Frame Level Classification of Speech

In this challenge you will take your knowledge of feedforward neural networks and apply it to a more useful task than recognizing handwritten digits: speech recognition. You are provided a dataset of audio recordings (utterances) and their phoneme state (subphoneme) labels. The data comes from articles published in the Wall Street Journal (WSJ) that are read aloud and labelled using the original text. If you have not encountered speech data before or have not heard of phonemes or spectrograms, we will clarify the problem further.

The training data comprises of:

1. Speech recordings (raw mel spectrogram frames)
2. Frame-level phoneme state labels

The test data comprises of:

1. Speech recordings (raw mel spectrogram frames)
2. Phoneme state labels are not given

Your job is to identify the phoneme state label for each frame in the test data set. It is important to note that utterances are of variable length. We are providing you code to load and parse the raw files into the expected format. You can find that code [here](https://gist.github.com/rbrigden/d611ef4f2eccddb5581e0ac617ce38eb).

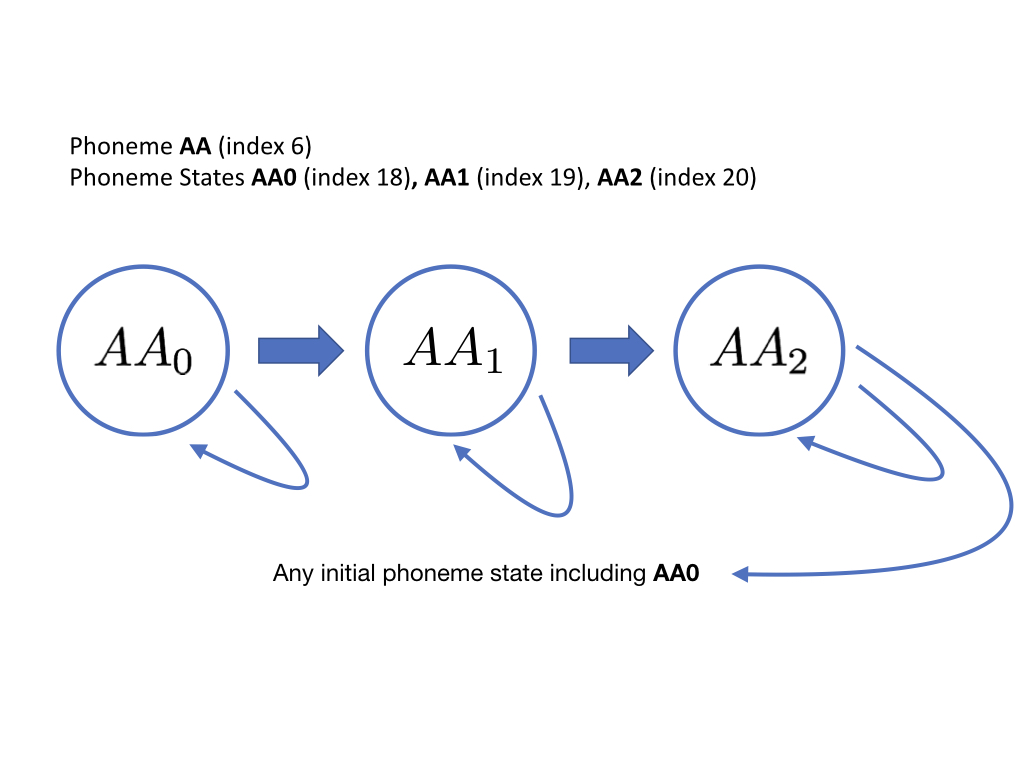
Phonemes and Phoneme States

As letters are the atomic elements of written language, phonemes are the atomic elements of speech. It is crucial for us to have a means to distinguish different sounds in speech that may or may not represent the same letter or combinations of letters in the written alphabet.

For this challenge we will consider 46 phonemes in the english language.

["+BREATH+", "+COUGH+", "+NOISE+", "+SMACK+", "+UH+", "+UM+", "AA", "AE", "AH", "AO", "AW", "AY", "B", "CH", "D", "DH", "EH", "ER", "EY", "F", "G", "HH", "IH", "IY", "JH", "K", "L", "M", "N", "NG", "OW", "OY", "P", "R", "S", "SH", "SIL", "T", "TH", "UH", "UW", "V", "W", "Y", "Z", "ZH"]

A powerful technique in speech recognition is to model speech as a markov process with unobserved states. This model considers observed speech to be dependent on unobserved state transitions. We refer to these unobserved states as phoneme states or subphonemes. For each phoneme, there are 3 respective phoneme states. Therefore for our 46 phonemes, there exist 138 respective phoneme states. The transition graph of the phoneme states for a given phoneme is as follows:



Hidden Markov Models (HMMs) estimate the parameters of this unobserved markov process (transition and emission probabilities) that maximize the likelyhood of the observed speech data. Your task is to instead take a model-free approach and classify mel spectrogram frames using a neural network that takes a frame (plus optional context) and outputs class probabilities for all 138 phoneme states. Performance on the task will be measured by classification accuracy on a held out set of labelled mel spectrogram frames. Training/dev labels are provided as integers [0-137].

Speech Representation

As a first step, the speech (audio file) must be converted into a feature representation (matrix form) that can be fed into the network.

In our representation, utterances have been converted to "mel-spectrograms" (you do not have to convert anything, just use the given mel-spectrograms), which are pictorial representations that characterize how the frequency content of the signal varies with time. The frequency domain of the audio signal provides more useful features for distinguishing phonemes.

For a more intuitive understanding, consider attempting to determine which instruments are playing in an orchestra given an audio recording of a performance. By looking only at the amplitude of the signal of the orchestra over time, it is nearly impossible to distinguish one source from another. But if the signal is transformed into the frequency domain, we can use our knowledge that flutes produce higher frequency sounds and bassoons produce lower frequency sounds. In speech, a similar phenomenon is observed when the vocal tract produces sounds at varying frequencies.

To convert the speech to a mel-spectrogram, it is segmented into little "frames", each 25ms wide, where the "stride" between adjacent frames is 10ms. Thus we get 100 such frames per second of speech.

From each frame, we compute a single "mel spectral" vector, where the components of the vector represent the (log) energy in the signal in different frequency bands. In the data we have given you, we have 40-dimensional mel-spectral vectors, i.e. we have computed energies in 40 frequency bands.

Thus, we get 100 40-dimensional mel spectral (row) vectors per second of speech in the recording. Each one of these vectors is referred to as a frame. The details of how mel spectrograms are computed from speech is explained [here](http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html).

Thus, for a T-second recording, the entire spectrogram is a 100\*T x 40 matrix, comprising 100\*T 40- dimensional vectors (at 100 vectors (frames) per second).

Data Files

You can find the training and test data under the 'Data' tab

Optional Readings

Read the Implementation Details section for some tips and advice on building your classifier.

Go through [this link](http://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html) to understand how Mel-Spectrograms are generated.

## Evaluation Metric

The evaluation metric for this competition is frame-level accuracy. There are a total of 223,593 frames in the test set. You will be ranked by unweighted accuracy on those phoneme state labels.

## Submission Format

Submission files should contain two columns: Id and Label.

 Id: the 0-based index of the frame in the test set [0-223592]

 Label: the predicted label of the phoneme [0-137]

Id=0 is the first frame in the first utterance. Id=223,593 is the last frame in the last utterance. A sample submission file is available on the Data page.

## Rules

Your model MUST use an MLP (no RNNs or ConvNets). We will check your code to make sure this is the case. However, you are free to use auto-differentiation libraries like PyTorch or Tensorflow.

(Note: Support will most likely only be provided for PyTorch.)