1. Train:

   The function ***train*** is used to train a neural network model in PyTorch. It takes in several parameters such as ***args, model, device,train_loader, optimizer, and epoch.*** The function sets the model to training mode using ***model. train().*** It then iterates over the data in the training loader and performs forward and backward passes through the model to compute gradients. The optimizer is then used to update the model parameters based on these gradients. Finally, the function computes the training accuracy and loss and returns them as output.

2. Test:

   The function ***test*** is used to train a neural network model in PyTorch. It takes in several parameters such as ***model, device, and test_loader.*** The function sets the model to evaluation mode using the ***model.eval().*** It then iterates over the data in the test loader and performs forward passes through the model to compute predictions. The function computes the testing accuracy and loss and returns them as output.

3. Plot:

   The function ***plot*** is used to plot the performance of a neural network model in PuTorch. It takes in several parameters such as ***epochs, performance, title,and config.*** The function plots the performance of the model using Matplotlib. It saves the plot as an image file and displays it using ***plt. show().***

4. Plot_mean:

   The function ***plot_mean*** is used to plot the mean results after three runs of a neural network model in PyTorch. It reads the recorded results and plots the mean results using Matplotlib. The function saves the plot as an image and displays it using ***plt. show().***

5. Multiprocessing:

   This code is used to train a neural network on MNIST data sets using multiple processes. The first few lines of code check that CUDA is available and set up the device accordingly. It then sets the batch size for training and testing. After that, it downloads the MNIST dataset and applies some transformations to it. The next few lines of code use PyTorch's multiprocessing module to run the training process on multiple processes. It creates a pool of processes with different seeds and runs them in parallel. Finally, average results are plotted.