Detecting speaker gender according to speaker diarization results

Jiaxin Yuan

University of Mannheim, Germany
https://github.com/Tttthea/speaker.git
{jiyuan}@students.uni-mannheim.de

1 Introduction

Our project focuses on detecting speakers and their gender from audio recordings, which can be useful for identifying points of emotion change. Emotions are often reflected in speech, and individuals generally have different emotional baselines. Accurately detecting emotions in speech can be influenced by the speaker's gender. Certain acoustic features, such as fundamental frequency (also known as pitch), have been found to vary significantly between genders. These gender-based differences in speech characteristics can impact the performance of emotion detection models. Thus, detecting gender beforehand can enhance overall accuracy. Our project is targeted to solve this gap, which consists of two main parts: Speaker Diarization and Gender Classification.

In the Speaker Diarization part, we will explain the concept of Speaker Diarization, various approaches used in this field, and the current state-of-the-art method, specifically Pyannote, which we have adopted for our project. We will also discuss existing benchmarks, evaluation criteria, and datasets commonly used for evaluating speaker diarization systems.

Moving on to the Gender Classification part, we will explore machine learning-based and deep learning-based methods to address the task of gender classification. We will discuss different techniques for preprocessing the data, feature engineering, model selection and inference. Additionally, we will present experimental results comparing the effectiveness of various approaches.

Then, we will provide you with detailed instructions on how we implemented these methodologies and how you can utilize our tool.

1.1 Speaker Diariztaion

An explicit review can be found here[1]. Easily speaking, Speaker Diariztaion solves problem of "Who Speaks When". Given a complete input audio(.wav, .mp3...), it would notice and segment when there is speaker talking/not talking, and when the speakers change, and finally output Diariazation Output(.RTTM[2]). Speaker Diarization techniques can be roughly divided to traditional ones(without Deep Learning) and Deep learning based ones.

Traditional diarization systems can be illustrated in this picture 1, combined with following parts:

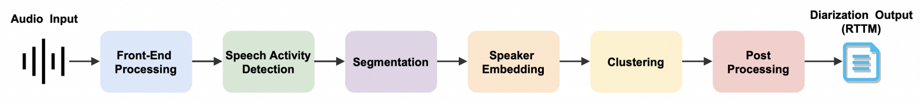

Fig. 1. Traditional speaker diarization system.

Front-end processing-feature extraction Preprocessing of inputs, including Speech Enhancement and Denoising

Speech activity detection Distinguishes speech from non-speech such as background noise

Segmentation Exactly identify the location of speaker change point

Embedding Speaker specific representation, prominent approaches such as i-Vector(SOTA before Deep Learning), d-Vector and x-vector(embedding layers from Neural Networks)

Clustering Group similar clips together, different clustering algorithms can be utilised here

Post processing-Re-segmentation Correction of previous errors

Commonly used evaluation criteria in benchmarks is DER(Diarization error rate).

$DER = \frac{false\ alarm\ +\ missed\ detection +\ confusion}{Total}$

JER(Jaccard error rate) is also mentioned in this industry, with the goal allocating each speaker with identical weight.

$JER = \frac{1}{N}\sum_i^{N_{ref}} \frac{false\ alarm_i\ +\ missed\ detection_i}{Total_i}$

Detailed explanation and additional evaluation matrices can be found here:

DER and JER explanation: https://github.com/nryant/dscore

DER realised by Pyannote: https://pyannote.github.io/pyannote-metrics/reference.html

There are some public free datasets that we found:

TOEFL Listening practice: https://www.kaggle.com/datasets/wiradkp/mini-speech-diarization

ICSI:https://groups.inf.ed.ac.uk/ami/icsi/download/

AMI: https://groups.inf.ed.ac.uk/ami/download/

There are some open source libraries and APIs that focus on this task:

EEND: https://github.com/hitachi-speech/EEND

Spectral clustering: https://github.com/wq2012/SpectralCluster

PyAnnote: https://github.com/pyannote/pyannote-audio

Essentially, current research in the field of speaker diarization and related tasks often involves the utilization of machine learning and deep learning methods. For instance, speech activity detection, which aims to identify whether there is voice present or not in an audio signal, can indeed be framed as a binary classification task.

1.2 Pyannote

Pyannote is a Python library that facilitates the construction of speaker diarization pipelines using the PyTorch machine learning framework [3]. With its capabilities, Pyannote has demonstrated state-of-the-art performance on three benchmark datasets [4]. The library offers pre-trained models that are readily available for use and also supports fine-tuning for further customization and optimization.

Technical background Pyannote offers an extensive solution that surpasses traditional models by incorporating overlapped speech detection. The standaord pipeline built a neural network called PyanNet, which addresses various tasks within the pipeline. The pipeline itself comprises three key components: speaker segmentation, speaker embedding, and clustering (referenced as [5]). As shown in Figure 2, we have expanded upon the standard pipeline by integrating a well-tested gender classifier, which can potentially be used to refine segmentation (although this has not been implemented yet, it holds theoretical promise).

For our speaker diarization project, we leveraged publicly available data, which had already been covered by the pre-trained model. Hence, we did not employ fine-tuning techniques. Nevertheless, the performance on the AMI and ICSI datasets yielded satisfactory results. Specifically, we utilized the pre-trained model to perform speaker diarization. Based on these outcomes, we segmented the complete audio into individual clips and organized them by each speaker, as detailed in the code "SpkrDiarization.py".

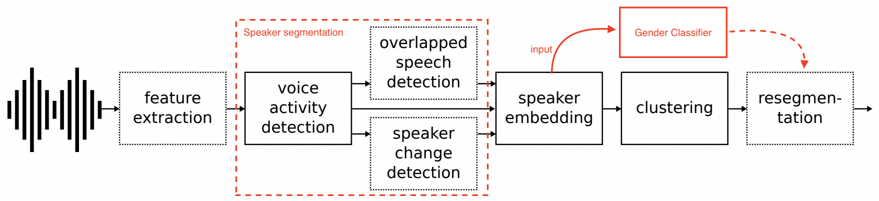

Fig. 2. Traditional speaker diarization system.

Notice that you have to downupdate pytorch version, because pytorch 1.11 is required for speechbrain compatibility. In my practice, installing as pyannote github shown would arise error later during inference. Therefore, installing as this notebook written might be the only work way. Explanation for each function can be found in commands.

2 Speaker gender classification

Compared to Speaker Diarization, gender classification is relatively an easier task. In contrast, gender classification can often be approached using a more straightforward classification pipeline. The process typically involves extracting relevant features from the audio, such as pitch(F0), formants(f1/f2/f3 of vowels), or other acoustic characteristics(mfcc coefficients, Spectrogram...), and feeding them into a classification model. This model can be a simple binary classifier, such as logistic regression, support vector machines (SVMs), or even a deep neural network, depending on the complexity of the gender classification task.

The classification pipeline usually consists of the following steps:

Pre-processing and feature extraction : It involves several steps, including denoising, format transformation (e.g., converting from m4a to wav), and windowing. We experimented with different input formats, such as raw sequential data, MFCC coefficients, and Embedding. We encountered a challenge with the varying lengths of the raw wav sequences. To address this, we extracted a fixed duration of 6 seconds for each audio, which was the minimum length in the dataset. Considering the gender imbalance in our dataset, we employed different windowing strategies to augment the datasets. However, we found that the raw wav input alone was not directly comparable to the other two methods mentioned. To overcome this, we performed waveform separation and extracted MFCC coefficients, following the blog [6]. Additionally, we extracted the Fundamental Frequency as an additional feature. A lot of research have conducted working on the features, you can find additional information as below:

https://www.sfu.ca/sonic-studio-webdav/cmns/Handbook%20Tutorial/SpeechAcoustics.html
https://newt.phys.unsw.edu.au/jw/voice.html
https://www.youtube.com/watch?v=TWRB443YrHI
https://erikbern.com/2017/02/01/language-pitch.html

We also explored an interesting approach proposed in a paper (referenced as [7]), which claimed to achieve 98.5% precision using a graph as input. However, we faced challenges in implementing this method due to a lack of practical details in the paper(input data).

Model training : The extracted features are used to train a gender classification model using supervised learning techniques. The model learns to map the input audio features to the corresponding gender labels. During our experiments and comparisons, we trained our models using the VoxCeleb Dataset. We explored different settings and methods, and you can find the details and results through the provided link[8].

Among the single models we tested, XGBoost demonstrated the most stable and high performance. When considering all the ensemble methods, we found that stacking, which involves combining predictions from multiple well-performed models, yielded the overall best results. However, we noticed that this stacking approach required a significant long excution time. In addition, selecting which models can be also problematic. As a result, we decided to select XGBoost as our final model for the pipeline, taking into account factors such as time efficiency, performance, and generalization.

Model evaluation : To assess the performance of our models on data that they had not been trained on, we conducted testing on unseen datasets, namely SVD [9] and RAVDESS [10], which have been used in paper [7]. During the testing process, we discovered that certain models, such as SVM and Logistic Regression, decrease drastically in performance on these datasets. This finding led us to exclude these models from further consideration. However, it is important to note that SVD and RAVDESS datasets differ significantly from our target dataset, which means that achieving high generalization on these datasets was not our primary focus. Instead, our main objective was to select models that exhibited strong performance and generalization capabilities on our target dataset. While the performance on SVD and RAVDESS provided valuable insights, we prioritized models that demonstrated consistent and reliable performance on our primary dataset over those that performed well solely on the additional datasets.

Inference : After segmenting the audio clips, we proceed to infer the gender of each clip. To determine the speaker's gender, we adopt a voting mechanism where we consider the most probable prediction based on the votes obtained. It is important to highlight that the original method described earlier was not feasible to execute on either a local CPU or the available free GPU resources provided by Google Colab. To overcome this limitation, we developed an alternative approach. In this alternative approach, we utilize Pyannote for feature extraction by extracting embeddings from each training audio. These embeddings capture essential information from the audio signals. Subsequently, we use XGBoost to process the extracted embeddings. This alternative approach allows us to overcome the computational challenges associated with the original method. By extracting embeddings using Pyannote and feeding them as input to XGBoost, we can take advantage of Pyannote's feature extraction capabilities and harness the modeling strengths of XGBoost for accurate gender classification.

Although gender classification is generally considered an easier task compared to speaker diarization, it still requires careful consideration of feature selection, model architecture, and training data to achieve optimal results. In practice, several challenges may arise in gender classification. These challenges include variations in speech patterns and characteristics across different languages, dialects, and accents. Moreover, gender is a complex and multifaceted concept that can be influenced by societal, cultural, and individual factors, making it challenging to capture accurately through speech analysis alone. Additionally, handling real-world audio data with noise, background sounds, and low-quality recordings can also pose challenges in achieving accurate gender classification.

3 How to use this tool

The Google Colab notebook ”tool.ipynb” is available for running the pipeline with the convenience of free GPU settings. You can modify the path in the notebook to specify the dataset you want to detect. Here's a brief overview of how the pipeline works:

1. Use the Pyannote pre-trained model to extract an Rttm file, which contains information about the speaker segments in the audio.

2. Group the audio clips based on the identified speakers using the Rttm file.

3. Train the gender classifier using the embedding arrays obtained from Pyannote. The embeddings capture important features from the audio.

4. Group the audio embeddings for each speaker and predict the gender for each sample.

5. Finally, apply a voting mechanism to determine the final gender prediction. Since the training set may be imbalanced, the pipeline takes into account the relative ratios of predicted genders instead of simple counting.

It's important to note that you have flexibility in readjusting the pipeline according to your needs. For example, you can explore using different classifiers for gender classification if desired.

If you prefer to use the original method of inputting wav clips, you can refer to the provided repository and access the ”sd_classfifier_pipeline.ipynb” notebook.

By utilizing the notebook and adapting the pipeline as necessary, you can run the gender classification process and obtain the final results for your specific dataset.

References

1. Park, T. J., Kanda, N., Dimitriadis, D., Han, K. J., Watanabe, S., & Narayanan, S. (2022). A review of speaker diarization: Recent advances with deep learning. Computer Speech & Language, 72, 101317.
2. Explanation can be found in this Appendix: https://web.archive.org/web/20170119114252/http://www.itl.nist.gov/iad/mig/tests/rt/2009/docs/rt09-meeting-eval-plan-v2.pdf
3. H. Bredin et al., ”Pyannote.Audio: Neural Building Blocks for Speaker Diarization,” ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 7124-7128, doi: 10.1109/ICASSP40776.2020.9052974.
4. https://paperswithcode.com/task/speaker-diarization
5. https://huggingface.co/pyannote/speaker-diarization/resolve/main/technical_report_2.1.pdf
6. https://www.primaryobjects.com/2016/06/22/identifying-the-gender-of-a-voice-using-machine-learning/
7. Abeer Ali Alnuaim, Mohammed Zakariah, Chitra Shashidhar, Wesam Atef Hatamleh, Hussam Tarazi, Prashant Kumar Shukla, Rajnish Ratna, ”Speaker Gender Recognition Based on Deep Neural Networks and ResNet50”, Wireless Communications and Mobile Computing, vol. 2022, Article ID 4444388, 13 pages, 2022. https://doi.org/10.1155/2022/4444388

8. https://docs.google.com/spreadsheets/d/1fKaLLKsh4Cqa8BTXkp7vex$_{o}xn45TgMBWBzuAmYsA68/edit?usp = sharing$
9. W. J. Barry and M. Pützer, Saarbrücken Voice Database, Institute of Phonetics, Univ. of Saarland, 2022, http://www.stimmdatenbank.coli.uni-saarland.de/.
10. S. R. Livingstone and F. A. Russo, "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English," PloS one, vol. 13, no. 5, article e0196391, 2018.