Name: lan-gauge

Vladimir Ermoshkin(ermov), James Riggleman(jamesfr3), Edward Wang(ejaywang)

Description: This dataset contains 800 wav files of people reading individual sentences in a variety of languages. 400 are in English, and there are 100 recordings for each of the following languages: German, Russian, Turkish, and Spanish. The goal of the classifier is to label a given audio file as "English" or "Not English" which could have applications in automatic translation, voice recognition, or dictation software.. This is not a trivial problem because speech in the same language can vary greatly between different speakers, contexts, and enunciation. However, different languages do have differing phonetic inventories, cadence, and tempo, and a machine learning algorithm could identify such patterns. Since the classifier is deciding between English and non-English speech, we included several languages from different language families, both similar to English (e.g. German) and completely linguistically distinct (e.g. Turkish). This should allow the classifier to identify linguistic features that are unique to English. Because the source database had a substantial gender imbalance, we chose to focus the dataset specifically on male speakers, as languages with more female speaker examples would skew the data. To ensure that the features extracted are actually representative of the language, we hand-picked the files with the least recording noise and no heavy pronunciation issues.

Format/feature extraction: Each audio sample contains one speaker speaking one sentence in a single language. The files are all in wav format with a sampling rate of 16kHz at 16-bit resolution. In order to perform the classification problem, a feature space will have to be generated from these files. A feature space which would provide a good starting point is the Fourier series of the audio sample which distills the audio sample into a frequency composition. This analysis can be done for the entire audio file to represent the total frequency component, or can be done for short, consecutive, overlapping segments (i.e. 1 second segments with 0.5 second overlaps) to show the frequency component as it changes over time. Features could then come from the amplitude of each frequency in one segment, or from changes in frequency components with consecutive segments. To read a .wav file in Python, one can use scipy.io.wavfile. After loading the wav files, one can apply the Fourier transform in Python using numpy's fft functions over arrays generated from the scipy output.

Division criteria: We chose to split the data 80/20 into training and test sets. In order to mitigate the consequences of overfitting to a specific speaker, we took several precautions. Rather than divide the individual files into "training" and "test", we assigned each speaker to one of the categories. This way, a classifier would not be able to identify a speaker's language simply based on seeing that speaker in a training set, and would have to rely on the actual features of the language. Furthermore, we ensured that the training and the test sets have the same number of speakers of each language in order to prevent a situation where the classifier mislabels one specific speaker and as a result fails on a substantial portion of the test set. There are a total of 160 speakers in the dataset with 80 unique speakers in each set. There are 8 samples for each speaker in the training set and 2 samples per speaker in the test set. In each set, there are 40 English speakers and 10 speakers of each other language.

Ethics: We used voxforge.org, an open online database, to gather our data files. All the files we used were voluntarily submitted by the speakers and are licensed under GPL. As the speakers are reading from text prompts, the speech does not contain any sensitive information.