Robust End-to-end Speaker Diarization with Generic Neural Clustering

Chenyu Yang¹, Yu Wang¹,²*¹
¹Cooperative Medianet Innovation Center, Shanghai Jiao Tong University ²Shanghai AI Laboratory
{yangcy, yuwangsjtu}@sjtu.edu.cn

Abstract

End-to-end speaker diarization approaches have shown exceptional performance over the traditional modular approaches. To further improve the performance of the end-to-end speaker diarization for real speech recordings, recently works have been proposed which integrate unsupervised clustering algorithms with the end-to-end neural diarization models. However, these methods have a number of drawbacks: 1) The unsupervised clustering algorithms cannot leverage the supervision from the available datasets; 2) The K-means-based unsupervised algorithms that are explored often suffer from the constraint violation problem; 3) There is an unavoidable mismatch between the supervised training and the unsupervised inference. In this paper, a robust generic neural clustering approach is proposed that can be integrated with any chunk-level predictor to accomplish a fully supervised end-to-end speaker diarization model. Also, by leveraging the sequence modelling ability of a recurrent neural network, the proposed neural clustering approach can dynamically estimate the number of speakers during inference. Experimental show that when integrating an attractor-based chunk-level predictor, the proposed neural clustering approach can yield better Diarization Error Rate (DER) than the constrained K-means-based clustering approaches under the mismatched conditions.

Index Terms: Speaker diarization, end-to-end neural diarization, neural clustering

1. Introduction

Speaker diarization, which is often referred to as “who speaks when” problem, is an important speech processing task. The goal of a speaker diarization system is to estimate the temporal boundary of each talking speakers in real audio recordings [1, 2]. Traditional speaker diarization approaches first segment the audio streams into speech and non-speech regions, then the embeddings representing speaker characteristics are extracted from each speech frame. Popular used speaker embeddings include i-vectors [3], d-vectors [4, 5], and x-vectors [6]. After the speaker embeddings are extracted, the embeddings belonging to the same speaker are clustered using unsupervised clustering methods to yield the diarization results, where agglomerative hierarchical clustering (AHC), K-means clustering, and spectral clustering are commonly used methods. Various speaker embeddings and clustering techniques have been explored for speaker diarization tasks in [7, 8]. Although the traditional approaches can give good performance on various datasets, the main disadvantages of them are twofolds: first, in these approaches multiple components need to be optimixed using separate criteria and second, it is difficult for them to deal with overlapped speech as the unsupervised clustering normally assumes that each segment can only be assigned to one speaker. Although recently some approaches are proposed leveraging the supervised neural clustering models to improve the diarization performance, such as the unbounded interleaved-state recurrent neural network (UIS-RNN) [9] and Discriminative Neural Clustering (DNC) [10] approaches, they still suffer from the aforementioned two problems.

To solve those problems, end-to-end joint optimization-based models have been proposed which form the speaker diarization problem as a multi-label classification problem. The goal of these end-to-end frameworks is to achieve the diarization by a single neural network that is trained to directly optimize the diarization performance. The self-attentive EEND approach proposed in [11] was optimized to calculate diarization results for every speaker in a mixture from input audio features using permutation invariant training (PIT) [12]. EEND can only handle scenarios where numbers of speakers are fixed. It was later extended to deal with variable number of speakers by the end-to-end diarization model with encoder-decoder based attractor (EEND-EDA) [13] and speaker-wise conditional EEND (SC-EEND) [14]. Although previous experiments have shown that these end-to-end models can have a better or similar performance as the traditional methods, they fail to handle some complicated situations, for example, long-form speech. In order to address these issues, some approaches have been proposed to combine the advantages of both end-to-end models and unsupervised clustering algorithms as a two-stage framework [15, 16]. These two-stage approaches normally divide the complete audio sequence into chunks. Each chunk is processed by a neural-network-based model separately. An unsupervised clustering algorithm is then applied to determine the speaker correspondence among chunks. EEND-vector [15, 16] extends the EEND so that it can both diarize the subsequence and extract corresponding chunk-level speaker embeddings. These embeddings are clustered using the COP-K-means algorithm [17]. This approach enables the original model to process long overlapped speech. Thus, it combines the advantages of both clustering and the EEND-based method. The approach proposed in [18] incorporates local attractors with the EEND-EDA model which splits the sequence after feeding it into the encoder and replaces the linear decoder with a local encoder-decoder-based attractor calculation module, in order to deal with the mismatch of the number of speakers between training and evaluation.

However, these methods use a constrained unsupervised clustering algorithm as the post-processing, which have some limitations: 1) Some constrained clustering algorithms like COP-K-means are not robust enough. They can not handle conflicts between constraints. These conflicts may be caused by incorrect estimation of the number of speakers. 2) The unsupervised clustering algorithms are only used in the inference stage. Although extra loss functions have been applied to make

*Corresponding author.

This work was supported by National Natural Science Foundation of China under Grant No. 62106140, Shanghai Science and Technology Committee under Grant No. 21511101100, and Shanghai Key Lab of Digital Media Processing and Transmission (No. 18D2270700)
the training to be consistent with the inference, there is still un-
avoidable mismatch between the supervised training and unsu-
A neural end-to-end diarization model receives a sequence of
with local attractors \([13, 18]\) to deal with
the number of clusters which is required by
to convert each attractor to a hidden state vector with the op-
Although the attractors are still out of order at this stage,
\(\hat{z^n}\) is the corresponding existence probabilities. The chunk-level active probability is computed by
The chunk-level predictor is trained with two losses, namely the attractor existence loss and diarization loss. The at-
the targets for training such a sequential clustering model are formed as:
\[\phi_{n i} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}, \quad n = 1, \ldots, N_s, \quad i, j = 1, \ldots, N\]  
\[a^n = \text{TransformerDec}(\hat{a^n}, E).\]  
2. Proposed Framework: EEND with
2.1. Overview of two-stage frameworks
A neural end-to-end diarization model receives a sequence of
A two-stage diarization framework norm-
it is able to dynami-
Finally, the proposed neural clustering approach is generic
in the sense that it can be integrated with any chunk-level pre-
neural clustering approaches that have been explored in \([9, 10]\).

2.2. Chunk-level diarization and attractor extraction
Similar to \([18]\), the chunk-level predictor is a local encoder-
decoder-based attractor (EDA) calculator aiming to diarize sub-
separate. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2.3. RNN-based clustering methods
On the basis of the chunk-level predictor, we get pairs of
and output of the encoder is then split into \(N\) chunks,
where \(L\) is the size of chunk.
A chunk-level predictor is applied on each chunk separately.
A chunk-level predictor is predicted on each chunk separately. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2. Proposed Framework: EEND with
2.1. Overview of two-stage frameworks
A neural end-to-end diarization model receives a sequence of
\(L = L_{\text{pre}} + L_{\text{post}}.\)
Here \(L_{\text{pre}}\) is the loss function of the predictor, which mainly
focuses on the internal diarization results of blocks. And \(L_{\text{post}}\)
aims to train the clustering module among different blocks.
Such a proposed neural clustering approach can be inte-
genital, in this paper we will focus on its integration with the the
EEND-EDA predictor with local attractors \([13, 18]\) to deal with
the number of speakers in the same way as train-
end-to-end optimization and achieving more efficient use of the
The proposed approach can leverage the advantages of both the Transformer-based end-to-end framework and the neural
clustering approaches that have been explored in \([9, 10]\).

\[a^n = \text{TransformerDec}(\hat{a^n}, E).\]  
2.3. RNN-based clustering methods
On the basis of the chunk-level predictor, we get pairs of
and output of the encoder is then split into \(N\) chunks,
where \(L\) is the size of chunk.
A chunk-level predictor is applied on each chunk separately. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2.2. Chunk-level diarization and attractor extraction
Similar to \([18]\), the chunk-level predictor is a local encoder-
decoder-based attractor (EDA) calculator aiming to diarize sub-
separate. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2.3. RNN-based clustering methods
On the basis of the chunk-level predictor, we get pairs of
and output of the encoder is then split into \(N\) chunks,
where \(L\) is the size of chunk.
A chunk-level predictor is applied on each chunk separately. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2. Proposed Framework: EEND with
2.1. Overview of two-stage frameworks
A neural end-to-end diarization model receives a sequence of
\(L = L_{\text{pre}} + L_{\text{post}}.\)
Here \(L_{\text{pre}}\) is the loss function of the predictor, which mainly
focuses on the internal diarization results of blocks. And \(L_{\text{post}}\)
aims to train the clustering module among different blocks.
Such a proposed neural clustering approach can be inte-
genital, in this paper we will focus on its integration with the the
EEND-EDA predictor with local attractors \([13, 18]\) to deal with
the number of speakers in the same way as train-
end-to-end optimization and achieving more efficient use of the
The proposed approach can leverage the advantages of both the Transformer-based end-to-end framework and the neural
clustering approaches that have been explored in \([9, 10]\).

\[a^n = \text{TransformerDec}(\hat{a^n}, E).\]  
2.3. RNN-based clustering methods
On the basis of the chunk-level predictor, we get pairs of
and output of the encoder is then split into \(N\) chunks,
where \(L\) is the size of chunk.
A chunk-level predictor is applied on each chunk separately. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2.2. Chunk-level diarization and attractor extraction
Similar to \([18]\), the chunk-level predictor is a local encoder-
decoder-based attractor (EDA) calculator aiming to diarize sub-
separate. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.

2.3. RNN-based clustering methods
On the basis of the chunk-level predictor, we get pairs of
and output of the encoder is then split into \(N\) chunks,
where \(L\) is the size of chunk.
A chunk-level predictor is applied on each chunk separately. The details of this predictor will be given in Section 2.2. After
the chunk-level prediction stage, the proposed neural clustering
module then re-concatenates the outputs together, which will be
described in Section 2.3.
Because the optimal permutation has been estimated using Equation (2), permutation-invariant training is not required at this stage. Thus, a cross entropy (CE) loss is used to optimize the RNN clustering model:

\[
L_{post} = \frac{1}{N} \sum_{n=1}^{N} \left( \frac{1}{S_n} \sum_{i=0}^{C-1} \sum_{j=0}^{N} -y^o_{ij} \cdot \log(p^o_{ij}) \right).
\]

After the prediction step, the goal of the update step is to adaptively adjust the hidden states at each step given the estimate of the attractors. Because \( r^o_{i,j} \) provides a mapping from the \( i \)-th hidden state, \( h^o_{ij} \), the hidden states can be updated using a teacher-forcing strategy [21]:

\[
h^o_{nj} = \begin{cases} 
    h^{o-1}_{n} & \forall i \ r^0_{ij} = 0 \\
    \text{GRU}(a^o_{n}, h^{o-1}_{nj}) & \exists i \ r^0_{ij} = 1.
\end{cases}
\]

For the inference, the number of speakers is unknown. To dynamically estimate the attendance of new speakers, an initialized \( h_0 \) is concatenated with the previous states at every step. The optimal speaker permutation \( \phi^o \), is estimated by:

\[
\phi^o = \arg \max_{\phi \in \Phi'} \prod_{n=0}^{N-1} p^{\phi}_{n}.
\]

It should be noticed that a cannot-link exists between every pair of speakers in the same chunk, therefore each hidden state except for the new one can be linked to no more than one attractor. Then the hidden states are updated according to Equation (4) and (5).

After obtaining each \( \phi^o \), the final output of the diarization is reordered by replacing the chunk-level output \( y^o \) with \( y^o_{\phi^o} \).

3. Experiments

3.1. Data

To evaluate the performance of the proposed RNN neural clustering integrated EEND-EDA approach, namely the EDA-RC approach, training and test datasets with different number of speakers were generated. The simulated datasets were generated using the data from the LibriSpeech [22] 360 hours of clean speech. The sampling frequency is 8kHz. We firstly selected \( n \) speakers from this dataset. Then for each speaker, [30/n, 60/n] utterances were randomly selected. This simulating procedure was identical to the one that was introduced in [11]. It is worth noting that the average length of silence intervals between utterances varied in order to maintain the overlap ratio in a reasonable range. The simulated training set includes 100,000 samples with 3 speakers. The details of the test sets for the simulation data, LS_3, LS_4 and LS_5, are given in Table 1.

The CALLHOME dataset, i.e., NIST SRE 2000 (LDC2001S97, Disk-8) [23], was used to further evaluate the performance on the real data. This dataset has been widely used for speaker diarization experiments, which contains 500 sessions of multilingual telephonic speech. Although each session has 2 to 6 speakers, there are 2 dominant speakers in each conversation. For the experiments, it was split into training and test sets using the tools provided in the Kaldi Toolkit diarization recipe [24]. Samples with \( \leq 3 \) speakers in the training set were picked to train the models. The test set was further split with various numbers of speakers in order to evaluate the performance of approaches for both matched and unmatched conditions.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Source</th>
<th>#Spk</th>
<th>#Sample</th>
<th>Overlap(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS_train</td>
<td>LibriSpeech</td>
<td>3</td>
<td>100,000</td>
<td>49.6</td>
</tr>
<tr>
<td>LS_3</td>
<td>LibriSpeech</td>
<td>3</td>
<td>500</td>
<td>42.7</td>
</tr>
<tr>
<td>LS_4</td>
<td>LibriSpeech</td>
<td>4</td>
<td>500</td>
<td>44.6</td>
</tr>
<tr>
<td>LS_5</td>
<td>LibriSpeech</td>
<td>5</td>
<td>500</td>
<td>45.2</td>
</tr>
<tr>
<td>CH_ft</td>
<td>CALLHOME</td>
<td>2-3</td>
<td>216</td>
<td>16.2</td>
</tr>
<tr>
<td>CH_test</td>
<td>CALLHOME</td>
<td>2-6</td>
<td>250</td>
<td>16.7</td>
</tr>
</tbody>
</table>

3.2. Experimental Setup

We use 23-dimensional log-Mel-filterbanks with a 25-ms frame length and 10-ms frame shift as the input feature. Feature vectors were concatenated with their neighboring [−7, 7] frames and the concatenated features were subsampled by a factor of ten. Therefore, the total input dimension was 345 and the interval between frames was 100ms. This configuration was the same as that used in [11]. The training samples were fed into models every 500 frames while the test samples remain un-split.

The baseline approaches include both one-stage and two-stage end-to-end approaches: the EEND approach [11], the EEND-EDA approach [13] and the EEND-EDA with a COP-K-means unsupervised clustering on the local attractors (EDA+COP-K-means) [18]. For the EEND and EEND-EDA approaches, a 4-layer Transformer with 256 hidden units were used as the encoder. Each layer had 4 heads and a dropout rate of 0.1. The end-to-end models were trained for 50 epochs with a batch size of 32. An Adam optimizer [25] with the same learning rate scheduler in [26] was used during training.

For the EDA+COP-K-means and EDA-RC approaches, the
chunk size was set to 50. Parameters of the EDA chunk-level predictor were the same as those in EEND-EDA. The models were trained for another 20 epochs by an Adam optimizer with a learning rate of $10^{-3}$. Diarization Error Rate (DER) was used as the evaluation metric. Errors less than 0.25s around boundaries were tolerated. Please note that for the DER computation, all of the errors were evaluated including the overlapping speech segments. For EDA-RC, a beam search strategy was applied during decoding and the beam size was set to 3. Models of the last 1/10 epochs were averaged to obtain a better performance.

3.3. Experimental Results

Table 2 shows the DERs for the Librispeech-based simulated datasets. Please note that the switching strategy described in [18] is not adopted here, because the main purpose is to compare the unsupervised and supervised clustering methods instead of an overall performance. As expected, it can be seen that EEND and EEND-EDA approaches can yield performance for matched conditions but get worse with the increase of speakers. On the other hand, the EDA+COP-K-means approach takes an obvious advantage over the EEND-EDA approach and it outperforms the EEND-EDA by about 10% DER under mismatched conditions. For our proposed EDA-RC approach, it further gives a DER reduction of 1.66% and 1.56% compared to the baseline approaches under mismatched conditions while the performance does not degrade much for the matched conditions. Additionally, a refine method is applied to the EDA-RC, where an extra decoding process was conducted with the hidden states initialized as the first-pass decoded hidden states. Table 2 shows that this strategy can yield small performance gains when the number of speakers is large. Finally, the performance of the EDA-RC with the oracle permutations (oracle) is also given as a reference. This shows the performance upper bound of the clustering integrated EEND approaches as the correct clustering is used to reorder the speakers of each chunk.

Table 2: Diarization Error Rate (%) on simulated datasets. All models are trained with the 3-speaker training set.

<table>
<thead>
<tr>
<th>Models</th>
<th>two-stage</th>
<th>match</th>
<th>mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>EEND [11]</td>
<td>✓</td>
<td>5.19</td>
<td>-</td>
</tr>
<tr>
<td>EEND-EDA [13]</td>
<td>✓</td>
<td>4.93</td>
<td>33.79</td>
</tr>
<tr>
<td>+ COP-K-means [18]</td>
<td>✓</td>
<td>6.93</td>
<td>22.48</td>
</tr>
<tr>
<td>EDA-RC (ours)</td>
<td>✓</td>
<td>5.17</td>
<td>20.82</td>
</tr>
<tr>
<td>EDA-RC+refine (ours)</td>
<td>✓</td>
<td>5.64</td>
<td>20.92</td>
</tr>
<tr>
<td>EDA-RC (oracle)</td>
<td>✓</td>
<td>4.39</td>
<td>8.70</td>
</tr>
</tbody>
</table>

In order to compare the performance on real audio recordings, models were also finetuned on the CALLHOME real datasets. Table 3 shows the DERs on CALLHOME. The proposed EDA-RC approach yields 25.46% and 35.79% DER for the 4 and 5 speakers’ conditions, which corresponds to about 12.8% and 9.0% relative reduction compared to the EDA+COP-K-means approach. This validates the effectiveness and robustness of the proposed RNN-based clustering model.

To make a more comprehensive comparison, a switching strategy that can dynamically decide whether to decode at chunk level or global level according to the potential number of speakers is also applied [18] in this case. Thus, a global EDA loss is further added during the pretraining and finetuning of the EDA-RC model.

We notice that the gap between the clustering-integrated approaches and the conventional ones are rather small. This may be caused by the mismatch of data. As mentioned in Section 3.1, CALLHOME consists of real conversational speech where there are only two dominant speakers while all speakers have the same priority in the simulated data. Even though, with the switching method, clustering-integrated approaches can also have a better performance in some cases.

Considering an unsupervised clustering algorithm does not require a temporal order, it would be interesting to investigate the sensitivity of the RNN-based neural clustering to the order of the input chunks. The results are given in Table 4 with various shuffled ratios of samples. The DER decreased by about 1% when the order is completely shuffled, which indicates the EDA-RC is not sensitive to the order of inputs. This may be resulted from the position independence of the Transformer encoder. In addition, table 4 also shows that a global EDA loss has little influence on our EDA-RC.

Table 4: Performance of EDA-RC with the order of chunks being shuffled

<table>
<thead>
<tr>
<th>Models</th>
<th>w/ $L_{global}$</th>
<th>shuffled ratio (%)</th>
<th>Test data</th>
</tr>
</thead>
<tbody>
<tr>
<td>match</td>
<td>mismatch</td>
<td>LS_3</td>
<td>LS_4</td>
</tr>
<tr>
<td>+ COP-K-means [18]</td>
<td>✓</td>
<td>5.17</td>
<td>20.82</td>
</tr>
<tr>
<td>✓</td>
<td>100</td>
<td>5.79</td>
<td>21.88</td>
</tr>
<tr>
<td>✓</td>
<td>10</td>
<td>5.79</td>
<td>21.88</td>
</tr>
<tr>
<td>✓</td>
<td>10</td>
<td>5.78</td>
<td>21.27</td>
</tr>
</tbody>
</table>

4. Conclusions and Future Work

In this paper, we propose a generic neural clustering method for two-stage end-to-end diarization models. We evaluate our model by solving the speaker mismatch problem of overlapped speech recordings. And it can be extended to more applications in the future, e.g., online EEND systems [27, 28]. Compared to unsupervised clustering algorithms, our method is more robust and has a better performance. However, there is still some room for improvement. For example, such a method severely relies on the quality of local speaker representations, so a good chunk-level predictor is important. And it is sensitive to data adaptation, which should be solved in the future work. Apart from this, some advanced neural architectures like self-attention mechanism can also be used to improve the performance.
5. References


