Neural Network-Training And Evaluation

The specific task we are trying to solve is image classification. We will be using a very commong dataset called CIFAR-10 that has 60,000 images (50,000 for training and 10,000 for testing) separated into 10 classes (link for Keras , link for Pytorch). The classes are

- airplane
- automobile
- bird
- cat
- deer
- dog
- frog
- horse
- ship
- truck

In this task, you will:

- Use a modern machine learning library (Pytorch/TF/Keras)
- Writing the chosen library-specific code.
- Evaluating neural network models.

Create three neural network models and train each for the classsification task above, where each of the models described should be a different Python class with the specified name.

`NetA`

The first neural network will be the simplest, in that it has no hidden layers. It should take the image and flatten it to a vector for the input, and then have 10 outputs, one for each class.

There should be no non-linearities for this network and is just a very simple linear classifier.

`NetB`

The second neural network will be slightly more complicated in that it has a hidden layer with 300 nodes and adds a non-linearity between the layers. It should use the following operations in this order:

- Flatten the image to a vector for the input
- Use a fully-connected linear layer with 300 hidden-neurons
- Use the ReLU activation function
- Use a fully-connected linear layer to the 10 outputs.

`NetC`

This third neural network will be a convolutional neural network. It should use the following operations in this order:

- Use a convolution layer with kernel-width 5 and depth 25
- Use the ReLU activation function
- Use a max-pool operation with kernel-width 2 and stride 2
- Flatten the image to a vector for the next step's input
- Use a fully-connected linear layer to the 10 outputs.

Deliverables:

You need to submit a google collab notebook which

1. imports all the important modules for the specific machine learning library you choose,
2. loads the training and test dataset,
3. trains the given network on the training data for **50** epochs, evaluates the train and test accuracy of the network at the end of each epoch and creates a visualization of the training/test accuracy as the training progresses.
4. do this for the three neural network models (NetA, NetB, NetC) described above.

The last cell should print the training and test accuracy of each of the three models, including a visualization of the accuracy histories for each model during the training. An example of the last cell is given below *for reference only*, which can be different from your submission as long as the required outputs are printed.

```
nets = [NetA(), NetB(), NetC(), NetD()]
histories = []

for net in nets:
    net_name = type(net).__name__
    print(f'==== Training {net_name} ====')
    train_history, test_history = train(net, train_loader, test_loader,
                                        num_epochs=NUM_EPOCHS,
                                        learning_rate=LEARNING_RATE,
                                        compute_accs=True)
    histories.append({
        'name': net_name,
        'net': net,
        'train_accs': train_history,
        'test_accs': test_history
    })
plot_history(histories)
```