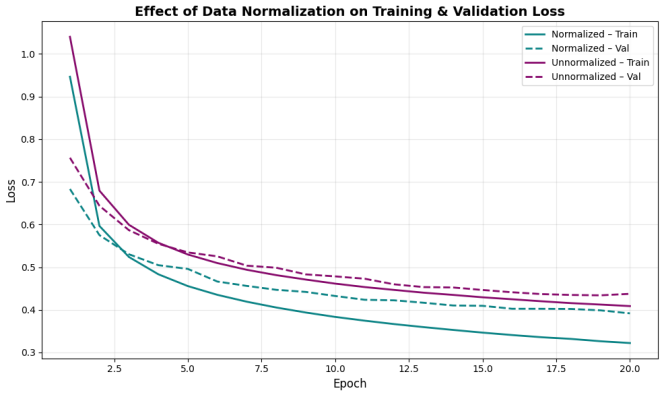

Figure 7: MLP loss for normalized vs. unnormalized inputs.

3.6 Effect of Data Augmentation on MLPs

We evaluated whether random shifts and rotations improve MLP generalization. As shown in Figure 8, augmentation significantly reduced test accuracy, from 85.20% (no augmentation) to 78.43%. This sharp drop occurs because MLPs operate on flattened pixel vectors, so small spatial transformations (translations, rotations) reshuffle pixel positions in ways that do not preserve local structure. To the network, an augmented image appears as a noisy or unrelated input, harming both optimization and generalization.

In this setting, augmentation provides no benefit: it increases training loss, slows convergence, and introduces harmful variance. For image tasks where spatial arrangement matters, but the model cannot exploit locality, augmentation becomes counterproductive. This illustrates a broader drawback: data augmentation can be harmful whenever the model architecture cannot interpret spatial transformations meaningfully, as with fully connected networks on image data.

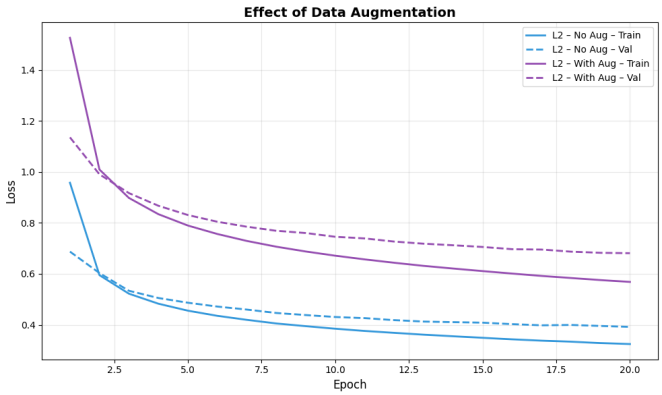

Figure 8: MLP performance with and without augmentation.

3.7 CNN Hyperparameter Search

We ran a small grid search over learning rate 0.001, 0.01, 0.1 and batch size 32, 64, 128 (Figure 9). A clear trend emerged: LR = 0.001 was the only stable setting, with all batch sizes reaching around 92% validation accuracy, while larger learning rates frequently diverged to ≈10%. Batch size had a smaller effect; at LR = 0.001, batch size 64 achieved the best accuracy (92.36%), and the gap plot (Figure 10) shows that larger batches slightly reduced overfitting.

Overall, CNN performance was highly sensitive to learning rate but relatively robust to batch size, so we selected LR = 0.001 and batch size 64 for later experiments.

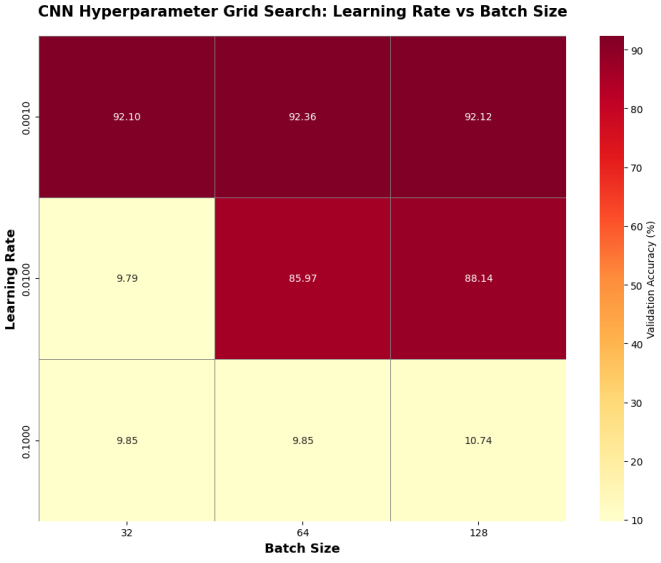

Figure 9: CNN validation accuracy by configuration.

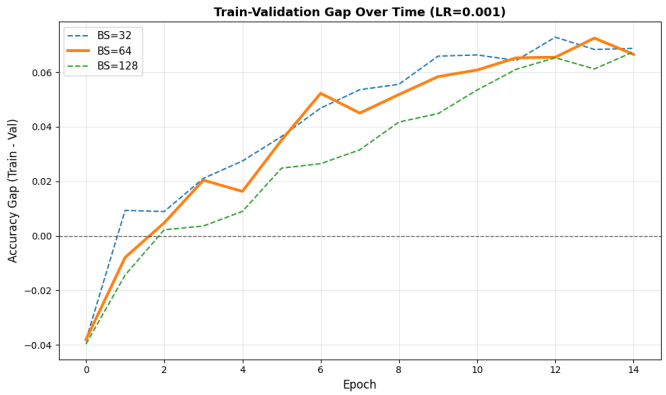

Figure 10: Train - validation accuracy gap at learning rate 0.001.

3.8 Convolutional Neural Network Performance

To compare fully connected models with convolutional architectures, we trained a CNN with two convolutional layers followed by a fully connected layer. As shown in Table 1, the CNN reached 91.57% test accuracy, outperforming the best MLP (85.20%) by 6.37%. This gap reflects the benefit of convolutional feature extraction: by preserving spatial structure and learning local patterns, CNNs capture information that flattened MLP inputs cannot.

While the CNN shows slight overfitting, it still generalizes far better than any MLP, confirming that convolutional inductive biases are well suited for Fashion-MNIST.

Table 1: Comparison of MLP and CNN Performance

Model	Accuracy (%)	Improvement
MLP (2 hidden, L2)	85.20	0.00
CNN (2 conv, 1 FC)	91.57	+6.37

3.9 CNN with Data Augmentation

Applying random shifts, rotations, and flips slightly improved test accuracy (from 91.57% to 92.22%), but at a significant computational cost: training time increased from 107 seconds to over 360 seconds. As shown in Figure 11, augmentation also reduced overfitting by narrowing the gap between training and validation loss. Overall, augmentation provides a small accuracy boost and more stable learning, but at the expense of much slower training.

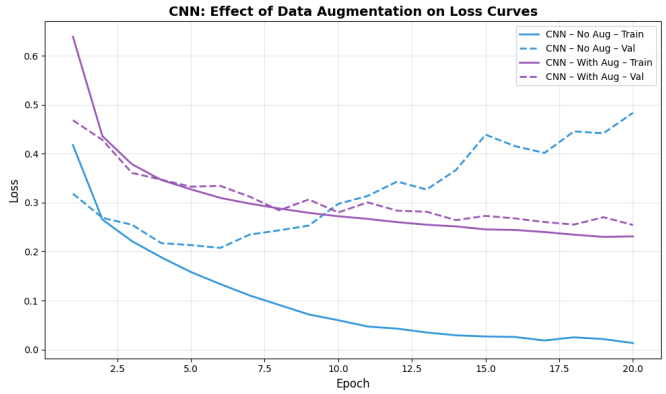

Figure 11: CNN performance with and without augmentation.

3.10 CNN Architecture Search

We explored how varying the number of filters (16, 32, 64) and kernel size (3×3 vs. 5×5) affects CNN performance. As shown in Figure 12, accuracy increases with more filters, since larger feature maps allow the model to capture richer spatial patterns. Smaller 3×3 kernels consistently outperformed 5×5, aligning with standard CNN design principles that favor deeper stacks of small kernels over larger ones.

The best configuration, 64 filters with 3×3 kernels, reached 92.17%, slightly above the baseline. Overall, moderate increases in model capacity help, but the gains taper off due to the simplicity of Fashion-MNIST.

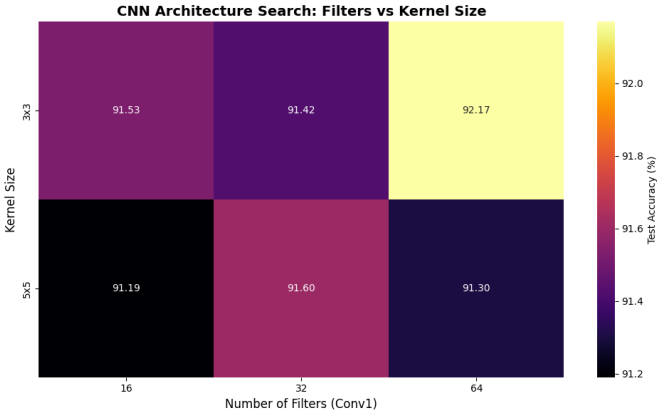

Figure 12: CNN test accuracy with different filters and kernels.

3.11 Dropout Regularization

We evaluated dropout rates between 0.0 and 0.5 to study their effect on overfitting. As shown in Figure 13, increasing dropout consistently reduced the train-validation accuracy gap. Without dropout, the gap was above 7%, but it fell steadily with stronger regularization, reaching about 2.5% at a dropout rate of 0.5. These results show that moderate to high dropout effectively suppresses overfitting in our CNN. Extremely strong dropout begins to limit learning capacity, but within this range the dominant effect is improved generalization.

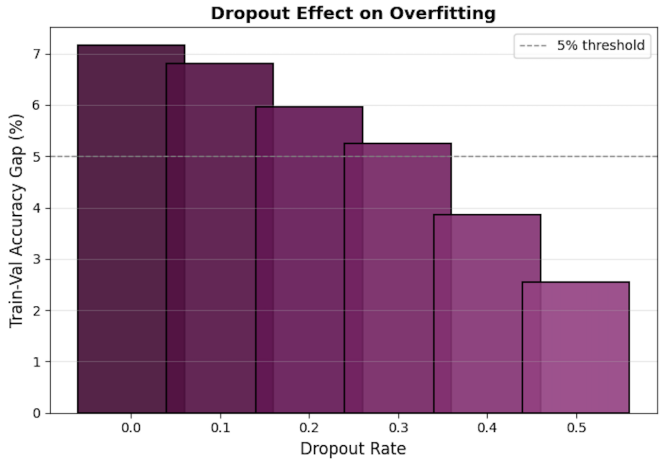

Figure 13: Train-validation accuracy gap for different dropout rates.

3.12 Transfer Learning with ResNet18

We fine-tuned a pretrained ResNet18 by adding 0, 1, or 2 fully connected layers on top of its feature extractor. Because ResNet18 is designed for 224×224 RGB ImageNet images, our 28×28 grayscale inputs had to be aggressively upsampled, making training far more computationally expensive. A full 20-epoch run would take close to one hour per epoch, so all experiments were limited to 5 epochs.

Across the three variants, the model with one FC layer achieved the lowest validation loss (Figure 14), indicating the best transfer performance. The two-FC model was slightly worse, and the zero-FC variant performed the worst. However, even the best ResNet configuration (≈83% accuracy) remained below both our best MLP (85.20%) and especially the CNN trained from scratch (91.57%). This shows that ImageNet features transfer poorly to low-resolution grayscale clothing images, and the overhead of upsampling outweighs any benefit of pretrained features.

In terms of training time, ResNet18 was by far the slowest model, even with only 5 epochs, making it less practical than both MLPs and CNNs. Based on these results, one added FC layer offers the best trade-off between performance and complexity within this transfer-learning setup.

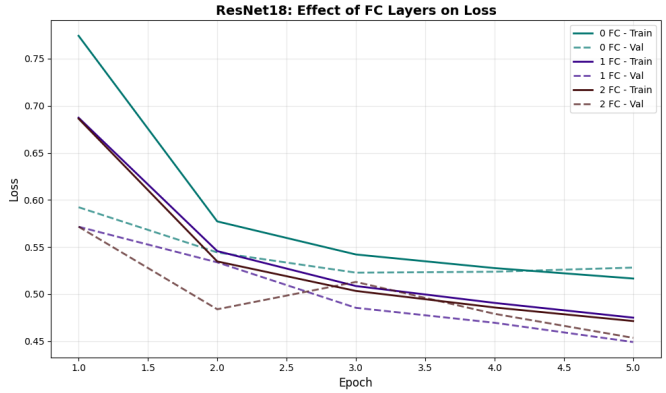

Figure 14: Loss for ResNet18 models with different FC layers.

4 Discussion and Conclusion

This project provided a systematic comparison of several neural network architectures on the Fashion-MNIST dataset. Our results show a clear pattern: adding non-linearity and depth to MLPs significantly improves performance, though gains taper off beyond the first hidden layer. Activation functions and regularization had only modest impact, while normalization played a meaningful role in stabilizing training. Data augmentation was ineffective for MLPs due to the loss of spatial structure when flattening images.

Convolutional networks performed substantially better, highlighting the value of spatial feature extraction for image classification. The baseline CNN already surpassed all MLP configurations by a large margin, and simple adjustments, such as moderate dropout, further improved performance and reduced overfitting. A lightweight hyperparameter search revealed small but consistent gains, while transfer learning from ResNet18 underperformed due to domain mismatch between ImageNet and Fashion-MNIST.

Overall, the best-performing model was the CNN with 0.4 dropout, achieving 92.40% accuracy. This suggests that compact CNNs trained from scratch are well-suited for Fashion-MNIST and can outperform deeper or pretrained models when the domain is simple.

Future Work. Several natural extensions could further improve or refine these results.

(i) Running more training epochs may reveal whether some models plateau early or continue improving.

(ii) Trying additional CNN variations, such as adding one more convolutional layer or adjusting filter sizes, could clarify how much capacity is actually needed for this dataset.

(iii) Exploring different, simpler forms of augmentation (e.g., slight rotations or shifts) may help, if tuned more carefully.

(iv) Performing cross-validation or repeating experiments with different random seeds would strengthen the reliability of the findings.

(v) Finally, comparing against a few well-known lightweight architectures (e.g., a slightly deeper CNN or a small residual block) would help determine whether the performance ceiling is architectural or dataset-limited.

Table 2: Summary of configurations across experiments.

Model / Setting	Best Accuracy (%)
MLP (0 hidden layers)	73.80
MLP (1 hidden layer)	84.69
MLP (2 hidden layers, L2)	85.20
MLP + Augmentation	78.43
CNN (baseline)	91.57
CNN + Dropout (0.4)	**92.40**
CNN Hyperparam (64 filters, 3×3)	92.17
ResNet18 Transfer Learning	83.26

5 Statement of Contributions

Nusaibah Binte Rawnak implemented the MLP models and experiments on depth, activation functions, regularization, and normalization, as well as the initial CNN setup and MLP-CNN comparison.

Kazi Ashhab Rahman conducted the subsequent experiments, including CNN vs. MLP analysis, augmentation studies, transfer learning with ResNet18, the CNN architecture search, and dropout regularization.

Both authors contributed equally to experiment design, figure preparation, interpretation of results, and report writing.

References

[1] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[2] Roboflow Blog. Resnet-18 explained: Architecture, uses, and performance. `https://blog.roboflow.com/resnet-18/`, 2023.

[3] Torchvision Contributors. Resnet-18 - torchvision models. `https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html`, 2025.

[4] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[5] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3320–3328, 2014.