Impact of CNNs on Image Classification

Elvar Þór Sævarsson <elvars20@ru.is>
Guðjón Hafsteinn Kristinsson <gudjonk24@ru.is>
Kjartan Már Andersen <kjartan23@ru.is>

T-809-DATA
Data Mining and Machine Learning

November 8, 2024

Contents

List of Tables

List of Figures

1 Introduction

This project focuses on studying the effects of Convolutional Neural Networks (CNNs) on image classification tasks. We will investigate the ResNet architecture and evaluate its performance. We will use two common benchmark datasets CIFAR-10 [2] and Fashion-MNIST [3] to train and evaluate our model.

CNNs are fundamental in computer vision, particularly for image classification, making this study highly relevant to understanding the foundations of deep learning.

The main objectives of this project are to study CNN basics and their impact on image classification, implement and experiment with ResNet, evaluating its performance on both datasets and to identify challenges and analyze how ResNet addresses them.

In this report, we begin with a review of the relevant literature, followed by a detailed explanation of our methodology. We then present our experimental setup and preliminary results, concluding with a discussion of our findings.

2 Literature Review

We base our project on the paper "Deep Residual Learning for Image Recognition" [1] in which the ResNet architecture is introduced. This paper explores some of the problems of very deep networks like vanishing and exploding gradients but mainly it explores the degradation problem which is observed when network depth increases, accuracy gets saturated and then degrades rapidly [1]. This problem of degradation is solved in the paper by introducing skip connections between layers as can be seen on Figure 1.

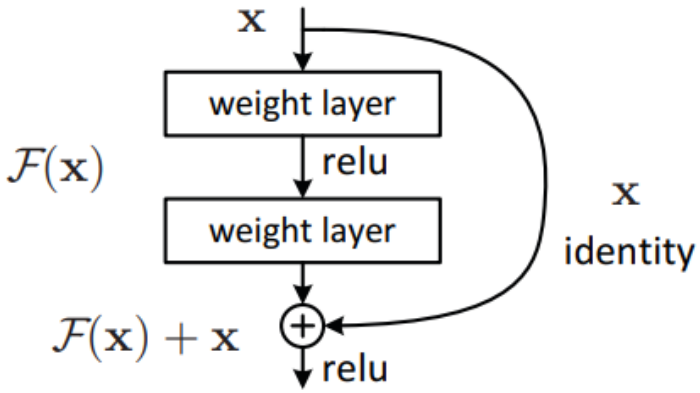

Figure 1: Residual learning building block. Figure from [1].

This was shown to be easily implemented with standard libraries and the network is still trainable using stochastic gradient descent with backpropagation [1] [4]. This approach preforms better than simply adding more layers to a basic networks and was shown to be a general solution to the problems of very deep networks [1]. This property of ResNet makes it an interesting algorithm to explore and get to know better.

3 Methodology

3.1 Data Preprocessing

Data preprocessing involved obtaining the datasets, performing data augmentation, and splitting the data into training and testing sets. Data augmentation is done to increase the performance of the model by expanding

the dataset. This is done by taking images from the original dataset and changing them in some way for example by rotating them, flipping them or changing colours.

As discussed in Section 1 we will be using two common benchmark datasets CIFAR-10 and Fashion-MNIST. These datasets are often used in image classification tasks, they are rather small but due to our limited computational resources they are preferable to larger dataset such as ImageNet [5].

3.1.1 CIFAR-10

The CIFAR-10 dataset contains 60,000 32x32 color images across 10 classes which are: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. The CIFAR-10 dataset was downloaded, augmented and then split into training and testing sets with minimal code. This augmentation is very straightforward with PyTorch due to its built-in functionality using `transforms.Compose()`. The augmentations we applied to CIFAR-10 included random rotations of up to 40 degrees, horizontal flips with a 50% probability, changes in brightness, contrast, and saturation, as well as sharpness adjustments. Lastly, the images were converted into tensors and normalized.

3.1.2 Fashion-MNIST

The Fashion-MNIST dataset contains 70.000 28x28 grayscale images of fashion products with 10 categories that are: T-shirt/top, trousers, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot.

To make the fashion dataset comparable with the CIFAR-10 dataset, we had to process it somewhat differently. One incompatibility was the grayscale 1 channel width of the images. Another incompatibility was the image size 28x28 vs. the 32x32 size of CIFAR-10. To fix these incompatibilities we used the `transforms.Lambda()` function to add two more channels that were repeats of the first single channel, and `transforms.Pad()` to pad the images with 4 pixels.

The dataset was then further augmented by with all of the transformations that the CIFAR-10 used, including rotations, horizontal flips, color adjustments, and sharpening. And finally the images were converted into tensors and normalized.

3.2 Model implementation

For implementing the ResNet model we used PyTorch, a machine learning library for the Python programming language which is based on the Torch library developed for Lua. This allows us to implement the model by inheriting the `torch.nn` model from PyTorch, and then defining the sequence of operations in the model's `forward()` function. The ResNet model accepts the network size n as a parameter that dictates the total number of layers in the ResNet, calculated as $6n + 2$. This enables the creation of various ResNet configurations such (e.g., ResNet-18, ResNet-34, etc.) just by adjusting the n value.

The model is comprised of an input layer, three stacks of residual blocks (Stack1, Stack2, and Stack3) and an output layer. Each stack is configured dynamically by the provided n value.

- **Stack1 (low-level features):** Has n residual blocks with 16 filters, all working at the same spatial resolution to capture low-level features.
- **Stack2 (mid-level features):** Begins with downsampling residual block, reducing spatial dimensions (using stride 2), and increasing the filter size to 32. This transition enables the focusing of more abstract features by reducing the spatial resolution, and the capturing of more complex patterns by increasing the number of filters. This stack increases network depth by $n - 1$ blocks

- **Stack3 (high-level features):** We downsample again to further reduce spatial dimensions and increase the number of filters to 64, enabling the extraction of higher level features and increasing network depth once again by $n - 1$ blocks.

The total layer depth contributions are $2n$ (Stack1) + $2n$ (Stack2) + $2n$ (Stack3) = $6n$. By adding the input and output layers this aligns with our $6n + 2$ formula for the final number of layers.

The Residual blocks are defined within the *blocks* class, and each block consists of two convolutional layers with batch normalization and ReLU activation, facilitating residual learning. The Shortcut (Skip) connections are controlled by the *shortcuts* parameter and are fundamental in ResNet's ability to generate and learn through deep networks. The weights are initialized according to the Kaiming normal initialization method, which is suitable for ReLU activations.

3.3 Experiments

Our goal is to investigate and gain insights into the ResNet model. We will experiment with both layer depth and number of epochs. For layer depth we will compare three models with 20, 32, and 44 layers, respectively. We trained each model for 20, 50, and 100 epochs to observe how training duration impacts accuracy.

For each combination of depth and number of epochs, the same dataset splits and hyperparameters were used to ensure a fair comparison. We also train and test on both datasets CIFAR-10 and Fashion-MNIST.

3.3.1 Training Setup and Hyperparameters

We will conduct our experiments with similar parameters as our source paper [1]. Our optimizer uses Stochastic Gradient Descent (SGD) with momentum. The learning rate starts from 0.1 and is divided by 10 when the milestone of 82 epochs is reached. A batch size of 128 is used for training. Cross-entropy loss is used for classification.

3.3.2 Evaluation Metrics

To evaluate the performance of the models, we used accuracy and test error as our two primary metrics.

Accuracy, as shown in Equation (1), is the ratio of correctly classified images to the total number of images:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{1}$$

Similarly, test error, presented in Equation (2), is calculated as the complement of accuracy, representing the ratio of incorrectly classified images to the total number of images:

$$\text{Test Error} = (1 - \text{Accuracy}) \times 100 \tag{2}$$

We also used a Confusion Matrix to provide insight into the classification performance on each class.

Finally we measured the training time of each model. This is done to assess the computational cost of each model.

3.3.3 Comparison Criteria

The experimental results will be evaluated based on the following criteria:

- **Accuracy Across Epochs:** How accuracy improves or plateaus over time as the number of epochs increases.

- **Layer Depth vs. Accuracy:** The impact of increasing layer depth on accuracy.
- **Training Time vs. Accuracy:** The trade-off between training time and performance.

Additionally, we will compare our findings with the results from the original ResNet paper [1], including both ResNet configurations and plain networks, to provide a comprehensive investigation into the ResNet architecture.

4 Results

Figure 2 highlights the error vs. epochs of ResNet models with 20, 32 and 44 layers using CIFAR-10 across 100 epochs.

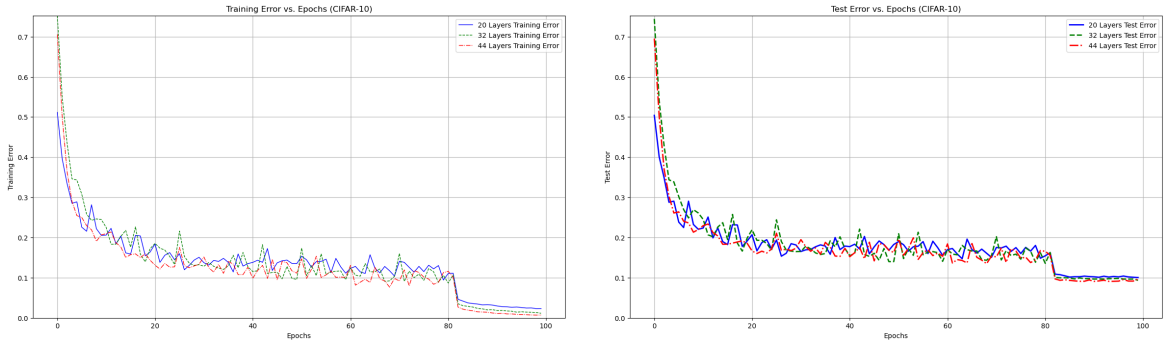

Figure 2: Training and Test Error results from the CIFAR-10 dataset

Figure 3 highlights the error vs. epochs of ResNet models with 20, 32 and 44 layers using Fashion-MNIST across 100 epochs.

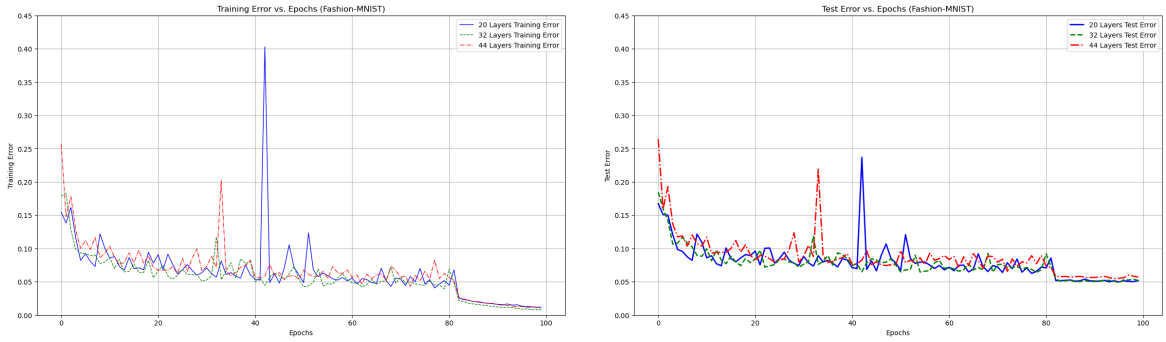

Figure 3: Training and Test Error results from the Fashion-MNIST dataset

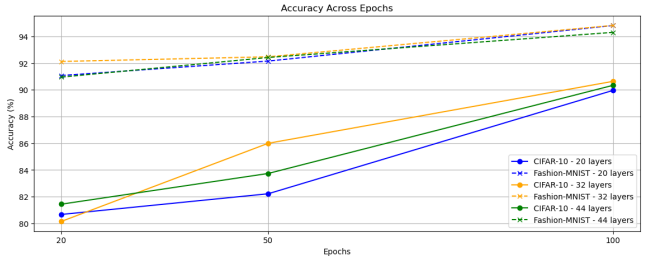

Figure 4: Accuracy Across Epochs

Table 1: Results for CIFAR-10 and Fashion-MNIST with varying layers and epochs.

Epochs	20 layers			32 layers			44 layers		
	20	**50**	**100**	**20**	**50**	**100**	**20**	**50**	**100**
CIFAR-10									
Training time (seconds)	838	2074	4113	973	2429	4851	984	2452	4855
Test error (%)	19,33	17,78	10,04	19,85	14	9,36	18,56	16,26	9,66
Fashion-MNIST									
Training time (seconds)	633	1586	3172	686	1718	3447	729	1819	3633
Test error (%)	8,92	7,84	5,18	7,87	7,52	5,17	9,04	7,57	5,69

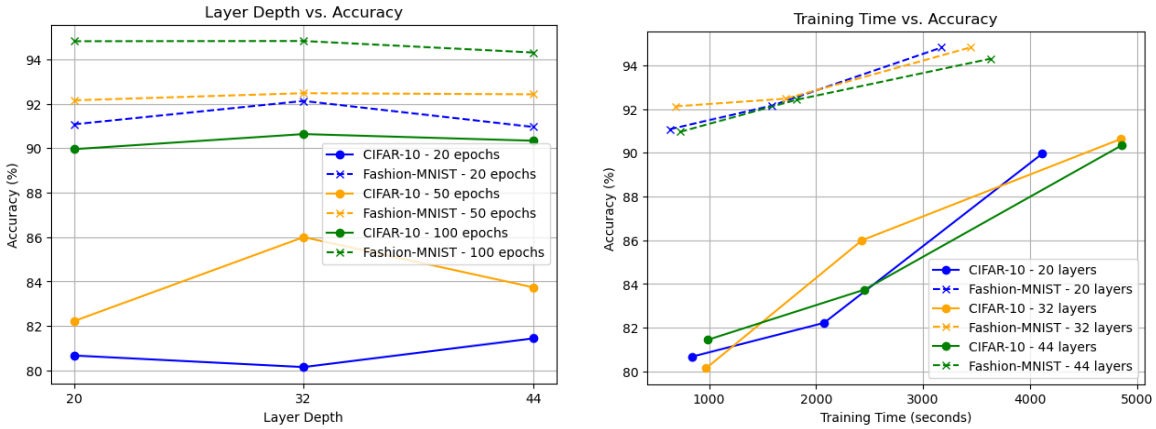

Figure 5: Layer Depth (left) & Training time (right) vs. Accuracy

5 Discussion

Note that the model training on the CIFAR-10 and Fashion-MNIST datasets were done on different computers, so comparing training time differences between them would be inaccurate.

5.1 CIFAR-10

We can see from Figure 2 that there is not much difference in training and test error across models with different layer sizes. We can also see that after about 20-30 epochs the accuracy plateaus for while, then after 82 epochs the learning rate decreases and learning becomes more stable, slowly increasing accuracy until termination.

Looking deeper at the scores from Table 1 we see that the final epoch scores from the model with 32-layers are actually better than the ones from the 44-layer model. This could be an indicator that the deeper model is beginning to exhibit signs of slight overfitting to the training data, and highlighting that a deeper model is not always better. Figure 5 also highlights that the 32-layer model performs better than the 44-layer model around 50 & 100 epochs but is slightly worse in the beginning. We also see that the training time is (obviously) the fastest for the 20-layer model but almost exactly the same for the 32 & 44-layer models.

5.2 Fashion-MNIST

Like with the CIFAR-10 dataset, we can see in Figure 3 that the training and test errors across the models with different layer sizes are very similar. We can however see some spikes in the training error with the 20 & 40-layer models at around the 40 epoch mark. We suspect that this may be due to some parameter that covers

an important, but subtle feature that may have been altered at the spike, and then corrected immediately after. Additionally we see again the learning rate get decreased after 82 epochs, causing the learning to be more stable until termination.
Looking at Table 1, we can see that the 20 & 32-layer models have the best accuracy scores in the end, indicating that the 44-layer model is starting to overfit. We can see this happening across Figure 4 and Figure 5 as well where the shallower models are outperforming the deeper one. This might be indicating that the features required to classify this dataset are fewer overall than the ones defined in deeper models.

5.3 Differences between datasets

Some notable differences that we noticed when training the models on the two datasets are:

- **Spikes in training error**
  The Fashion-MNIST had some major spikes in the training error around the 40 epoch mark, indicating some important, subtle features while the CIFAR-10 had no major spikes.
- **Training time**
  The training time differences between layer depths of the Fashion-MNIST dataset increase more linearly than on the CIFAR-10, which appeared to plateau with the 32 & 44-layer models, which had almost the exact same training time, with only minimal differences.
- **Accuracy floors and longer lasting gains**
  As seen by the left graph on Figure 5, the accuracy score floor for the Fashion-MNIST dataset is much higher, especially in the beginning. The Fashion models started with an accuracy score around 90, while the CIFAR-10 models started with a score at around 82. The model trained on the CIFAR-10 observed longer lasting accuracy gains while the model trained on the Fashion-MNIST plateaued pretty harshly and observed no accuracy gains after the decreasing the learning rate at 82 epochs.

6 Conclusion

In this project we experimented with the impact of ResNet models with different depths on image classification tasks using the CIFAR-10 and Fashion-MNIST datasets.

Some key takeaways from the result of our experimentation include:
Layer Depth vs. Accuracy: The results from our experiments indicate that increasing layer depth only increases accuracy up to a certain point. For the CIFAR-10 models the 32-layer outperformed the 44-layer model by a small margin. This suggests that deeper models may be overfitting on the dataset. For the Fashion-MNIST dataset the accuracy plateaued accross both the 20 and 32-layer model, indicating that the features required to model the dataset may be captured by shallower models.
Training Time vs. Accuracy: The accuracy scores for both datasets consistently increased over longer training time, with the most impact being observed in the beginning. However, after some while, the deeper layered models start to overfit on the data and start to get worse results on the validation set. This highlights the importance of stopping early when the accuracy plateaus to prevent overfitting and that deeper models do not necessarily mean better results.
Dataset-specific differences: Some notable differences between the datasets emerged, some indicating a more subtle features space like the spikes in accuracy when training on the Fashion-MNIST dataset, and some highlighting different accuracy floors and training time dynamics. These findings highlight the need to tune the models differently when employing ResNet architectures.

Future work could explore different hyperparameters tuning, training techniques, or applying similar analyses to more complex datasets to further understand the scalability and limitations of deep ResNet architectures.

This project provides valuable insights into how network depth in convolutional neural networks influences classification performance and highlights the importance of balancing complexity with accuracy.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 770–778. [Online]. Available: https://paperswithcode.com/paper/deep-residual-learning-for-image-recognition

[2] A. Krizhevsky and G. Hinton, "Cifar-10 (canadian institute for advanced research)," 2009, https://www.cs.toronto.edu/~kriz/cifar.html.

[3] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms," 2017, https://github.com/zalandoresearch/fashion-mnist.

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 248–255.

A Appendix - Code Availability

The full implementation of the project, including all scripts for data processing, model training, and evaluation, can be found in the following GitHub repository:

https://github.com/kjartanandersen/DATA_Project.

B Appendix - Confusion Matrices

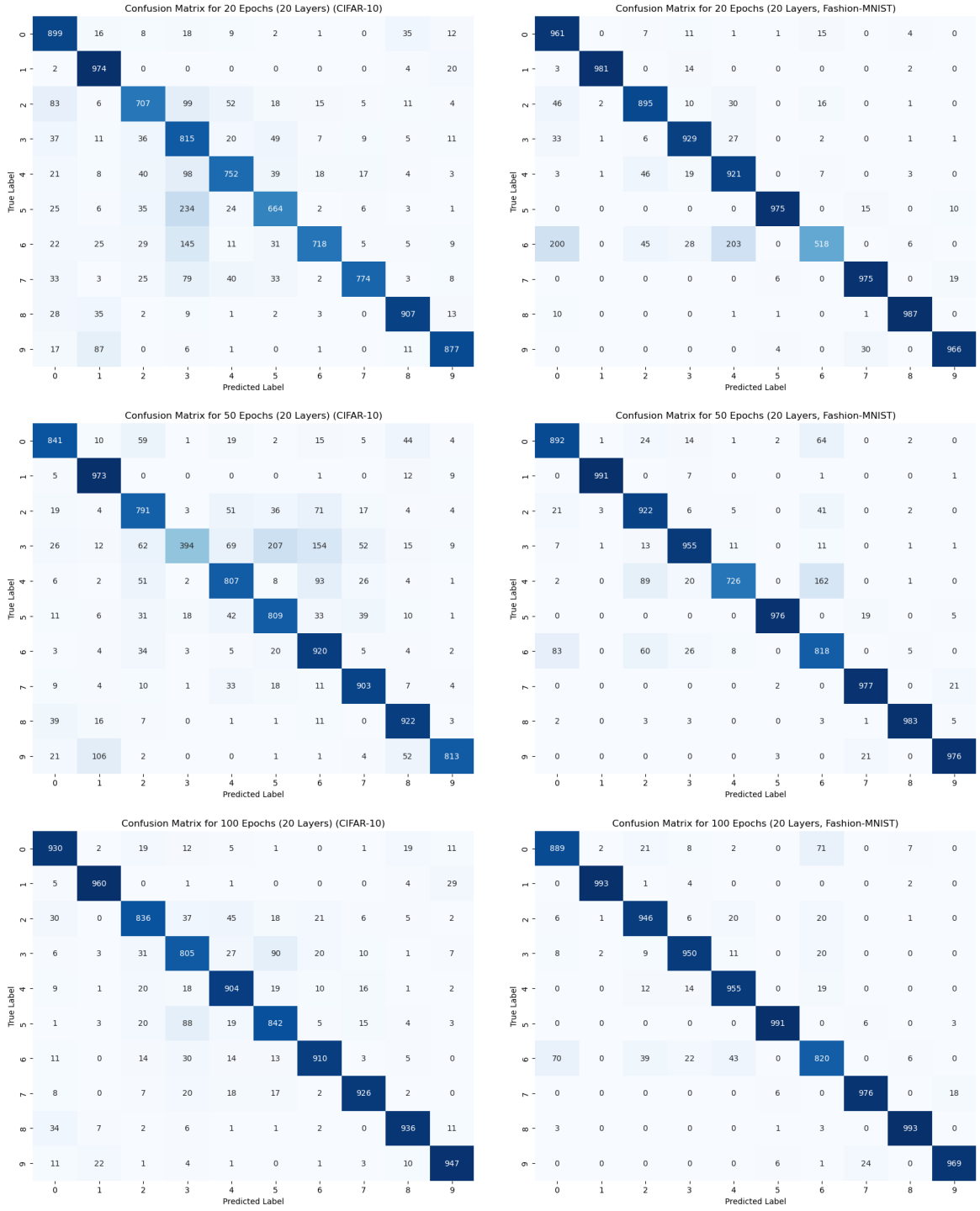

Figure 6: Confusion Matrix Plots - ResNet 20 layers - Left: CIFAR-10 Right: Fashion-MNIST

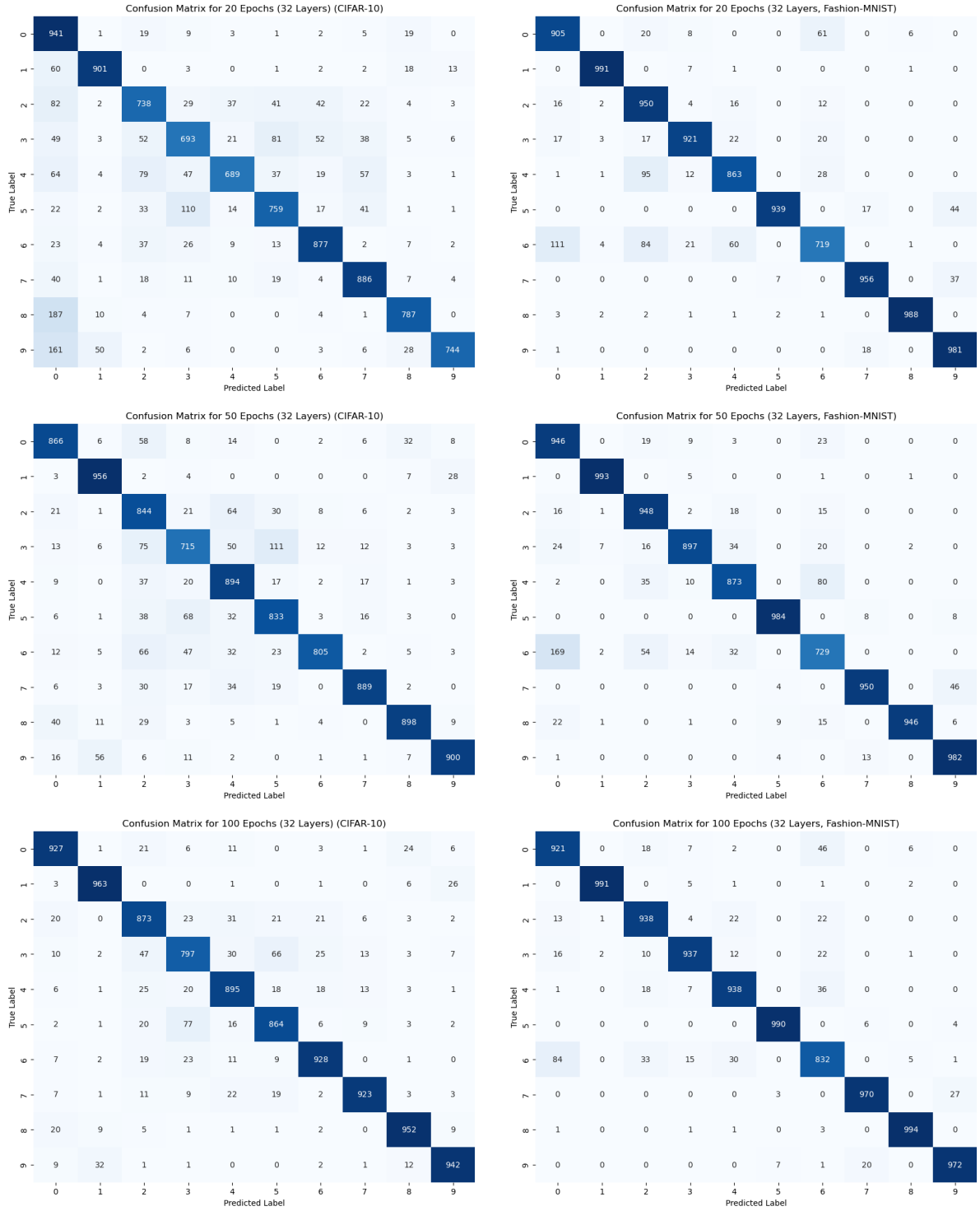

Figure 7: Confusion Matrix Plots - ResNet 32 layers - Left: CIFAR-10 Right: Fashion-MNIST

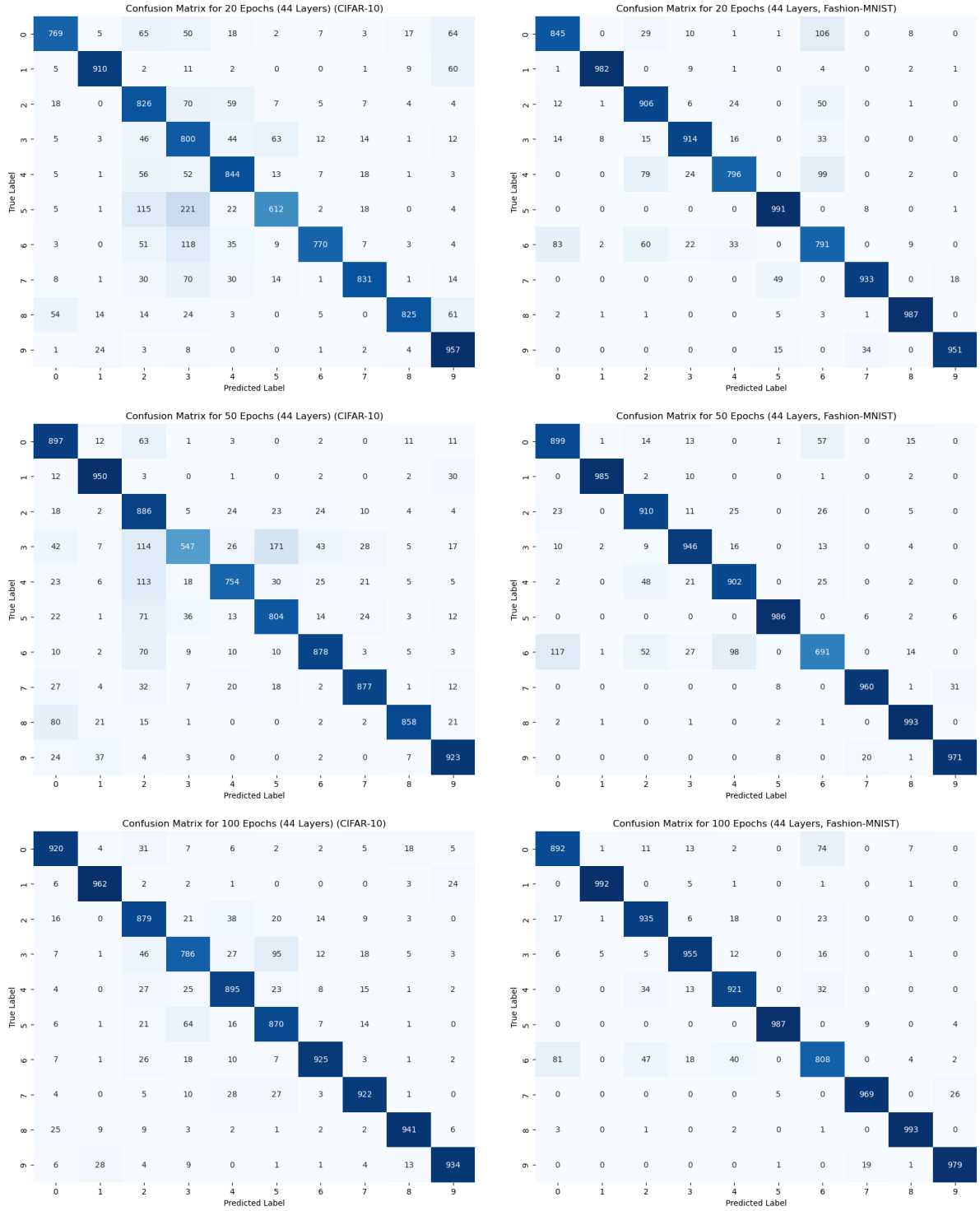

Figure 8: Confusion Matrix Plots - ResNet 44 layers - Left: CIFAR-10 Right: Fashion-MNIST