Methods for Spoken Language Identification

Julien Boussard, Andrew Deveau, Justin Pyron

{julienb, adeveau, pyron}@stanford.edu

December 16, 2017

Abstract

In this paper, we explore several machine learning techniques for classifying spoken language. In particular, we construct algorithms which utilize various spectral features derived from English and Mandarin Chinese phone call audio to predict the language to which the phone call belongs. We investigate multiple feature sets and modeling approaches, and find that Gaussian Mixture Models, combined with shifted delta cepstra (SDC) features, achieve the best performance.

1 Introduction

In recent years, advances in machine learning have tremendously expanded the extent to which people can interact with technology using only their voice. However, these technologies often require being explicitly told which language to process. The ability to dynamically process input speech streams of different languages would expand the usefulness of existing speech processing technology and open up a wide array of additional functionality.

The field of Spoken Language Identification addresses this issue by exploring how to extract information from the audio signal of speech and use it to predict the language in which the speech was spoken. In this paper, we explore various approaches to identifying the language of speech, restricting our focus to the specific challenge of discriminating between English and Mandarin Chinese speech.

The input to our algorithm is an audio segment in either English or Mandarin. We then use Gaussian mixture models to output a prediction of which language the segment was spoken in.

2 Related work

There exists a wide array of audio-based machine learning applications, along with a correspondingly rich set of literature outlining techniques for extracting pertinent information from an audio signal.

Guided by the hypothesis that attributes such as tempo and pitch register that are useful for music-related tasks would also be valuable for language discrimination, our attention was initially drawn to the subfield of music genre classification.

Here, songs are sliced into several short frames from which features are extracted using signal processing techniques such as Fourier Transforms. Features are generally separated into two classes: timbre and temporal. Timbre features convey information about the quality or "color" of a sound, while temporal features convey information about how timbre features evolve over time. Temporal features are often taken as statistics of the distribution of timbre features, such as the mean, variance, and covariance. A variety of learning algorithms, such as Support Vector Machines, K-nearest neighbors, and Logistic Regression are then trained on the constructed feature sets. See [ZFZ] and [TC].

In recent years, there have also been attempts to use neural networks to learn from a frequency representation of a signal, typically as a spectrogram or a time series of MFCCs. These efforts have typically used convolutional or recurrent architectures [BHYM17] [RZ] , although feed-forward architectures have been used as well [FRD] [ILM]. When large data sets are available, high accuracies have been obtained using this approach; a convolutional recurrent neural network trained on 1508 hours of audio by Bartz, et al. [BHYM17] achieves 98% accuracy when discriminating among French, English, Spanish, and German, for instance.

3 Dataset and Features

3.1 Dataset

We used the OGI Multilanguage Corpus as our dataset [CM]. This dataset consists of phone calls in eleven languages, with durations ranging from a few seconds to approximately one minute. These calls are stored as `.wav` files sampled at 8 KHz. We used calls of at least three seconds in length, of which 60% percent were assigned to the training set and 20% were assigned to each of the validation and test sets. This resulted in a training set containing 3960 examples across all languages, 892 of which were in either English or Mandarin. We also experimented with using audio from audiobooks, but preferred the OGI Corpus because it contains a greater diversity of speakers.

3.2 Features

Because of the large number of samples in a raw audio signal, it is typically difficult to work with raw audio in the time domain. All of our features are therefore derived from representations of the signal in the frequency domain.

Chief among these are Mel Frequency Cepstral Coefficients (MFCCs), which are perhaps the most widely utilized feature in digital signal processing for capturing speech information. Roughly speaking, the MFCCs quantify how the energy of a signal is distributed by frequency, adjusted for how humans perceive sound. Humans do not perceive changes in pitch linearly, and MFCCs account for this aspect of human auditory perception [ZFZ].

We additionally constructed features typically used in the music genre classification realm. These included features such as Spectral Centroids, which measures the “center of gravity” of the frequencies which make up the signal, and Spectral Contrasts, which measures the distance between the spectral peaks and troughs of a signal [TC].

Each phone call in the dataset was split up into several overlapping 25 millisecond frames, and the features described above were constructed for each of these frames. Shifted Delta Cepstral (SDC) features were additionally constructed utilizing frame-level MFCC features [EAS]. For each frame, the difference between the next frame’s MFCC vector and the previous frame’s MFCC vector was taken to capture information about the time dynamics of the signal. The SDC feature vector for a particular frame is constructed by concatenating together the difference vectors from several future frames. See figure 1 for an illustration of how SDC are constructed. Also see figure 2 for an illustration of entire feature construction process.

The Python libraries `scipy`, `librosa`, and `python_speech_features` were used to process audio files and construct features.

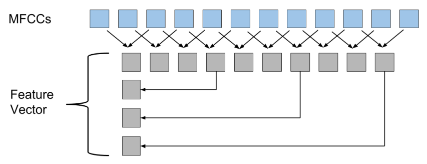


Figure 1: Shifted Delta Cepstra

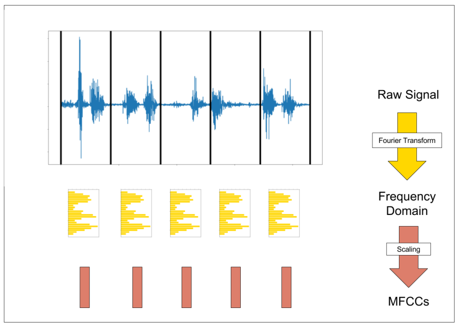


Figure 2: Feature Construction Process

4 First methods and algorithms

4.1 Music-genre motivated approach

Motivated by music genre classification, we first attempted prediction utilizing features described in section 3.2, namely the mean, variance, and covariance of MFCCs, the Spectral Centroid, and Spectral Contrast from all frames within a phone call. Note that a single feature vector corresponds to a single phone call. Feed-forward neural networks were trained on the resulting dataset, but without success – such methods often performed worse than random guessing.

By aggregating information from all of the frame-level features into statistics like the mean and variance when constructing call-level features, critical properties of the sound wave are obscured and lost. The poor performance of this attempt made us realize the importance of maintaining granular frame-level features, which guided our next attempts.

4.2 Feed-Forward Neural Network

After obtaining poor results utilizing aggregated features, we next attempted to classify individual frames, after which call-level prediction would be performed by aggregating predictions on each of a call's constituent frames. More precisely, we gathered all (labeled) frame-level MFCC vectors into one dataset, ignoring the phone calls from which each came. This dataset was then used to train a feed-forward neural net which would predict the language from which an MFCC vector from a single frame came. However, poor results were again obtained, with test accuracies only slightly better than random guessing.

This method utilized only static MFCC features, and as a result, did not capture any of the time dynamics of the speech signal. In order to capture information about how the audio signal evolves over time and relates to nearby utterances, we next trained a feed-forward neural network on a feature set that additionally contained Shifted Delta Cepstra, which yielded improved results.

To form a prediction on a phone call level, prediction was first conducted on each frame contained inside a call. The language receiving the highest number of frame-level predictions was taken as the prediction for the call.

A variety of architectures were tested, and shallower architectures with a large number of neurons generally achieved the best performance, although performance was not especially sensitive to variations in architecture. The optimal architecture obtained call-level training set accuracy of 72.8% and test set accuracy of 67.5%.

4.3 Recurrent Neural Network

To make predictions directly on the call level, we thought that considering each call as a time-series of MFCCs and training a Recurrent Neural Network would be a way to take into account the dynamics of speech, as RNNs are designed to handle sequence dependence. We used a Long Short-Term Memory network. The LSTM neurons remember data from a previous time steps, which make them effective for understanding time series and their dynamics.

We constructed a model using the `Keras` function `LSTM()`. We trained a Neural Network constituted of one `LSTM` layer followed by a `Dense` layer. We tried different numbers of neurons in each layers, from 10 to 128 neurons in the `LSTM` layer. All models were fit using Adam for 5 epochs.

We obtained poor results, with at most 65% test accuracy, with similar results for every model. Our interpretation was that the Recurrent Neural Networks were capturing some of the dynamics shared by all the calls (English and Mandarin), such as the beginning and the end of the calls. We thought that a convolutional neural network might solve this problem thanks to parameter sharing.

4.4 Convolutional Neural Network

We thus trained a convolutional neural network to make predictions directly on the call level. Training examples were two dimensional arrays consisting of all the MFCCs for a particular call. In principle, this model is equipped to capture time dynamics of speech by linearly combining MFCCs from different frames, and could extract features similar to Shifted Delta Cepstra.

A convolution neural network is similar to a feed-forward neural network, but shares parameters–the output of a convolutional layer is constructed by convolving the same relatively small learned filters with each section of the input. This parameter sharing encodes translation invariance (which is relevant for this task as utterances at the beginning and end of a speech segment should likely be treated the same way), guards against overfitting, and makes the network less expensive to train.

The convolutional neural networks we experimented with tended to overfit the training data. Our initial architectures and hyperparameters achieved perfect training accuracy but validation accuracy of approximately 68%. We experimented with a variety of approaches to remedy this. We first directly regularized the model, using both dropout layers and L^2 regularization on the network's weights. We performed a grid search over the shapes of the filters in the network, the weight-decay parameter λ, and the dropout rate. All models were fit using Adam with a learning rate of .001 for 5 epochs with batch size 32. We did not include these parameters in the grid search because of compute constraints, but did experiment with them on an ad-hoc basis. The validation accuracy of the models trained in this grid search was remarkably stable, ranging from approximately 68% to 74% with a few outliers above and below. See Figure 3.

We also experimented with using data from other languages to improve performance on English/Mandarin classification. We began by training a CNN with the same hyperparameters and architecture found via grid search in the two-class case on ten languages, simply swapping in a softmax output layer. We used this network to classify calls as either English or Mandarin by simply classifying to the language with larger predicted probability. This gave 68% validation accuracy. We additionally tried training a binary classifier on top of the vectors in $\mathbb{R}^{10}$ outputted by this network. It seems natural to incorporate the full predictions into the model, especially since some languages in the data set, e.g. German and English, are fairly similar. We used logistic regression as our binary classifier. Logistic regression is a generalized linear model that learns a parameter θ^* satisfying

$$\theta^* = \arg\min_{\theta} \sum_{i}^{m} \log(1 + e^{-y^{(i)}\theta^T x^{(i)}})$$

and predicts probabilities via

$$h_{\theta^*}(x) = \frac{1}{1 + \exp(-\theta^{*T} x)}$$

This gave worse validation accuracy (65.7%) than our simple initial classification scheme, however.

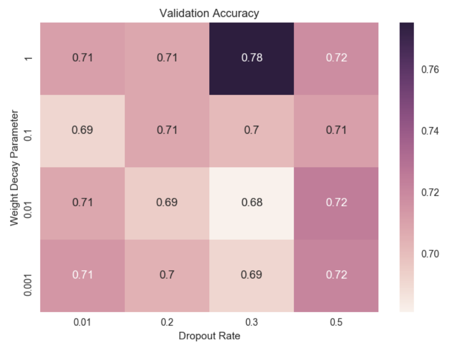


Figure 3: Validation Accuracy with 5 x 1 Filters

Finally, we trained a CNN on a feature set that contained Shifted Delta Cepstra as well as the raw MFCCs to incorporate more explicit information about time variation; this, too, did not give better results.

Given that neural networks have achieved good performance on larger data sets, it seems that the relatively small size of our data set was a significant problem for both RNNs and CNNs. The fact that validation error was largely insensitive to the various regularization and data augmentation techniques we tried was somewhat surprising, but may be explained in part by our consistent choice of architecture.

5 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) was the final model we tried, as well as the most successful. This was particularly satisfying, since the GMM framework is very intuitively reasonable in the context of language classification.

Each language contains a fixed number of distinct phonetic expressions. Under the GMM modeling approach, it is assumed that each of these distinct expressions has associated with it a Gaussian distribution, and that each utterance is drawn from one of them. For any given utterance, the phonetic expression to which it belongs is an unobserved latent variable, and the EM algorithm must be employed in order to obtain the probability of that utterance occurring [TCD]. This probability, conditional on the utterance coming from language ℓ, is given by

$$\begin{aligned} \mathbf{P}(x^{(i)}|\ell) &= \sum_{z} \mathbf{P}(x^{(i)}, z|\ell) \qquad (1) \\ &= \sum_{z} \mathbf{P}(x^{(i)}|z,\ell)\mathbf{P}(z|\ell), \end{aligned}$$

where $x^{(i)}$ is the i^{th} utterance from a phone call, and z is a phonetic expression "cluster", which is a latent variable that determines the Gaussian distribution from which $x^{(i)}$ is drawn.

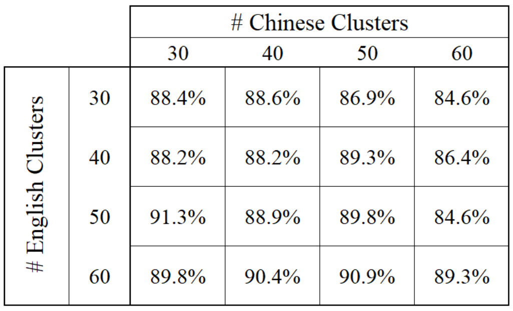
To make call-level predictions, a mixture model is first fitted for each language separately. For each utterance in a phone call (an MFCC vector) the probability of that utterance occurring in each language is computed via equation (1). We tried two approaches for aggregating frame-level predictions into call-level predictions. We first used a majority vote, where a call was classified to the language with higher probability in a majority of the frames. We then tried making the assumption, akin to that of Naive Bayes, that the features for each frame in a call are generated by independently sampling from the learned Gaussian mixture

$$\mathbf{P}(x^{(1)}, x^{(2)}, \ldots, x^{(m)}|\ell) = \prod_{i=1}^{m} \mathbf{P}(x^{(i)}|\ell)$$

We then classified via

$$\arg\max_{\ell \in \{\text{English}, \text{Mandarin}\}} \prod_{i=1}^{m} \mathbf{P}(x^{(i)}|\ell)$$

An important hyperparameter impacting the performance of the GMM model is the number of phonetic expression "clusters," i.e. the number of latent variables, that are set for each language. The validation accuracy resulting from the various parameter sets can be found in Figure 4.



		# Chinese Clusters			
		30	40	50	60
# English Clusters	30	88.4%	88.6%	86.9%	84.6%
	40	88.2%	88.2%	89.3%	86.4%
	50	91.3%	88.9%	89.8%	84.6%
	60	89.8%	90.4%	90.9%	89.3%

Figure 4: GMM validation accuracy

Using 50 clusters for English and 30 for Chinese as suggested by these results, we obtained 90.3% test accuracy.

6 Discussion/Conclusions

Our preliminary testing involved making predictions utilizing only static MFCC feature vectors, combined with simple statistics of their distribution in each call (e.g. mean, variance). The resulting models performed particularly poorly, and often performed worse than random guessing. The failure of these models highlights the importance of granular time dynamics. To a large extent, a language is not distinguished by the presence of certain sound waves, but rather by the patterns they form and the sequence in which they are produced. By explicitly incorporating this information into our features via SDC features and using GMM and neural network models, we were able to effectively capture this crucial information, leading to improved predictive power. The modeling assumptions of the GMM ultimately turned out to be vital since we had only limited data.

7 Future

There exists a rich set of literature describing techniques for extracting information from a signal, but their successful implementation requires an advanced and thorough knowledge of signal processing. Given more time, we would like to explore this area in greater depth, which would allow for enhanced data cleaning and normalization, as well the inclusion of additional features. Helpful features would capture information such as the rhythm and modulation of speech.

We also experimented briefly with pretraining a CNN on samples from the learned Gaussian mixtures before fine-tuning using real data in an attempt to combine the strong performance of the GMM with the modeling flexibility of a CNN. This was computationally intensive, but could be interesting to explore further.

Contributions

Everyone contributed equally to all parts.

References

[BHYM17] C Bartz, T Herold, H Yang, and C Meinel. Language identification using deep convolutional recurrent neural networks. *CoRR*, abs/1708.04811, 2017.

[C+15] François Chollet et al. Keras. https://github.com/fchollet/keras, 2015.

[CM] R.A Cole and Y.K Muthusamy. "The OGI Multilanguage Telephone Speech Corpus". *Proceedings International Conference on Spoken Language Identification*, vol. 2 pp. 895-899 Oct. 1992.

[EAS] L. Wang B. Yin E. Ambikairajah, H. Li and V. Sethu. "Language Identification: A Tutorial". *IEEE Circuits and Systems Magazine*, vol. 11, no. 2, pp. 82-108, 2011.

[FRD] D. Reynolds F. Richardson and N. Dehak. "Deep Neural Network Approaches to Speaker and Language Recognition". *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671-1675.

[ILM] O. Plchot D. Martinez J. Gonzalez-Rodriguez-P. Moreno I. Lopez-Moreno, J. Gonzalez-Dominguez. Automatic language identification using deep neural networks. *2014 IEEE ICASSP.*

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[RZ] J. Gonzalez-Dominguez D. T. Toledano J. Gonzalez-Rodriguez R. Zazo, A Lozano-Diez. "Language Identification in Short Utterances

Using Long Short-Term Memory (LSTM) Recurrent Neural Networks". *PLoS One*.

[TC] G. Tzanetakis and P. Cook. "Musical Genre Classification of Audio Signals". *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293-302, 2002.

[TCD] Reynolds Torres-Carrasquillo and Deller. "Language Identification Using Gaussian Mixture Model Tokenization". *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 757-760, 2002.

[ZFZ] K. M. Ting Z. Fu, G. Lu and D. Zhang. "A Survey of Audio-Based Music Classification and Annotation". *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303-319, 2011.