A short report on the model we have implemented in this exercise.

```
  def __init__(self):
      super(model_with_pooling, self).__init__()
      self.conv1 = nn.Conv2d(1, 15, 10, stride=1)
      nn.init.xavier_uniform_(self.conv1.weight)
      self.conv1_bn = nn.BatchNorm2d(15)

      self.pooling2d = nn.MaxPool2d(2,2)
      self.conv2 = nn.Conv2d(15, 1, 5,stride=1)
      nn.init.xavier_uniform_(self.conv2.weight)
      self.conv2_bn = nn.BatchNorm2d(1)

      self.dropout = nn.Dropout2d(p=0.5)
      self.rnn_module = nn.LSTM(dropout=0.2,input_size=72, hidden_size=72, num_layers=2,
batch_first=True,
                          bidirectional=bidirectional)
      self.rnn2fc = nn.Linear(72*2, 72*2)
      self.h2o = nn.Linear(72*2, 27)
      self.softmax = nn.LogSoftmax(dim=2)
```

Above you may see the model code description. We will go over the layers one by one and later will discuss other parameters.
the first layer is a 2d convolutional filter, we decided to use this 2d filter since we know there is a strong connections between relatively close frequencies.

we have used a 15 channels output and kernel size of 15 because empirically they produced the best result.
later we used a batch normalization.

as to the pooling layer, initally this layer wasnt part of the network we had, but when we analyzed the model loss
performance on the train and dev we have seen a big gap between the dev and train loss.
Our conclusion was that we should insert a layer that will help us generalize better, for that we added the pooling layer.

second convolutional layer would filter much more complex feature, its out is a single channel.
The time sequence length after this second convolutional is 42, each time frame has 72 features.

Then we took this tensor and pushed it into a two layers BiLSTM.
We believe that this rnn model dramatically increased the the performance,
this is essentially due to the context that the rnn can take into account.

lastly the rnn output went into two fully connected layers which their output was the size if the number of classes.

we transformed this score vetor into a log probability with a log softmax layer.

Basically we chose most of the hyper parameters after tweaking them and checking which was give us the best result on the dev dataset.