**Report**

In order to train an AlexNet model on the MNIST dataset I followed the following steps:

1. First, I Loaded the MNIST dataset: The **MNIST dataset is loaded** using the torchvision library. The train\_dataset contains the training images and labels, and the test\_dataset contains the test images and labels.
2. **Create data loaders:** The train\_loader and test\_loader are created using the torch.utils.data.DataLoader method. The batch\_size is set to 100, which means that the model will be trained on 100 images at a time, and the shuffle parameter is set to True for the training data and False for the test data.
3. **Define the AlexNet model:** The AlexNet model is defined as a subclass of nn.Module in PyTorch. It consists of two convolutional layers, two fully connected layers, and uses the ReLU activation function.
4. **Define the loss function and optimizer**: The cross-entropy loss is defined as the loss function using nn.CrossEntropyLoss(), and stochastic gradient descent is used as the optimizer with a learning rate of 0.001 and a momentum of 0.9.
5. **Train the model**: The model is trained for 2 epochs, where an epoch is a full pass through the training data. In each iteration, the gradients are zeroed out using optimizer.zero\_grad(), the output of the model is computed using the forward pass outputs = model(images), the loss is computed using loss = criterion(outputs, labels), and the gradients are computed using loss.backward(). Finally, the model parameters are updated using ***optimizer.step()***.
6. **Test the model:** The model is tested on the test data and the accuracy is computed.
7. **The final step is to apply the audio transformation,**

The torchaudio.load function is used to load the audio file as a waveform and its sample rate. Then, the torchaudio.transforms module is used to apply the STFT, Mel Spectrogram, and MFCC transformations one after the other.

**matplotlib.pyplot** is used to visualize the output of each transformation as an image plot. The numpy method is used to convert the tensors to NumPy arrays for plotting, and the *cmap parameter* is used to specify the color map for the image plot.