– Independent Work Report Fall 2021 –

Speaker diarization via adaptive classification

Cathy Teng

Adviser: Adam Finkelstein

Abstract

Captioning and transcripts are available for most forms of audio and audiovisual media today. Transcription is often done by hand, and the accuracy of manual transcription is currently superior to automated techniques, but deep learning methods applied to speaker diarization have reduced error rates considerably. Speaker diarization (SD), or the problem of "who spoke when" given an audio recording, can be reframed into an adaptive speaker classification problem. A neural network can be trained on labeled speaker segments from a recording by extracting speaker features known as speaker embeddings and grouping similar embeddings together. This method is an improvement upon the traditional SD framework in that the only information required from the user is the number of speakers, and it performs similarly to Resemblyzer, the package used to extract the embeddings.

1. Introduction

In recent years, there has been a growing focus on making online content, whether video-, audio-, or text-based, more accessible to those with disabilities. On a federal level, efforts towards increasing accessibility have been of concern since 1998, when Congress first "amended the Rehabilitation Act of 1973 to require Federal agencies to make their electronic and information technology (EIT) accessible to people with disabilities" [3]. Under this legislation and other rulings since then, federal agencies and their partners must ensure that their digital communications (emails, documents, website content, etc.) are Section 508 compliant [21]. Section 508 compliance means ensuring that content is accessible for those with disabilities [21], whether it be through screen readers, video transcriptions, or other means, and the U.S. Access Board has created standards to be utilized by federal agencies with digital content [6].

The content that regular people consume daily is also more frequently being captioned and transcribed. Most creators on content-sharing and other streaming sites include captioning and transcripts alongside their videos and podcasts, and many (if not most) hearing people also enjoy content with subtitles [7]. Youtube provides an automated captioning feature, and it also used to have a community captions feature where viewers could submit their own transcript files for videos [13]. When Youtube's community captions was discontinued in 2020, creators began encouraging their subscribers to submit captions or provided the captions themselves [15]. More podcasts are now being accompanied by web pages that provide full transcripts of the content, and Spotify has even made it a goal to provide automatic transcriptions of podcasts on its platform – a feature which is now in its beta version [10].

Transcription takes work, and costs money. Companies such as Rev offer transcription and captioning by hand for $1.25 per minute as well as automatic transcription for only $0.25, but caution that the latter method produces "80%+ accuracy" [2]. Automatic transcription – which can be done using speech recognition technology – has come a long way in providing a fast and labor-free solution to generating captions. However, generating accurate and reliable transcriptions is a process that is sensitive to background noise and other issues that may stem from the quality of an audio recording and overlapping speakers [14]. Speech recognition systems also require a substantial amount of data to be able to recognize various pronunciations. In the specific case of podcast transcription, it would also be helpful to know *who is speaking when*, because there may be multiple speakers in a podcast and a speech recognition system cannot distinguish that.

2. Background

Speaker diarization (SD) is the problem of "who spoke when" given an audio recording. SD has generally been done in two steps (Figure 1 [4]). First, the different speakers are separated in the speaker segmentation step. The recording is partitioned into segments by speaker, and the system finds potential speaker change points using specific characteristics of the audio to create different segments. Next, the different segments are clustered in the speaker clustering step. Speech segments are clustered together and labeled as a particular speaker if they share specific characteristics. [4]

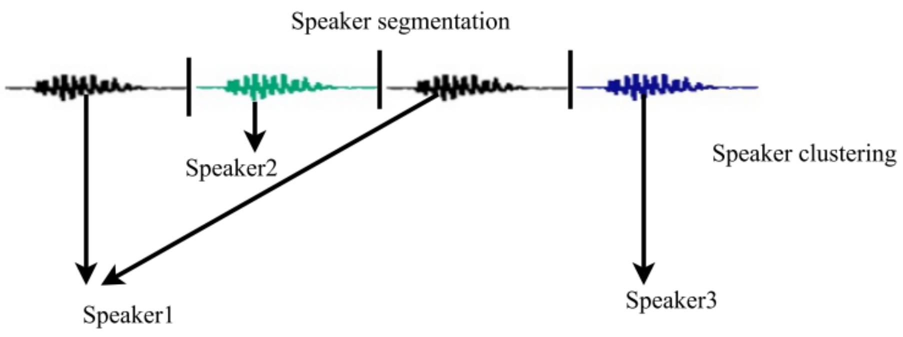


Figure 1: Speaker diarization typically consists of speaker segmentation and speaker clustering.

SD can be an automatic transcription tool that also labels the speakers in the transcript. Recent advances in the field have utilized deep learning to improve speaker segmentation – such as using neural networks to create representations of speakers and audio segments [12]. The neural networks themselves are typically either created with specific architecture to fit the problem at hand (using a specific audio dataset) or are pre-trained with speech data, and many of those pre-trained neural networks are long short term memory (LSTM) networks [20] or convolutional neural networks [11]. But despite all of these advances in recent years, the accuracy of speaker diarization models is still not ideal for reliable automatic transcript generation. Current state-of-the-art systems have diarization error rates, or the fraction of time that is not attributed correctly to a speaker or to non-speech [8], of 6.5-9.0 on the NIST-2000 CALLHOME dataset [9], and likely perform worse on multi-speaker recordings with overlapping voices, background noise, and poor sound quality.

3. Related Work

Deep learning approaches have been applied towards the feature extraction task in speaker diarization, and it is the deep learning that is of interest here. In 2015, Rouvier et. al first proposed speaker embeddings as a "set of high-level feature representations through deep learning" that can be used to identify a particular speaker [17]. The method learns high-level speaker features using a deep neural network (DNN) that is trained for a speaker identification task, and the feature vector itself is extracted from the hidden layer neuron activations. In the context of speaker diarization, speaker embeddings can be used in the clustering step, such that similar embeddings are clustered together and labeled as the same speaker. In a similar vein, x-vectors are another type of embedding that are created from a DNN [18]. The DNN is trained to classify speakers in the training data, and embeddings are later extracted from an intermediate hidden layer. X-vectors appear to be similar to speaker embeddings, but the main difference seems to be in the DNN architecture and the subsequent hidden layer from which the vector is extracted.

The particular method of speaker clustering using speaker embeddings utilized in this paper comes from the Resemblyzer package [16]. Resemblyzer generates 256-dimensional speaker embeddings from a pre-trained LSTM network by retrieving the final hidden state of the last layer, and also L2-normalizes the embedding vectors so that vectors can be compared for similarity by simply taking the dot product. The L2-normalization removes the need for further clustering methods – in the diarization demo file, providing segments of the audio that are known to be certain speakers is enough to diarize the entire recording.

Most research has focused on improving the individual parts of the traditional SD framework, such as better feature extraction for speaker segmentation and better clustering algorithms. Further improvements on the current SD framework may slow after a certain point, which is why it may be worth considering other approaches to the SD problem. There are two core components of the traditional framework that I will be focusing on in a new framework for SD. As seen with the Resemblyzer package, being able to compare embeddings without having to undertake a more comprehensive speaker clustering step is simpler and could even be faster than traditional clustering methods. Additionally, many SD frameworks approach an audio recording without any prior information. The system must try and distinguish not only the number of speakers but the features belonging to each speaker on its own, though information like the number of speakers would be easy for a user to input. Some SD frameworks, like Microsoft's Speaker Recognition and Conversation Transcription [19] do require the user to enroll their speakers by providing sample recordings. But when the number of speakers is very large, this may become tedious.

The general idea for my framework would be to transform speaker diarization into a speaker classification problem. Speaker classification is a much simpler problem and has very high accuracy even with less complex models. For instance, in our deep learning seminar, we observed that simple models can achieve high accuracy and can classify spoken numbers given data from 23 different speakers as well as the reverse — classifying audio segments from 23 different speakers. A simple fully connected model can achieve over 80% accuracy on a training dataset and over 60% accuracy on a testing dataset, with the entire dataset containing 23 speakers and 20 1-second audio segments per speaker (Figure 2, 3). Since speaker diarization is the problem of who spoke when, given that an audio recording can be segmented into different sections, a network could simply classify the person speaking at a certain time step after gathering speaker embeddings from the recording itself. Such a method may be more accurate than the traditional SD framework, since the networks are specific to each recording rather than using one network for all audio inputs. In this paper, I propose an alternative framework called *adaptive classification*, where I reframe SD into a speaker classification problem by creating a specific neural network to each audio recording using speaker embeddings and ultimately use the neural network to classify the speaker segments rather than cluster them.

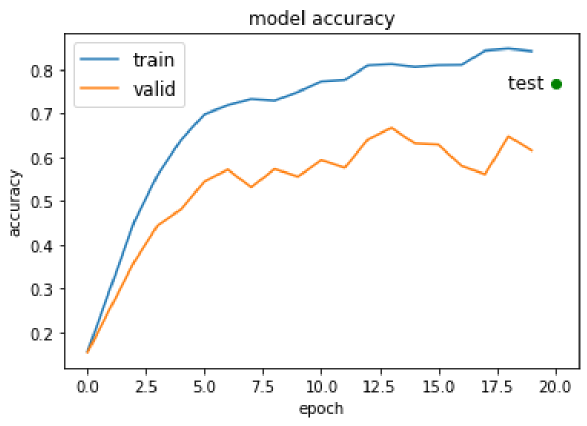
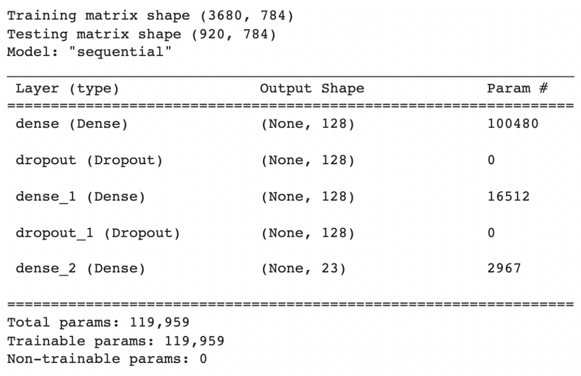


Figure 2: Simple fully connected model accuracy.



```
Training matrix shape (3680, 784)
Testing matrix shape (920, 784)
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 128)               100480

dropout (Dropout)            (None, 128)               0

dense_1 (Dense)              (None, 128)               16512

dropout_1 (Dropout)          (None, 128)               0

dense_2 (Dense)              (None, 23)                2967

=================================================================
Total params: 119,959
Trainable params: 119,959
Non-trainable params: 0
```

Figure 3: Simple fully connected model summary.

4. Approach

The framework I am proposing for the speaker diarization (SD) problem is called *adaptive classification*. SD can be transformed into a speaker classification problem by labeling speaker embeddings and then training a neural network specific to each recording. This framework is *adaptive* because each recording will have its own small neural network. The neural network predicts the probability of an input audio segment belonging to each speaker, respectively. The number of nodes in the output layer of the network can be fixed and have 100% accuracy because the user can input the number of speakers *N* as the number of nodes in the final layer.

While using the neural network after using labeled embeddings as training data is fairly straightforward, the crux of the framework lies in the process of assigning embeddings to speakers with a high level of confidence. As previously mentioned, Resemblyzer embeddings are L2-normalized and can be compared for similarity by taking the dot product. The main idea in my approach is to use speaker embedding similarities as a "first pass" to then generate labeled data for the neural network. Resemblyzer also can take an audio recording and output partial embeddings by sliding a window of a certain size across the recording and generate embeddings from the audio within each window. If one assumes that the first few seconds of a recording is the first speaker, then by subsequently comparing across the partial embeddings of the entire recording, one can then assign embeddings of high similarity to this first speaker. When the similarity begins to taper off, it is likely that the first speaker is no longer speaking, and one assumes that the next few seconds are the second speaker. The process repeats in this way, also making note of the fact that sections of the recording that are already "assigned" should be bypassed. By setting a high threshold for similarities, one can be fairly confident that the embeddings belonging to a speaker actually belong to that speaker. The embeddings with high confidence are the ones passed to the neural network as data, which should then be useful in identifying who is speaking for segments where the embedding similarities are less confident.

It is important to note here that this framework only applies to recordings with non-overlapping speakers. For simplicity, when creating the framework, I experimented with artificial multi-speaker data that I constructed myself, so the framework has not been tested on overlapping speech. But given the framework's architecture, if the first few seconds of the recording are in fact two speakers speaking at once, then the embedding will capture the characteristics of those two voices in tandem and the framework will assume that it is a single speaker. Likewise, it is also possible that an embedding with two or more speakers in it will be similar to one or more speakers and be labeled as a single speaker because each embedding is only assigned to one speaker. However, the framework is able to capture speakers speaking at different times in the recording, and more than once.

5. Implementation

5.1. Model Construction

This part of this section is based on part on the diarization demo from Resemblyzer [16]. First, a continuous embedding is generated for the whole audio recording (where partial embeddings are generated using the sliding window technique). Partial embeddings are embeddings created by sliding a window of a specific size, but the stride length is less than the size of the window so there is some overlap in audio between embeddings. All embeddings are 256-dimensional arrays, including partial embeddings. I used an encoding rate of 16, meaning that an embedding is generated every 0.0625 seconds (16 embeddings per second). Each partial embedding from Resemblyzer is taken from a 1.6 second slice of the original recording.

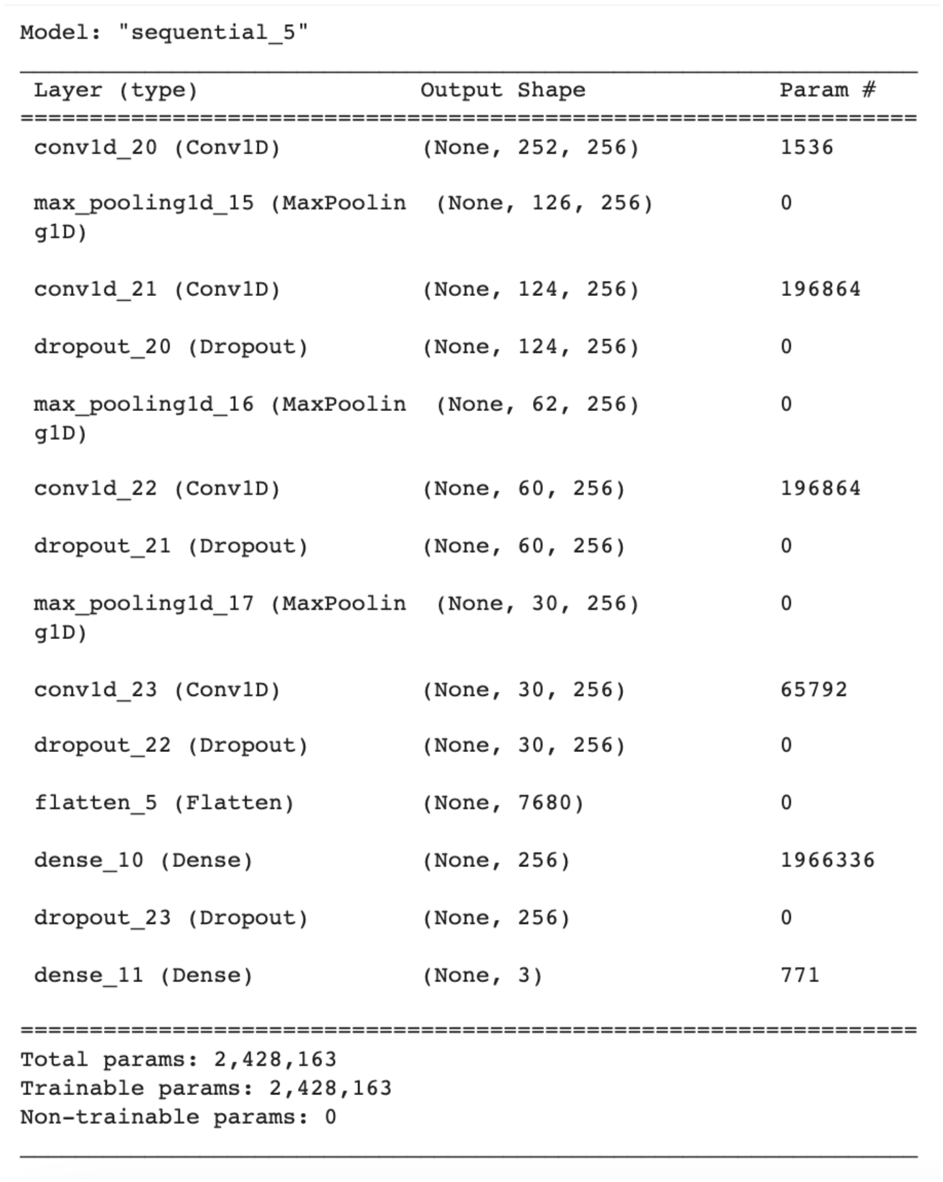
Next, in addition to providing the number of speakers, I assume that the first 3 seconds of the recording belongs to the first speaker. This is then used to create a representative embedding of a speaker, which I then use to calculate similarities with every partial embedding from the original recording. Similarities are calculated by taking the dot product of two embeddings (two 256-dimensional arrays), resulting in a scalar value. Thus for the first speaker I can create a similarity array containing the similarity scores between the representative embedding and the original recording.

The iterative process below follows:

1. All embeddings not yet assigned with a similarity greater than the similarity threshold of 0.7 in relation to the current speaker (the similarity array) are assigned to that speaker.
2. The newly assigned embeddings are transformed into time segments, and continuous segments are collated. This is possible because I have set a specific encoding rate and the partial embeddings also exist in an array. All segments belonging to a speaker are also padded with 0.5 seconds on both sides to avoid possible errors when transitioning between speakers.
3. The next time segment that has not yet been assigned and has a length of at least 3 seconds is assumed to be the next speaker. If all segments have been assigned, then the loop exits.
4. A representative embedding is generated from those 3 seconds and is used to calculate similarities with every partial embedding from the original recording and generate a similarity array.

The data and labels are then organized to feed into the neural network for training. The majority of audio recordings will have people speaking for different amounts of time, and the similarity threshold approach will also result in different amounts of data for each speaker. The above process outputted time segments assigned to each speaker, and to create data for the network I created continuous embeddings for each of the segments (partial embeddings over the entire length of each segment). However, since it is necessary to have the same amount of data per speaker for training, I chose to make the number of embeddings for each speaker equal to the number of embeddings for the speaker with the smallest amount of data. Embeddings were chosen randomly if the speaker had more data than needed. I also considered data augmentation to create equal-sized datasets, but for simplicity, I chose the former method, because continuous embeddings generate enough data for the 1-minute recordings I had created, and would generate more for podcasts and interviews.

The data is only used for training. There is no validation or testing set, because the partial embeddings from the original recording will then be fed into the network to be classified. I used an Alex-like network (AlexNet) and trained for 10 epochs with a batch size of 16. Further details on the network can be found in Figure 4. Other networks can be used, but AlexNet was used here because it was fast to train and provided some complexity in the different layers.


```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_20 (Conv1D)          (None, 252, 256)          1536

 max_pooling1d_15 (MaxPoolin  (None, 126, 256)         0
 g1D)

 conv1d_21 (Conv1D)          (None, 124, 256)          196864

 dropout_20 (Dropout)        (None, 124, 256)          0

 max_pooling1d_16 (MaxPoolin  (None, 62, 256)          0
 g1D)

 conv1d_22 (Conv1D)          (None, 60, 256)           196864

 dropout_21 (Dropout)        (None, 60, 256)           0

 max_pooling1d_17 (MaxPoolin  (None, 30, 256)          0
 g1D)

 conv1d_23 (Conv1D)          (None, 30, 256)           65792

 dropout_22 (Dropout)        (None, 30, 256)           0

 flatten_5 (Flatten)         (None, 7680)              0

 dense_10 (Dense)            (None, 256)               1966336

 dropout_23 (Dropout)        (None, 256)               0

 dense_11 (Dense)            (None, 3)                 771

=================================================================
Total params: 2,428,163
Trainable params: 2,428,163
Non-trainable params: 0
_________________________________________________________________
```

Figure 4: The Alex-like network architecture used in adaptive classification.

The Alex-like network was able to achieve over 90% accuracy on the training set for multiple runs (Figure 5).

5.2. Model Application

After training, the model can now be applied on the partial embeddings from the original recording. The model outputs the probabilities that a single embedding belongs to each speaker, respectively. I assigned an embedding to a speaker by taking the maximum of the predictions over all speakers at each time step. This can be done with high confidence because there are no overlapping speakers in

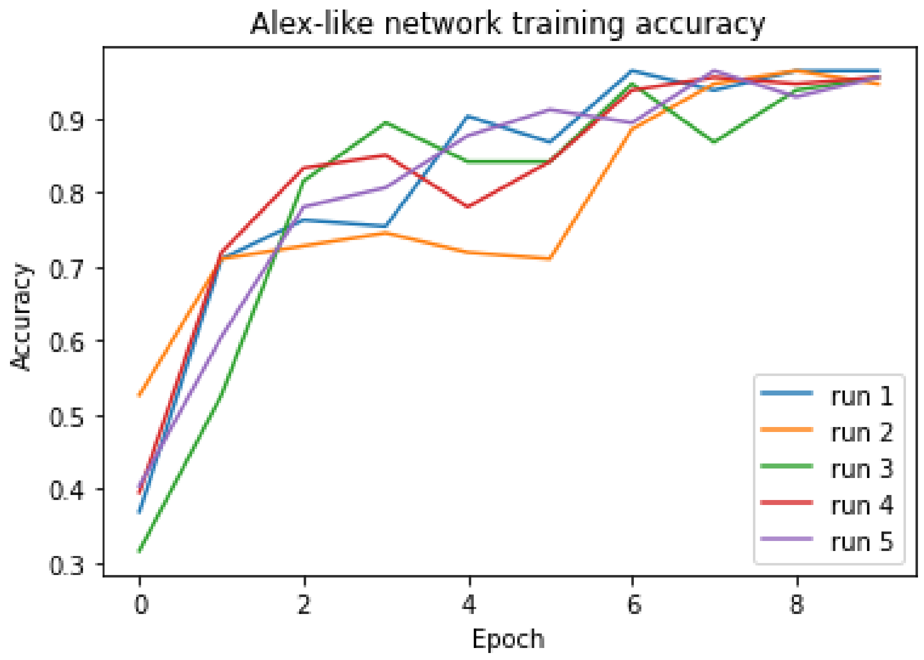


Figure 5: Over five training runs, the Alex-like model was always able to achieve over 90% accuracy.

the recording.

Time segments of when each speaker is speaking can be generated from the labeled embeddings and the order that they occur in. However, I often found that it was the case that within a large time segment belonging to one speaker, there would be a single embedding that would be labeled as a different speaker. After inspecting the model output, it appeared to be that the the embedding had very similar probabilities of belonging to correct speaker and the outputted speaker (most often in the 0.4-0.5 range), and the outputted speaker was slightly higher. This necessitated an optimization step to correctly label those single embeddings as the same speaker as the surrounding time segments.

The time segments can then be used to generate a transcript. I used the SpeechRecognition Python package [5] and its accompanying Google Speech Recognition functionality to create text from speech. I used the time segments to also segment the original recording and feed into the speech-to-text package, and also prepended each segment with the speaker (in this case, the number that identified the speaker; this can be easily changed afterwards).

6. Evaluation

Because my framework has no precedents, it is difficult to directly compare and evaluate my results with a comparable process. However, since I used Resemblyzer embeddings in my method, comparing my framework with the original seemed appropriate. Resemblyzer's diarization demo outputs a dictionary with the probability of each speaker speaking at every time step. I opted for comparing my output with the results obtained from taking the maximum probability at each time step from Resemblyzer's method. Resemblyzer asks the user to input time segments from the original recording that are known to belong to different speakers (speaker registration) and then uses those segments as representative embeddings to compare with partial embedding from the original recording.

To test my framework, I used the LibriSpeech dataset [1] to create artificial 3-speaker data. More specifically, I used the dev-clean dataset and the speakers numbered 84 (Christie Nowak), 174 (Peter Eastman), and 251 (Mark Nelson). I created multi-speaker recordings that were about a minute long, and with each speaker speaking for at least 5 seconds before switching to another.

To evaluate the model, I compared the transcripts generated using the modified Resemblyzer method and using adaptive classification against transcription by hand (see Appendix A). Adaptive classification may also lead to varying results even with the same recording, because a new neural network is trained each time the method is executed and may have different weights. Thus, the example provided is one possible outcome of a transcript. However, I observed over multiple runs that the speaker transition points did not change, which indicates that the variation in the network's weights between runs is not significant enough to be of concern.

A measure of success for my model would be if it meets or exceeds the performance of the modified Resemblyzer model. The adaptive classification model does fairly well overall, but does miss the mark on a few of the transitions between speakers (Appendix A). For instance, in the middle of the recording, the manual, adaptive classification, and Resemblyzer models are clearly different in the speaker translation in Table 1 (newlines are included for comparison purposes), and the adaptive classification model performs the poorest in this case.

Model	Line
Manual	speaker 2: now you may play a surana singer a dream of night for an Apple Gold lady
Adaptive classification	speaker 2: I dream of night for an Apple Gold lady
Resemblyzer	speaker 2: play it's a raining singer I dream of night for an Apple Gold lady

Table 1: Both adaptive classification and Resemblyzer do not capture the accurate speaker transition point, but adaptive classification misses more of the words than does Resemblyzer.

An instance in this particular recording where adaptive classification performs similarly to Resemblyzer (but still poorer than the manual transcription) can be seen in Table 2.

Model	Line
Manual	speaker 0: three maidens at the right wheel in a circle...
Adaptive classification	speaker 2: ... 3 Maiden speaker 0: the right wheel in a circle...
Resemblyzer	speaker 2: ... 3 made speaker 0: at the right wheel in a circle...

Table 2: Both adaptive classification and Resemblyzer do not capture the accurate speaker transition point, and they both miss portions of words at that point.

As noted earlier, adaptive classification can only work with non-overlapping speech at the moment. The limited evaluation provided above suggests that it can provide labeled transcripts with slightly less accuracy at speaker transition points compared with its Resemblyzer cousin. Adaptive classification can achieve this through only the input of the number of users in the recording as opposed to requiring the additional task of finding segments of the audio that are known to belong to certain speakers. Overall, adaptive classification appears perform comparably with the original Resemblyzer method, when accounting for slightly less accuracy and greater ease of use. It is also likely that adaptive classification will outperform Resemblyzer on longer recordings due to increased amount of training data that will improve the network's accuracy.

7. Conclusion

Speaker diarization via adaptive classification has the potential to be a framework that can provide accurately labeled transcripts for video and audio. The current iteration of the framework presented in this paper has addressed multi-speaker recordings with non-overlapping voices. The framework is an improvement upon other speaker diarization methods in that it requires some information from the user, but that information (the number of speakers in the recording) is simpler than the speaker registration required by other state-of-the-art systems, such as Resemblyzer or Microsoft's Speaker Recognition and Conversation Transcription. Adaptive classification was tested with 1-minute, 3-speaker recordings and performed similarly to Resemblyzer's method, which requires speaker registration, and may perform better with longer audio recordings.

Further research into adaptive diarization can go in many directions. The most obvious extension of the research presented here would be to test the framework on longer length recordings, with different neural network architectures, and using different parameter values (transition buffer time, minimum number of new speaker seconds, encoding rate, etc). Further experimentation and testing would help provide a more robust understanding of the limitation and strengths of the current system, and may reveal insights that would help improve it to be able to handle the overlapping speakers case. Additionally, I would be interested in calculating the diarization error rate [8] with these models with varying architectures and larger datasets.

8. Acknowledgements

I (Cathy Teng) pledge my honor that this paper represents my own work in accordance with University regulations. I would also like to thank my advisor, Prof. Adam Finkelstein, and our seminar's TA, Pranay Manocha, for their advice and support during this project.

References

[1] "Papers with code - librispeech dataset." [Online]. Available: https://paperswithcode.com/dataset/librispeech

[2] "Rev speech-to-text services: Convert audio video to text." [Online]. Available: https://www.rev.com/

[3] "Section508.gov." [Online]. Available: https://www.section508.gov/

[4] "Speaker diarization." [Online]. Available: https://wiki.aalto.fi/display/ITSP/SpeakerDiarization

[5] "Speechrecognition." [Online]. Available: https://pypi.org/project/SpeechRecognition/

[6] "U.s. access board - information and communication technology." [Online]. Available: https://www.access-board.gov/ict.html

[7] "Lights, camera, caption! why subtitles are no longer just for the hard of hearing," Jul 2019. [Online]. Available: https://www.theguardian.com/tv-and-radio/2019/jul/21/subtitles-tv-hearing-no-context-twitter-captions

[8] X. Anguera, "Diarization error rate." [Online]. Available: http://www.xavieranguera.com/phdthesis/node108.html

[9] H. Aronowitz, W. Zhu, M. Suzuki, G. Kurata, and R. Hoory, "New advances in speaker diarization." in *INTERSPEECH*, 2020, pp. 279–283.

[10] A. Carman, "Spotify will auto-transcribe podcasts over the coming weeks," May 2021. [Online]. Available: https://www.theverge.com/2021/5/18/22441886/spotify-podcast-transcription-accessbility-app-update

[11] P. Cyrta, T. Trzciński, and W. Stokowiec, "Speaker diarization using deep recurrent convolutional neural networks for speaker embeddings," in *International Conference on Information Systems Architecture and Technology*. Springer, 2017, pp. 107–117.

[12] G. Li and Y. Gu, "Distance-preserving property of random projection for subspaces," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 3959–3963.

[13] K. Lyons, "Youtube is ending its community captions feature and deaf creators aren't happy about it," Jul 2020. [Online]. Available: https://www.theverge.com/2020/7/31/21349401/youtube-community-captions-deaf-creators-accessibility-google

[14] J. Markoff, "From your mouth to your screen, transcribing takes the next step," Oct 2019. [Online]. Available: https://www.nytimes.com/2019/10/02/technology/automatic-speech-transcription-ai.html

[15] L. O'Dell, "Youtube pulled its community captions feature, so now more creators are making their own," May 2021. [Online]. Available: https://www.theverge.com/2021/5/21/22443577/youtube-captions-increased-deaf-campaigners

[16] Resemble-Ai, "resemble-ai/resemblyzer: A python package to analyze and compare voices with deep learning," Nov 2020. [Online]. Available: https://github.com/resemble-ai/Resemblyzer

[17] M. Rouvier, P.-M. Bousquet, and B. Favre, "Speaker diarization through speaker embeddings," in *2015 23rd european signal processing conference (eusipco)*. IEEE, 2015, pp. 2082–2086.

[18] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[19] E. Urban, "Conversation transcription (preview) - speech service - azure cognitive services." [Online]. Available: https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/conversation-transcription

[20] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker diarization with lstm," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5239–5243.

[21] B. Works, "What is section 508? who needs to be 508 compliant?" Dec 2021. [Online]. Available: https://brailleworks.com/508-compliance-needs-compliant/

A. Appendix: Results

A.1. Adaptive Classification

speaker 0: some apprehension keeps you marveling but the song delicate hace giveth light which has the power to unCloud your intellect
speaker 2: forgotten to the name of Jillian the lovely captive
speaker 1: a flood light was on in the room inside and Latimer was going around looking at things while a space horse officer stood by the door
speaker 0: I do not think they're shown so great a light under the lids of Venus when transfixed by her own son Beyond his usual custom Martha
speaker 1: remember the closed-door on the first survey they hadn't attempted opening it
speaker 2: I dream of night for an Apple Gold lady for the fruit is now on the Applebaum and the Moon is up and the lawn is Shady singer singer wandering singer oh my honey sweet singer 3 Maiden
speaker 0: the right wheel in a circle came on were dancing one so very read that in the fire she hardly had been noted

A.2. Resemblyzer

speaker 0: some apprehension keeps you marveling but the song delicate hace giveth light which has the power to unCloud your intellect
speaker 2: forgotten to the name of Jillian the lovely captive
speaker 1: a flood light was on in the room inside and Latimer was going around looking at things while a space horse officer stood by the door
speaker 0: I do not think they're shown so great a light under the lids of Venus when transfixed by her own son Beyond his usual custom
speaker 1: Arthur remember the closed-door on the first survey they hadn't attempted opening it
speaker 2: play it's a raining singer I dream of night for an Apple Gold lady for the fruit is now on

the Applebaum and the Moon is up and the lawn is Shady singer singer wandering singer oh my honey sweet singer 3 made

speaker 0: at the right wheel in a circle came on were dancing one so very read that in the fire she hardly had been noted

A.3. Manual

speaker 0: some apprehension keeps you marveling but the psalm delictacy giveth light which has the power to uncloud your intellect

speaker 2: forgotten too the name of Jillian the lovely captive

speaker 1: a floodlight was on in the room inside and Latimer was going around looking at things while a space force officer stood by the door

speaker 0: I do not think they're shown so great a light under the lids of Venus when transfixed by her own son beyond his usual custom

speaker 1: Martha remembered the closed door on the first survey they hadn't attempted opening it

speaker 2: now you may play a surana singer a dream of night for an Apple Gold lady for the fruit is now on the apple bow and the moon is up and the lawn is shady singer singer wandering singer oh my honey sweet singer

speaker 0: three maidens at the right wheel in a circle came onward dancing one so very red that in the fire she hardly had been noted