Assignment 3: Classification of Image Data with Multilayer Perceptrons and Convolutional Neural Networks

COMP 551: Applied Machine Learning

Audréanne Bernier (261100643)

Anjara Trachsel-Bourbeau (261119884)

Ben Coull-Neveu (261116508)

McGill University

Department of Computer Science

November 24, 2025

Abstract

In this project, we investigate the performance of multilayer perceptrons (MLPs) and convolutional neural networks (CNNs) on the Fashion-MNIST image classification dataset. We implement an MLP class supporting 0–2 hidden layers, user-defined activation functions, and L1/L2 regularization. A systematic hyperparameter search across batch size, learning rate, regularization strength, and hidden layer size was performed, using early stopping to reduce overfitting and avoid extensive tuning of training epochs. Simple MLPs achieve test accuracies up to 88% on the normalized dataset, but their performance is sensitive to small variations in initialization and validation splits. Regularization and data augmentation provide minimal benefit, indicating that fully connected networks struggle to capture spatial features. By contrast, CNN models generally outperform MLPs, achieving test accuracies above 92% and generalizing better under augmentation. These results highlight the limitations of MLPs for image classification and the advantage of CNNs in capturing image structure.

1 Introduction

Image classification is a common task in machine learning. Classical approaches rely on handcrafted features, where the focus is traditionally on detecting for edges and specific patterns, textures and shapes. Images are very complex due to their large size and all of the information that is carried. To reduce this complexity,the raw pixels are converted into a numerical representation of feature vectors as working with the raw pixels is computationally taxing. However, with the advent of deep learning, neural networks have become dominant tools for image classification given their ability to learn hierarchical features in images, especially convolutional neural networks (CNNs). However, there is still some debate to this day about which tool is best for different scenarios as CNNs require large samples for accuracy. For example, when comparing the two for liver MRI identification, the handcrafted features performed better than the CNNs due to the limited data availability (Lin et al., 2020). This highlights the need to compare these two methods and evaluate in which conditions they work well.

In this project, we evaluate the performance and limitations of MLPs on the Fashion-MNIST dataset (Xiao et al., 2017) made up of 70,000 grayscale images across ten clothing categories. We implement an extensible MLP framework supporting variable depth (0–2 hidden layers), multiple activation functions, and L1/L2 regularization. Through comprehensive hyperparameter searches, we analyze how depth, width, and learning rate influence convergence and generalization. We also compare them against a baseline CNN model implemented for this project. Related work consistently shows that CNNs substantially outperform MLPs on image tasks due to convolutional filtering, and our experimental results support this. Overall, this study situates MLPs within the broader landscape of image classification tasks, illustrating both their pedagogical value and their practical shortcomings relative to CNNs.

2 Datasets

We used the Fashion-MNIST dataset (Xiao et al., 2017) which is comprised of 60,000 training images and 10,000 test images, each with their corresponding labels. All are 28x28 grayscale images. Each image corresponds to one of 10 classes, all of which have 6,000 samples in the training set. A validation set was made using 10% of the training data, leaving 54,000 training images and 6,000 validation ones. The pixel intensity distribution for each class are shown in Figure 1. Since the images in the dataset are in the form of greyscale images, we normalized the data by subtracting off the mean and dividing by the standard deviation of the whole training set.

3 Results & Discussion

3.1 Hyperparameter Searches

Grid searches were performed for hyperparameter tuning in the MLP, specifically for batch size, learning rate, and regularization coefficients. The number of epochs was set to 50 for most training, but early stopping was implemented such that training stops once validation loss starts to worsen. The models used for this tuning are

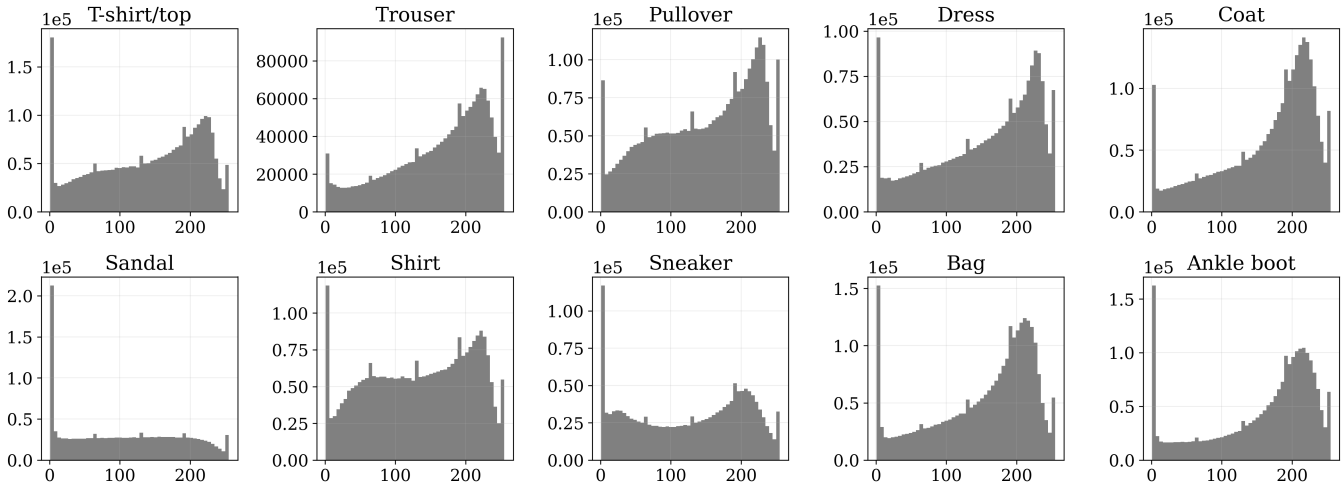

Figure 1: Distribution of raw pixel intensity per class. Pixels of value 0 (background) were omitted to prevent them from dominating the distribution.

the same as the first three presented in Section 3.2, namely the 0 layer model, the MLP with 1 hidden layer and ReLU activation, and the MLP with 2 hidden layers and ReLU activation. We also performed tuning of the batch size and earning rate for the CNN model. Optimal hyper-parameters are shown in Table 1, with the full resulting plots shown in Appendix B.

Model	Batch Size	Learning Rate	Regularization Strength
0 Hidden Layers	32	0.01	–
1 Hidden Layer	256	0.1325	–
2 Hidden Layers	64	0.1325	–
2 Hidden Layers & L1 Reg.	–	0.01	0.001
2 Hidden Layers & L2 Reg.	–	0.04	0.001
CNN	16	0.001	–

Table 1: Optimal hyperparameters found for different MLP & CNN models. The optimal batch size found for the MLP model with 2 hidden layers was used for all subsequent models with the same amount of layers.

Additional hyperparameter tuning was done to find the optimal number of hidden units for a 2 layer MLP with ReLU activation. Doing so, $M_1 = 256$ and $M_2 = 256$ were found to be the optimal values for the first and second hidden layers, which happen to be the number of hidden units required to be used for all models trained.

Certainly, more tuning could have been done, such as doing greater dimensional grid searches, but this was our approach given limited time and resources.

3.2 Multilayer Perceptron Models

For the MLP models, we use the categorical cross-entropy loss function, and implement the softmax classification in the output layer since there are 10 classes in the dataset. The MLP class was designed to allow for 0, 1, and 2 hidden layers, with the number of hidden nodes being a definable quantity. Additionally, the activation functions are user-defined; for our tests we used the ReLU, leaky ReLU, and tanh functions. L1 (lasso) and L2 (ridge) regularization were also implemented.

In all MLP tests, we used the optimal model hyperparameters as shown in Table 1. The hidden layer size is held at 256 for all tests, as required by the work instructions and in agreement with the optimal values founds for L=2. The performance of all models described here are shown in Table 2. In our first tests, we compare MLP models with 0, 1, and 2 hidden layers. A plot showing the losses and validation accuracy during training is shown in Figure 2. The MLP with 2 hidden layers required fewer epochs to converge, while still achieving better test accuracy than 0 hidden layers, and similar performance to the single hidden layer model. These results are

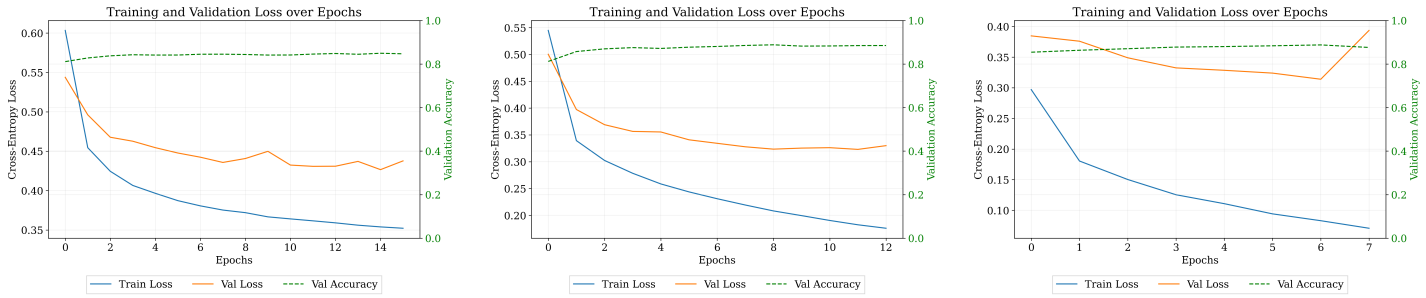

Figure 2: Training loss, validation loss, and validation accuracy across epochs for MLP models with 0 (left), 1 (center), and 2 (right) hidden layers. All used the ReLU activation function.

expected, since deeper MLPs generally achieve better performance due to their ability to learn more complex patterns. Curiously, the validation loss is consistently larger than training loss, suggesting overfitting, though this was largely dependent on the validation split. We also trained two additional 2-layer MLPs using tanh and leaky-ReLU activations. Their test accuracies were very similar to the ReLU model, with no activation showing a consistent advantage. The primary upside to using ReLU is that it is simple to compute, as opposed to tanh, and it doesn't require an additional hyperparameter, unlike leaky ReLU.

Model	Test Loss	Test Accuracy	Epochs
0 & 1 Hidden Layers MLP			
0 Hidden Layers	0.464	0.841	16
1 Hidden Layer & ReLU	0.364	0.873	13
2 Hidden Layers MLP Variants			
ReLU	0.415	0.869	8
tanh	0.386	0.869	8
Leaky ReLU	0.363	0.879	9
L1 Reg.	0.443	0.845	11
L2 Reg.	0.348	0.880	21
Unnormalized Data	0.493	0.844	9
L1 Reg. & Data Augmentation	0.521	0.781	14
L2 Reg. & Data Augmentation	0.441	0.844	21
CNN Models			
CNN	0.2375	0.9205	15
CNN & Data Augmentation	0.2321	0.9222	15
ResNet18 CNN & Data Augmentation	0.8733	0.6963	15

Table 2: Test loss and test accuracy for all 8 different MLP models and 3 different CNN models trained. For the bottom five rows in the 2 hidden layers MLP section, the models were trained using a ReLU activation.

L1 and L2 regularization were applied during training on both the original and augmented datasets, with the corresponding training curves shown in Figures 3 and 4. Neither regularization significantly improved performance and generally increased training time, suggesting that these techniques are not well-suited to generalize MLPs for image classification. Using unnormalized data also resulted in slightly worse performance, as expected for most machine learning models. Finally, data augmentation was applied by rotating and flipping images, without cropping to preserve important features. However, this approach provided limited benefit, with L1 and L2 regularization yielding significantly lower test performance for both L1 and L2 regularization in the augmented setting.

All of these tests suggest that MLPs aren't ideal for generalized image classification, likely because they can't consider contextual information and thus don't pick up on image features. Although earlier tests supported this conclusion, the data augmentation test was the final nail in the coffin, since it shows that the MLP models struggle to classify *transformed* images of the same objects that the model was trained on!

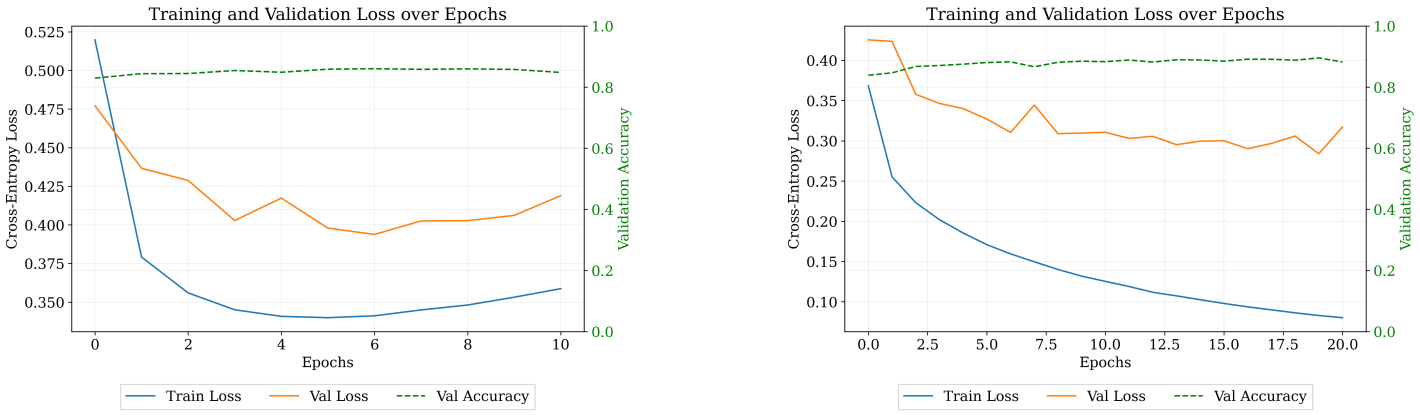

Figure 3: Training loss, validation loss, and validation accuracy across epochs for models with 2 hidden layers, L1 regularization (left), and L2 regularization (right).

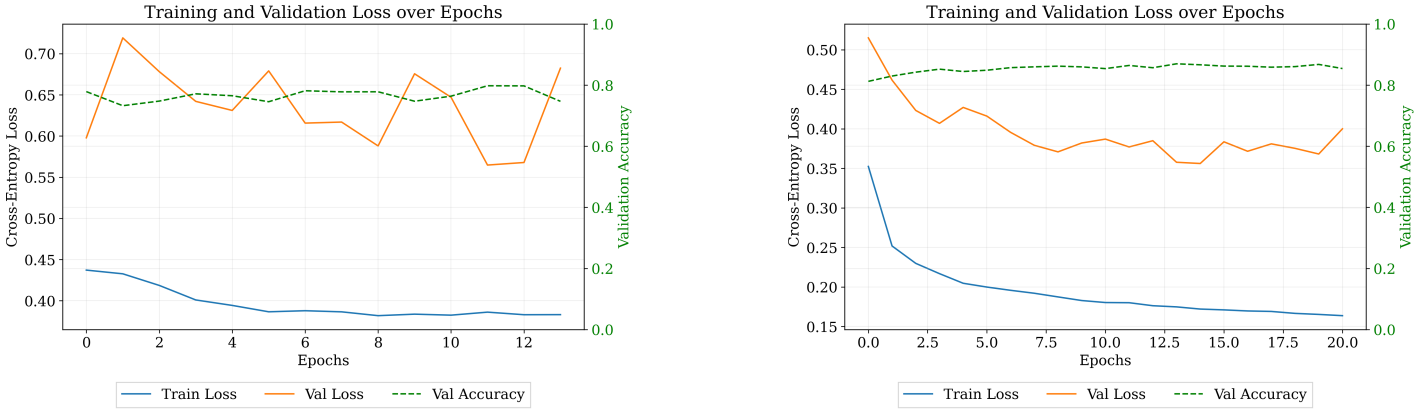

Figure 4: Training loss, validation loss, and validation accuracy across epochs for models with 2 hidden layers, L1 regularization (left), and L2 regularization (right). Both models were trained on augmented data.

3.3 Convolution Neural Network Models

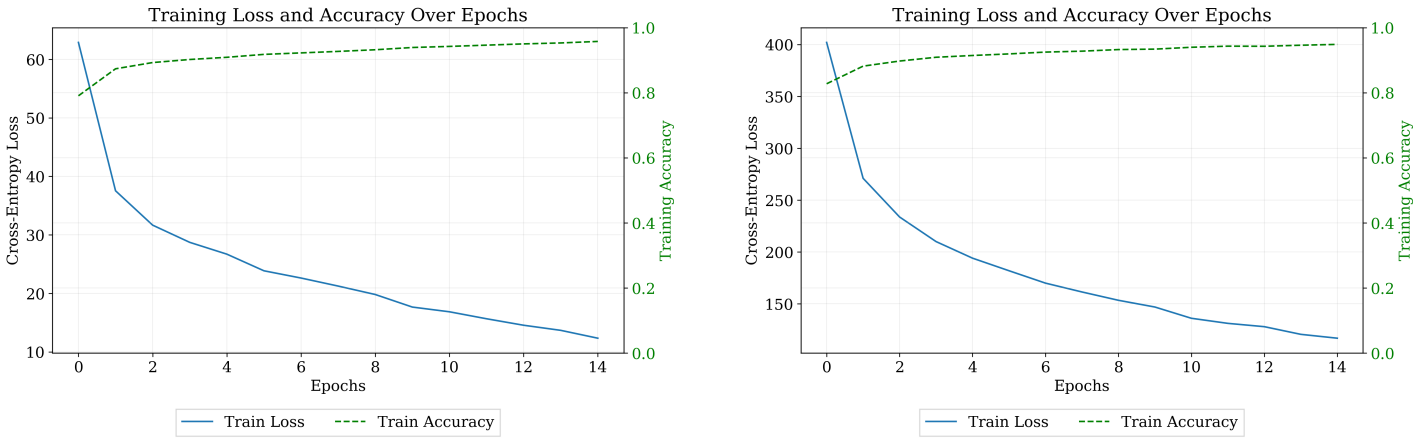

Figure 5: Training loss and accuracy over epochs for the CNN on the Fashion-MNIST dataset (left) and on the augmented dataset (right). The same augmented dataset was used for the MLP with 2 hidden layers.

We trained a convolutional neural network (CNN) with two convolutional layers, one fully connected hidden layer, and a 256-unit output layer. Padding was applied to the convolutional layers to preserve input dimensions, and grid searches (see Table 1) were used to optimize learning rate and batch size. The CNN was trained on

both the original and augmented datasets, with test loss and accuracy reported in Table 2 and training curves shown in Figure 5. The test accuracy for both CNN models is around 92%, consistently outperforming the MLP. Although training was limited to 15 epochs for the CNN due to time constraints, higher accuracy could likely be achieved with more training. 15 epochs is, however, comparable to the number of epochs used for the MLPs, which typically converged in even fewer epochs than that. This shows that, for a comparable training time, the CNN reaches significantly better performance. This result is expected since CNNs are much more effective than MLPs for image classification tasks. More specifically, in the data-augmented setting, the CNN performs noticeably better but both CNNs and MLPs converge in roughly the same number of epochs (on average 15).

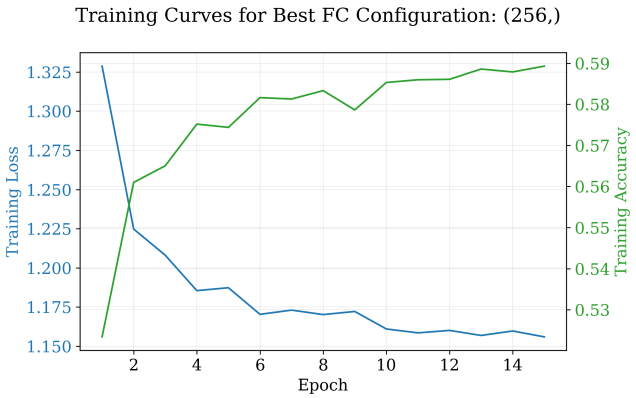

Figure 6: Training loss and accuracy across epochs for the ResNet model with 1 fully connected layers.

The pre-trained ResNet18 model (He et al., 2015) was used with the Fashion-MNIST dataset and with data augmentation. All convolution layers remained fixed while the fully connected layers were varied and trained on. We opted for 3 different fully connected layer configurations: 1 layer with 256 units, 2 layers with 512 & 256 units, and 3 layers with 1024 & 512 & 256 units. These were trained for 5 epochs in order to determine the optimal configuration. The results of this search are presented in Table 3 of Appendix A. The optimal configuration (1 fully connected layer with 256 units) was then used to train the model for 15 epochs, with the test metrics reported in Table 2 and the training curves shown in Figure 6. This model showed the weakest performance among all models in this project, achieving a test accuracy of 69.63%. This is 14.77% below the best MLP with regularization and data augmentation, and 22.59% below the CNN with data augmentation. This substantially lower performance may be due to the more limited search conducted for the optimal fully connected architecture. In terms of raining time, the accuracy of this model was reached in either a comparable or a higher number of epochs that all other models, reinforcing that all other models perform better.

4 Conclusion

We implemented multilayer perceptrons (MLPs) and convolutional neural networks (CNNs) on the Fashion-MNIST image classification dataset. We tested the effects of various models including an MLP class supporting 0–2 hidden layer, user-defined activation functions, and and L1/L2 regularization. We performed hyperparameter search over batch size, learning rate, regularization strength, and hidden layer size and obtained optimal values for each parameter. We also implemented early stopping to lower overfitting. In a further analysis, we should perform a more extensive hyperparameter search over the parameters we included and also perform searches on other parameters such as the number of filters, kernel size, stride or padding. When evaluating for different number of hidden layers, we found that our MLP model with two layers required the smallest number of epochs to converge and resulted in better test accuracy. However, the validation loss was larger than the test loss, suggesting an overfit, but this was not consistent across all validation splits. Our MLPs achieve test accuracies up to 88%, but their performance was sensitive to small variations in parameter initialization and validation splits. The addition of regularization and data augmentation had a minimal and sometime negative effect. We then tested our CNNs with the same data, for which test accuracies got as high as 92.2% with a similar number of epoch, thus performing better than the MLPs with similar training times, largely due to they ability to use contextual information by capturing spatial patterns extending beyond individual pixels.

5 Statement of Contributions

All members contributed equally to the analysis and presentation of the data, to the implementation and training of the model, and to the writing of this report.

References

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Wenyi Lin, Kyle Hasenstab, Guilherme Moura Cunha, and Armin Schwartzman. Comparison of handcrafted features and convolutional neural networks for liver MR image adequacy assessment. *Scientific Reports*, 10 (1):20336, November 2020. ISSN 2045-2322. doi: 10.1038/s41598-020-77264-y. URL https://doi.org/10.1038/s41598-020-77264-y.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

A Fully Connected Layer Configurations for the ResNet18 CNN

FC Configuration	Test Loss	Test Accuracy
(256)	0.9406	0.6687
(512, 256)	0.9882	0.6668
(1024, 512, 256)	1.0824	0.6387

Table 3: Test loss and accuracy for different configurations of fully connected layers in the ResNet model while searching for best configuration. The FC Configuration column gives the number of hidden units for each fully connected layer used. We performed training for 1-3 layers and on 5 epochs.

B Hyperparameter Search Grids

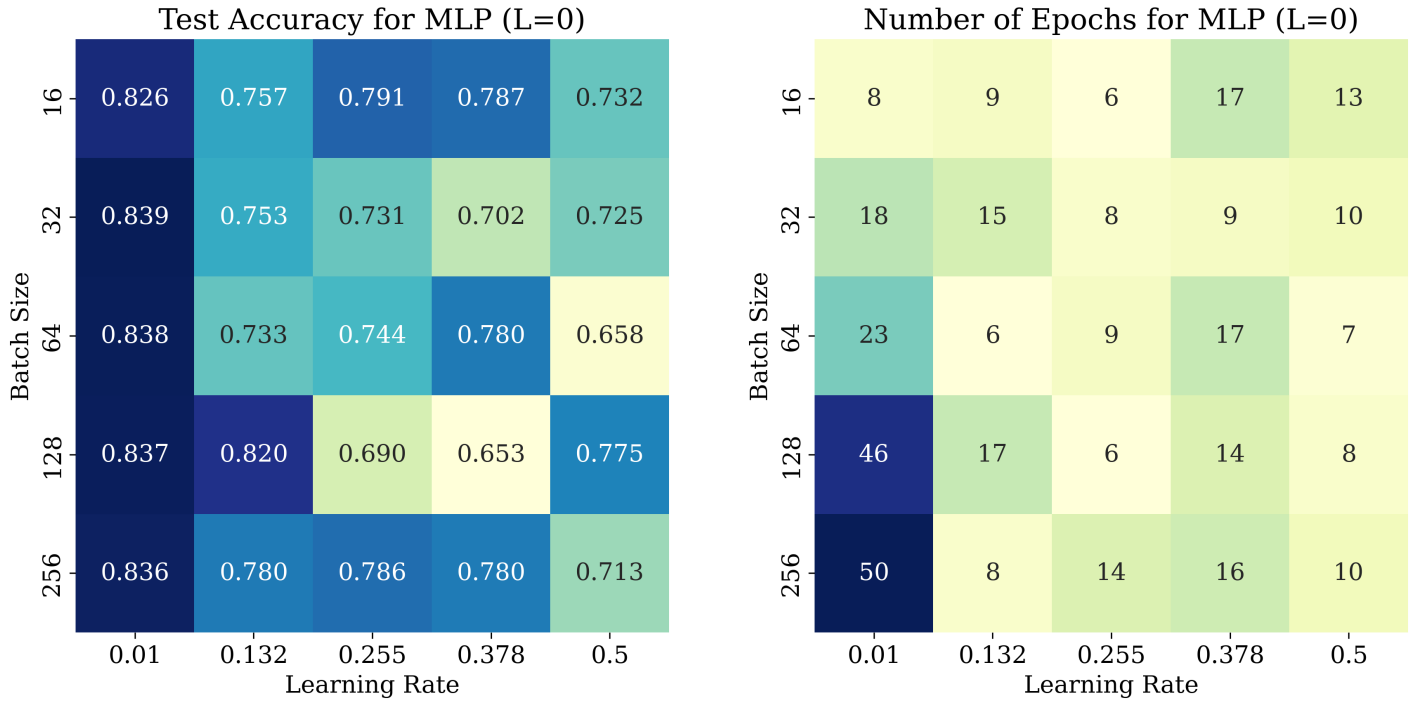

Figure 7: Hyperparameter grid search for an MLP model with **0 hidden layers**. Plots show the test accuracy (left) and the number of epochs required for convergence (right) of the final trained model.

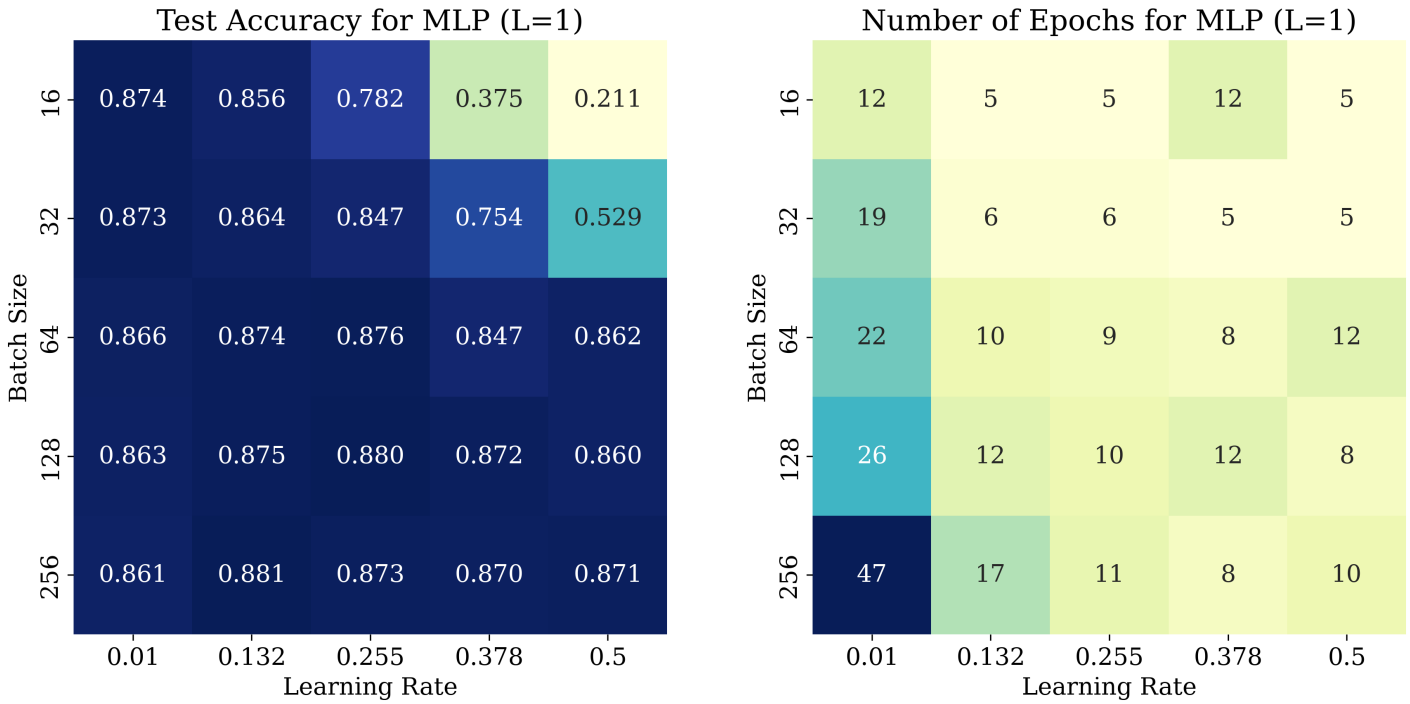

Figure 8: Hyperparameter grid search for an MLP model with **1 hidden layer and ReLU activation**. Plots show the test accuracy (left) and the number of epochs required for convergence (right) of the final trained model.

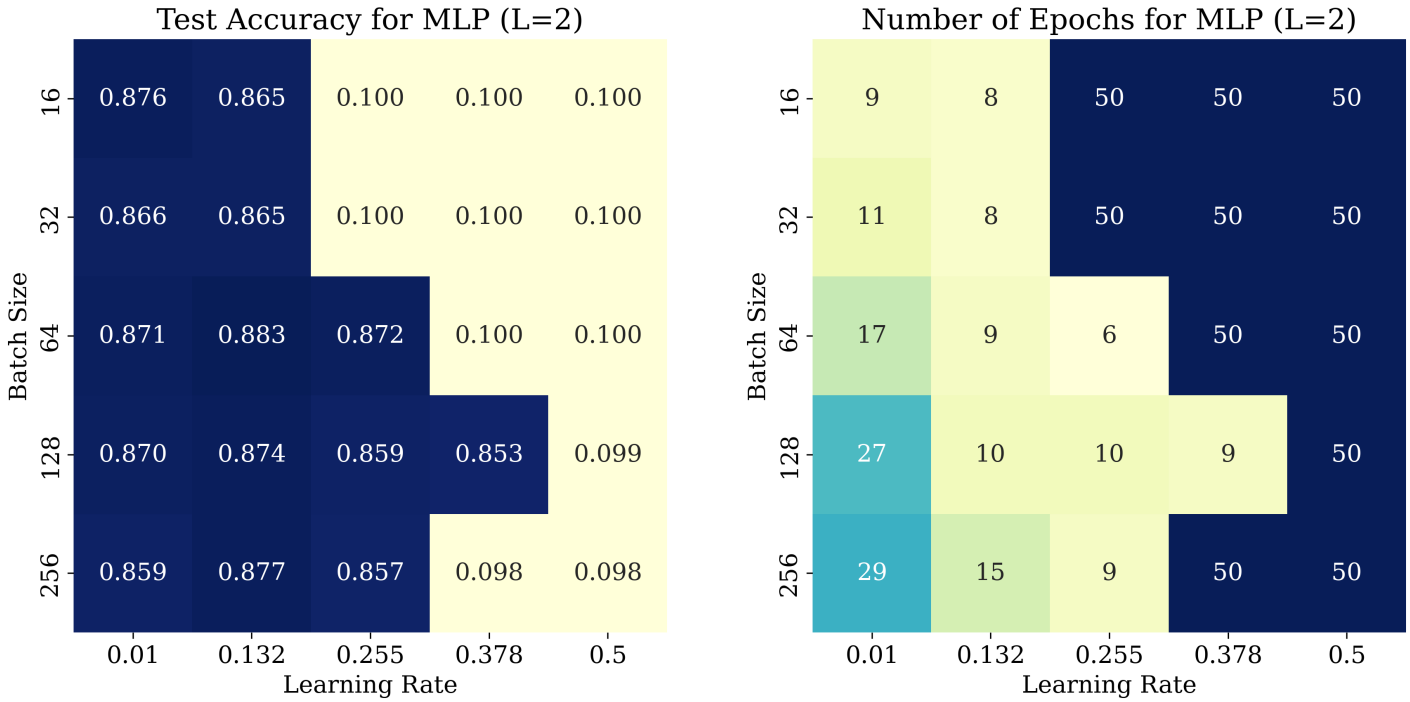

Figure 9: Hyperparameter grid search for an MLP model with **2 hidden layers and ReLU activations**. Plots show the test accuracy (left) and the number of epochs required for convergence (right) of the final trained model. Notice that with sufficiently large hyperparameters, the loss blows up, suggesting in particular that more hidden layers requires smaller learning rates.

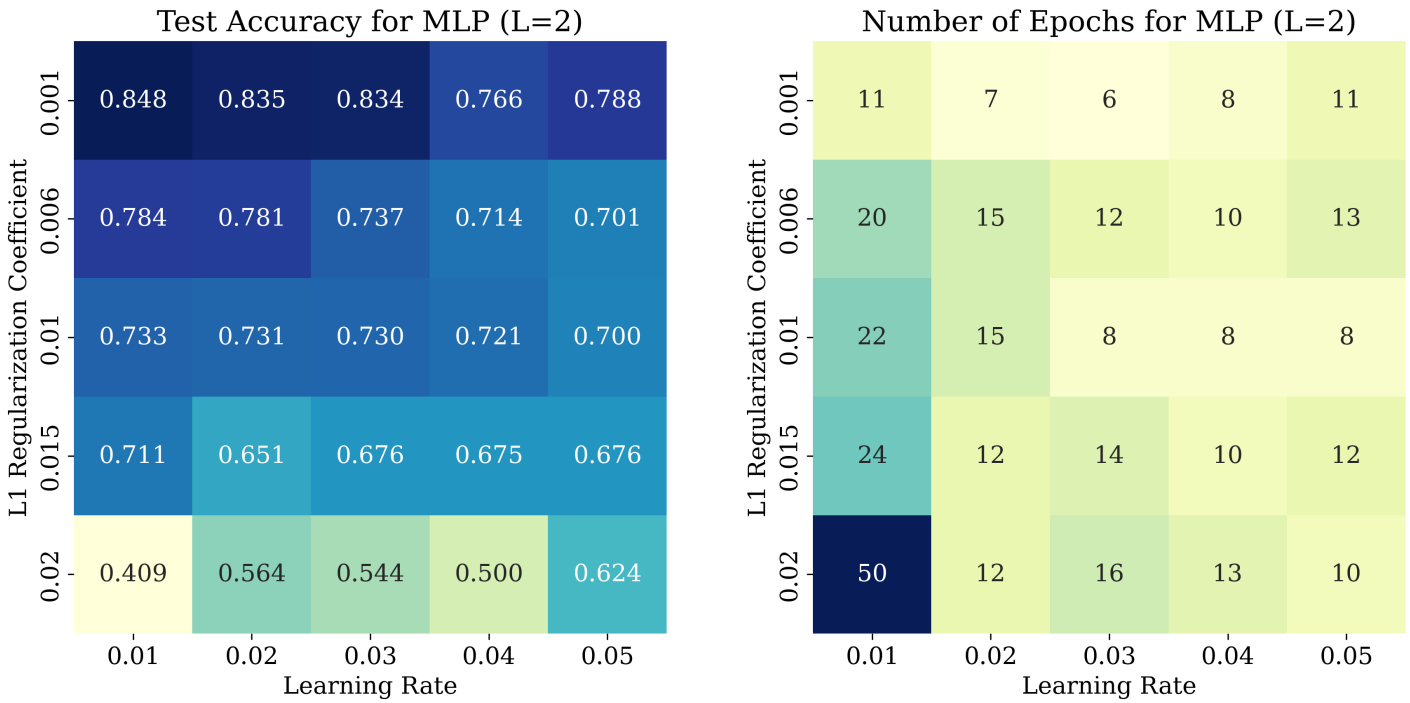

Figure 10: Hyperparameter grid search for an MLP model with **2 hidden layers and ReLU activations, and L1 (lasso) regularization**. Plots show the test accuracy (left) and the number of epochs required for convergence (right) of the final trained model.

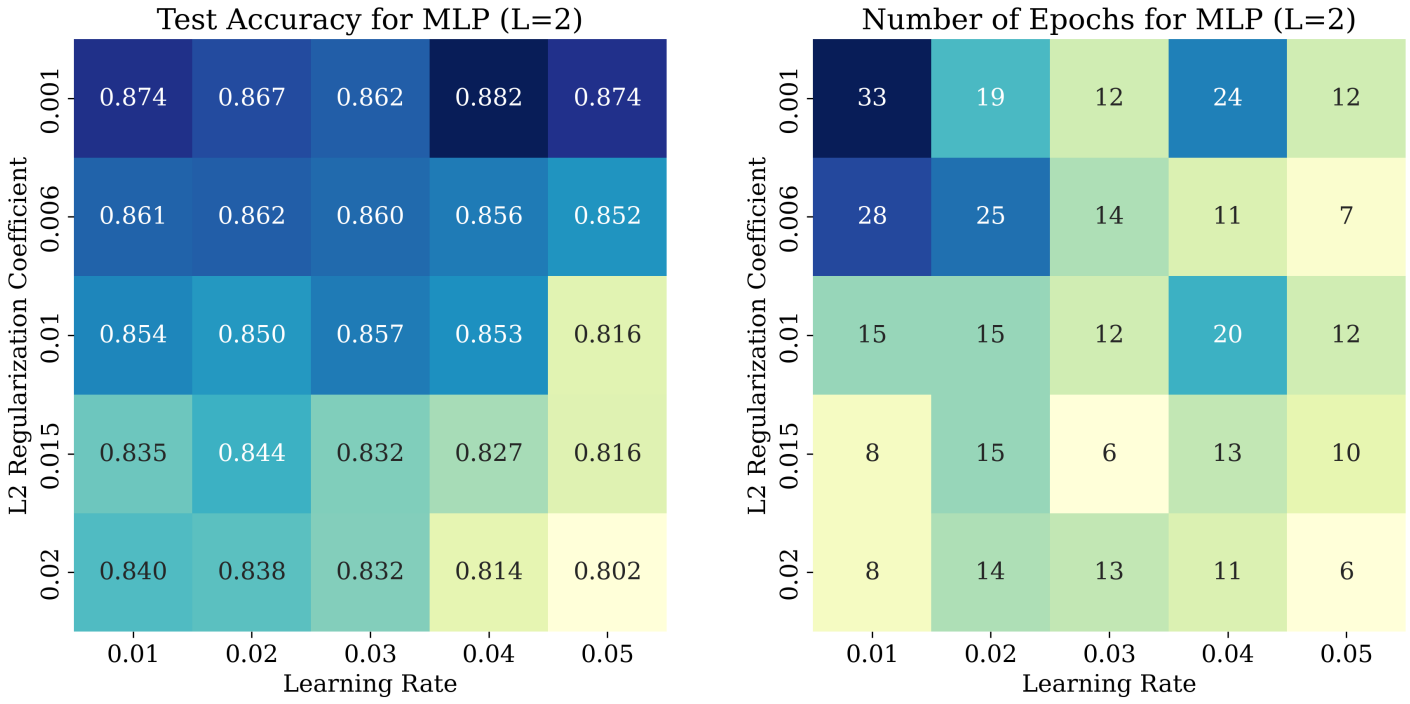

Figure 11: Hyperparameter grid search for an MLP model with **2 hidden layers and ReLU activations, and L2 (ridge) regularization**. Plots show the test accuracy (left) and the number of epochs required for convergence (right) of the final trained model.

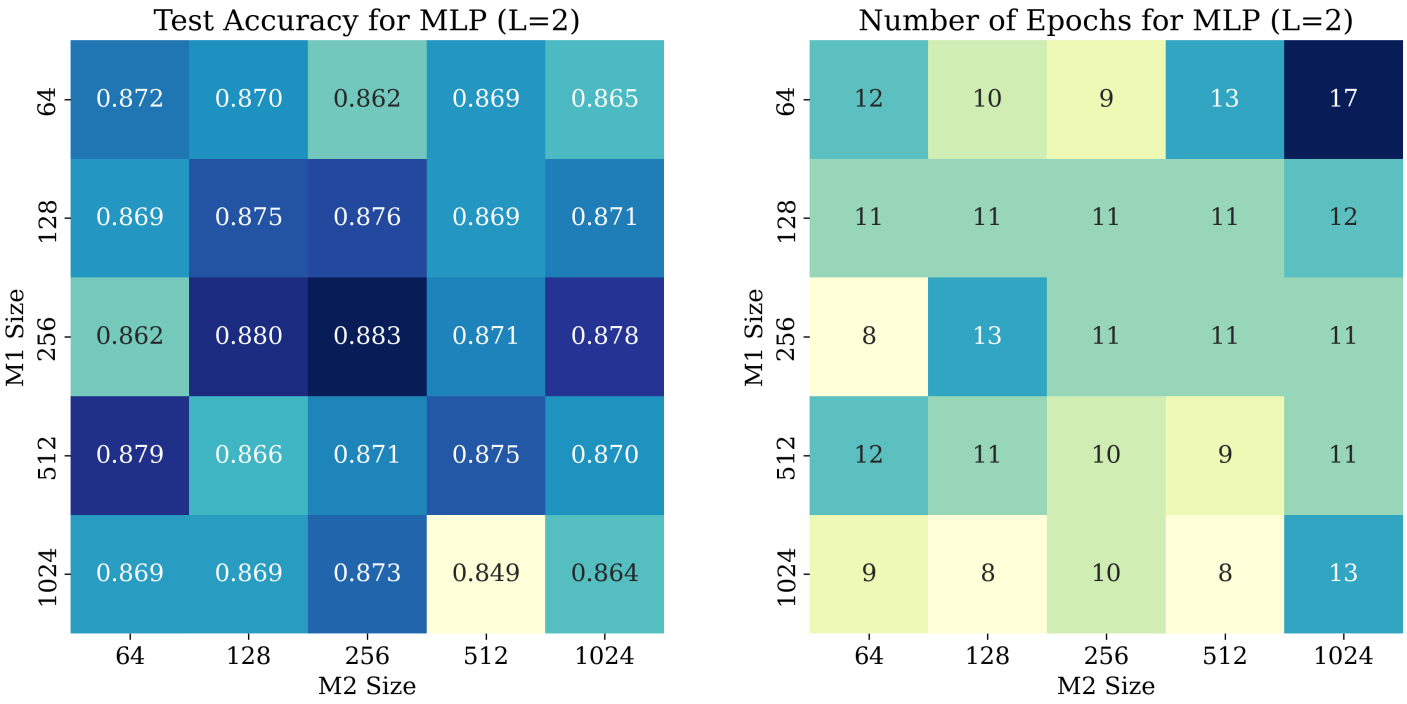

Figure 12: Hyperparameter grid search over the **number of hidden units** for a two-hidden-layer MLP with ReLU activations. The plots show the test accuracy (left) and the number of epochs needed for convergence (right) of the final trained model.

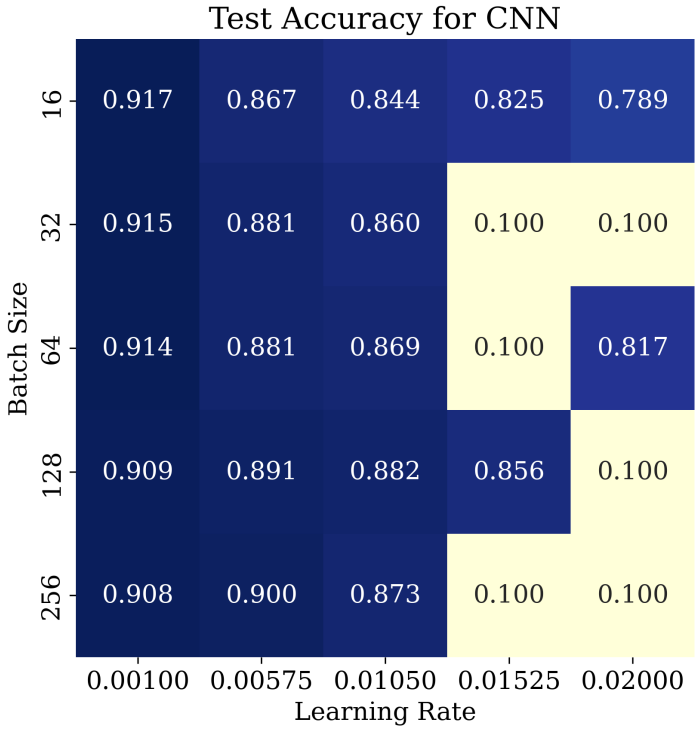

Figure 13: Hyperparameter grid search for a **CNN model**. The plot shows the final test accuracy of the trained model.