1. **Basic Skeleton**

**File Name : EVA4\_S5\_Basic Skeleton.ipynb**

Target:

* Get the set-up right
* Set Transforms
* Set Data Loader
* Set Basic Working Code
* Set Basic Training & Test Loop

Results:

* Parameters: 194 K
* Best Training Accuracy: 99.36
* Best Test Accuracy: 98.85

Analysis:

* Heavy Model for a hand-written digit (MNIST) recognition problem.
* Model is over-fitting, but we are changing our model in the next step.

1. **Reduce Parameters + Add Batch-norm**

**File Name :** EVA4\_S5\_Reduce parm + Batch-Norm.ipynb

Target:

* Reduce the parameters
* Add batch-normalization

Results:

* Parameters: 17,428
* Best Training Accuracy: 99.86
* Best Test Accuracy: 99.29

Analysis:

* Model still over-fitting. Training accuracy almost close to 100%.
* Parameters reduced and both accuracies improved.

1. **Reduce parameters, modify Max-Pooling interval, add dropout**

**File Name :** EVA4\_S5\_Dropout + Adjust MP + Reduce parm.ipynb

Target

* Further reduce parameters to bring it close to 10K.
* Modify convolution layers to adjust the MP based on image complexity.
* Adding dropout to make network more robust and remove overfitting.

Results:

* Parameters: 11, 864
* Best Training Accuracy: 99.48
* Best Test Accuracy: 99.26

Analysis:

* Over-fitting reduced. Training accuracy reduced to 99.48% opening-up room for further improvement.
* Doing Max-Pooling at RF of 5 improved the model.
* Fine-tuning dropout values might reduce over-fitting further and make model more robust thereby making test accuracy more consistent.
* Introducing GAP and 1x1 layer instead of 6x6 convolution might improve the model as it will shift one-hot encoding from the final 6x6 layer. Will reduce parameters as well.

1. **Introduce GAP + FC in final layers**

**File Name :** EVA4\_S5\_GAP\_FC\_Dropout Fine Tuning.ipynb

Target

* Replace 6x6 with GAP (nn.AdaptiveAvgPool2d) followed by fully connected layer (nn.Linear).
* Fine-tune dropout values.

Results:

* Parameters: 9,680
* Best Training Accuracy: 99.17
* Best Test Accuracy: 99.34

Analysis:

* Over-fitting removed. Training and test accuracy showing a steady increase which is good.
* Drop-out value of 0.1 used.
* Model is becoming robust during training with dropout which is evident from fact that training accuracy slightly lesser than testing accuracy.
* Introducing image augmentation will help this further as it can learn from augmented images as well.

1. **Use image augmentation and fine-tune LR**

**File Name :** EVA4\_S5\_Image Aug\_LR Change.ipynb

Target

* Use image augmentation to give more variety of images while training the model
* Fine-tune LR using stepLR

Results:

* Parameters: 9,680
* Best Training Accuracy: 99.23
* Best Test Accuracy: 99.40

Analysis:

* RandomHorizontalFlip reduces accuracy and hence shouldn’t be used.
* With random rotation & color jitter able to get steady accuracy improvement.
* Used steplr for step-size of 6 with a multiplication factor of 0.1.
* Test accuracy reached 99.40 but just once. Model needs further improvement to consistently touch >= 99.40%.

1. **Use AvgPool2D for GAP and 1x1 for FC**

**File Name :** EVA4\_S5\_GAP = AvgPool2D + FC = 1x1.ipynb

Target

* Replace nn.AdaptiveAvgPool2d with **nn.AvgPool2d** for GAP
* Replace nn.Linear with **1x1 convolution** for FC

Results:

* Parameters: 9,680
* Best Training Accuracy: 99.25
* Best Test Accuracy: 99.47

Analysis:

* In AdaptiveAvgPool2d, kernel size and stride are automatically selected whereas in AvgPool2d we have the option to supply these as hyper parameters.
* While connecting AdaptiveAvgPool2d with Linear we let go height & width dimensions whereas while connecting AvgPool2d with 1x1 h& w are retained.
* This improved test accuracy and also helped achieve consistency.
* Model was able to achieve >= 99.4% accuracy 5 times out of 15 epochs

*nn.AvgPool2d + 1x1 convolution -> Shape change as below*

nn.AvgPool2d **:** torch.Size([2, 16, 1, 1])

1x1 conv :torch.Size([2, 10, 1, 1])

x.view(-1, 10) : torch.Size([2, 10])

*nn.AdaptiveAvgPool2d + nn.Linear -> Shape Change as below*

nn.AdaptiveAvgPool2d : torch.Size([2, 16, 1, 1])

x.view(-1, 16) : torch.Size([2, 16])

nn.Linear : torch.Size([2, 10])

x.view(-1, 10) : torch.Size([2, 10])

1. **Further Improvements**

**File Name :** EVA4\_S5\_GAP = AvgPool2D + FC = 1x1.ipynb

Target

* Further improve the model either by:
  + Staying within 10K parameters improving accuracy and make consistent (Part-1)

OR

* + Further reduce the parameters and get >= 99.40% consistently (Part-2)

Results:

* Parameters: 9,680
* Best Training Accuracy: 99.25
* Best Test Accuracy: 99.47

Analysis:

Part-1

* Adding one more FC by 1x1 while keeping everything else same as in 6 reduces accuracy. Not helping the target.
* Increasing dropout from 0.05 to 0.08 not helping.
* Decreasing dropout from 0.05 to 0.02 not helping. So 0.05 can be retained as best value.
* Increasing LR to 0.03 from 0.025 while reducing dropout to 0.01 helped achieve accuracy >= 99.40% two times but couldn’t beat 6. Also best accuracy achieved was only 99.42%.

Part -2

* Reduced parameters to 7344 by doing GAP on 8x8x16 layer size.
* LR as 0.25 with StepLR of 0.1 every 6 steps is not helping.
* LR as 0.3 with StepLR of 0.1 every 6 steps is not helping.
* LR as 0.2 or 0.3 without StepLR is not helping.
* Reduced batch\_size to 64 but not improving.
* Retained GAP at 6x6 but reduced channel size from 16 to 10. Still Not improving.

**Conclusion : Take model 6 as the best model for this problem**