Joint Speaker Diarization and Recognition Using Convolutional and Recurrent Neural Networks

Conference Paper · April 2018

DOI: 10.1109/ICASSP.2018.8461666

3 authors:

Zhihan Zhou
Northwestern University
7 PUBLICATIONS **321** CITATIONS
SEE PROFILE

Yichi Zhang
University of Rochester
12 PUBLICATIONS **149** CITATIONS
SEE PROFILE

Zhiyao Duan
University of Rochester
112 PUBLICATIONS **2,597** CITATIONS
SEE PROFILE

JOINT SPEAKER DIARIZATION AND RECOGNITION USING CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS

Zhihan Zhou, Yichi Zhang, Student Member, IEEE, and Zhiyao Duan, Member, IEEE

Dept. of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627, USA

ABSTRACT

Speaker diarization and speaker recognition are important tasks in speech processing field. But speaker diarization cannot determine the speaker's absolute identity. Also speaker recognition is often refrained by one speaker recognition scenario, which lacks timing boundaries for multi speaker settings. Hence, in this paper, to overcome these limitations, we introduced an approach using convolutional neural network(CNN) and recurrent neural network(RNN) to simultaneously implement speaker diarization and recognition, in which we employ a CNN(CNN1) to do a segment-level classification and another CNN2 to detect corresponding speaker change probability between adjacent segments, and feed these results into a RNN to effectively integrate them together. Experiments on different datasets show that our approach gives promising and robust performance for practical applications.

Index Terms— Speaker diarization, speaker recognition, convolutional neural network, recurrent neural network, speak change detection

1. INTRODUCTION

Speaker recognition aims to recognize the identity of a speaker from his/her utterances, yet the time boundaries of such utterances do not need to be detected. Speaker diarization, on the other hand, aims to detect "who spoke when" during a conversation, yet speaker identities can be relative within the conversation (e.g., Speaker No. 1 vs. John Smith). In many scenarios, however, speaker recognition and speaker diarization are both needed. Take the call center as an example, it may want to recognize a caller's identity based on the paralinguistic parameters (e.g., emotion) of the caller's speech so that it can quickly direct the caller to a specialized agent to improve the caller's satisfaction. In this case, the call center would need a system that is able to diarize the conversation between the caller and the initial agent and recognize the caller's identity against a pretrained model. Therefore, jointly diarizing a conversation and recognizing the identity of conversational partners is an interesting and useful problem to investigate.

One naive way to achieve joint speaker diarization and recognition in a conversation is to segment the conversation into short segments and recognize the speaker(s) (if any) in each segment independently. This, however, does not fully exploit the many useful properties of the problem, which allow the two tasks to benefit each other. On one hand, speaker diarization ideas helps speaker recognition. First, within a conversation, the identity of an active speaker is likely to be stationary within a short period of time; this is a property that speaker diarization techniques often exploit (e.g., speaker change detection [1]), and can help smooth speaker recognition results. Second, within a conversation, there are usually only a few speakers, i.e., the identity of an active speaker at a moment can only come from a small set of people of a large identity database; this can help reduce the search space of speaker recognition significantly. On the other hand, speaker recognition techniques explicitly or implicitly learn speaker models from many recordings of many different speakers. This cross-speaker, cross context learning helps the speaker models to capture highly discriminative features of speech. When they are applied to speaker diarization, the clustering of the same speaker within a conversation can also be benefited.

In this paper, we develop a method to jointly diarize and recognize speakers from a collection of conversations. It not only estimates the timing boundaries of the utterances of each speaker, but also recognizes the absolute identity of a set of speakers of interest, provided that training speech of these speakers are available. Our method exploits the the unique properties of the problem and allows the two tasks to benefit each other.

Specifically, we first use one Convolutional Neural Network (CNN), which is first introduced in [2] and obtains great achievement in image classification and audio recognition domains [3, 4, 5]. A CNN1 is to classify the absolute speaker identity on equally spaced segments of each conversations. We incorporate a sparsity term in the loss function to account for the fact that only a few speakers are present in each conversation. We also use another CNN, CNN2, to perform Speaker Change Detection (SCD) on each conversation to model the temporal continuity of speaker identities, where we design a loss function to bias towards false alarms. Finally we combine the output of both CNNs and feed it into a Recurrent Neural Network (RNN) for joint speaker recognition and diarization. Through the RNN, the temporal con-

tinuity information captured by CNN2 can be utilized by the speaker recognition task, while the discriminative speech features captured by CNN1 can be utilized by the speaker diarization task.

2. RELATED WORKS

Many recent advances adopt i-vector extraction [6, 7, 8] for speaker diarization followed by a probabilistic linear discriminant analysis (PLDA) based scoring function [9] to cluster speakers. However, due to the clustering performance relying on the size of segments, such systems could not work well for short segment processing. Also, feature embedding was proposed to embed the speech utterance into a pre-defined anchor space [10]. Deep neural networks can also be used to create speaker embeddings [11]. However, most speaker diarization systems work for relative label identification. So in this paper, we propose to estimate the time boundaries of the utterances of each speaker, but also identify the real speaker's identity, which is basically a classification task. We further combine our predicted result with Speaker Change Detection(SCD), which is a task to determine the specific time of speaker change. An common way [8] for this task is calculating the distance between two sliding windows' contents, using Kullback-Leibler divergence [12] and Generalized Likelihood Ratio as distance metrics. Deep Neural Network(DNN) was also exploited in such task [13], they utilized pre-computed feature, which contains information about each segment, as the input of DNN. Using CNN to detect speaker change has been introduced by [14], in which divided a conversation file into continue windows with overlap and did a regression task to predict the speaker change probability by assigning each window a label between 0 and 1. In this paper, we further developed the model to exploit it into our work.

3. PROPOSED APPROACH

Our model achieves joint speaker recognition and diarization. The overall structure is shown in Figure 1. It has two CNN structures for segment-level speaker identity classification and Speaker Change Detection (SCD), respectively. Then it is followed by an RNN to integrate the information of classification and SCD together, to generate a more robust speaker identity prediction for each segment. The detailed description about each module is given in the following subsections.

3.1. CNN1 for classification

The first convolutional neural network (CNN1) acts as a segment-level classifier to predict each segment into a certain speaker identity label. It receives spectral segments from the original recording track spectrogram as input. Each spectral segment is computed over a 0.2 second window and there is no overlap between two adjacent windows. As showed

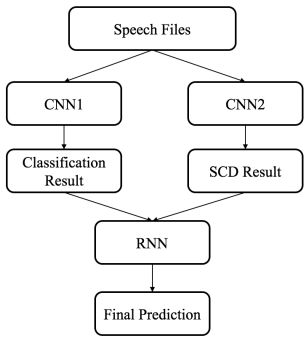

Fig. 1. The overall structure of our proposed method

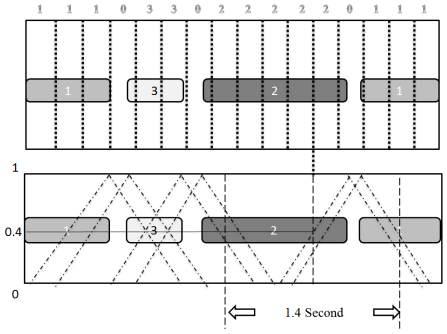

Fig. 2. Recording track segmentation and data preparation. It shows how we separately use different information of the same conversation segment to train CNN1 and CNN2.

in Figure 2, every spectrogram corresponds to a label from 0 to n (positive integer), where 0 means silence and 1 to n represent the n possible speakers respectively.

The CNN1 structure is specified in Figure 3. It consists of 4 convolutional layers and every two convolutional layers are followed by a max pooling layer. For each convolutional layer, zero padding and batch normalization [15] are adopted with the ReLU activation. Every fully connected layer was added a dropout as 0.5 to avoid over fitting [16]. For the output layer, softmax activation is used to generate n+1 dimensional probabilistic output.

CNN1 output could have various combinations of speaker identity prediction. However, considering there are only limited amount of speakers in a certain recording track (in most cases 2 speakers), CNN1 should be expected to have sparse output. Thus, we define a new loss function for CNN1 with sparsity constraint as follows:

$$loss = y_{true} \times log(y_{pred}) + \sqrt{y_{pred}}, \quad (1)$$

where the first term is cross-entropy [17] and the second term is a L-0.5 norm regularizer to make the output layer sparse. Stochastic Gradient Descent (SGD) is used as the optimizer.

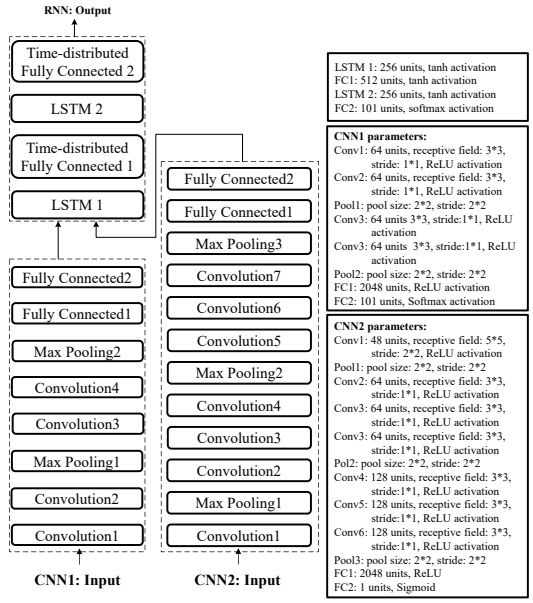

Fig. 3. Our proposed model: CNN1 + CNN2 + RNN

3.2. CNN2 for speaker change detection

CNN2 for Speaker Change Detection (SCD) performs the regression task. Its main purpose is to estimate the probability of speaker change at every time point between two adjacent time windows. Following [14] with slight modifications, the input of CNN2 are spectrogram segments computed over 1.4 second windows with a 0.2 second hop size and covering the entire frequency range. This is because 1.4 seconds covers long temporal evolution that helps to detect change points, also the 0.2-second hop size guarantees that CNN1 and CNN2 have synchronized time steps.

For the annotation, we utilized the fuzzy method in [14]. We take advantage of non-integer labels between 0 and 1 representing speaker change probabilities. We define speaker change time as the time when one speaker begin or stop speaking. As shown in Fig.2, for each speaker change time point, there is a tolerance of 0.6 seconds. For every sample with 1.4 second long, we define the specific time of the middle of the windows as t_{mid} and the nearest speaker change time as t_{SCD}, so the label of each sample can be written as:

$$label = max\left\{0, \frac{5}{3} \times (0.6 - |(t_{mid} - t_{SCD})|)\right\}. \quad (2)$$

The architecture of CNN2 is showed in Fig.3. Each convolution and fully connected layer is followed by Batch Normalization (BN) and ReLU activation. The output layer only has one unit with sigmoid activation to generate an output value between 0 and 1. We defined a new loss function as:

$$loss = (0.1 + y_{ture}) \times (y_{ture} - y_{pred})^2 \quad (3)$$

which gives the points with higher value a larger weights. It hence ensures CNN2 can find more speaker change point but may cause more false alarm. SGD is used as the optimizer.

3.3. RNN for combining results together

The RNN model is shown in Fig.3. We employ two LSTM layers [18] with 128 units and tanh activation, each followed by a time-distributed fully connected layer. Softmax activation is adopted for the output layer. Categorical cross-entropy serves as the loss function and we used RMSprop as the optimizer. After obtaining the predicted classification and SCD result, we feed them into RNN to fuse the results, which can further be classified into 1 out of n speaker identities.

4. EXPERIMENTAL RESULTS

4.1. Datasets

We first adopt the CallHome English conversation dataset, where 50 conversation recording tracks are used[19]. Each recording has two distinct speakers so there are 100 distinguished speakers in total. Every speaker get a label from 1-100 and label 0 represents the silence. We divide every recording into 3 parts containing 30%, 20%, and 50%. The first 30% of all the recordings are combined to train both CNN1 and CNN2. After the two models are trained, we could obtain the predicted classification results of the rest 70% part given by CNN1 and predicted SCD result of the last 50% part given by CNN2. Further, as mentioned in 3.3, the predicted classification result of middle 20% part combined with corresponding ground truth of SCD result were treated as the training data for RNN and the predicted classification result of the last 50% part combined with corresponding predicted SCD result given by CNN2 were considered as the test samples. Actually, we used first half of all the conversation recordings to train the neural networks and the second half of them were exploited as the test sample to examine the result of our approach.

We also use another dataset contains the two-party telephone conversation between prisoner and the other speaker. Each prisoner speaks with multiple different speakers in different recording tracks. In total there are 10 prisoners and each prisoner talks to 10 other different speakers in 10 different recording tracks. 10 prisoner are assigned labels from 1-10 respectively and all the other speakers are treated as one background class with label 11. The first 3 conversation file of the prisoner are used for training CNN1, then the following 2 conversation files to train RNN. We use PyAudioAnalysis [20] to detect speaker activities in order to generate annotations. Based on our own listening, we realized such method misses many speaker change point. Considering the speaker change detection is highly depending on the specific moment of speaker change, we used the same CNN2 model trained on call-home dataset to estimate the speaker change probability.

Table 1. Comparisons of the predicted mean accuracies

Method	Accuracy	std
Cross-entropy	0.711	0.0191
New loss	0.741	0.0092
With-zeros	0.743	0.0077
With predicted SCD	0.829	0.0042
With true SCD	0.867	0.0027
Baseline	0.847	0.0067

Table 2. The precision and recall for 10 prisoners

ID	Pre.	Rec.	ID	Pre.	Rec.
1	0.621	0.948	6	0.621	0.847
2	0.702	0.973	7	0.673	0.822
3	0.772	0.578	8	0.748	0.840
4	0.805	0.837	9	0.716	0.517
5	0.804	0.863	10	0.716	0.873

4.2. Baseline

We develop an effective baseline as the following. First, we obtain the classification result by exploited CNN1 with our own-defined loss function. Then, with the known information about which test samples belong to which conversation file, for the CallHome dataset with 100 speakers, we manually transfered the 101-dimensional result into 3 dimensions. Specifically, for every sample, we could directly find the only three possible classes it might belong to. If the predicted result of it is not within these classes, we will change the result to be one of the three classes with the highest predicted value. Intuitively speaking, we give the misclassified samples a chance to choose a more reasonable classification result. Such method, obviously, could significantly improve the classification accuracy since it exploit two many special information about the samples, but not practical.

4.3. CallHome dataset result

We test the model on the CallHome dataset for 10 times and report the mean accuracy with standard deviation in Table 1. We compare the predicted results of using (1) cross-entropy only as loss function, (2) our proposed loss function with sparsity constraint, (3) CNN1 + all zeros SCD, (4) CNN1 + predicted SCD, (5) CNN1 + ground truth SCD, and (6) the baseline.

Several conclusions can be made as follows. First, the newly proposed loss function with sparsity constraint can not only improve the prediction but also makes the results more stable (consider the high std value). As we notice the model occasionally gets trapped to a local minimum when using cross-entropy as loss. Second, integrating classification result with SCD result significantly improve the classification performance, compared with adopting sparsity loss function only, we achieved relatively 11.9% improvement of accuracy, which indicates that taking advantage of time continuity information by combining SCD result does make sense. Third, SCD information is important to combine with CNN1 classification results. By integrating CNN1 prediction with artificial all-zeros, CNN2 predicted SCD, and ground truth SCD, we observe the trend of increasing RNN classification accuracy. Fourth, the result of our proposed method (With predicted SCD) is quite close to the baseline, a method which is only useful when there is specific information to significantly decrease the possible range of every test sample. On the contrary, our proposed method is much more practical since it does not need any addition information but can achieve almost the same accuracy.

4.4. Prison dataset result

We only interest in when the prisoner is talking but not if the other people were correctly classified or not. Thus for such retrieval problem, we use precision and recall instead of predicted accuracy to evaluate our approach performance on the prison dataset. The experimental results for each prisoner are shown in Table 2. Both precision and recall differs greatly between different prisoners. After listening to the recordings, we believe such difference mainly comes from annotation inaccuracy as well as recording quality.

We use the same method as CallHome dataset except that CNN2 model is trained on the CallHome data for SCD, as speaker change detection model learns the speaking behaviors, patterns, styles, etc. from different speakers, so we assume such model could generalize well to the new prison dataset. RNN works for a 12 classes classification (prisoner + 10 other speakers + silence), so the input of RNN should be a 13-dimensional vector including the SCD result. We reduce the number of neurons in every layer to be 1/8 of the given structure in Figure 3 to avoid overfitting.

5. CONCLUSION AND FUTURE WORK

In this paper, we developed an approach with two CNNs and one RNN to realize joint speaker diarization and recognition. We first used CNN1 by incorporating a sparsity term in the loss function to classify the absolute speaker identity, and used another CNN2 to perform speaker change detection to model the temporal continuity of speaker identities. Then outputs from both CNNs are fed into an RNN for joint speaker recognition and diarization. Experiments show that our approach achieves satisfying speaker recognition and diarization results, and it is more practical than the baseline. It also shows that SCD plays an important role in final RNN classification results. For future work, we would like to explore better methods for sparsity constraint in the CNN1 output layer. Also, we would like to realize semi-supervised systems to alleviate the need of annotations.

6. REFERENCES

[1] Zhenhao Ge, Ananth N. Iyer, Srinath Cheluvaraja, and Aravind Ganapathiraju, "Speaker change detection using features through a neural network speaker classifier," .

[2] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[3] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013, pp. 6645–6649.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] Yoon Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[6] Mohammed Senoussaoui, Patrick Kenny, Themos Stafylakis, and Pierre Dumouchel, "A study of the cosine distance-based mean shift for telephone speech diarization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 1, pp. 217–227, 2014.

[7] Yan Xu, Ian McLoughlin, Yan Song, and Kui Wu, "Improved i-vector representation for speaker diarization," *Circuits, Systems, and Signal Processing*, vol. 35, no. 9, pp. 3393–3404, 2016.

[8] Mickael Rouvier, Grégor Dupuy, Paul Gay, Elie Khoury, Teva Merlin, and Sylvain Meignier, "An open-source state-of-the-art toolbox for broadcast news diarization," Tech. Rep., Idiap, 2013.

[9] Gregory Sell and Daniel Garcia-Romero, "Speaker diarization with plda i-vector scoring and unsupervised calibration," in *Proc. the IEEE Spoken Language Technology Workshop*, 2014.

[10] Mickael Rouvier, Pierre-Michel Bousquet, and Benoit Favre, "Speaker diarization through speaker embeddings," in *Proc. 2015 23rd IEEE European Signal Processing Conference (EUSIPCO)*, 2015, pp. 2082–2086.

[11] Pawel Cyrta, Tomasz Trzciski, and Wojciech Stokowiec, "Speaker diarization using deep recurrent convolutional neural networks for speaker embeddings," in *Proc. International Conference on Information Systems Architecture and Technology*, 2017, pp. 107–117.

[12] James M Joyce, "Kullback-leibler divergence," in *International Encyclopedia of Statistical Science*, pp. 720–722. Springer, 2011.

[13] Vishwa Gupta, "Speaker change point detection using deep neural nets," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4420–4424.

[14] Marek Hrúz and Zbyněk Zajíc, "Convolutional neural network for speaker change detection in telephone speaker diarization system," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4945–4949.

[15] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[16] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[17] Lih-Yuan Deng, "The cross-entropy method: a unified approach to combinatorial optimization, monte-carlo simulation, and machine learning," 2006.

[18] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] A Canavan, D Graff, and G Zipperlen, "Callhome american english speech ldc97s42," *LDC Catalog. Philadelphia: Linguistic Data Consortium*, 1997.

[20] Theodoros Giannakopoulos, "pyaudioanalysis: An open-source python library for audio signal analysis," *PloS one*, vol. 10, no. 12, 2015.