Smart Meeting Record System

Xiangyu Fan, Wenjia He, Yangming Ke, Tianyi Xie
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48105
xyfan, wenjiah, ymke, xietiany [at] umich.edu

Abstract

We develop a meeting record system which can automatically identify the speakers and create the transcript for the meeting. We use deep clustering to do speaker identification combining with open source speech recognition API. Specifically, we use deep embedding and standard clustering algorithms to solve the speaker identification problem. This is more efficient than normal approach to this kind of problem. We are currently working on embedding part of the problem. The evaluation is based on signal-to-distortion ratio and sparse non-negative matrix factorization is used as the baseline. We would expect the model be successful not only in dialogue data but also in the data of mixture of three or more speakers.

1 Introduction

Meetings are important modern activities in communication and collaboration. It is a great way to collect information, exchange opinions, resolve conflicts, and make decisions. By itself, meetings contain huge amount of rich information and the maximum level of capturing information in meeting becomes vitally important to improve work efficiency. The most common way to capture information during meeting is simply by taking notes. However, manual way of note taking is inefficient and it is barely possible to record everything said in the meeting by hand. An application that can record the meeting efficiently and accurately is in need. If this application can be developed, work efficiency can be hugely improved. People can recall the content in the meeting, review the decision made or revisit the deferred items with ease. Moreover, the recording can help people who do not attend the meeting get informed as well. Therefore, the objective of this project is to develop a model to record the meeting.

Meeting record problem can be divided into two sub-problems, speaker identification and speech recognition. The project aims to identify which person is speaking and create a transcript of the meeting. For the solution to speech recognition problem is widely studied and developed well, the project uses the open-source API of speech recognition directly. The methodology used in the project focuses on speaker clustering, which means to cluster the features from audio stream so that the audio streams from the same person is clustered together. It is not a simple clustering problem, as the objective types may be unknown. Hence, it cannot be solved by conventional clustering methods, such as k-means.

The problem of speaker identification can be approach as a partition-based segmentation problem. For audio data, we use a set of time-frequency coordinates, which define *"elements"*. *"Elements"* can be viewed as feature vectors storing fractional information of the signal. Particularly, for this problem, object class labels are required. The task is to learn directly from partition labels.

There have been tremendous studies on solutions to this segmentation problem. There are two typical approaches. The first approach is to segment the *"elements"* of audio data into a target speaker dominated regions. It can be based on classifiers, generative models, or deep neural networks. Although this approach is proven to performs well with known object classes, it does not address

problem raised in this paper with unknown object types. Another approach is *spectral clustering*. By constructing a similarity graph and forming the graph Laplacian, we use standard clustering methods on the relevant eigen-vectors of the Laplacian. Spectral clustering does not require the points to cluster tightly. However, it is computationally expensive unless the graph is sparse and the affinity matrix can be constructed efficiently. In this paper, we use "deep clustering", which uses deep learning to cluster, to solve the speaker identification problem.

Contribution:

2 Problem Statement

Speaker identification and speech recognition are the two main problems to solve.

2.1 Speaker identification

Speaker identification problem is to identify which speaker each audio segment belongs to and it involves two steps. The first step is embedding and the second step is clustering.

Embedding:

Clustering:

2.2 Speech recognition

3 Related Work

Automatic meeting record, converting audio recordings of meeting to text format, has long been studied [1]. In this problem, speaker identification on closed speaker set or open speaker set and speech recognition are two significant parts. According to [2], speaker identification is a good way of indexing meeting records for further work. Identifying speakers on an open set, which clusters each element defined in terms of time and frequency in audio recordings, is more practical and difficult than that on a closed set, which classifies each element.

Various identification methods have been investigated. [3] shows that some methods combine audio and video streams to identify speakers through face recognition or sound source location. For example, [4] applies a pre-trained probability distribution of source location features. On the other hand, if only audio stream is available, existing methods that use the limit information to realize speaker diarization are based on hidden Markov models [5] or spectral clustering [6]. However, a major disadvantage of spectral clustering is that it needs spectral factorization which causes high cost. Fortunately, deep network shows powerful computational ability with high speed in various applications. For example, with the development of deep learning, meeting recognition has been greatly improved by applying RNN and DNN [7, 8]. A few works have been done to identify speakers only from audio stream by combining traditional central clustering methods and deep networks, such as CNN [9], BLSTM [10] and ResCNN [11] to outperform previous works. Our work in speaker identification component analyzes and compares [10] and [11].

The second part of smart meeting record, speech recognition, has been well explored for a long time and further developed recently to almost achieve the level of humanity [12, 13]. In order to make speech recognition research easier for researchers and professionals, powerful and open-source library such as $SpeechRecognition$ [14] in Python can be utilized. It supports several engines and APIs, and can translate spoken language into texts.

4 Methodology

In our smart meeting record system (Figure 1), there are two major components to convert meeting record audio stream to text format indexed by speakers: speaker identification, which identifies each element of audio signals to each speaker in a meeting, and speech recognition, which converts words and phrases in spoken language to a text format.

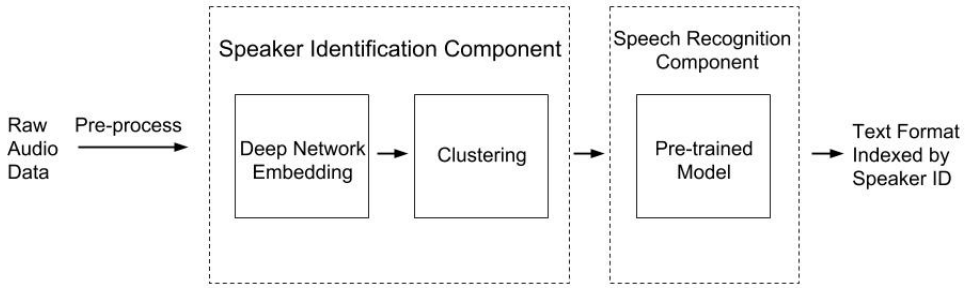

Figure 1: Flow chart of the smart meeting record system

4.1 Speaker Identification Component

In speaker identification component, in order to avoid the high overhead of spectral clustering, we combine traditional and simple clustering methods with deep neural networks to leverage their learning power and high speed. In the training process, given the speaker label of each segment, an embedding learning neural network model is trained offline. Then in the test process, a new piece of audio stream can be put into the model to output embedding for each segment. Without the labels for the new piece, a central cluster method is utilized here in order to cluster segments from the same speaker into one group.

4.1.1 Deep Network Embedding

Numerous embedding methods, which represents an encoder to map elements to a multidimensional embedding space, have been proposed. The object is that when two elements are from the same speaker, the Euclidean distance between the two embeddings is small, while when two elements are from different speakers, the distance is otherwise large. One commonly used method is to extract embeddings from a hidden layer in a deep network for classification. However, the class-based method is not suitable for this embedding task, because 1) there are a large amount of speakers in the world, thus obstructing well-defined class labels, 2) there are not enough and uniform data in each class to be trained in the training procedure, 3) it is not easy to generalize the model to new classes if they are not included in the training data. Therefore, we compare the other two embedding methods that utilize class connection of elements in [10] and [11], and implements the better method in our system.

For deep clustering model in [10], after data preprocessing procedure, the feature matrix X of segments is obtained from raw audio data. Each row of X is the feature vector of each element in the segment. Because the audio stream is sequential data, bidirectional LSTM, one robust variant of RNN, can be utilized in this case. There are four bidirectional LSTM layers, one fully connected layer and one normalization layer in the deep network architecture, and TanH is chosen as the activation function. The output of deep network is $V \in \mathbb{R}^{N\times K}$, each row of which is the learned K-dimensional unit-norm embedding of each element. At training time, parameters θ in the network are updated according to the derivation of loss function:

$$\begin{aligned} C(\theta) &= |VV^T - YY^T|_W^2 \\ &= \sum_{i,j:y_i=y_j} \frac{|v_i - v_j|^2}{d_i} + \sum_{i,j:y_i\neq y_j} \frac{(|v_i - v_j|^2 - 2)^2}{4\sqrt{d_i d_j}} - N \end{aligned} \tag{1}$$

where the matrix $Y \in \mathbb{R}^{N\times C}$ is an indicator whether an element belongs to a class (speaker), to be specific, the entry $y_{n,c} = 1$ when n-th element is in class c, otherwise $y_{n,c} = 0$, and v_i and y_i are the i-th row of matrix V and Y. And in the formula, $|\cdot|_W^2$ is defined as a weighted Frobenius norm, in which $W = d^{-\frac{1}{2}}(d^{-\frac{1}{2}})^T$ and $d = YY^T1$. In order to decrease the loss function, the distance between v_i and v_j will converge to 0 if i-th and j-th elements are in the same class, or will converge to $\sqrt{2}$ otherwise, which meets our goal above.

For deep speaker model in [11], the deep network architecture and the loss function are different from [10]. Because of CNNs' effect in reducing spectral variations and modeling spectral correlations and being inspired by the superior performance of ResNets [15], Residual CNN composed of four residual CNN blocks, one fully connected layer and one normalization layer is designed as the deep architecture. And the clipped rectified linear (ReLU) function is used as the activation function for all of the layers. In addition, the triplet loss that takes in three samples is applied here. For every triplet i, the feature vectors of an anchor segment x_i^a, a positive segment x_i^p from the same speaker and a negative segment x_i^n from a different speaker are put into the deep architecture, and accordingly the embeddings v_i^a, v_i^p and v_i^n are obtained. We intend to make updates to increase the similarity of the anchor and the positive segment and decrease the similarity of the anchor and the negative segment. The loss function for N triplets is:

$$\begin{aligned} C(\theta) &= \sum_{i=0}^{N} \max(s_i^{an} - s_i^{ap} + \alpha, 0) \\ &= \sum_{i=0}^{N} \max(\cos(v_i^a, v_i^n) - \cos(v_i^a, v_i^p) + \alpha, 0) \end{aligned} \tag{2}$$

where s_i^{an} is the cosine similarity between the anchor and the negative segment, s_i^{ap} is the cosine similarity between the anchor and the positive segment and α is a defined minimum margin. To even make the model better, softmax pre-training is applied to initialize the weights in the model, because the cross entropy loss can converge more stably than the triplet loss.

4.1.2 Clustering

Since audio elements from the same speaker are close to each other in the embedding space, traditional clustering methods are leveraged to cluster elements into groups, which represents a decoder to assign elements to speakers. There are various kinds of clustering algorithms such as centroid models and connectivity models. In these models, the typical representative algorithms are K-Means, which partition elements into clusters with minimum mean of distances. Given an audio meeting record, the number of participants in the meeting room may be not known in advance. In other words, it's ambiguous to choose the correct k for K-Means. In order to address the challenging task, two practical solution can be utilized here. W_k is defined as the summation of L_2 distance between every element and the center point of its cluster:

$$W_k = \sum_{l=1}^{k} \sum_{i:C(i)=l} ||x_i - c_l||^2 \tag{3}$$

If the graph of W_k against the number of clusters is plotted, the elbow point, which means when x-coordinate becomes larger, the change of y-coordinate becomes much smaller, can be chosen. Therefore, an appropriate number of clusters is determined.

4.2 Speech Recognition Component

In addition to cluster people's speech, we will also recognize people's speech. we will call existing API to complete this part of task, such as the pre-trained models in PyKaldi, and combine with result of clustering system and recognition system to record the content of meeting. We may design our own model to recognize speech if we perfectly complete the clustering job.

5 Experiment Setup

5.1 Dataset and Pre-processing

We run speaker identification experiments for both methods on an English Speech dataset called LibriSpeech. The details of the dataset are given as follows:

- It is prepared by Vassil Panayotov assisted by Daniel Povey. It is collected from audiobooks from the LibriVox project.

- Table 1 lists information about the number of speakers and hours of speech for training and testing dataset.
- Training-clean-360 consists of 921 speakers, among whom 439 are female and 492 are male, and 363.6 hours of speech. On average, each speaker speaks 25 minutes.
- The evaluation partition, testing-clean comprises 40 speakers with half female and half female. It comprises 40-hour speech. Each speaker talks 8 minutes on average.

	spkr	female	male	hrs	min/spkr
training-clean-360	921	439	492	363.6	25
testing-clean	40	20	20	5.4	8

Table 1: LibriSpeech Dataset

To construct the final training data for the model, we randomly join two speakers' speeches to form dialogue. Each file in the training set is a wav file consisting of two speakers each saying one sentence one after the other. The dialogue is randomly formed and we did not pay attention to genders or length of each speaker's talking.

When constructing the testing data, we pay special attention to gender-balance. One third of the testing files consisting only male speakers and another one third comprising only female speakers. The rest of one third is mixture of both female and male speakers.

For both training and testing data, we partition the data files into small segments and use each segment as the input X for the model.

5.2 Baseline

To determine whether we should use embedding and which embedding methods are better. We use K-means without any other layers as baseline methods. There's no training process for the baseline method. Raw input file was split into segments and we directly conduct K-means on these segments.

5.3 Training Methodology

For deep speaker model, the final input data were constructed by randomly picking one anchor positive sample and 99 negative sample globally for each anchor, called "pair". Audio is converted then to 64-dimensional Fbank coefficients, normalized to have zero mean and unit variance. As stated in section 3, deep speaker model is trained in two stages: softmax pre-training and triplet loss fine-tuning. In both stages, SGD was used with 0.99 momentum, with learning rate ranging from 0.05 to 0.005. The model is pre-trained for 10 epochs using minibatch of size 64 and fine-tuned with triplet loss for 15 epochs with minibatch of size 128. In each epoch, we re-shuffle the training pairs.

For deep clustering model, the final input data were the segments themselves. Deep clustering model is trained with BLSTM with loss function and architecture stated in Section 4. In the training stage, SGD was used with 1e-4 learning rate. The model is trained for 15 epochs using minibatch of size 128. In each epoch, we re-shuffle the training data.

6 Experiment Results

6.1 Embedding Comparison

First, we compare whether it is good to have embedding. Figure 1 shows the t-distributed stochastic neighbor embedding for no embedding, embedding with deep clustering model and embedding with deep speakers model. The input data for no embedding TSNE is the original feature vectors. Embedding with the model is the embedding results on the original feature vectors. From the figure we can see in general, embedding is better than no embedding and the deep speaker model performs better than deep clustering model.

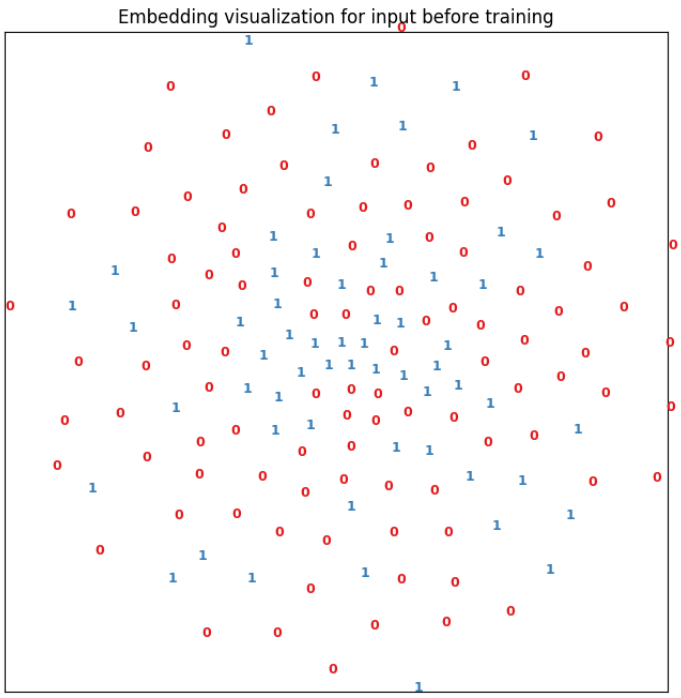

Figure 2: Visualization of input features before training

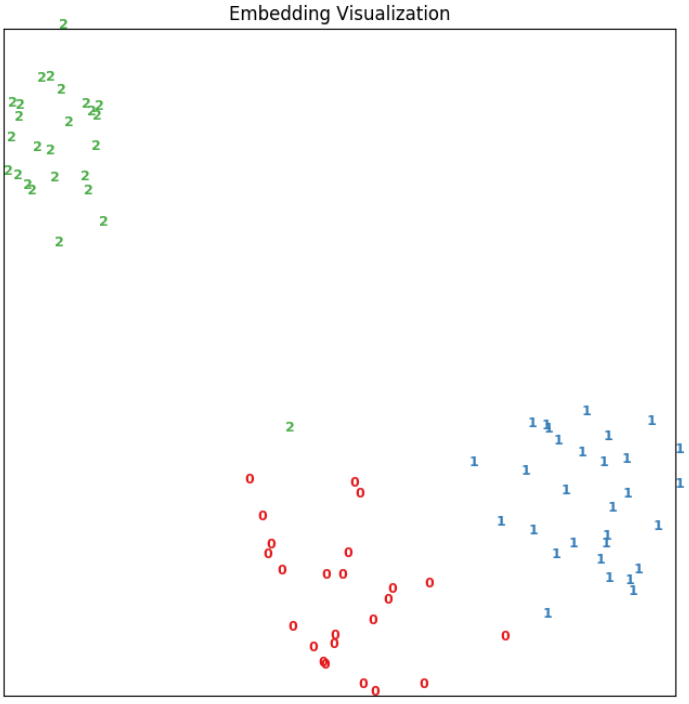

Figure 3: Embedding Visualization for three speakers data(two males and one female), accuracy = 78/80

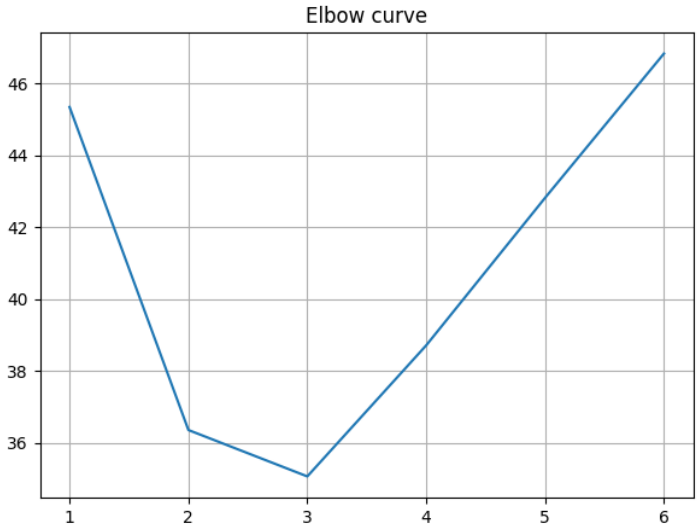

Figure 4: Elbowcurve for three speakers data

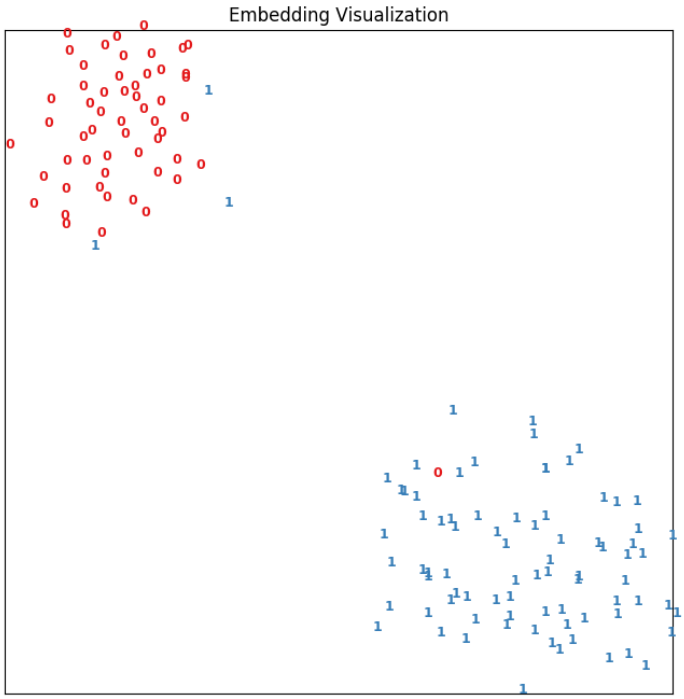

Figure 5: Embedding Visualization for two speakers data(one male and one female), accuracy = 133/137

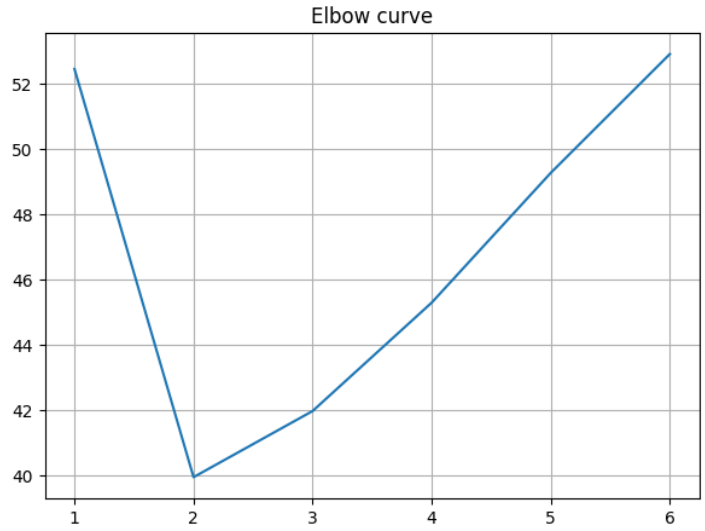

Figure 6: Elbowcurve for two speakers data(one male and one female)

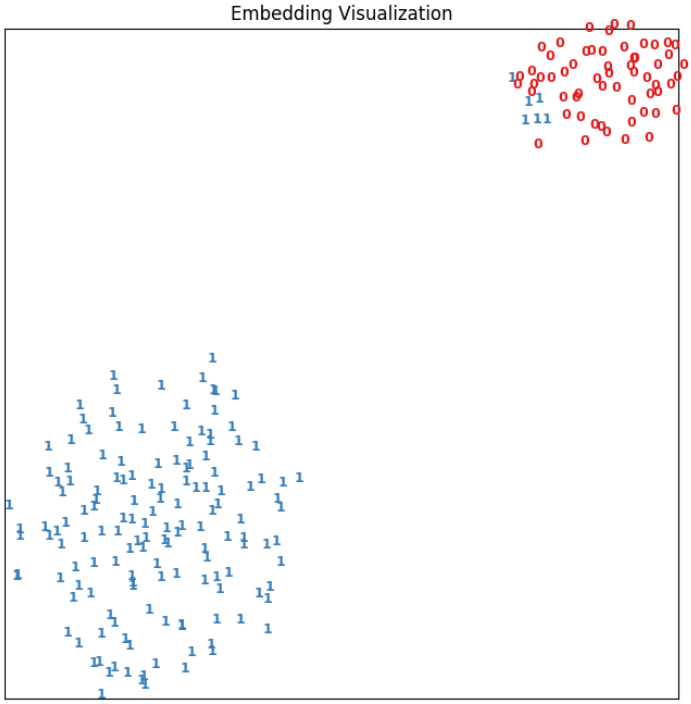

Figure 7: Embedding Visualization for two speakers data(two male), accuracy = 204/212

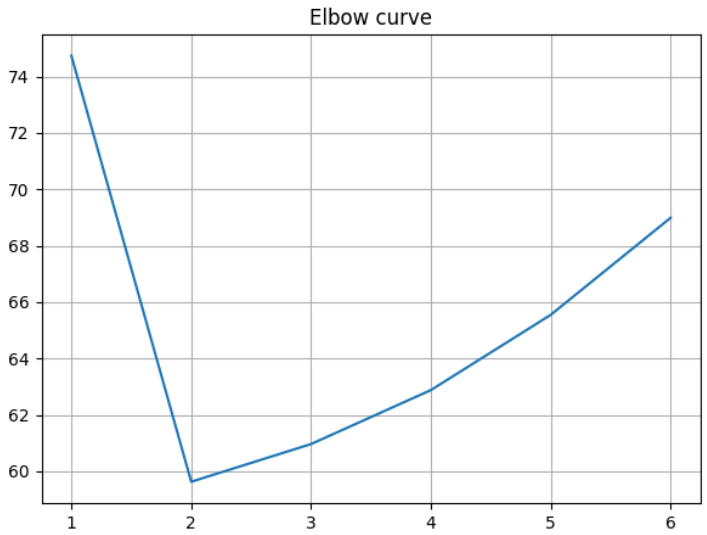

Figure 8: Elbowcurve for two speakers data(two male)

6.2 Gender Distribution

6.3 Determining the Number of Speakers (K)

7 Conclusion

Our project goal is to build a meeting record system which can automatically identify the speakers and record their speeches. So far, we have looked over some relative work and understanding the mathematical principle of methodology we will use in our model. And we will leverage the best architecture and determine the unknown number of speaker to outperform the previous model. We are currently working on mixtures of two people, finishing the dataset construction and data preprocessing. Our current mission is to write embedding code and train with different structures. Our future work includes evaluation and extension of dataset to mixture of 3 or more speakers.

References

[1] Ralph Gross, Michael Bett, Hua Yu, Xiaojin Zhu, Yue Pan, Jie Yang, and Alex Waibel. Towards a multimodal meeting record. In *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No. 00TH8532)*, volume 3, pages 1593–1596. IEEE, 2000.

[2] Werner Geyer, Heather Richter, and Gregory D Abowd. Towards a smarter meeting record—capture and access of meetings revisited. *Multimedia Tools and Applications*, 27(3):393–410, 2005.

[3] Zhiwen Yu and Yuichi Nakamura. Smart meeting systems: A survey of state-of-the-art and open issues. *ACM Computing Surveys (CSUR)*, 42(2):8, 2010.

[4] Shoko Araki, Nobutaka Ito, Marc Delcroix, Atsunori Ogawa, Keisuke Kinoshita, Takuya Higuchi, Takuya Yoshioka, Dung Tran, Shigeki Karita, and Tomohiro Nakatani. Online meeting recognition in noisy environments with time-frequency mask based mvdr beamforming. In *2017 Hands-free Speech Communications and Microphone Arrays (HSCMA)*, pages 16–20. IEEE, 2017.

[5] Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):356–370, 2012.

[6] Francis R Bach and Michael I Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7(Oct):1963–2001, 2006.

[7] Stefan Kombrink, Tomáš Mikolov, Martin Karafiát, and Lukáš Burget. Recurrent neural network based language modeling in meeting recognition. In *Twelfth annual conference of the international speech communication association*, 2011.

[8] Pengyuan Zhang, Yulan Liu, and Thomas Hain. Semi-supervised dnn training in meeting recognition. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 141–146. IEEE, 2014.

[9] Yanick Lukic, Carlo Vogt, Oliver Dürr, and Thilo Stadelmann. Speaker identification and clustering using convolutional neural networks. In *2016 IEEE 26th international workshop on machine learning for signal processing (MLSP)*, pages 1–6. IEEE, 2016.

[10] John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35. IEEE, 2016.

[11] Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*, 2017.

[12] Kyu J Han, Akshay Chandrashekaran, Jungsuk Kim, and Ian Lane. The capio 2017 conversational speech recognition system. *arXiv preprint arXiv:1801.00059*, 2017.

[13] Neil Zeghidour, Qiantong Xu, Vitaliy Liptchinsky, Nicolas Usunier, Gabriel Synnaeve, and Ronan Collobert. Fully convolutional speech recognition. *arXiv preprint arXiv:1812.06864*, 2018.

[14] A. Zhang. Speech recognition (version 3.8), 2017.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.