ECS 189G-001

Deep Learning

Winter 2024

Course Project: Stage 3 Report

Team Information

Enter Your Team Name Here (delete the extra rows if your team has less than 4 students)		
Denis Savytski	920000274	dsavytski@ucdavis.edu

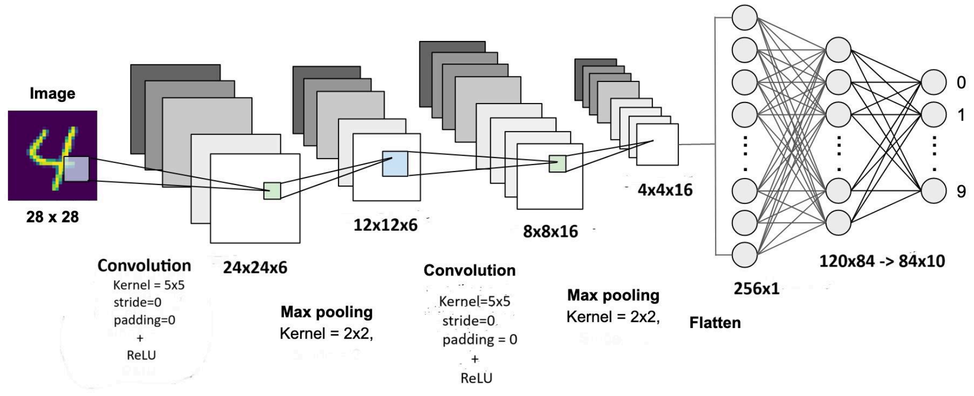
Section 1: Task Description

Train 3 CNN models for MNIST, CIFRA, and ORL datasets. Experiment with different activation functions, loss functions, adding dropouts, different learning rates, and batch sizes affect accuracy and loss.

Section 2: Model Description

All of the models are CNN

1) MNIST



2) CIFAR
   CIFAR which has been used has an identical model to MNIST with the same convolutions and pooling layers. The only difference is in dimensions.
   32x32x3 -> 28x28x6 -> 14x14x6 -> 10x10x16 -> 5x5x16 -> 400x1 -> 256x1 -> 128x1 -> 10x1
3) ORL has one more convolution layer, and one more Maxpool layer, and convolution layers use kernels of size 3 instead of 5. Dimensions: 112x92x1 -> 110x90x32 -> 55x45x32 -> 53x43x64 -> 26x21x64 -> 24x19x128 -> 12x9x128 -> Flatten -> Dropout(0.5) -> 13824x1 -> 256x1 -> Dropout(0.5) -> 128x1 -> 40x1

Section 3: Experiment Settings

3.1 Dataset Description

All of the datasets have been pre-partitioned into training and test datasets. All three datasets represent images.

1) MNIST - 60000 (train) + 10000(test) examples of handwritten digits represented as GrayScale, where each observation is a 28x28 matrix of grayscale values (between 0 and 255)
2) CIFAR - 50000 (train) + 10000(test) examples of 10 different objects represented in RGB, Where each observation is 3x32x32 tensor. Each of the three 32x32 matrices represented the respective RGB channel.
3) ORL - 360 (train) + 40 (test) examples of 40 different people represented in grayscale, each observation is 3x112x92 tensor. Each of the three 112x92 matrices represented identical grayscale channels, therefore data is transformed into a 1x112x21 tensor and ignores two channels.

3.2 Detailed Experimental Setups

All models initially used a similar setup. Mini-Batch GD (with ADAM optimizer) and batch_size=64,
CrossEntopyLoss, learning rate of 0.001. The difference was in the number of epochs. More parameter tuning is explored in 3.7

1) **MNIST**
   The model converges quickly and shows high accuracy, therefore only 10 epochs have been used.
2) **CIFAR**
   The model takes a long time to converge, therefore 100 epochs have been utilized. However, the problem is that the model becomes too biased for the training set, and achieves an accuracy of 90% +, but fails to perform better than 50% on the test set.
3) **ORL**
   The model converges quickly, therefore only 50 epochs were used. However, due to the small number of observations, it quickly becomes biased towards the training set and achieves 100% on all metrics, even with dropout.

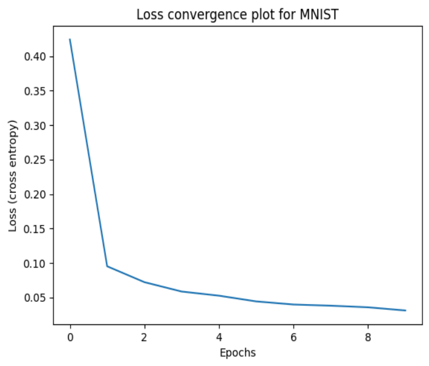
3.3 Evaluation Metrics

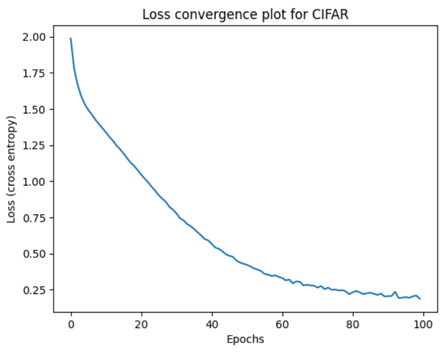
We will use Accuracy, Precision, Recall, F1 score in the experiment as our evaluation metrics.

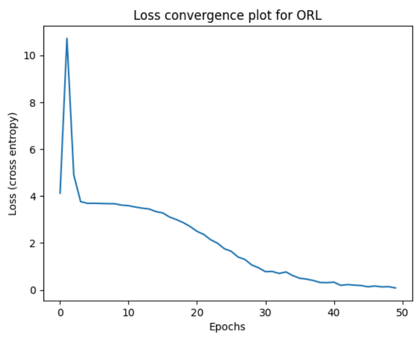
3.4 Source Code

https://github.com/BoyyyxD/ecs189_w2024

3.5 Training Convergence Plot



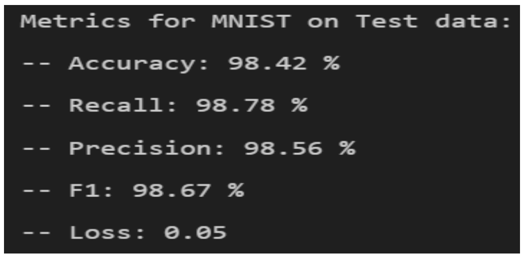




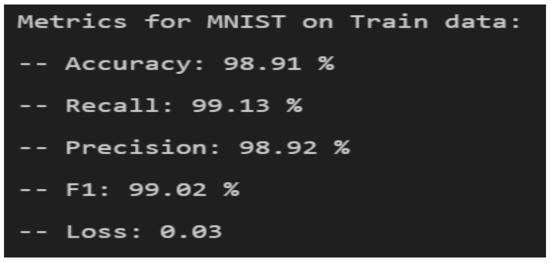
All plots show convergence of the loss.

3.6 Model Performance

1) MNIST:

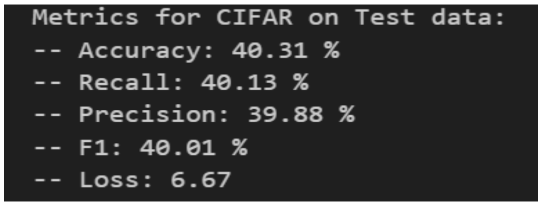


```
Metrics for MNIST on Test data:
-- Accuracy: 98.42 %
-- Recall: 98.78 %
-- Precision: 98.56 %
-- F1: 98.67 %
-- Loss: 0.05
```

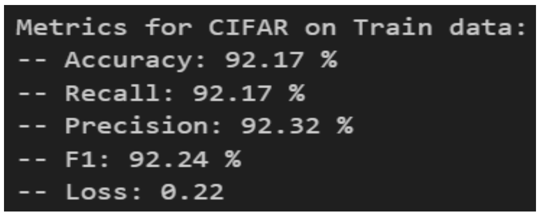


```
Metrics for MNIST on Train data:
-- Accuracy: 98.91 %
-- Recall: 99.13 %
-- Precision: 98.92 %
-- F1: 99.02 %
-- Loss: 0.03
```

2) CIFAR:

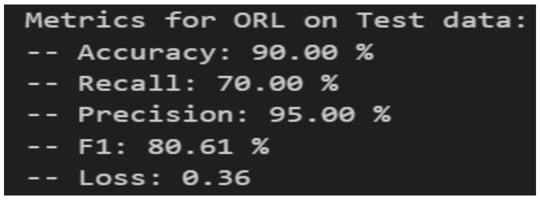


```
Metrics for CIFAR on Test data:
-- Accuracy: 40.31 %
-- Recall: 40.13 %
-- Precision: 39.88 %
-- F1: 40.01 %
-- Loss: 6.67
```

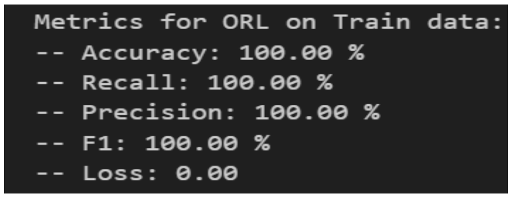


```
Metrics for CIFAR on Train data:
-- Accuracy: 92.17 %
-- Recall: 92.17 %
-- Precision: 92.32 %
-- F1: 92.24 %
-- Loss: 0.22
```

3) ORL:



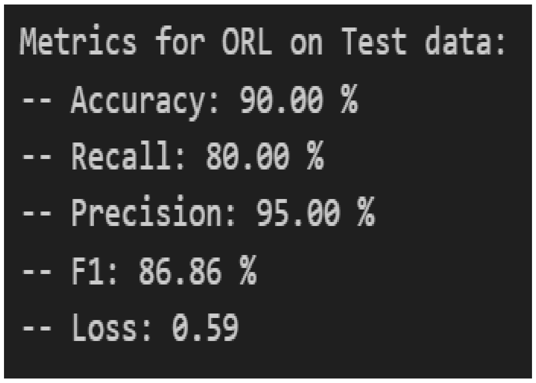
```
Metrics for ORL on Test data:
-- Accuracy: 90.00 %
-- Recall: 70.00 %
-- Precision: 95.00 %
-- F1: 80.61 %
-- Loss: 0.36
```



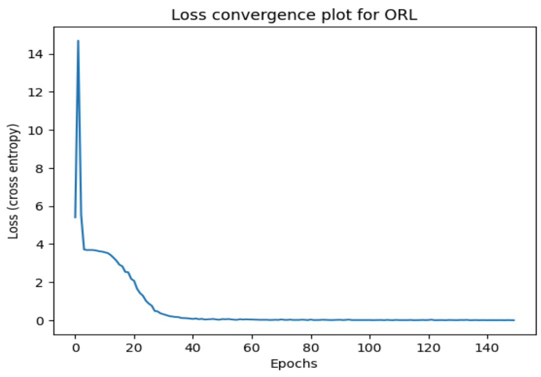
```
Metrics for ORL on Train data:
-- Accuracy: 100.00 %
-- Recall: 100.00 %
-- Precision: 100.00 %
-- F1: 100.00 %
-- Loss: 0.00
```

3.7 Ablation Studies

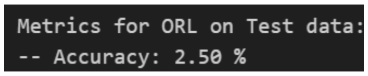
1) MNIST: the model shows very high accuracy and quickly converges, therefore we will not modify it and instead will play around with two other models.
2) ORL:
   Orl dataset has only 400 observations, therefore the first reasonable thing to do is perform full batch GD with an increased number of epochs - 150 on the same model. During training, loss seems to be converging fast enough, therefore loss and lr are kept the same



```
Metrics for ORL on Test data:
-- Accuracy: 90.00 %
-- Recall: 80.00 %
-- Precision: 95.00 %
-- F1: 86.86 %
-- Loss: 0.59
```



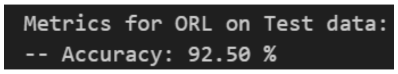
Changing batch size increased precision, f1, and recall. Next, we want to play around with activation functions. After changing AF to Sigmoid instead of ReLU:



```
Metrics for ORL on Test data:
-- Accuracy: 2.50 %
```

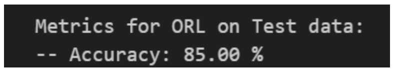
Surprisingly, changing activations functions from ReLU to Sigmoid results in the model's inability to learn, where it starts to randomly guess, with an accuracy of 2.5%. Therefore, we will stick to ReLU for this model.

Furthermore, we will change MaxPool to AvgPool:



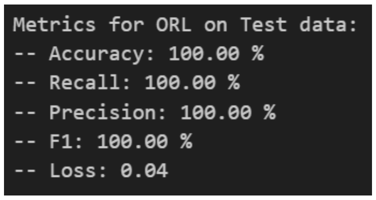
```
Metrics for ORL on Test data:
-- Accuracy: 92.50 %
```

After changing the inner layers of MaxPool to AvgPool, a slight increase in accuracy from 90% to 92.5% is seen. We also want to modify kernel sizes, after changing kernels in convolution layers from 3 to 5, we get an accuracy of 85%.



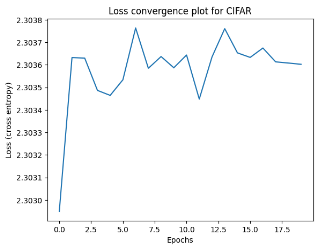
```
Metrics for ORL on Test data:
-- Accuracy: 85.00 %
```

Adding padding of 2 to the default model increased accuracy to 95% from 90 standard.
The last model provides the best accuracy of 100%, it used padding of 3 and kernels of size 5 with the default model.

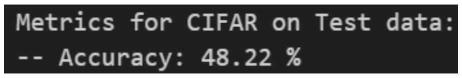


```
Metrics for ORL on Test data:
-- Accuracy: 100.00 %
-- Recall: 100.00 %
-- Precision: 100.00 %
-- F1: 100.00 %
-- Loss: 0.04
```

3) CIFAR:
From the graph error graph, one might think that the algorithm converges slowly, and should increased learning rate should be applied. However increasing the learning rate results in divergence of teh model:

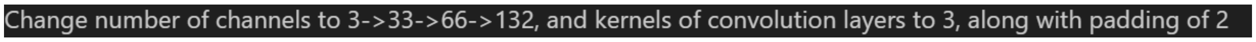


Changing batch size also didn’t have significant effect on the model. Adding padding in the previous dataset positively impacted test metrics. We will add padding of 2 to each Convolution layer:



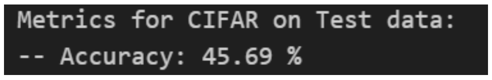
```
Metrics for CIFAR on Test data:
-- Accuracy: 48.22 %
```

Which had almost no effect on the model. Further, we will add another layer:



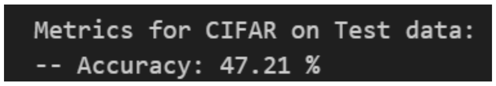
Change number of channels to 3->33->66->132, and kernels of convolution layers to 3, along with padding of 2

This model performs even worse:



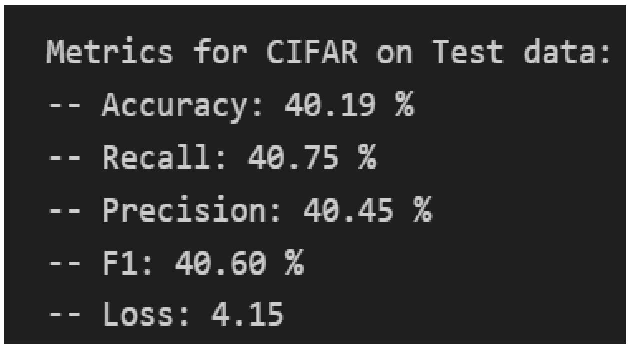
```
Metrics for CIFAR on Test data:
-- Accuracy: 45.69 %
```

Changing MaxPool to AvgPool slightly increases accuracy:

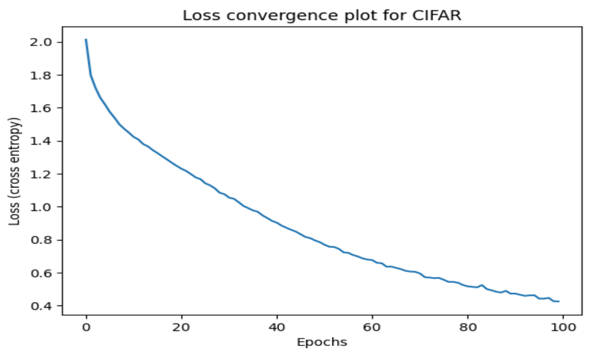


```
Metrics for CIFAR on Test data:
-- Accuracy: 47.21 %
```

Overall, the main issue is that the model itself converges on the training set, slowly but surely. However, it makes the model biased towards that set and has absolutely no effect on the test set.



```
Metrics for CIFAR on Test data:
-- Accuracy: 40.19 %
-- Recall: 40.75 %
-- Precision: 40.45 %
-- F1: 40.60 %
-- Loss: 4.15
```


Loss convergence plot for CIFAR
Loss (cross entropy)
Epochs