ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-6654-0540-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICASSP43922.2022.9747450

MULTI-SCALE SPEAKER EMBEDDING-BASED GRAPH ATTENTION NETWORKS FOR SPEAKER DIARISATION

Youngki Kwon[1], *Hee-Soo Heo*[1], *Jee-weon Jung*[1], *You Jin Kim*[1], *Bong-Jin Lee*[1], *Joon Son Chung*[2]

[1]Naver Corporation, South Korea
[2]Korea Advanced Institute of Science and Technology, South Korea

ABSTRACT

The objective of this work is effective speaker diarisation using multi-scale speaker embeddings. Typically, there is a trade-off between the ability to recognise short speaker segments and the discriminative power of the embedding, according to the segment length used for embedding extraction. To this end, recent works have proposed the use of multi-scale embeddings where segments with varying lengths are used. However, the scores are combined using a weighted summation scheme where the weights are fixed after the training phase, whereas the importance of segment lengths can differ within a single session.

To address this issue, we present three key contributions in this paper: (1) we propose graph attention networks for multi-scale speaker diarisation; (2) we design scale indicators to utilise scale information of each embedding; (3) we adapt the attention-based aggregation to utilise a pre-computed affinity matrix from multi-scale embeddings.

We demonstrate the effectiveness of our method in various datasets where the speaker confusion which constitutes the primary metric drops over 10% in average relative compared to the baseline.

Index Terms— Speaker Diarisation, Multi-scale, Graph Attention Networks.

1. INTRODUCTION

Speaker diarisation is a task of partitioning audio clips (i.e., sessions) into homogeneous speaker segments, essentially solving the problem of *"who spoke when"*. It is an important pre-processing step for many speech-related tasks, such as transcribing meetings or movie scripts.

Recent work on speaker diarisation can be divided into two strands. The first is end-to-end based methods [1–3], which have received increasing attention recently. Although these works have shown promise in limited environments, they have been reported not to generalise well to real-world conditions. The second strand employs multiple conventional modules, comprising a multi-stage pipeline. While the stages differ by the systems, they typically consist of speech activity detection, speaker embedding extraction, and clustering. Researches in this strand are focused on optimising the pipeline.

In particular, the quality of speaker embedding plays a crucial role in the performance of speaker diarisation systems. In general, speaker embeddings produced from longer segments are more discriminative. However, since these embeddings are more likely to contain multiple speakers, they are more vulnerable to rapid speaker transitions. Deriving speaker embeddings from a short duration mitigates this issue, but, their discriminative power is known to be weaker. Because of this trade-off, recent literature in speaker diarisation adopts a window of 1.5 seconds for extracting speaker embeddings.

A few works have addressed this issue by using *multi-scale* speaker embeddings where the different scales refer to the duration of audio for embedding extraction [4, 5]. Different scales are combined using weights assigned to each scale. Various techniques have been introduced to derive these weights. However, none of these works involves adaptive mechanisms that can assign weights dynamically. Thus, once the weights are fixed, it is applied to entire sessions. Because of the limitation of the existing framework, speaker embeddings can only be compared with embeddings where the scale is identical. This work proposes to address these limitations.

In this study, we propose several techniques on top of the existing multi-scale speaker embedding framework. Specifically, we introduce a graph attention network (GAT) [6] in place of weighted summation when merging multi-scale speaker embeddings. The graph attention networks can model non-Euclidean relations between speaker embeddings with different scales where each embedding is set as a node.

Our approach can compare embeddings with different scales where the weights are dynamically assigned via the attention mechanism. It can even assign different weights within a single session which improves the flexibility. A novel *scale indicator*, inspired by positional encoding [7, 8], is also proposed. We additionally extend the attention-based aggregation method, introduced in our previous work [9], by modifying the mechanisms towards multi-scale. We expect that it would be a novel application of multi-scale embeddings in the speaker diarisation.

We validate our approach using a wide range of datasets, DIHARD I, II and III test sets, as well as the VoxConverse dataset. Consistent improvement in all scenarios is observed, demonstrating the effectiveness of our approach.

2. MULTI-SCALE SPEAKER DIARISATION PIPELINE

In this section, we describe the overall process pipeline of speaker diarisation using multi-scale speaker embeddings. The overall structure is illustrated in Figure 1. Speech segments are extracted from the input audio via SAD. Uniform segmentation is performed for all speech segments extracted using different time scales. Speaker embeddings are extracted for each scale segment. We then construct an affinity matrix based on these speaker embeddings. The resulting affinity matrix is used for clustering, the final step in the pipeline. The details of each step are explained through subsections.

2.1. Speech activity detection

The speech activity detection (SAD) module is located at the front of the speaker diarisation pipeline. It performs segmentation regardless of speaker identity. The output of the SAD module is denoted using onsets and offsets of each segment where overlaps and speaker changes can be involved. We employ a convolutional recurrent neural network-based SAD module which has an identical architecture to that of our previous work [10]. We further apply a sliding window on the module's output to decide onsets and offsets where an onset is determined when more than 70% of a window is predicted as speech and vice versa for an offset.

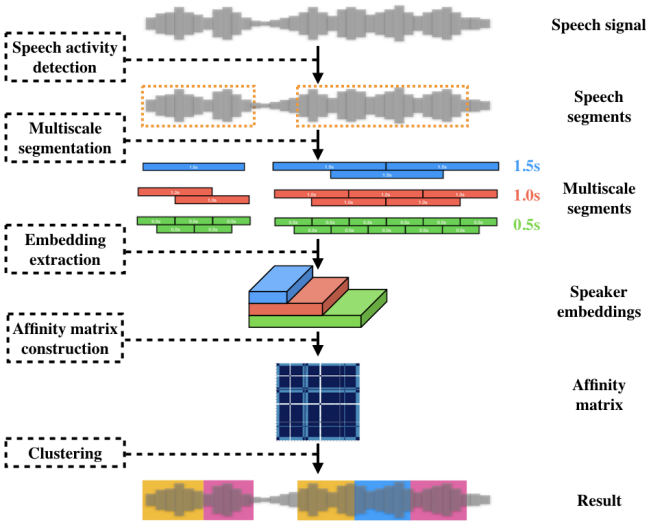

Fig. 1. Multi-scale speaker diarisation pipeline.

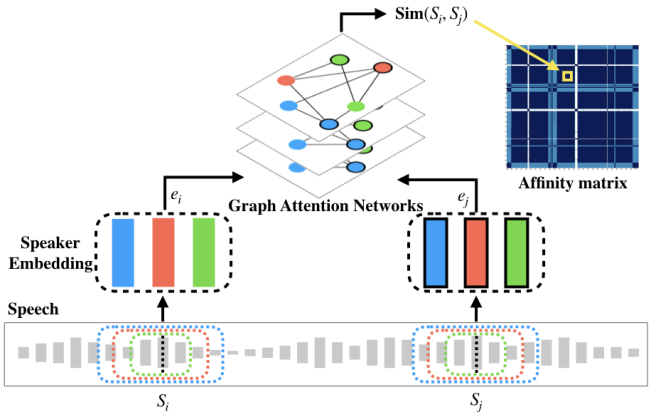

Fig. 2. Overview of proposed GAT-based similarity measure.

2.2. Multi-scale segmentation

Using the SAD module's output, we split each segment into groups of shorter segments where the shorter segments in each group have an equal duration (i.e., scale) and different groups have different scales. This multi-scale segmentation uses four hyper-parameters: number of scales, segment scale defined by window size and shift size, base-scale which determines the unit of clustering and labeling, and segment mapping criterion which defines how to map between segments from different scales. We employ the method introduced in [4]. We use three different scales. The window sizes for each scale are 0.5s, 1.0s, and 1.5s, and the shift sizes are 0.25s, 0.25s, and 0.16s, respectively. We use the scale of 0.5s window size as a base-scale. For segment mapping, we select the segment with the closest midpoint as suggested in [4].

2.3. Speaker embedding extraction

Speaker embeddings are derived for each and every segment divided via the multi-scale segmentation. We build our speaker embedding extraction module on top of that we used in our previous work [9], which is a variant of [11]. Three additional techniques are additionally employed in the training phase.

First, we use *mixup* [12] which generates augmented training samples using a weighted summation of two different speakers' utterances and a corresponding soft-label. Second, we propose a new augmentation technique by connecting two different speakers' utterances also trained with soft-labels. Lastly, each mini-batch is constructed using one of the scales among the utilised multi-scales.

The former two techniques aim to counteract towards segments that include overlaps or speaker changes whereas the last technique is applied to facilitate multi-scale speaker embedding extraction.

2.4. Clustering

We assign a speaker label for each segment via spectral clustering [13, 14] using the speaker embeddings. Spectral clustering has been widely adopted in speaker diarisation pipelines. It is considered as a manifold-based clustering technique, where results highly depend on the quality of the affinity matrix. Elements of the affinity matrix represent the similarity between two speech segments.

Various methods can be used to measure the similarity when constructing the affinity matrix $\mathbf{M}$ using multi-scale speaker embeddings. An existing work applies weighted summation of cosine similarities of each scale segment pair introduced in [4].

$$\begin{aligned} m_{ij} = Sim(S_i,S_j) = w_{0.5} * cos(\mathbf{e}_{i,0.5},\mathbf{e}_{j,0.5}) \\ + w_{1.0} * cos(\mathbf{e}_{i,1.0},\mathbf{e}_{j,1.0}) \\ + w_{1.5} * cos(\mathbf{e}_{i,1.5},\mathbf{e}_{j,1.5}), \end{aligned} \tag{1}$$

where S_i is the $i'th$ segment, $\mathbf{e}_{i,s} \in \mathbb{R}^d$ is the embedding extracted $i'th$ segment at scale s, and $w_s \in \mathbb{R}$ is weight factor for scale s.

This method is used as the baseline of our work.

3. PROPOSED GAT MODULE

In this section, we address our novel GAT module that inputs multi-scale speaker embeddings and outputs an integrated, single value composing the affinity matrix. We first introduce our variant of the original GAT architecture, which has been successfully adopted to speaker verification and spoofing detection in our previous works [15–18]. Then, we address how we derive an integrated affinity matrix using multi-scale speaker embeddings.

The purpose of the GAT used in this work is to calculate similarity $Sim(S_i,S_j)$ where S_i and S_j are the $i'th$ and $j'th$ segment, respectively.

$$Sim(S_i,S_j) = GAT(\mathcal{G}). \tag{2}$$

First, we construct a graph $\mathcal{G}$ using multiple embeddings from two segments as:

$$\mathcal{G} = \{MS_i, MS_j\}, \tag{3}$$

where $MS_i = \{\mathbf{e}_{i,0.5}, \mathbf{e}_{i,1.0}, \mathbf{e}_{i,1.5}\}$ is the set of multi-scale embeddings extracted from the $i'th$ segment.

However, since the existing GAT architecture cannot utilise the scale information of each embedding, a scale indicator is proposed. The scale indicator involves three vectors, each corresponding to one among three scales (0.5s, 1.0s, and 1.5s). These vectors are learnable parameters like the rest of the parameters within the model. We integrate scale indicators into the GAT model using an element-wise addition between each node and its corresponding scale indicator. Through this mechanism, we leverage our prior knowledge on which scale the speaker embedding (node) is extracted from. The dimension of the scale indicator is identical to that of input nodes to make element-wise addition applicable. This technique

is inspired by the positional encoding of the Transformer [7, 8]. With the scale indicator, the node representation $\mathbf{h}_u$ is defined as:

$$\mathbf{h}_u = \mathbf{e}_{i,s} \oplus \mathbf{l}_s, \tag{4}$$

where $\mathbf{h}_u$ refers the node representation of node u and vector $\mathbf{l}_s$ is scale indicator for each scale s.

Since it is difficult to pre-define the relationship between each node like in the previous research, it is assumed that all nodes are connected (i.e., all possible edges exist), including self-connections. Instead, the relations between nodes are defined by an attention mechanism assigning a weight for each edge using learnable parameters. In particular, the attention value $\alpha_{u,v}$ between node u and v is defined as:

$$\alpha_{u,v} = \frac{exp(g(\mathbf{h}_u, \mathbf{h}_v))}{\sum_{k \in \mathcal{G}} exp(g(\mathbf{h}_u, \mathbf{h}_k))}, \tag{5}$$

where the attention function $g(\cdot,\cdot)$ is defined as following:

$$g(\mathbf{h}_u, \mathbf{h}_v) = \begin{cases} (\mathbf{h}_u \otimes \mathbf{h}_v)\mathbf{w}_1, & \text{if } u \in \mathcal{U}(v) \\ (\mathbf{h}_u \otimes \mathbf{h}_v)\mathbf{w}_2, & \text{otherwise} \end{cases}$$

where $\mathbf{w}_1$ and $\mathbf{w}_2$ are the weight vectors for each case, $\otimes$ is the element-wise multiplication, and $\mathcal{U}(v)$ is a set of nodes from the same segment. Different from the previous approach which only compares pairs of the same scale, the attention function defined above allows comparisons between different scales because it considers all possible node pairs. In addition, by using two attention parameters ($\mathbf{w}_1$ and $\mathbf{w}_2$), it is possible to perform aggregation between discriminative nodes within the same segment while comparing nodes from the different segments.

Finally, the similarity between the two segments calculated through the above steps is used to construct the affinity matrix. After constructing the affinity matrix, clustering can be performed as described in the previous section 2.4. The node configuration and operation of the GAT is common to that of the GAT structure applied to speaker verification and spoofing detection [15–17].

4. ATTENTION-BASED FEATURE ENHANCEMENT

Our previous work [9] refines single-scale speaker embeddings before constructing the affinity matrix, referred to as *attention-based embedding aggregation* (AA). Although this technique brings performance improvement in a stable manner, it cannot be applied to multi-scale speaker embeddings. We thus adapt AA towards multi-scale framework as described in Algorithm 1.

The proposed algorithm utilises two affinity matrices: multi-scale $\mathbf{M}$, described in Section 2 and single-scale $\mathbf{C}$. Elements of both matrices represent the speaker similarity between two speech segments. However, there is a difference in how to measure similarity. Elements of $\mathbf{M}$ are derived using our proposed GAT model, whereas elements of $\mathbf{C}$ are derived via a cosine similarity.

However, affinity matrix ($\mathbf{C}$) is constructed by speaker embeddings in base-scale (0.5s) segments may not be as good due to its embedding quality. To overcome this issue, we match the shape by replacing base-scale speaker embeddings with larger-scale speaker embeddings $\mathbf{X}$ when constructing $\mathbf{C}$. For example, a base-scale embedding that represents 2.0s - 2.5s is replaced with a large-scale speaker embedding that represents 1.5s - 3.0s. The objective is to make $\mathbf{C}$ in Algorithm 1 credible.

We use each affinity matrix to construct an attention map. $\mathbf{A}_1$ is constructed from the affinity matrix $\mathbf{M}$ and $\mathbf{A}_2$ is constructed from the affinity matrix $\mathbf{C}$ (lines 5-6). We then create an attention map $\mathbf{A}$ by adding the weights of the two attention maps (line 7). The weight values of both matrices change with each step. We first construct the attention map $\mathbf{A}$ so that $\mathbf{A}_1$ has a higher weight, and then we gradually build up the attention map $\mathbf{A}$ so that $\mathbf{A}_2$ has a higher weight relative to $\mathbf{A}_1$.

Algorithm 1 Multi-scale attention based feature enhancement

1: **Input:** Larger-scale speaker embeddings $\mathbf{X} \in \mathbb{R}^{L \times 256}$, multi-scale affinity matrix $\mathbf{M} \in \mathbb{R}^{L \times L}$
2: **Hyper-parameters:** Number of repetition N, Temperature value τ
3: **for** $i = 0,1,...,N-1$ **do**
4: Construct affinity matrix $\mathbf{C}|c_{i,j} = cos(\mathbf{X}_i, \mathbf{X}_j)$
5: $\mathbf{A}_1$ = softmax($\mathbf{M}$ * τ)
6: $\mathbf{A}_2$ = softmax($\mathbf{C}$* τ)
7: $\mathbf{A}$ = ((N - i) * $\mathbf{A}_1$ + i * $\mathbf{A}_2$) / N
8: $\mathbf{X}$ = dot($\mathbf{A}$, $\mathbf{X}$)
9: **end for**

5. EXPERIMENTS AND RESULTS

We conduct experiments to evaluate the performance of multi-scale diarisation pipelines on various datasets: the test set of the first, second, third DIHARD challenge [19–21] and VoxConverse [22].

5.1. Evaluation protocol

We use the DER (Diarisation Error Rate) as a primary metric. The DER consists of three components: False alarm (FA, speech in prediction but not in reference), Missed speech (MS, speech in reference but not in prediction), Speaker confusion (SC, speech assigned to wrong speaker ID). We use the `dscore`[1] to measure the performance.

In order to test VoxConverse, we conduct experiments under the condition of using system SAD, and in this case, we use 250ms as the collar. For the DIHARD dataset, experiments are performed under the condition provided with a reference SAD, and we use 0 ms as the collar.

5.2. Implementation details

5.2.1. *Clustering*

The clustering stage of our diarisation pipeline requires a threshold for eigenvalues. We tune the threshold for each dataset based on empirical evaluations; the values are 48, 38, 48, and 80 for DIHARD I, DIHARD II, DIHARD III, and VoxConverse, respectively.

5.2.2. *Attention-based feature enhancement*

AA requires two hyperparameters: the temperature value and the number of repetitions. We use 0.30 as the temperature value for all datasets. Use a different value for each dataset for the number of repetitions. Actual values are 10 for DIHARD I, 20 for DIHARD II, 10 for DIHARD III, and 15 for VoxConverse.

5.2.3. *Graph attention networks*

Creating training data and label. As with the speaker verification datasets [27, 28], we construct a training dataset containing speaker pairs and labels. These pairs are extracted from RTTM files of the development data of speaker diarisation datasets. It uses speakers from RTTM to form a speaker pair. For each RTTM, all combinations of two speakers are selected and then multi-scale speech segments with the same midpoint are extracted. We use the DIHARD I, II, III development sets [19–21], VoxConverse development sets [22], ICSI datasets [29], AMI datasets [30], and internal conversation datasets as source data. We also conduct data augmentations following recipes: applying room impulse response [31, 32],

[1] https://github.com/nryant/dscore

Table 1. Results on the DIHARD I, II, III and VoxConverse test sets. (FA: false alarm, MS: miss, SC: speaker confusion, **lower is better for all four metrics**). Except for GAT + AA, all configuration with AA use the original version of AA [9].

Configuration	DER	FA	MS	SC
DIHARD I				
Baseline (0.5s)	31.40	0.0	8.71	22.69
Baseline (1.0s)	25.46	0.0	8.71	16.75
Baseline (1.5s)	24.60	0.0	8.71	15.88
Cosine Fusion	25.89	0.0	8.71	17.18
GAT	23.26	0.0	8.71	14.55
Baseline (0.5s) + AA	31.68	0.0	8.71	22.97
Baseline (1.0s) + AA	22.29	0.0	8.71	13.58
Baseline (1.5s) + AA	20.54	0.0	8.71	11.83
Cosine Fusion + AA	23.44	0.0	8.71	14.72
GAT + AA	**19.00**	0.0	8.71	10.29
Challenge Winner [23]	23.73	-	-	-
DIHARD II				
Baseline (0.5s)	35.61	0.0	9.69	25.93
Baseline (1.0s)	27.60	0.0	9.69	17.91
Baseline (1.5s)	27.87	0.0	9.69	18.18
Cosine Fusion	28.13	0.0	9.69	18.44
GAT	22.85	0.0	9.69	13.16
Baseline (0.5s) + AA	35.28	0.0	9.69	25.59
Baseline (1.0s) + AA	22.77	0.0	9.69	13.08
Baseline (1.5s) + AA	21.47	0.0	9.69	11.78
Cosine Fusion + AA	23.10	0.0	9.69	13.41
GAT + AA	**19.80**	0.0	9.69	10.12
Challenge Winner [24]	18.42	-	-	-
DIHARD III				
Baseline (0.5s)	25.80	0.0	9.52	16.28
Baseline (1.0s)	20.08	0.0	9.52	10.56
Baseline (1.5s)	19.93	0.0	9.52	10.40
Cosine Fusion	20.48	0.0	9.52	10.96
GAT	18.32	0.0	9.52	8.79
Baseline (0.5s) + AA	25.01	0.0	9.52	15.49
Baseline (1.0s) + AA	17.50	0.0	9.52	7.98
Baseline (1.5s) + AA	**17.04**	0.0	9.52	7.52
Cosine Fusion + AA	18.25	0.0	9.52	8.73
GAT + AA	17.35	0.0	9.52	7.83
Challenge Winner [25]	11.30	-	-	-
VoxConverse				
Baseline (0.5s)	28.94	1.38	3.29	24.27
Baseline (1.0s)	23.99	1.38	3.29	19.32
Baseline (1.5s)	24.81	1.38	3.29	20.12
Cosine Fusion	22.98	1.38	3.29	18.31
GAT	18.62	1.38	3.29	13.94
Baseline (0.5s) + AA	28.76	1.38	3.29	24.08
Baseline (1.0s) + AA	21.64	1.38	3.29	16.96
Baseline (1.5s) + AA	14.88	1.38	3.29	10.21
Cosine Fusion + AA	18.95	1.38	3.29	14.27
GAT + AA	**12.78**	1.38	3.29	8.1
Challenge Winner [26]	6.23	-	-	-

channel corruption using FFMPEG, masking frequency bin higher than 4k to simulate narrow-band signal.

Training protocol. In this dataset configuration, positive pairs are scarce. Thus, without further interference, GAT would tend to ignore positive pairs and output all inputs as negative. To reduce this tendency, we oversample positive pairs with an imbalanced dataset sampler [2] and construct a mini-batch of evenly composed pairs of both types.

Binary cross-entropy loss is used to train the GAT model. The model is trained for 50 epochs using a mini-batch size of 50. While the GAT model is being trained, the speaker embedding model is fixed. We train the model using Adam optimizer [33]. A learning rate of 0.0001 is used as the initial value, after which it is adjusted through the cosine annealing scheduler [34].

5.3. Baselines

We test two types of baselines: single-scale diarisation systems and multi-scale diarisation systems. The baseline for a multi-scale diarisation system is described in Section 2. Cosine Fusion in Table 1 represents this system. A single-scale diarisation system that has the same structure in a multi-scale baseline except using only a single-scale segment. Baseline (0.5s), Baseline (1.0s), and Baseline (1.5s) in Table 1 represent these systems. And we also test those mentioned systems in combination with AA.

5.4. Results analysis

Table 1 shows the diarisation results in DIHARD I, II, III, and VoxConverse. All configurations are evaluated with and without the proposed AA technique where top five rows denote performance without AA. A single-scale system shows better numbers when using a larger time scale segment. Cosine Fusion slightly underperforms compared to the best single-scale systems.

The multi-scale pipeline with the GAT module shows significant performance improvements relative to the baselines. This performance improvement can be seen on all datasets that we tested. Compared to the single-scale baseline of the best performance, the speaker confusion is improved by an average of 19.56%, and compared to the multi-scale baseline (Cosine Fusion), there is a performance improvement of 21.90%.

The above-mentioned trend can be confirmed even when feature enhancement (AA) is used. Compared to the single-scale baseline, an average improvement of 10.91% is shown and compared to the multi-scale baseline, there is a performance improvement of 27.04%. Also, the modified feature enhancement has an effect. There is an average increase of 26.30% (GAT vs. GAT + AA).

Challenge winner results, depicted in the last row for each dataset, leverage numerous ad-hoc modules as well as ensembles, making direct comparisons infeasible. Despite these handicaps, our system outperforms the winning system of DIHARD I and shows competitive results in DIHARD II. There is a performance gap for DIHARD III and VoxConverse, however, because our system does not consider overlapped speech and is not a fusion of multiple different systems, we argue that our system still has potentials.

6. CONCLUSION

We proposed a graph attention network for multi-scale speaker diarisation. This module enabled the construction of an affinity matrix by using segments of varying lengths to compute the similarity between the two segments. For a more robust similarity calculation, we designed scale indicators that provide scale information for each multi-scale speaker embedding. We also revamped the previous feature enhancement technique to take advantage of the pre-computed affinity matrix, which was constructed from multi-scale embeddings. We evaluated our proposed method on various datasets and demonstrated consistent performance improvement across a range of datasets.

[2] https://github.com/ufoym/imbalanced-dataset-sampler

7. REFERENCES

[1] Y. Fujita, N. Kanda, S. Horiguchi, K. Nagamatsu and S. Watanabe, "End-to-End Neural Speaker Diarization with Permutation-Free Objectives," in *Proc. Interspeech*, 2019.

[2] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu and S. Watanabe, "End-to-end neural speaker diarization with self-attention," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019.

[3] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue and K. Nagamatsu, "End-to-End Speaker Diarization for an Unknown Number of Speakers with Encoder-Decoder Based Attractors," in *Proc. Interspeech*, 2020.

[4] T. J. Park, M. Kumar and S. Narayanan, "Multi-scale speaker diarization with neural affinity score fusion," in *Proc. ICASSP*, 2021.

[5] K. Wang, X. Mao, H. Wu, C. Ding, C. Shang et al., "The bytedance speaker diarization system for the voxceleb speaker recognition challenge 2021," *arXiv preprint arXiv:2109.02047*, 2021.

[6] Petar Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Li and Y. Bengio, "GRAPH ATTENTION NETWORKS," in *Proc. ICLR*, 2018.

[7] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer et al., "Image transformer," in *Proc. ICML*, 2018.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones et al., "Attention is all you need," in *NIPS*, 2017.

[9] Y. Kwon, J. Jung, H. Heo, Y. J. Kim, B. Lee and J. S. Chung, "Adapting speaker embeddings for speaker diarisation," in *Proc. Interspeech*, 2021.

[10] J. Jung, H. Heo, Y. Kwon, J. S. Chung and B. Lee, "Three-class overlapped speech detection using a convolutional recurrent neural network," in *Proc. Interspeech*, 2021.

[11] Y. Kwon, H.-S. Heo, B.-J. Lee and J. S. Chung, "The ins and outs of speaker recognition: lessons from voxsrc 2020," in *Proc. ICASSP*, 2021.

[12] H. Zhang, M. Cisse, Y. N. Dauphin and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. ICLR*, 2018.

[13] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[14] H. Ning, M. Liu, H. Tang and T. S. Huang, "A spectral clustering approach to speaker diarization," in *Ninth International Conference on Spoken Language Processing*, 2006.

[15] J. Jung, H. Heo, H. Yu and J. S. Chung, "Graph attention networks for speaker verification," in *Proc. ICASSP*, 2021.

[16] H. Tak, J. Jung, J. Patino, M. Kamble, M. Todisco and N. Evans, "End-to-end spectro-temporal graph attention networks for speaker verification anti-spoofing and speech deepfake detection," in *Proc. ASVspoof workshop*, 2021.

[17] H. Tak, J. Jung, J. Patino, M. Todisco and N. Evans, "Graph attention networks for anti-spoofing," in *Proc. Interspeech*, 2021.

[18] J. Jung, H. Heo, H. Tak, H. Shim, J. S. Chung et al., "Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks," *arXiv preprint arXiv:2110.01200*, 2021.

[19] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du et al., "First dihard challenge evaluation plan," *2018, tech. Rep.*, 2018.

[20] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du et al., "The second dihard diarization challenge: Dataset, task, and baselines," in *Proc. Interspeech*, 2019.

[21] N. Ryant, P. Singh, V. Krishnamohan, R. Varma, K. Church et al., "The third dihard diarization challenge," *arXiv preprint arXiv:2012.01477*, 2020.

[22] J. S. Chung, J. Huh, A. Nagrani, T. Afouras and A. Zisserman, "Spot the conversation: Speaker diarisation in the wild," in *Proc. Interspeech*, 2020.

[23] G. Sell, D. Snyder, A. McCree, D. Garcia-Romero, J. Villalba et al., "Diarization is hard: Some experiences and lessons learned for the jhu team in the inaugural dihard challenge.," in *Proc. Interspeech*, 2018.

[24] F. Landini, S. Wang, M. Diez, L. Burget, P. Matějka et al., "But system description for dihard speech diarization challenge 2019," *arXiv preprint arXiv:1910.08847*, 2019.

[25] Y. Wang, M. He, S. Niu, L. Sun, T. Gao et al., "Ustc-nelslip system description for dihard-iii challenge," *arXiv preprint arXiv:2103.10661*, 2021.

[26] X. Xiao, N. Kanda, Z. Chen, T. Zhou, T. Yoshioka et al., "Microsoft speaker diarization system for the voxceleb speaker recognition challenge 2020," in *Proc. ICASSP*, 2021.

[27] J. S. Chung, A. Nagrani and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proc. Interspeech*, 2018.

[28] A. Nagrani, J. S. Chung and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Proc. Interspeech*, 2017.

[29] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart et al., "The icsi meeting corpus," in *Proc. ICASSP*. IEEE, 2003.

[30] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot et al., "The ami meeting corpus: A pre-announcement," in *International workshop on machine learning for multimodal interaction*. Springer, 2005, pp. 28–39.

[31] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. ICASSP*. 2017, pp. 5220–5224, IEEE.

[32] D. Povey, G. Boulianne, L. Burget, P. Motlicek and P. Schwarz, "The Kaldi Speech Recognition Toolkit," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2011.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[34] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.