3.2.1. Image classification

1. Datasets

In this report, we present the results of image classification using deep neural networks on two datasets, FashionMNIST and CIFAR10. The purpose of using these datasets is to evaluate the performance of feature extraction with shallow algorithms and deep neural network algorithms.

FashionMNIST is a dataset containing 70,000 images of fashion products, such as shirts, shoes, and bags. Each image is a 28x28 grayscale picture and is labeled with one of 10 classes. The dataset is intended to be a replacement for the original MNIST dataset, which is a benchmark for machine learning algorithms. It has a training set of 60,000 images and a test set of 10,000 images, which allows for a fair evaluation of the model's performance.

CIFAR10 is a dataset that includes 60,000 32x32 color images of various objects, such as airplanes, cars, birds, and cats, divided into 10 different classes. Each class has 6,000 images. The dataset is known for its diversity and complexity, making it a suitable benchmark for image classification tasks. The goal of using this dataset is to evaluate the model's ability to generalize to real-world images and handle the variations in lighting, pose, and background.

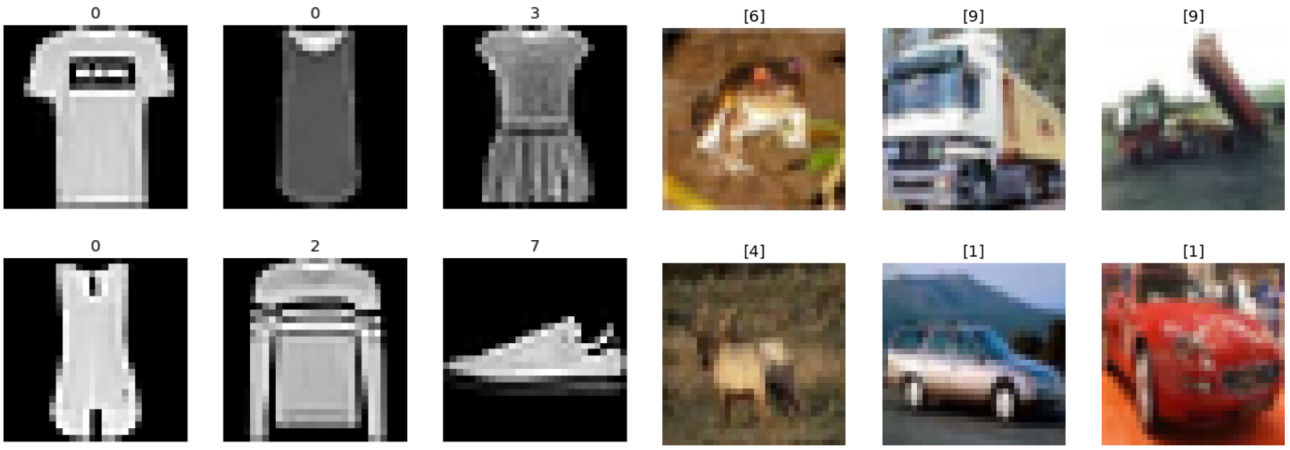


Image 1.1. Left: Fashion-MNIST dataset, right CIFAR-10 dataset

2. Feature extraction and representation

On both datasets, the Fashion-MNIST and the CIFAR-10, we start by extracting some key features using exactly the same procedure - we start by creating some 3-channel color histograms and then we create some image patches from the original ones, to use them in the training part after this one and on the Bag of Visual Words.

So we start by loading one of the datasets and plotting the red, green, and blue channels of a random image, creating a 3-channel histogram. We also show it and print it onto the screen. After that we store all of those color channels so that we can use them later and we do this for each image of both datasets. It also takes approximately less than 0.6 seconds for both datasets to load in memory, showing that there's no difference between the datasets. We can also note that the histograms are radically different for images of different datasets since the MNIST's images are on grayscale by default.

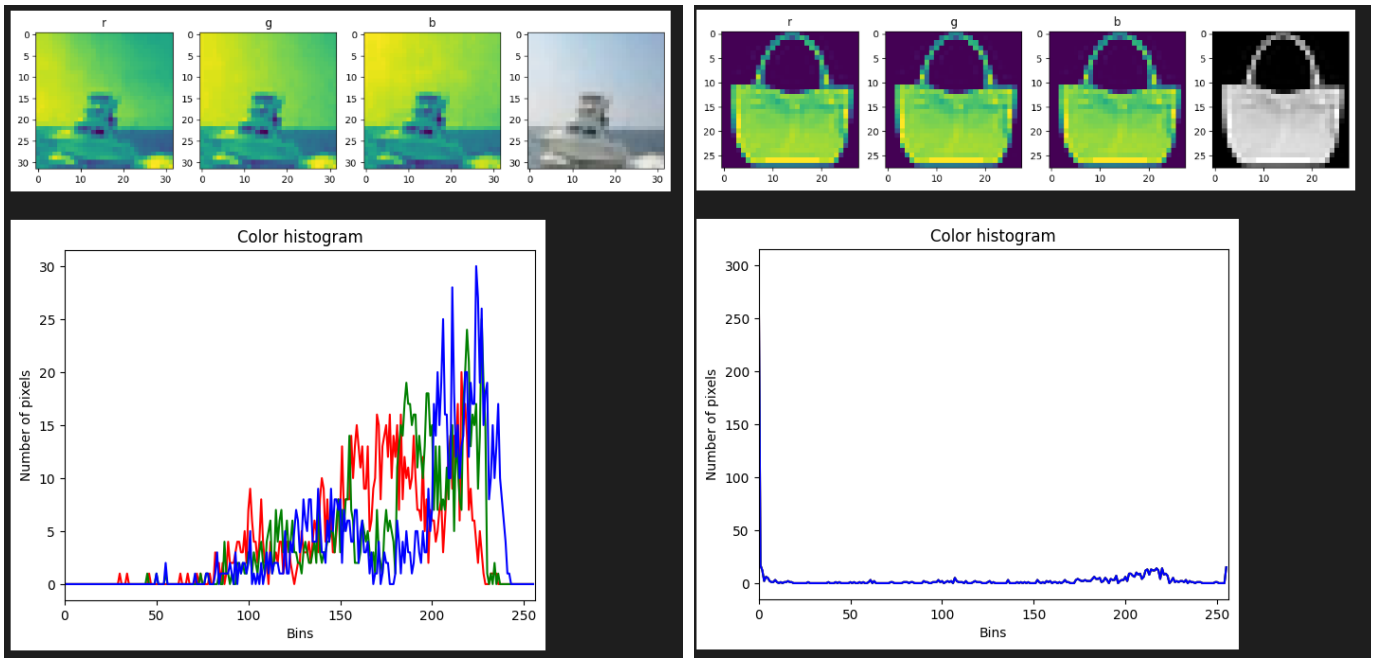


Image 2.1. Histogram CIFAR-10 on the left and fashion-MNIST on the right

On the next step we define some functions that return a list of patches from the images on the datasets, and we store them for later use. In this case we use both a random sample function to test the functionalities of it and a KMeans algorithm to get more reliable patches.

Some of the patches taken from the original images on the last page, right corner.

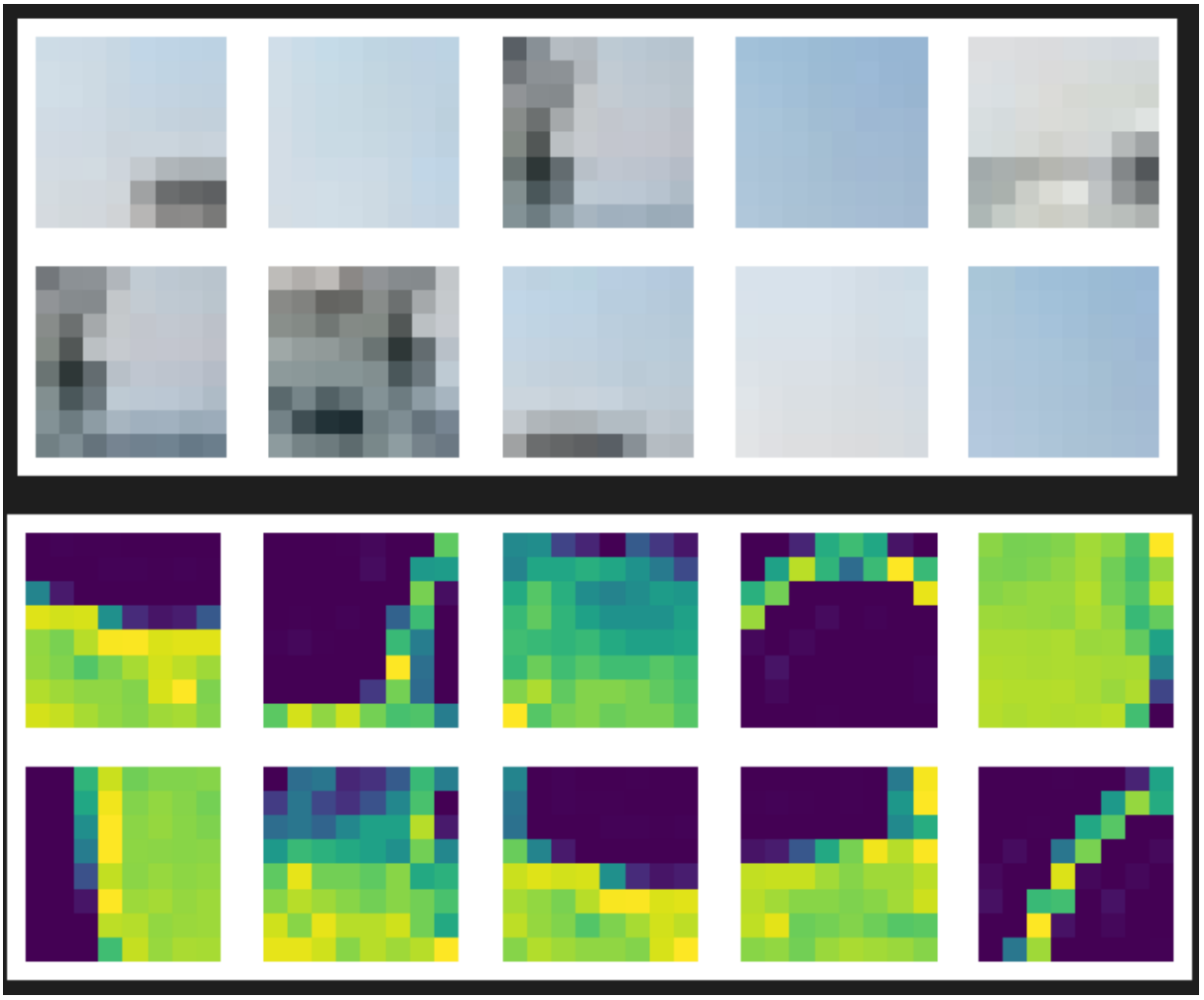


Image 2.2. Example of patches

Finally, we use the SIFT feature detection algorithm from OpenCV to extract features from the patches. Here we calculate the keypoints and descriptors for each of the patches. This part also takes quite some time to load since we have to resize all the images by a significant amount to be able to see the key points better.

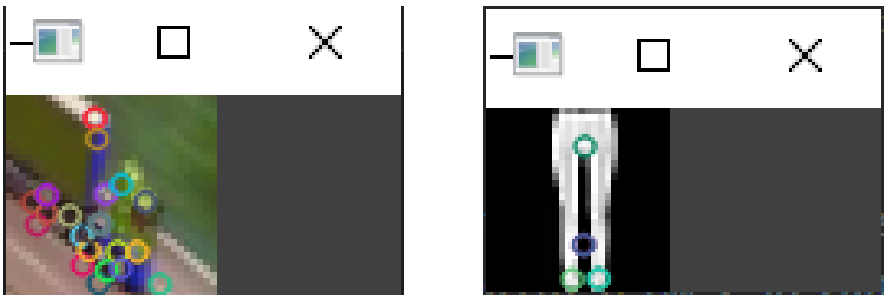


Image 2.3. Visualization of keypoints

To end this part, we will now create a Bag of Visual Words Representation of the SIFT features we just found and plot the newly learned vocabulary. We chose 20 as the ideal number of clusters for the learning step. Also important to note, the CIFAR10 took around 3 minutes to learn all of the vocabulary while Fashion-MNIST took basically 1 minute in the same task.

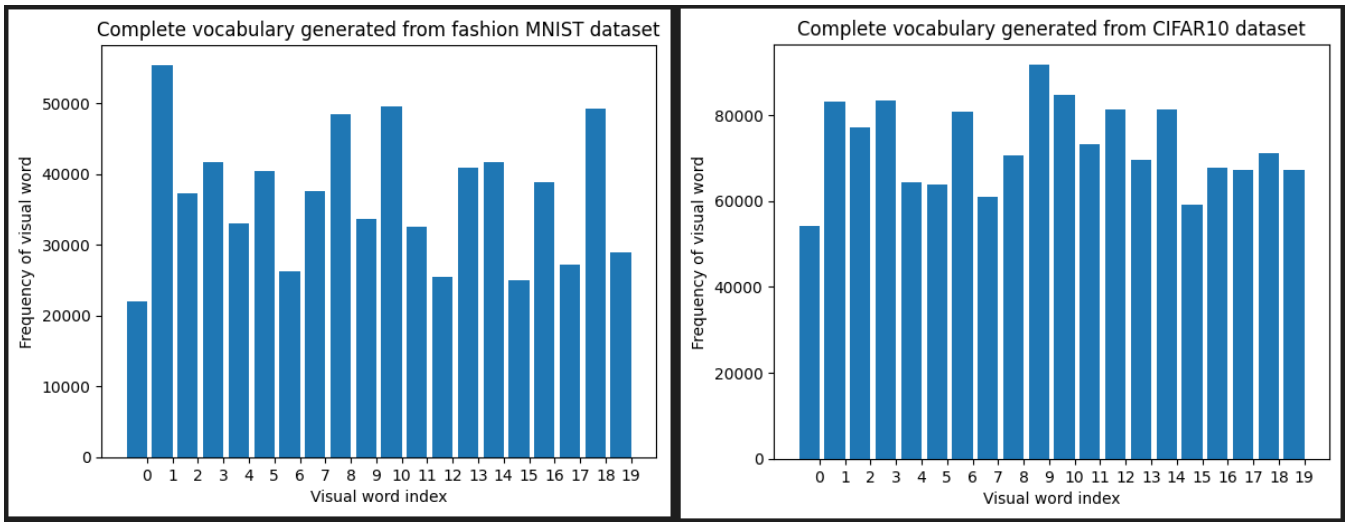


Image 2.4. Histograms of dataset vocabulary

3. Shallow algorithm

3.1. Models

In this project two shallow algorithms were used to classify images using the extracted features: support vector machine and decision tree. Support vector machines classify data by transforming the data to a higher dimension and then finding a decision boundary in this dimension. It finds an optimal decision boundary by maximizing the distance between training data points of the different classes. For this exercise the sklearn support vector machine was used. Decision trees classify data by choosing the best feature to split on at each split in the tree. The best feature is defined using some criterion such as information gain or Gini impurity. Again the sklearn implementation of decision trees was used.

3.2 Fashion-MNIST

In the Fashion-MNIST training dataset there were about 400 images for which no SIFT features were found, these were excluded from training. Another option would have been to add another bin to the histogram for missing features.

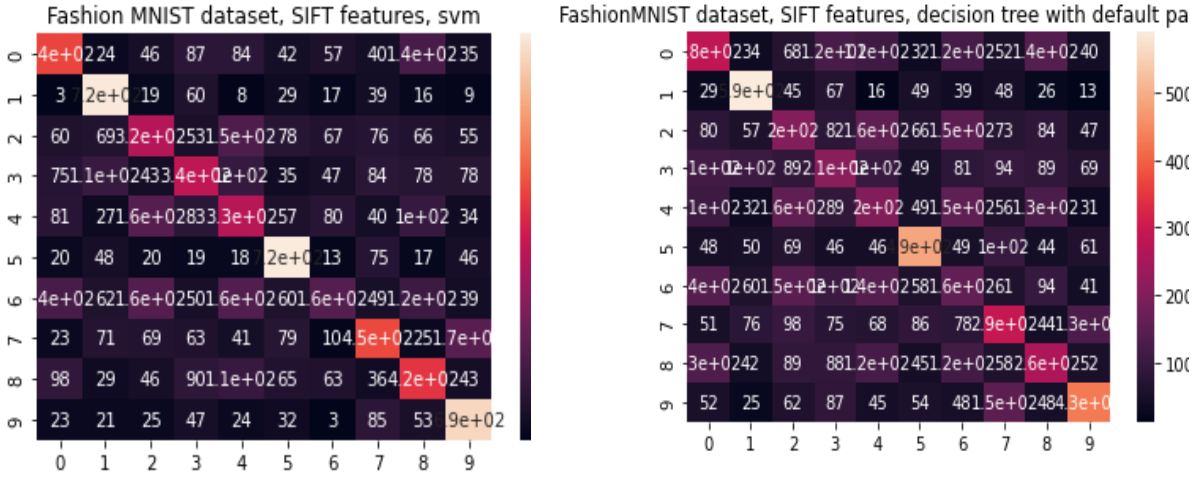


Image 3.1. Confusion matrices when using SIFT features

Confusion matrices for the Fashion-MNIST dataset show similar patterns for both algorithms when using the SIFT features. Class 6 (shirt) seems to be the hardest to classify correctly and is often mistaken for similar classes. For example the number of times it has been classified as class 4 (coat) is the same as the number of times it has been classified correctly. The easiest classes to identify based on SIFT features are 1 (trousers), 5 (sandals) and 9 (ankle boot) for both algorithms. The diagonal is clearly visible in both matrices, more so when using a support vector machine. This means that with support vector machines higher accuracy was reached, as can also be seen from table 1 later.

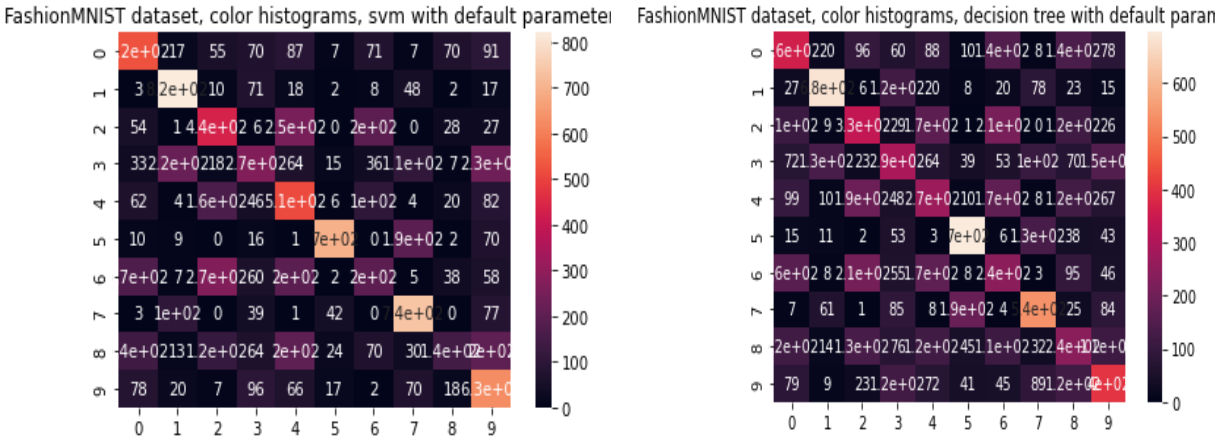


Image 3.2.

Image 3.2. Confusion matrices when using color histograms

The confusion matrices when using the color histograms as features again show similar patterns for both algorithms. There is more confusion when using decision trees for classification. Again the clothing items that resemble each other the most are most difficult to classify, see for example classes 4 (coat) and 6 (shirt). Similarly

to SIFT features, trousers and sandals were the easiest to classify correctly, but instead of ankle boots the third easiest was class 7 (sneakers).

3.2. CIFAR10

In the CIFAR10 training dataset there were 13 images for which no SIFT features were found, these were excluded from training.

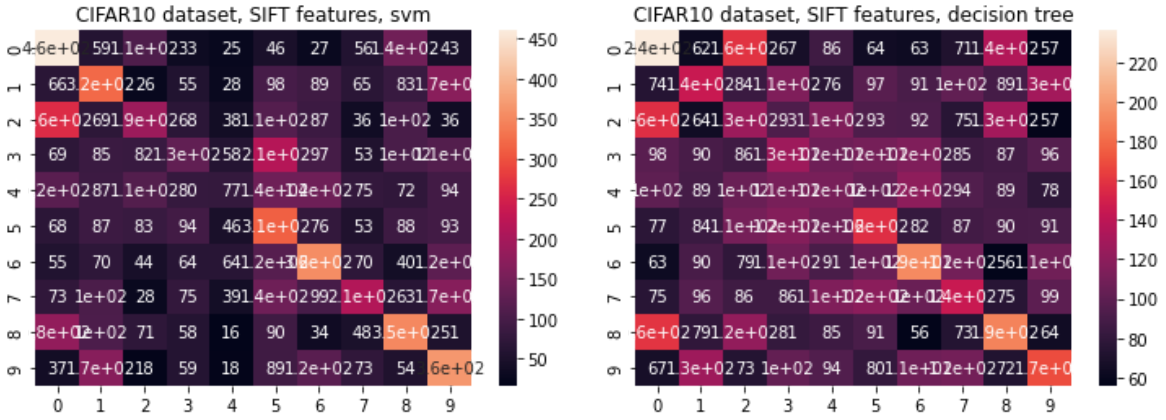


Image 3.3. Confusion matrix when using SIFT features

Compared to the other dataset, the confusion matrices for CIFAR10 show more confusion and the diagonal is not as clear. Notable confusions are class 2 (birds) being classified as class 0 (airplanes), which shows up clearly in both matrices. The most difficult class to classify correctly appears to be class 4 (deer) and the easiest class 0 (airplanes). In general distinctions between the two categories of animals and inanimate objects are easier and confusion happens within these two categories: for example mistaking class 3 (cats) for class 5 (dogs). The already mentioned exception is birds being classified as airplanes.

Table 1: comparison of shallow algorithms	Runtime in seconds: training	Runtime in seconds: testing	Accuracy
CIFAR10, SIFT, svm	237.029	59.585	0.287
CIFAR10, SIFT, decision tree	0.435	0.0038	0.158
CIFAR10, color histogram, SVM	2131.360	509.816	0.352
CIFAR10, color histogram, decision tree	29.300	0.0133	0.199
FashionMNIST, SIFT, SVM	357.244	121.353	0.464
FashionMNIST, SIFT, decision tree	0.4033	0.004	0.314

FashionMNIST, color histogram, SVM	1057.079	261.546	0.497
FashionMNIST, color histogram, decision tree	5.0758	0.007	0.406

Table 1 shows the same conclusions as the confusion matrices: the dataset FashionMNIST has higher classification accuracy with both algorithms and feature extraction techniques. Runtimes are higher for support vector machines, but still fast, which is the advantage for using simple, shallow algorithms. The support vector machine performs better than the decision tree, but the performance of both of these shallow algorithms shows that they are not ideal for this task and other techniques, such as deep learning, should be considered.

Table 2: feature extraction times	Feature extraction time is seconds (training set)	Time in seconds it took to make histogram (training set)
CIFAR SIFT	144.386	562.880
CIFAR10 colors	0.444	20.283
FashionMNIST SIFT	82.505	250.756
FashionMNIST colors	0.200	6.963

Table 2 shows the feature extraction times. From the table it is clear that SIFT features take a longer time to extract and creating a histogram from them also took a longer time.

4. Deep neural network

4.1. Models

In this project, three different models were used to classify images from the FashionMNIST and CIFAR10 datasets. The first model is LeNet, which is a well-known model developed by Yann LeCun in 1998. LeNet is a simple convolutional neural network (CNN) architecture that consists of two convolutional layers, followed by two fully connected layers. It is a relatively shallow model, with only around 60,000 parameters, making it suitable for small datasets such as FashionMNIST.

The second model is a custom-implemented model with 2 convolutional and 2 fully connected layers. This model has 250,902 parameters, which is significantly more than LeNet. The goal of using this model is to evaluate the effect of increasing the model's capacity on the classification accuracy.

The third model is ResNet, which is a deep residual neural network developed by Microsoft Research in 2015. ResNet is a state-of-the-art model that has been trained on the ImageNet dataset, which is a large dataset containing millions of images. We used transfer learning to fine-tune ResNet on the FashionMNIST and CIFAR10 datasets. The goal of using this model is to evaluate the performance of a state-of-the-art model on the datasets and to compare it with the custom-implemented model and LeNet.

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
dropout (Dropout)	(None, 15, 15, 32)	0
conv2d_4 (Conv2D)	(None, 13, 13, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 64)	0
dropout_1 (Dropout)	(None, 6, 6, 64)	0
flatten_1 (Flatten)	(None, 2304)	0
dense_2 (Dense)	(None, 100)	230500
dense_3 (Dense)	(None, 10)	1010

Total params: 250,902
Trainable params: 250,902
Non-trainable params: 0

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 6)	456
average_pooling2d (AveragePooling2D)	(None, 14, 14, 6)	0
activation (Activation)	(None, 14, 14, 6)	0
conv2d_1 (Conv2D)	(None, 10, 10, 16)	2416
average_pooling2d_1 (AveragePooling2D)	(None, 5, 5, 16)	0
activation_1 (Activation)	(None, 5, 5, 16)	0
conv2d_2 (Conv2D)	(None, 1, 1, 120)	48120
flatten (Flatten)	(None, 120)	0
dense (Dense)	(None, 84)	10164
dense_1 (Dense)	(None, 10)	850

Total params: 62,006
Trainable params: 62,006
Non-trainable params: 0

Image 4.1. Comparison between own and LeNet

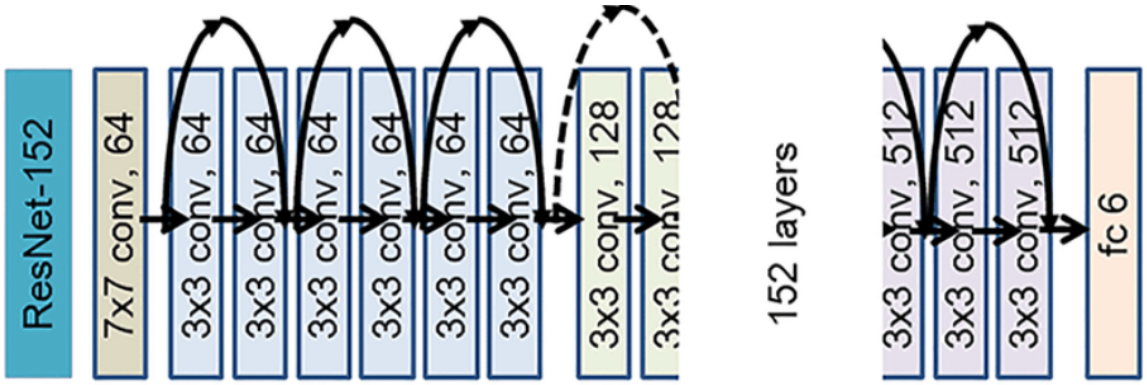


Image 4.2. ResNet architecture

4.2. Fashion-MNIST

For this dataset to be compatible with model input sizes we had to first resize the images to 32x32.

LeNet had the fastest runtime, completing the training and testing in a total of 134 seconds. In terms of accuracy, LeNet achieved an accuracy of 82% on the test set.

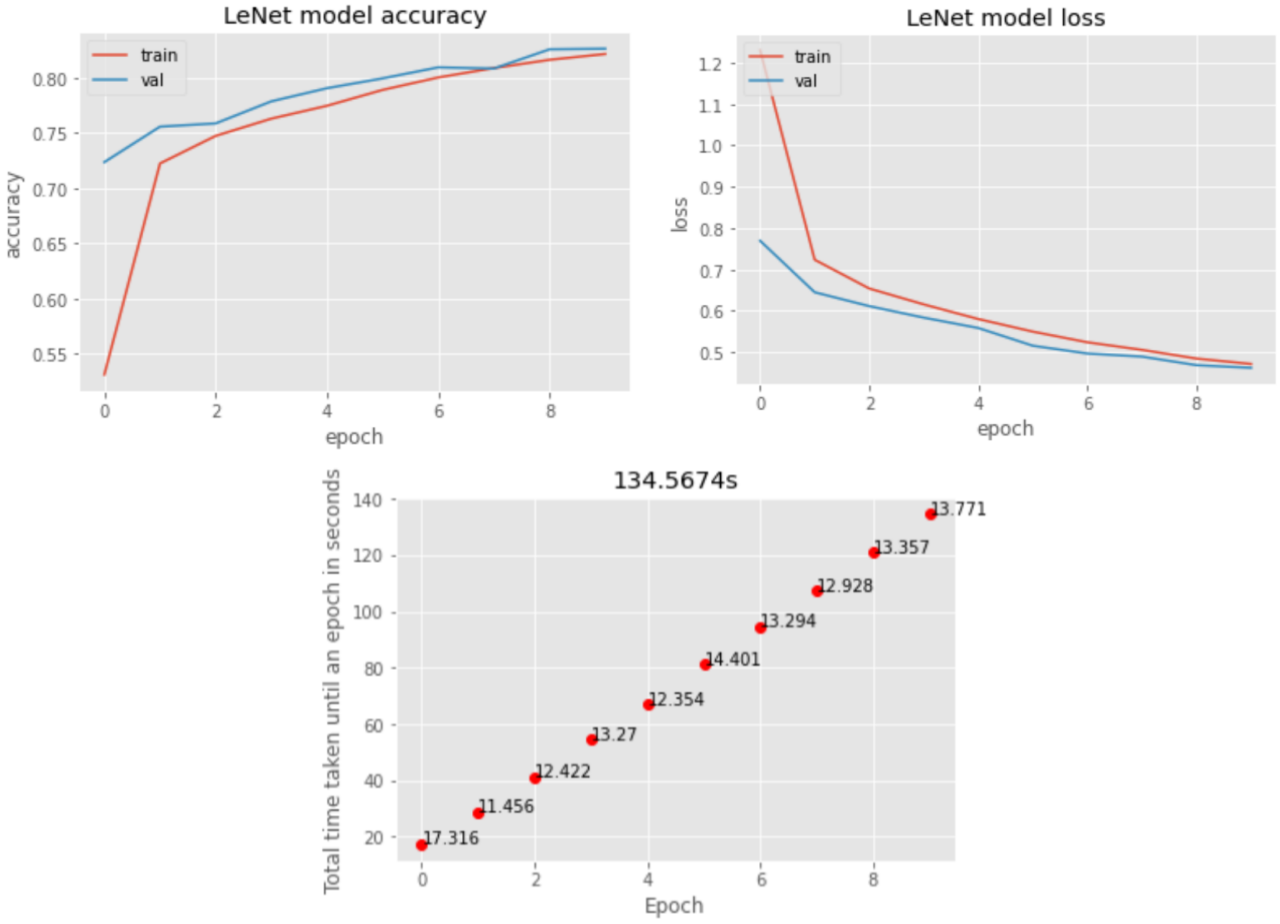
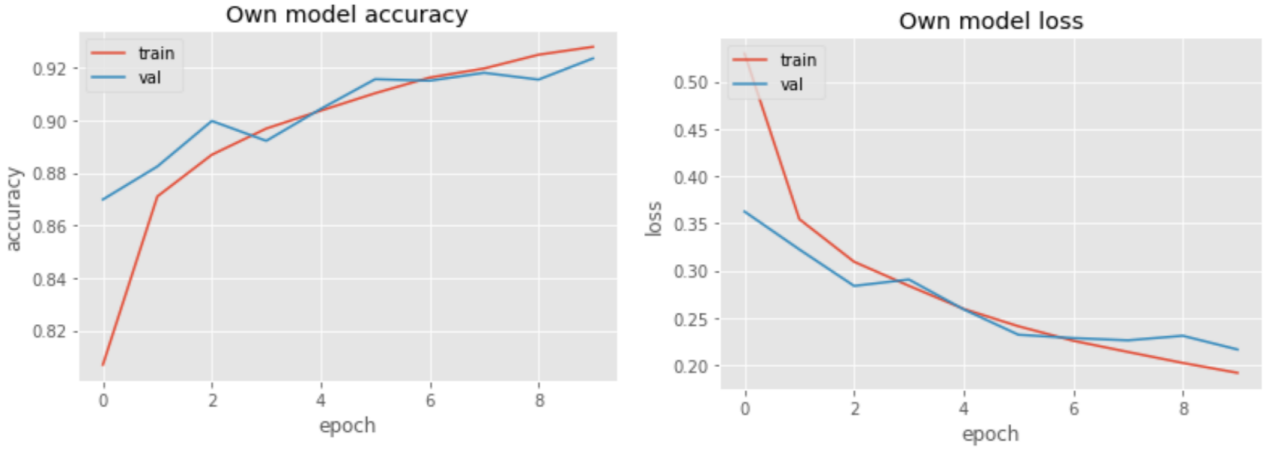


Image 4.3. Fashion-MNIST training overview for LeNet

The custom-implemented model had a slower runtime, taking 344 seconds to complete the training and testing. However, it achieved a higher accuracy of 91% on the test set.



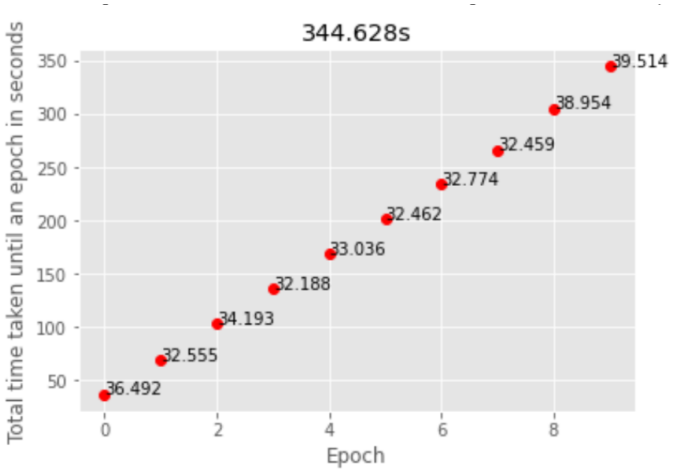


Image 4.3. Fashion-MNIST training overview for own model

The ResNet model had the slowest runtime, taking almost 6000 seconds to complete the training and testing. In terms of accuracy, ResNet achieved an accuracy of 75% on the test set.

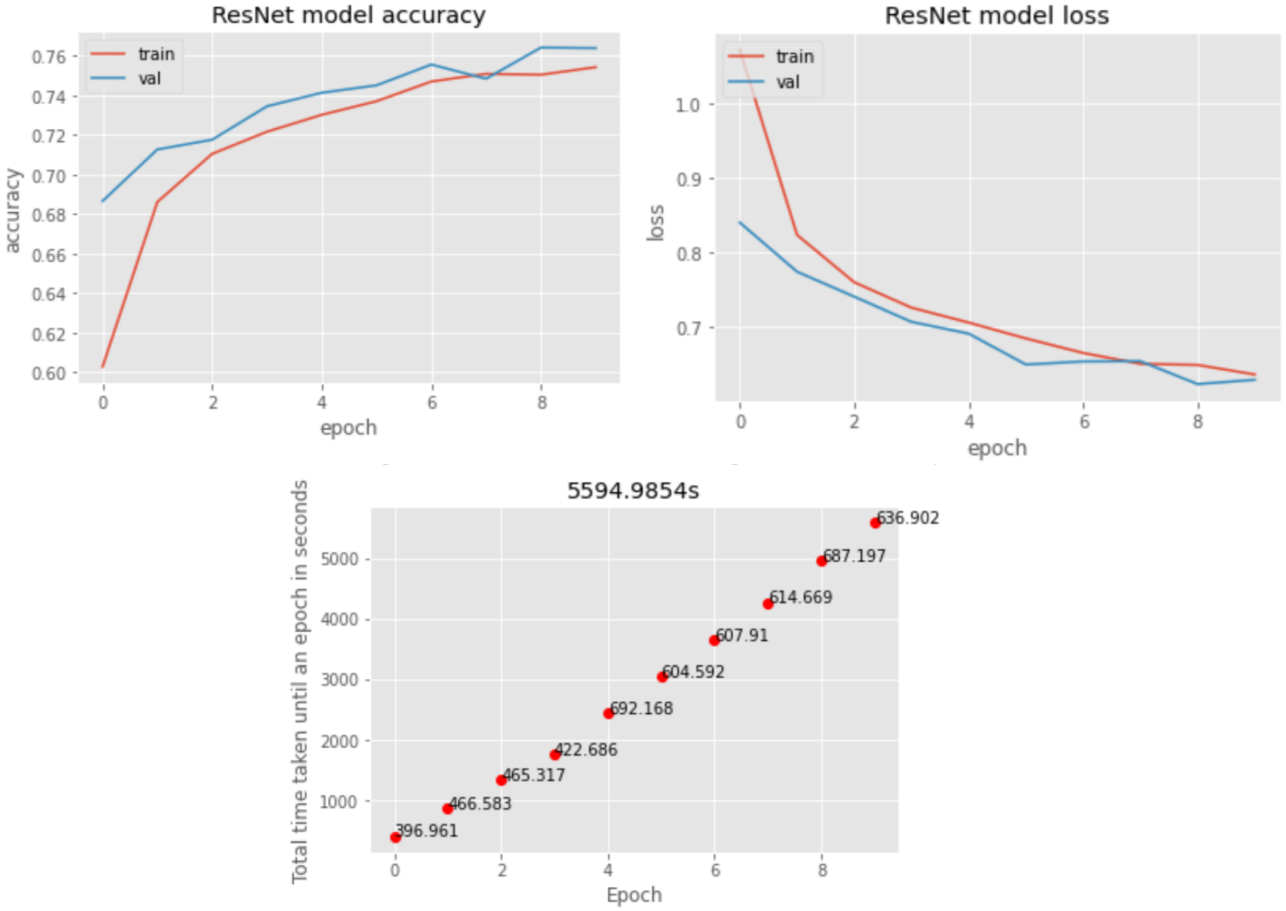


Image 4.4. Fashion-MNIST training overview for ResNet

Overall, the custom-implemented model had the best trade-off between runtime and accuracy, with a slower runtime but the best results in terms of accuracy on the test set. This suggests that increasing the capacity of the model can lead to better performance, but at the cost of longer training and testing times.

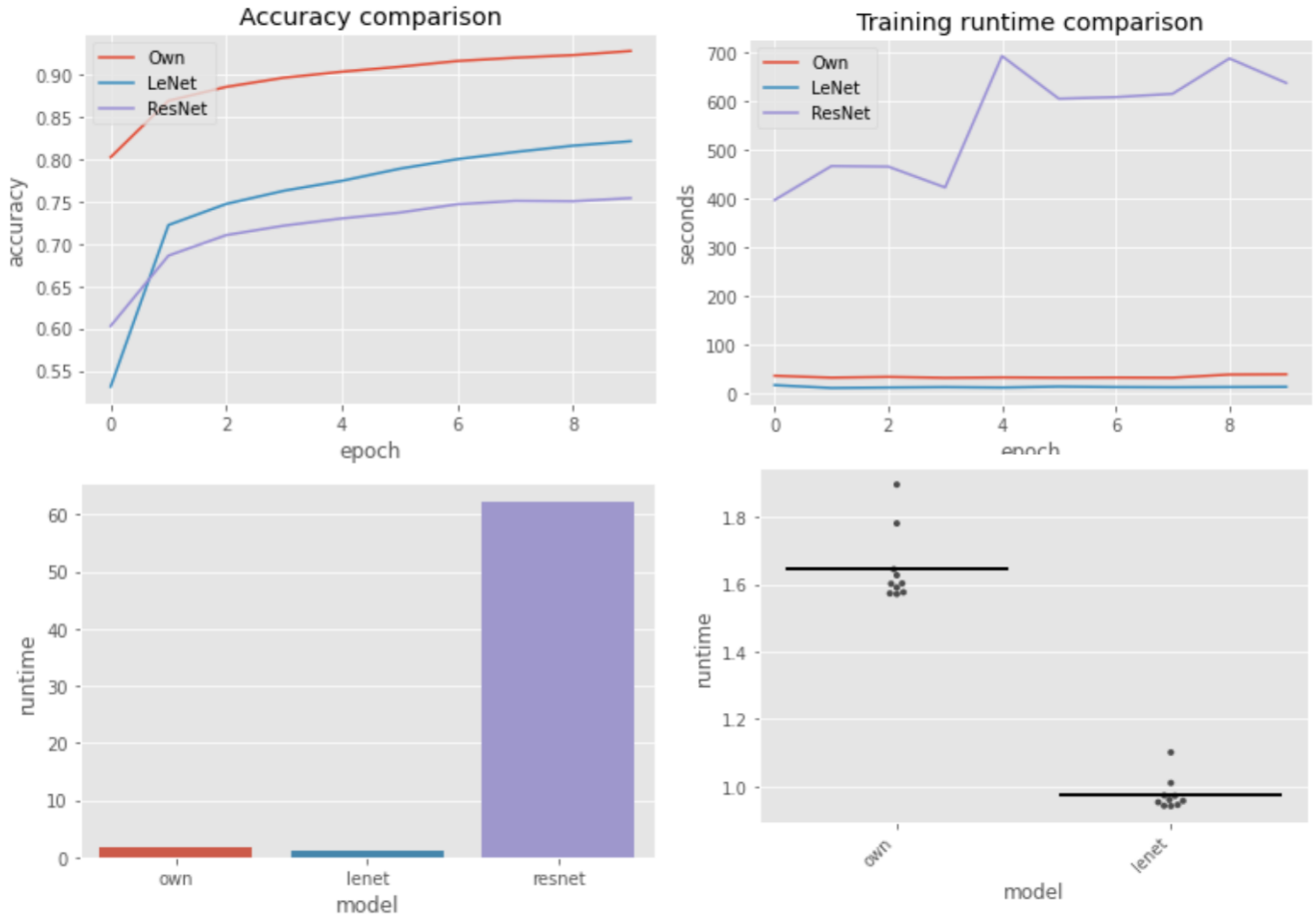


Image 4.5. Overview of all models for training and testing

4.3. CIFAR10

In the experiments on the CIFAR10 dataset, several methods were used to improve the performance of the models. Initially, the inputs were resized to 100x100 using interpolation to increase the resolution of the images. However, this approach proved to be too computationally expensive for the available hardware and was not pursued further.

Next, the inputs were resized to 50x50 using interpolation, but this did not result in any significant improvement in performance.

Different models were also tried for transfer learning, including Xception, Inception, and ResNet. ResNet was found to be the best performing model, with an accuracy of

8% on the training set. However, when tested on the test set, the accuracy dropped to 59%, which suggests overfitting.

Additional fully connected layers were added to the end of the ResNet model, but this only resulted in a further decrease in performance.

In terms of runtime, LeNet was the fastest model, completing the training and testing in a total of 85 seconds and achieving an accuracy of 41% on the test set.

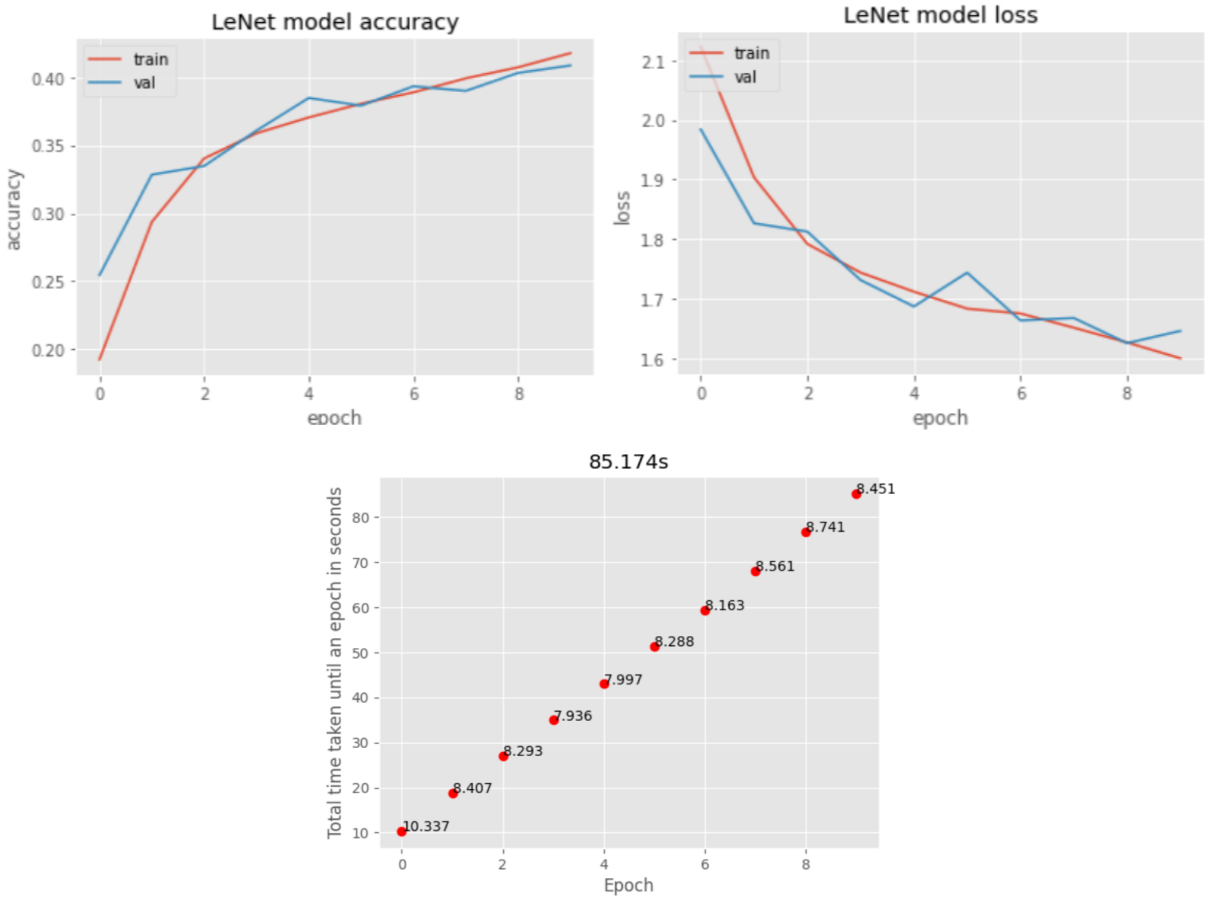
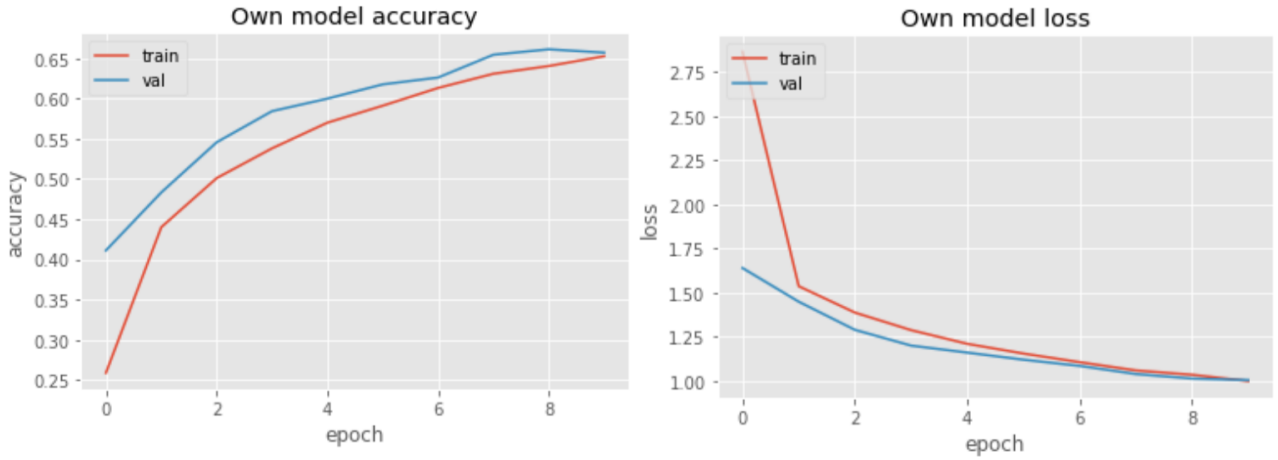


Image 4.6. CIFAR-10 training overview for LeNet

The custom-implemented model was slower, taking 288 seconds to complete the training and testing and achieving an accuracy of 65% on the test set.



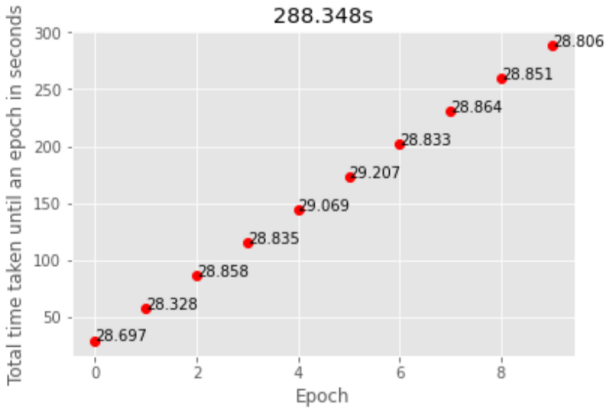


Image 4.7. CIFAR-10 training overview for own model

ResNet, which showed the most promising performance on the training set, had the slowest runtime, taking 4352 seconds to complete the training and testing.

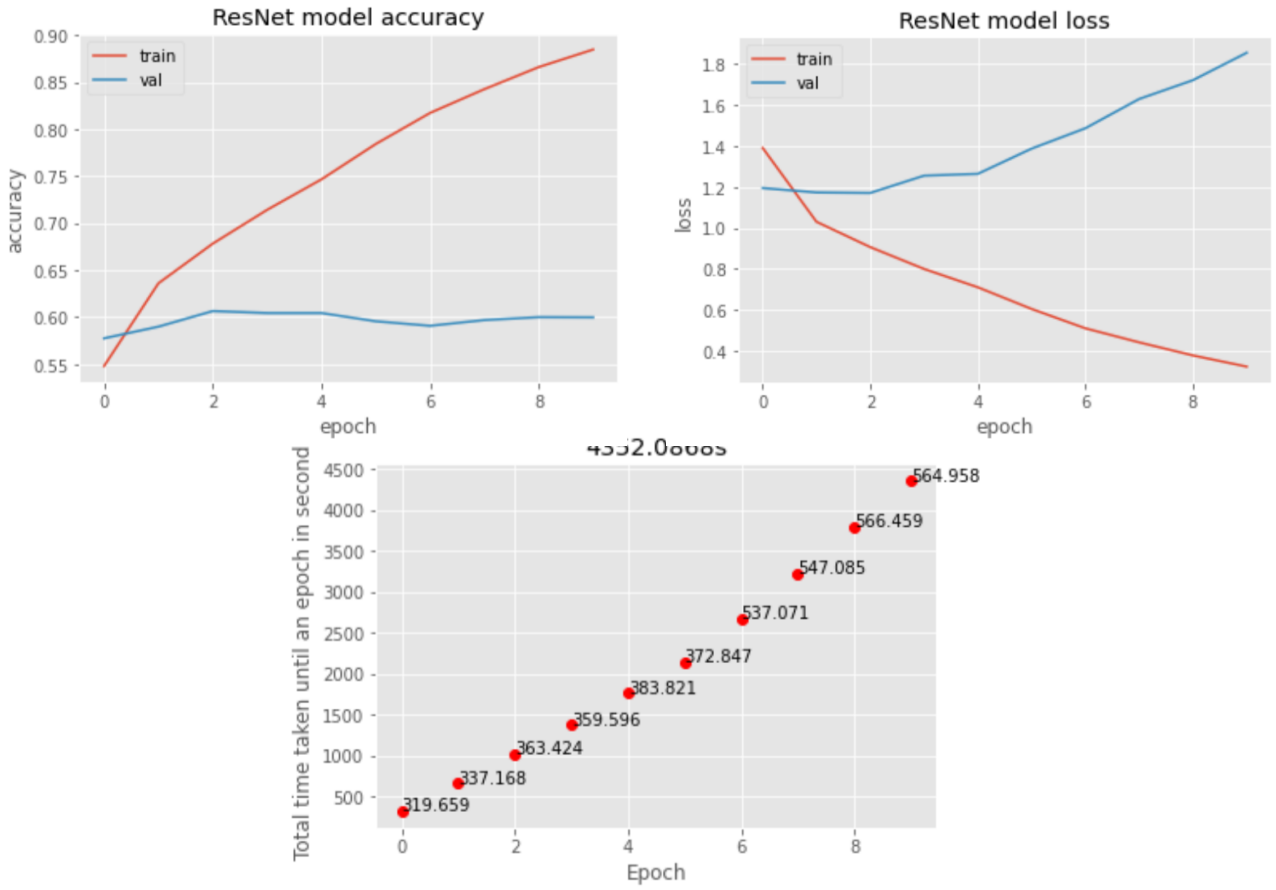


Image 4.8 CIFAR-10 training overview for ResNet

In conclusion, the ResNet model has a lot of potential for the CIFAR10 dataset, but it requires addressing the issue of overfitting and its high computational cost.

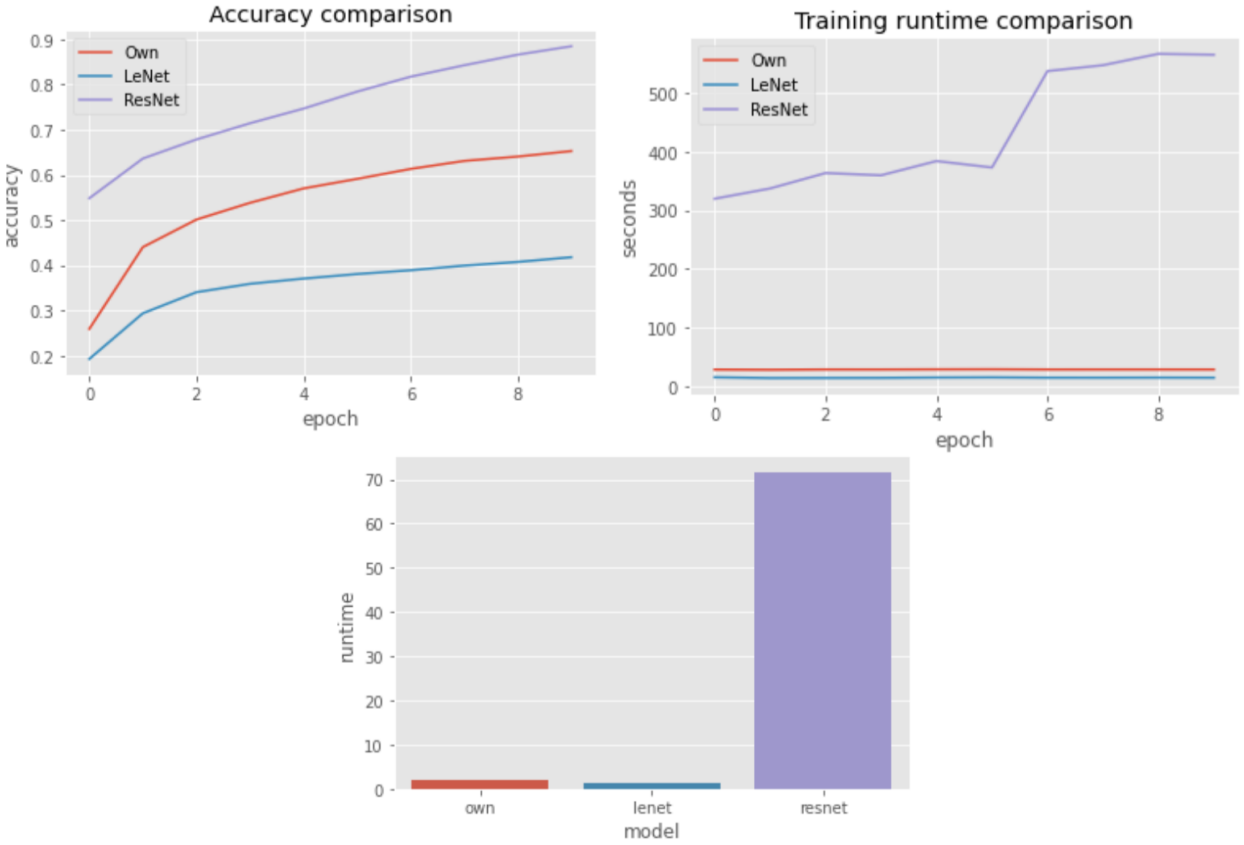


Image 4.9. Overview of all models for training and testing

5. Conclusion

In conclusion, the results of the experiments show that both shallow and deep learning algorithms were evaluated for image classification on the FashionMNIST and CIFAR10 data sets.

For shallow algorithms, it was found that SVM is slower than decision trees but more accurate, and that color histograms are a better feature representation than SIFT descriptors. The best result for CIFAR10 was 35% using color histograms and SVM. Similarly, the best result for FashionMNIST was almost 50% using color histograms and SVM.

For deep learning algorithms, the custom-implemented model had the best trade-off between runtime and accuracy for the FashionMNIST dataset, with a slower runtime but the best accuracy of 91%. On the other hand, for CIFAR10, the ResNet model showed a lot of potential, but it requires addressing the issue of overfitting and its high computational cost. The custom-implemented model was the best with 65% accuracy.

Overall, it is important to consider the trade-offs between runtime, accuracy, and model complexity when choosing a suitable algorithm for image classification tasks.