Q1. 3 channels - Red, Green, Blue, (RGB)
Q2. The model will be biased toward the classes with larger proportionality in the distributional representation.
Q3. Batching: Since GPUs can perform parallel computation, it will be faster if we provide batch_size worth of the input data to the forward_pass.
Q4. Original LeNet5 didn't have 2D batch normalization and used Average Pool 2D instead of Max Pool 2D.
Q5. The loss function tells how far away the model's prediction is from the ground truth value.
Q6. The validation step course corrects training.
Q7. Validation loss and accuracy give us an idea about how well the model will fit the test/unforeseen data.
Q8. Training loss and accuracy show signs of plateauing. We could train them for more epochs. But it doesn't seem that they will show drastic improvement as it's already quite high in accuracy.
Q9.

	Original LeNet5	**Modern LeNet5**
Training	Avg Loss: 0.01741030763560166, Avg Accuracy: 0.9939213446475196 on 10th epoch	Avg Loss: 0.016529451854944276, Avg Accuracy: 0.9944516971279374 on 10th epoch
Validation	Best Validation Loss: 0.052773135150728276 after epoch 8 Best Validation Acc: 0.9859446347031964 after epoch 8	Best Validation Loss: 0.03646008961911557 after epoch 8 Best Validation Acc: 0.9899400684931506 after epoch 8
Testing	Original LeNet-5 Inference Performance Test dataset contains 7000 samples. Testing Avg Loss: 0.06176949884890277 Testing Avg Acc: 0.9837328767123288	Modern LeNet-5 Inference Performance Test dataset contains 7000 samples. Testing Avg Loss: 0.04903900079214459 Testing Avg Acc: 0.98787100456621
Comments	Val accuracy and test accuracy differ by .00221 .00206	Val accuracy and test accuracy differ by .00206

Q10. Depending on the requirement any of the following:
%%time/%%timeit/ import time -> time.time()/time.timeit()

Ran code in Kaggle and have it in GitHub:
https://github.com/Initiated0/NeuromorphicCompute/blob/main/Assignments/assignment-1-0.ipynb