Real-Time Gender Identifier based on Voice

Antonio De Lima Fernandes, Andrew Ho

1.Abstract

The accuracy of speech recognition and speaker identification systems improve when the gender of the speaker is known. Though gender identification from speech is a well understood problem, there does not exist any open-source, real-time gender identification system freely available. This paper describes a real-time gender classifier based on voice and implemented in Python. Given a real-time audio stream or audio file in .wav format (which contains speech), the classifier labels the speech segments as male, female, or non-speech. The classifier achieves >90% accuracy using Random Forests and MFCC features.

The project is available on github here: https://github.com/antoniorohit/CS289A/tree/master/Final%20Project

2. Introduction

Speech recognition is one of the most efficient ways humans can communicate with a machine. If acoustic waves could perfectly map to words, a variety of exciting applications could be realized. However, speech recognition is difficult in reality. One reason is that there is much variability in people's voices. In addition, many applications of speech-recognition require voice authentication or speaker-recognition in order to be practically useful.

If speech recognition systems account for gender variability, the accuracy of speaker-recognition and even speech detection improves. Therefore, we built a real-time gender identifier based on voice. We can identify in real time the gender of the speakers with >90% accuracy on a rigorous test protocol we defined. Our work also studies the variation in performance based on factors like training database, recording device, and real-time tradeoffs. The gender identifier we implemented is completely open-source, and meant to be easy to use, so that it could be integrated into other speech recognition systems.

3. Related Work

- **Capstone Project on Speaker ID [1]**
  In his capstone project, Antonio [1] worked on implementing a speaker recognition system for mobile applications. The gender classifier in this work arose as a way to boost speaker recognition accuracy in that capstone project. Most of the code for this work was developed independently, though we relied heavily on the intuition and methods gained from Antonio's work [1].
- **Gender recognition from speech [2]**
  Results of this paper showed that, for gender recognition, most acoustic parameters lead to good

performance. This paper provides solid background to support our hypothesis that gender recognition is essential in improving speech recognition system.

- **Voice Activity Detection [3]**
  This paper discuss VAD technique for VoIP. It gives a quantitative measurement for speech quality. Studying the paper enabled us to implement an efficient energy-based method to detect speech.
- **Pitch Extraction [2]**
  This paper categorizes a speaker's gender based on his/her average pitch. Just by thresholding the pitch frequency, the shows significant improvement in word recognition accuracy.
- **Mel Frequency Cepstral Coefficient (MFCC) Extraction [5]**
  This tutorial explains MFCCs and provides well-documented Python code for MFCC extraction. We re-use this code almost as-is, with only minor modifications.
- **Emotion recognition via gender differentiation [6], [7], [9]**
  The papers here provide solid background to understand emotion recognition from speech. The first article focus more on the influence gender differentiation has on emotion recognition and the second paper emphasize on models to evaluate it statistically. The third paper gives some motivation to using an ensemble method for gender detection. Since gender recognition and emotion detection are closely related, many conclusions from these papers apply to our problem as well.

4. Methods

Overview of the Pipeline

Our training/test datasets consist of wav files containing the voice of an adult male or female speaker. For each file, we preprocess it first by framing it into smaller chunks, and classifying each chunk as speech or non-speech. The next step is featurization. We use Mel Frequency Cepstral Coefficient (MFCC) as the primary features because MFCCs outperformed Linear Prediction Cepstral Coefficients (LPCCs) in our tests. We also append the pitch of the speech as a feature since this is a key differentiator between male and female voices. After featurization, we train our random forest classifier. We measure the performance of the gender-identifier by running it on test data and additionally evaluate its real-time response. Finally, we integrate our program into a speaker recognition system that has four different speakers.

Dataset Used

Test Protocol

We reused a test protocol (developed for a separate project) that described the audio files to be used to train and test the system. At a high level, the dataset consists of 2 files, 24 minutes each, one to train and the second to test the system. The recordings were done in different controlled environments: clean (silence), with background noise (BGN), with background voices (BGV), and with echo. Recordings were captured on both an Android device (Nexus 5) as well as an Apple device (Macbook Pro). The recordings were then labeled for speakers and conditions using Audacity (software). Figure 1 below shows one of the audio files recorded and labeled according to this protocol. This protocol ensured that we had a repeatable, reliable and complete method of testing the system.

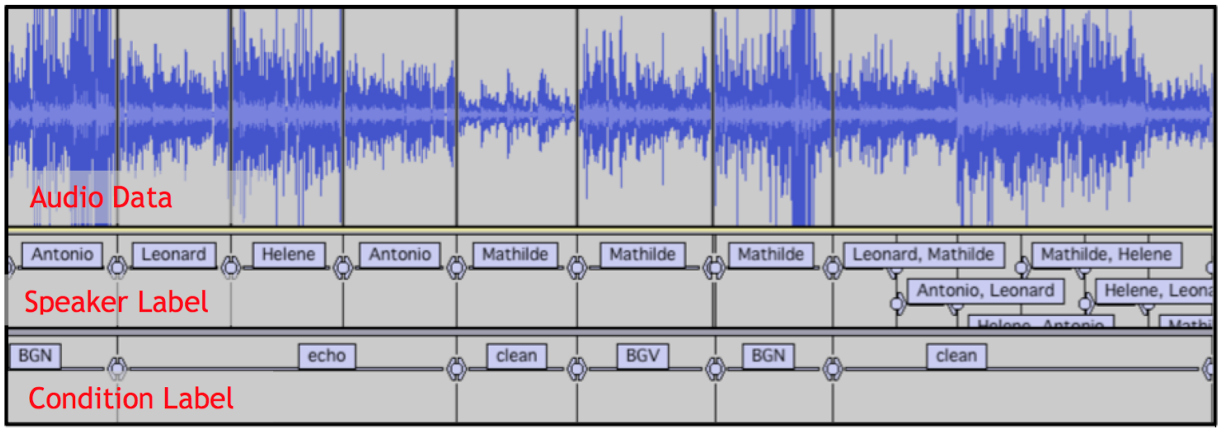

Figure 1. Sample of Audio File generated According to the Test Protocol

Voxforge

The second training dataset we used is the 16kHz_16bit subset of the openly available voxforge dataset [9].

Signal Processing Stages

- **Framing:** As described in the overview, the first step in processing an audio stream is to frame it into small chunks in order to be able to generate real-time results. We used a hamming window to do this.
- **Speech Detection:** After the signal is framed into chunks, each chunk is classified as speech or non-speech based on the Adaptive Energy Detection (AED) method described in [4]. Note that AED is performed on ~10ms frames from the chunk repeatedly, and chunks that retained more than half the signal after AED were retained as speech.
- **Feature Detection:** If the chunk is speech, it is then featurized. We used Mel Frequency Cepstral Coefficients (MFCCs) appended with the pitch of the voice. For the MFFCs, we used 55 filters and retained 30 coefficients. In addition, the high frequency was limited to 6000 Hz.
  For pitch detection, an open-source implementation [10] was used. However, this is simple but not very accurate, leaving room for improvement.

Note: Since pitch is an important feature in gender identification, we appended multiple (30) copies of the feature so that it would be present in most of the decision trees in our ensemble.

Machine Learning Model

Once we have the feature vector for each chunk, we train a Random Forest (RF) classifier of depth 10 and with 100 estimators. Ensemble methods like random forests are actually well suited to small sized data set with a high number of features [8] because of the (a) automatic feature selection that naturally occurs and (b) Their efficiency. Since this is the case with our gender classifier as well, we selected a broad and shallow random forest to get the best performance, yet avoid overfitting. The precise number of estimators and depth were chosen based on the outcome of cross-validation.

Implementing the RF classifier was very easy using the sklearn library, and provided the best performance. We tried out SVM and KNN classifiers but obtained worse results both in accuracy and speed.

Testing

After training is complete (either on the test protocol or on the voxforge data), we need to test the models built. All tests in this work were performed on the test files from the test protocol described above. We followed the same signal processing pipeline as for test, and predicted gender for each chunk.

In order to test real-time performance of our system, we wrote a python script to stream in audio from the laptop microphone, process it as above, and then print out the predicted gender to the console.

Tools and Methods

Python was used to implement the gender detector, along with major dependencies on supporting packages: numpy, pickle, pyaudio and sklearn.

5. Results

Variables Under Study

Here are the parameters we tested on:

- **Chunk Size**: For real-time detection, the audio stream needs to be framed into chunks. The smaller the chunks, the more real time the performance, but lower the accuracy. We tested with chunk sizes of 0.25, 0.5, 1 and 2 seconds.
- **Device**: As described earlier, we created a test protocol consisting of training and testing files recorded under different conditions. Each of these recordings were carried out simultaneously on a

Macbook laptop (Mac) and a Nexus 5 phone (Android). The variation in test accuracy for these two devices was studied

- **Machine Learning Model**: We tried Random Forests (RF) and K-Nearest Neighbors (KNN)
- **Training Dataset**: Different combinations of training on the voxforge data, Macbook training data, Android training data, and then testing on the Macbook and Android test data.
- **Features**: Contribution of MFCC features and the pitch feature to accuracy

Outcomes

Overall Accuracy vs chunk size

To begin with, here is the overall accuracy of our Random-Forest based gender detector when Android and Mac data was lumped together both for training and testing.

Chunk Size (s)	**Accuracy**
0.25	82.3%
0.5	85.7%
1	87.0%
2	86.2%

Effect of Device, Machine Learning Model and Training Dataset

Next, we tested the following scenario: training and testing based on recordings from the same type of device (training for Mac and testing for Mac, same for Android).

We repeated this for both a Random Forest Classifier (100 estimators, depth 10) as well as a K-Nearest Neighbour Classifier (20 neighbours) implemented using scikit-learn. The parameters for each classifier were previously selected using cross-validation.

Device	**Chunk Size (s)**	**Accuracy**	
		RF	KNN (20)
Mac	0.25	84.6 %	75.8%
	1	90.4 %	72.4%
Android	0.25	85.0 %	81.8%
	1	**92.1 %**	82.7%

Clearly the highest achieved accuracy is >92% for the Android device, RF classifier and 1s chunk size. The Android device outperformed the Mac device in most cases, and this is attributed to a higher quality mic on the Nexus 5.

Conducting a similar test, but training on the voxforge dataset yielded the following results:

Device	**Chunk Size (s)**	**Accuracy**
Mac	0.25	67.8%
	1	71.2%
Android	0.25	79.7%
	1	**88.8%**

Thus the best accuracy in this case (89%) is reduced somewhat from 92%, but is still good. This is a useful result, because this could allow us to build a universal gender model and predict genders for un-enrolled speakers.

Effect of Features

Since pitch is such a large factor in gender estimation, we wanted to know how much of a contribution it made to the overall accuracy. So we tested the RF with and without pitch features on the test protocol data (chunk size of 1s):

Device	**Accuracy for RF**		
	MFCC Only	MFCC+Pitch	Pitch Only
Mac	81.9%	90.4%	76.7%
Android	85.6%	92.1%	80.6%

From above, clearly the pitch is not the sole distinguishing characteristic for gender detection.

Speaker ID with and without Gender Detection

Finally, we integrated this gender detector into the speaker identification project described in [1], and the gender detector contributed an average improvement of ~5% in accuracy!

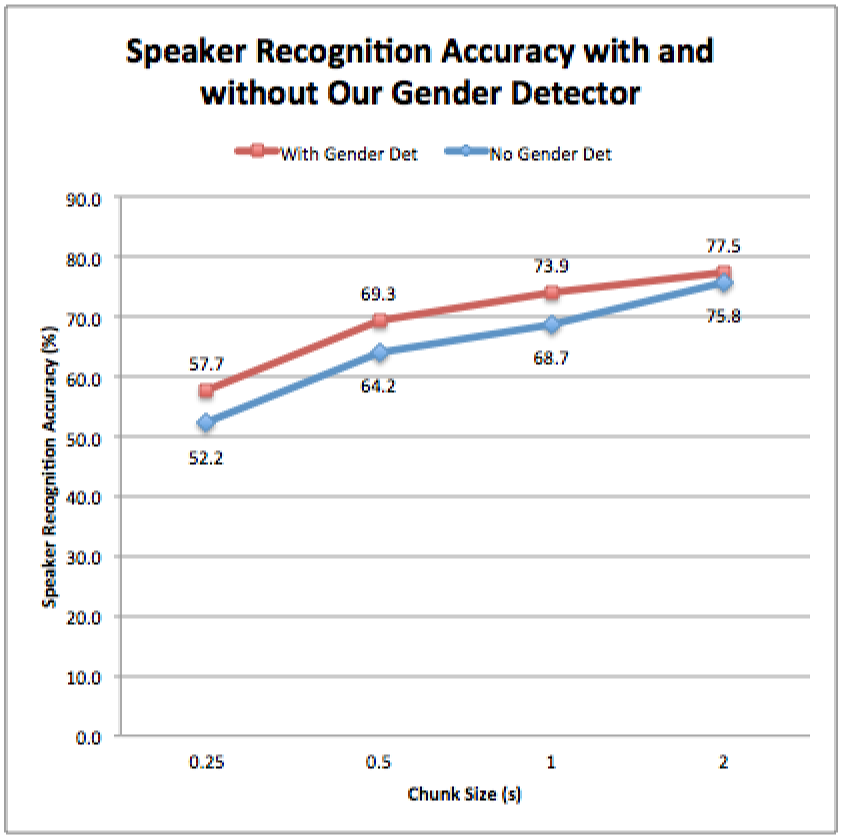

Figure 2. Accuracy of Speaker Recognition with the Gender Classifier

Real Time Testing

Only qualitative tests were performed to evaluate the real-time gender detector script that we wrote. Results showed good gender detection accuracy (~80%) for most situations, with near real time (1 second lag) performance.

7. Conclusion and Future Work

This paper presented a real-time open-source implementation of a gender classifier with >90% accuracy on a rigorous test set. As hypothesized, this gender-classifier boosted the accuracy of the speaker recognition system in [1]. Though these initial results are promising, there are much room for improving the system. The first would be to implement a more robust VAD algorithm to classify speech/non-speech. Next would be to visualize the principle features, and use them to build stronger trees. Studying the misclassified audio chunks could provide further insight into how to improve the RF classifier. It would be also interesting to test out the classifier on more datasets such as TIMIT [11], and compare it to state-of-the-art systems in literature. Finally, no Machine Learning problem can claim to be thoroughly investigated without throwing a Neural Network at it, so this could be an additional path to try out.

8. References

[1] Antonio Rohit. Caltranscense, empowering people with hearing disabilities. Master’s thesis, University of California, Berkeley, 2015.

[2] Wu, Ke, and Donald G. Childers. "Gender recognition from speech. Part I: Coarse analysis." *The journal of the Acoustical society of America* 90.4 (1991): 1828-1840.

[3] Sangwan, Abhijeet, et al. "VAD techniques for real-time speech transmission on the Internet." *High Speed Networks and Multimedia Communications 5th IEEE International Conference on*. IEEE, 2002.

[4] Abdulla, W. H., N. K. Kasabov, and Dunedin–New Zealand. "Improving speech recognition performance through gender separation." *changes* 9 (2001): 10.

[5] Python-based MFCC Tutorial:
http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

[6] Vogt, Thurid, and Elisabeth André. "Improving automatic emotion recognition from speech via gender differentiation." *Proc. Language Resources and Evaluation Conference (LREC 2006), Genoa*. 2006.

[7] Iliou, Theodoros, and C-N. Anagnostopoulos. "Statistical evaluation of speech features for emotion recognition." *Digital Telecommunications, 2009. ICDT'09. Fourth International Conference on*. IEEE, 2009.

[8] Rong, Jia, Gang Li, and Yi-Ping Phoebe Chen. "Acoustic feature selection for automatic emotion recognition from speech." *Information processing & management* 45.3 (2009): 315-328.

[9] Voxforge Voice Dataset

http://www.repository.voxforge1.org/downloads/SpeechCorpus/Trunk/Audio/

[10] Python based pitch estimation:

http://stackoverflow.com/questions/2648151/python-frequency-detection

[11] TIMIT Speech Corpus

https://catalog.ldc.upenn.edu/LDC93S1