ICS 471, Term 201
Artificial Neural Networks and Deep Learning

HW# 3

Due date: Tuesday, Oct. 27, 2020

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 LibriSpeech corpus which is derived from audiobooks that are part of the LibriVox project, and contains 1000 hours of speech sampled at 16 kHz. If you have not encountered speech data before or have not heard of phonemes or spectrograms, we will clarify the problem further. For more information, see the paper ”LibriSpeech: an ASR corpus based on public domain audio books”, Vassil Panayotov, Guoguo Chen, Daniel Povey and Sanjeev Khudanpur, ICASSP 2015 (submitted) (pdf).

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

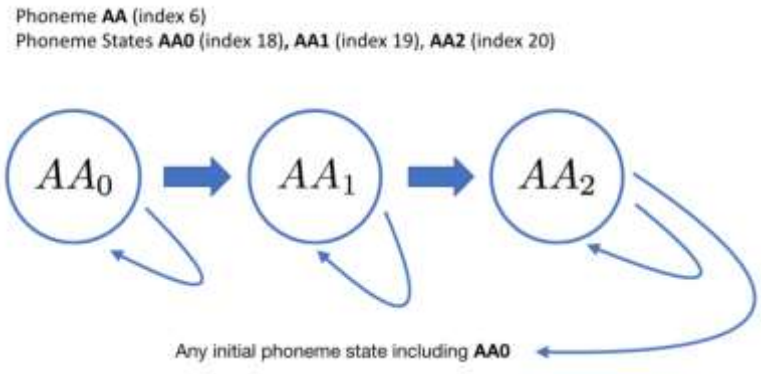
Your job is to identify the phoneme state label for each frame in the test data set. It is important to note that utterances are of variable length.

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 a total of 346 phonemes in this language.
Example: ["+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. In conventional design each phoneme is expanded

to three (sometimes five states). Basically states correspond to beginning of the phoneme, middle of the phoneme and the end of the phoneme. This helps to model dynamics of the phoneme sound.
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 likelihood 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 346 phoneme states. Performance on the task will be measured by classification accuracy on a held out set of labelled mel spectrogram frames. Training/Validation labels are provided as integers [0-345].

You will be evaluated on the accuracy of the prediction of the phoneme state labels for each frame in the test set.

Speech Representation

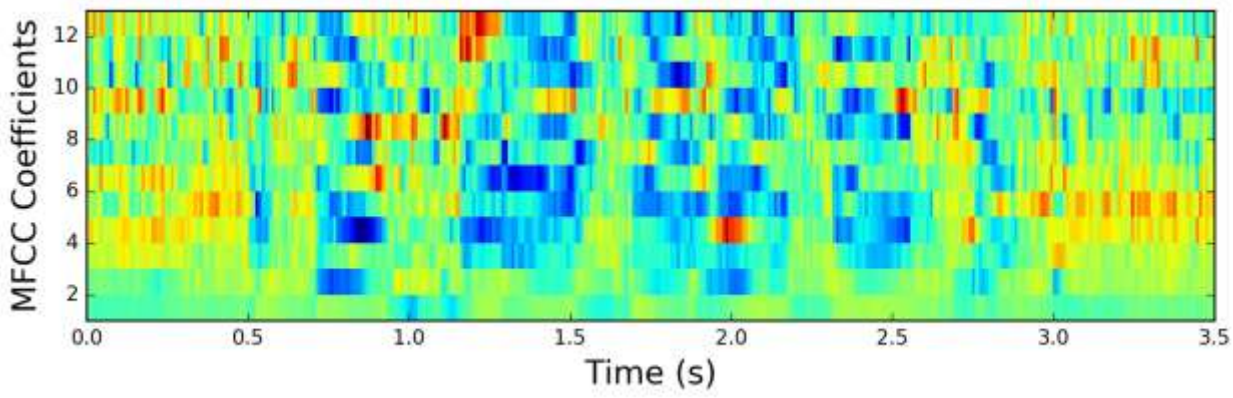
As a first step, the speech (audio file) must be converted into a feature representation (matrix form) that can be fed into the network.

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" (non-overlapping region) 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 13-dimensional mel-spectral vectors, i.e. we have computed energies in 13 frequency bands. The figure below shows an example of mel-spectral vectors for a duration of 3.5 s.



Thus, we get 100 13-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.

Thus, for a T-second recording, the entire spectrogram is a 100*T x 13 matrix, comprising 100*T 13-dimensional vectors (at 100 vectors (frames) per second).

Data

The audio data has been transcribed into mel spectrograms. You have 100 13-dimensional mel spectral (row) vectors per second of speech in the recording. The following tables summarizes the data characteristics.

Data File	Description	Number of Utterances	Total Number of Frames
ntrain.npy	Training Samples	19670	23628221
ntrain_labels.npy	Training Labels	19670	23628221
nval.npy	Validation Samples	2332	1598404
nval_labels.npy	Validation Labels	2332	1598404
ntest.npy	Testing Samples	2332	2813174

You can access the data using the following link

Implementation Tips

Network Architecture

You MUST use MLP network (e.g. no RNN or CNN).

Feel free to experiment with different architectures. The simplest approach is to build a feedforward network that takes in a single frame (input size 13) and outputs probabilities across phoneme states using a softmax output to normalize the logits.

Context

Temporal context is important for distinguishing elements of speech and you can experiment with adding context to your ANN model. For example, you can try concatenating k mel spectrogram frames around the current time step. This technique would make the size of the input vector 13 * (2k + 1). You will need to adjust your data preprocessing and batching logic to adapt for frame context. Writing code that will let you experiment with variable context sizes easily is recommended.

Intuition: Since each mel spectrogram frame is only 25ms of the speech audio data, a single frame is unlikely to represent a complete phoneme. Concatenating nearby "k" frames will thus be helpful. Here "k" can be viewed as a hyperparameter for you to try out.
(Hint: This is the key to boost your score!)

Data Batching

It is important to consider that the data is provided in the form of utterance samples and not frame samples. The property of mini-batch SGD that gradients over all batches in an epoch approximate the true gradient only holds if each batch is IID (Independent and Identically Distributed). If we sample uniformly over utterances and also sample uniformly over frames within the utterance, our batches will surely not be IID. For example, different utterances will have considerably different numbers of phoneme instances. Sampling uniformly biases the training data by over-representing phonemes in the utterance with fewer phonemes (and therefore its phoneme states). You can try a different sampling technique to batch your training data to mitigate this issue.

Many approaches are possible. See what can get you to the top of the competition.

Evaluation Metric

The evaluation metric for this competition is frame-level accuracy. There are a total of 2,813,174 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 [0-2,813,173]
- Label: the predicted label of the phoneme [0-345]

Id=0 is the first frame in the first utterance. Id=2,813,173 is the last frame in the last utterance. A sample submission file is available on the Data folder.

In addition, you should submit your notebook which has your code.