AASIST: AUDIO ANTI-SPOOFING USING INTEGRATED SPECTRO-TEMPORAL GRAPH ATTENTION NETWORKS

Jee-weon Jung[1], *Hee-Soo Heo*[1], *Hemlata Tak*[2], *Hye-jin Shim*[3], *Joon Son Chung*[4], *Bong-Jin Lee*[1], *Ha-Jin Yu*[3], *Nicholas Evans*[2]

[1]Naver Corporation, South Korea, [2]EURECOM, Sophia Antipolis, France
[3]School of Computer Science, University of Seoul, South Korea
[4]Korea Advanced Institute of Science and Technology, South Korea

ABSTRACT

Artefacts that differentiate spoofed from bona-fide utterances can reside in specific temporal or spectral intervals. Their reliable detection usually depends upon computationally demanding ensemble systems where each subsystem is tuned to some specific artefacts. We seek to develop an efficient, single system that can detect a broad range of different spoofing attacks without score-level ensembles. We propose a novel heterogeneous stacking graph attention layer that models artefacts spanning heterogeneous temporal and spectral intervals with a heterogeneous attention mechanism and a stack node. With a new max graph operation that involves a competitive mechanism and a new readout scheme, our approach, named *AASIST*, outperforms the current state-of-the-art by 20% relative. Even a lightweight variant, AASIST-L, with only 85k parameters, outperforms all competing systems.

Index Terms— audio spoofing detection, anti-spoofing, graph attention networks, end-to-end, heterogeneous

1. INTRODUCTION

Spoofing detection solutions can be an important consideration when automatic speaker verification systems are deployed in real-world applications. They help to protect reliability by determining whether an input speech utterance is genuine (*bona-fide*) or spoofed. Practical spoofing detection systems are required to detect spoofed utterances generated using a wide range of different techniques. The ASVspoof community has led research in the field with a series of challenges accompanied by public datasets [1–4]. Two major scenarios are being studied, namely logical access (LA) and physical access. The focus in this paper is LA, which considers spoofing attacks mounted with voice conversion and text-to-speech algorithms.

Recent studies show that discriminative information (i.e., spoofing artefacts) can reside in specific temporal and spectral intervals [5–9]. Artefacts tend to be dependent upon the nature of the attack and the specific attack algorithm. Adaptive mechanisms, which have the flexibility to concentrate on the domain in which the artefacts lie, are therefore crucial to reliable detection. In our recent works [10, 11], we proposed end-to-end systems leveraging a RawNet2-based encoder [12, 13] and graph attention networks [14]. We model both temporal and spectral information concurrently using a pair of parallel graphs and then apply element-wise multiplication to the two graphs to combine and exploit the information each provides. While we achieve state-of-the-art performance [11], we believe that there is still room for further improvement. Because the two graphs are heterogeneous, integrating them using a heterogeneity-aware technique should be beneficial.

We propose four extensions to our previous work, RawGAT-ST [11] where the first three compose the proposed model named *AASIST*. The fourth relates to a lightweight, efficient version of AASIST. First, we propose an extended variant of the graph attention layer, referred to as a "*heterogeneous stacking graph attention layer*" (HS-GAL). It facilitates the concurrent modelling of heterogeneous (temporal and spectral) graph representations. A HS-GAL includes a modified, heterogeneity-aware attention mechanism and an additional stack node, each inspired from [15] and [16] respectively. HS-GAL directly models two arbitrary graphs where the two graphs can have different numbers of nodes and different dimensionalities. Second, we propose a mechanism referred to as "*max graph operation*" (MGO) that mimics the max feature map [17]. MGO involves two branches where each branch comprises two HS-GALs and graph pooling layers, followed by an element-wise maximum operation. The underlying objective here is to enable different branches to learn different groups of artefacts. Components of branches that better capture artefacts will then persist following the max operation. Third, we present a new readout scheme that utilises the stack node. Finally, given the application of anti-spoofing solutions in *aasist*ing automatic speaker verification systems [18, 19] and given the associated requirement for practical, lightweight models, we further propose a lightweight variant of AASIST which comprises only 85k parameters.

2. PRELIMINARIES

In this section, we summarise the components that are common to the RawGAT-ST [11] and new AASIST models. We describe: i) the RawNet2-based encoder used for extracting high-level feature maps from raw input waveforms; ii) the graph module which includes graph attention and graph pooling layers. These two components are the "encoder" and "graph module" in Fig. 1, respectively.

2.1. RawNet2-based encoder

A growing number of researchers are adopting models that operate directly upon raw waveform inputs. The work described in this paper utilises a variant of the RawNet2 model that was introduced in [12] for the task of automatic speaker verification and applied subsequently to anti-spoofing [11, 13]. It extracts high-level representations F, $F \in \mathbb{R}^{C \times S \times T}$ directly from raw waveform inputs

Code and models are available at:
https://github.com/clovaai/aasist

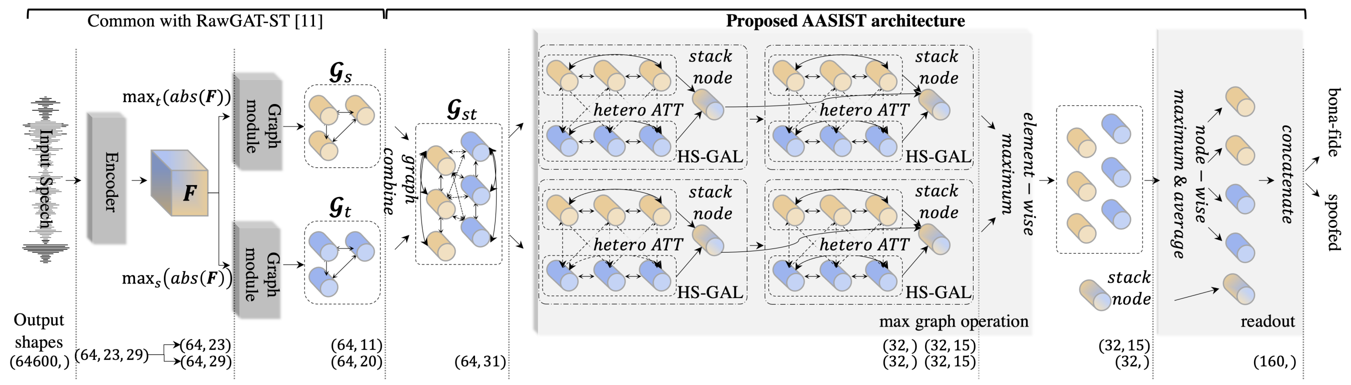

Fig. 1. The AASIST framework. *Identical to* [11]: the encoder extracts F; two graph modules are used to model temporal and spectral domains in parallel. *Proposed*: we construct a combined heterogeneous graph using two graph module outputs; a max graph operation technique is applied to two branches that model heterogeneous graphs in parallel before the application of an element-wise maximum; each branch includes two HS-GAL layers and two graph pooling layers (graph pooling layers and one HS-GAL layer is omitted in the illustration); the readout scheme concatenates node-wise maximum and average, and the stack node which is then followed by an output layer.

where C, S, and T are the number of channels, spectral (frequency) bins, and the temporal sequence length respectively.

In contrast to the original RawNet2 model, we interpret the output of the sinc-convolution layer as a 2-dimensional image with a single channel (akin to a spectrogram) rather than a 1-dimensional sequence. This is achieved by treating the output of each filter as a spectral bin. A series of six residual blocks with pre-activation [20] is used to extract the high-level representation. Each residual block comprises a batch normalisation layer [21], a 2-dimensional convolution layer, SeLU activation [22], and a max pooling layer. Further information can be found in [11].

2.2. Graph module

Graph attention network. Recent advances in graph neural networks have brought performance breakthroughs in a number of tasks [10, 14, 23] where a graph is defined by a set of nodes and a set of edges connecting different node pairs. Using high-dimensional vectors as nodes, graph neural networks can be used to model the non-Euclidean data manifold between different nodes. We have shown that graph attention networks [14] can be applied to both speaker verification [23] and spoofing detection [10, 11].

The graph attention layer used in our work is a variant of the original architecture [14]. In our work, graphs are fully-connected in the sense of there being edges between each and every node pair. This is because the relevance of each node pair to the task at hand cannot be predetermined. Instead, the self-attention mechanism in a graph attention layer derives data-driven attention weights, assigned to each edge, to reflect the relevance of each node pair. Before deriving attention weights, an element-wise multiplication is utilised to make edges symmetric. The reader is referred to [11] (Section 3) for further details.

Graph pooling. Various graph pooling layers have been proposed to effectively reduce the number of nodes [24, 25]. This has the aim of reducing complexity and improving discrimination. We apply a simple attentive graph pooling layer to the output of each graph attention layer. Except for the omission of projection vector normalisation, our implementation is identical to that in [25].

Let $\mathcal{G}, \mathcal{G} \in \mathbb{R}^{N \times D}$ be the output graph of a graph attention layer where N is the number of nodes and D refers to the dimensionality of each node. Note that the order of nodes is meaningless; the relationships between them are defined via the attention weights assigned to each edge. Attention weights are derived via $\mathcal{G} \cdot P$ where $\cdot$ is the dot product and $P, P \in \mathbb{R}^D$ is a projection vector that returns a scalar attention weight for each node. After the multiplication of a sigmoid non-linearity with the corresponding k nodes, the nodes with the top-k values are retained while the rest are discarded.

3. AASIST

AASIST builds upon the foundation of our previous work, RawGAT-ST, whereby two heterogeneous graphs, one temporal and the other spectral, are combined at the model-level. However, instead of using trivial element-wise operations and fully-connected layers, AASIST relies upon a more elegant approach using the proposed HS-GAL, in addition to the proposed MGO and the new readout technique.

The AASIST framework is illustrated in Fig. 1. High-level representations F are extracted by feeding raw waveforms into the RawNet2-based encoder (Section 2.1). The pair of graph modules first model the temporal and spectral domains in parallel, giving $\mathcal{G}_t$ and $\mathcal{G}_s$ (Section 2.2). The results are combined into $\mathcal{G}_{st}$ (Section 3.1) and processed by the MGO (Section 3.3) which includes four HS-GAL layers (Section 3.2) and four graph pooling layers. The readout operation is then performed, followed by an output layer with two nodes.

3.1. Graph combination

We first compose a combined heterogeneous graph ($\mathcal{G}_{st}$ in Fig. 1) from the pair of temporal and spectral graphs. Let $\mathcal{G}_t, \mathcal{G}_t \in \mathbb{R}^{N_t \times D_t}$ and $\mathcal{G}_s, \mathcal{G}_s \in \mathbb{R}^{N_s \times D_s}$ be temporal and spectral graphs respectively, each derived according to:

$$\mathcal{G}_t = graph\_module(max_s(abs(F))), \quad (1)$$
$$\mathcal{G}_s = graph\_module(max_t(abs(F))), \quad (2)$$

where graph_module refers to the combination of graph attention and graph pooling layers and where $F \in \mathbb{R}^{C \times S \times T}$ is the encoder output feature map. We then formulate a combined graph $\mathcal{G}_{st}$ which has $N_t + N_s$ nodes by adding edges between every node in G_t and every node in G_s and vice versa (dotted arrows under $\mathcal{G}_{st}$). The

Table 1. Results in terms of EER (%) for all 13 attacks in the ASVspoof 2019 LA evaluation set, pooled min t-DCF (P1), and pooled EER (%, P2) for the baseline RawGAT-ST [11] and the proposed AASIST models. RawGAT-ST results are regenerated using the same GPU environment used for AASIST, and from experiments with three different random seeds. Results in parentheses show the single best pooled result. The best performance for each separate attack and the best pooled results are highlighted in boldface.

System	A07	A08	A09	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	P1	P2 (%)
RawGAT-ST	1.19	**0.33**	0.03	1.54	0.41	1.54	0.14	**0.14**	1.03	**0.67**	**1.44**	**3.22**	**0.62**	0.0443(0.0333)	1.39(1.19)
AASIST	**0.80**	0.44	**0.00**	**1.06**	**0.31**	**0.91**	**0.1**	**0.14**	**0.65**	0.72	1.52	3.40	**0.62**	**0.0347(0.0275)**	**1.13(0.83)**

new edges in the combined graph G_{st} allow for the estimation of attention weights between pairs of heterogeneous nodes which each span both temporal and spectral domains. Despite the combination, G_{st} remains a heterogeneous graph in that nodes in each of the constituent graphs lie in different latent spaces; N_t and D_t are normally different to N_s and D_s.

3.2. HS-GAL

The new contribution is based upon a *heterogeneous stacking graph attention layer* (HS-GAL in Fig. 1). It comprises two components, namely *heterogeneous attention* and a *stack node*. Our approach to heterogeneous attention is inspired by the approach to the modelling of heterogeneous data described in [15].

The input to the HS-GAL is first projected into another latent space to give each of the two graphs with node dimensionalities D_t and D_s a common dimensionality D_{st}. Two fully-connected layers are utilised for this purpose, each projecting one of the constituent sub-graphs to a dimensionality of D_{st}.

Heterogeneous attention. Whereas the homogeneous graphs use a single projection vector to derive attention weights, we use three different projection vectors to calculate attention weights for the heterogeneous graph. They are illustrated inside $\mathcal{G}_{st}$ of Fig. 1 and are used to determine attention weights for edges connecting: (i) $\mathcal{G}_t$ to $\mathcal{G}_t$ (edges between blue nodes); (ii) nodes in $\mathcal{G}_s$ to $\mathcal{G}_t$ and $\mathcal{G}_t$ to $\mathcal{G}_s$ (dotted edges); (iii) nodes in $\mathcal{G}_s$ to nodes in $\mathcal{G}_s$ (edges between orange nodes). The projection vector in the case of (ii) above applies to edges in both directions; this is possible because the graph attention layer we use applies element-wise multiplication between two nodes making attention weights symmetrical, whereas the original graph attention layer concatenates two nodes [14].

Stack node. We also introduce a new, additional node referred to as the "*stack node*". The role of the stack node is to accumulate heterogeneous information, namely information or the relationship between temporal and spectral domains. The stack node is connected to the full set of nodes (stemming from $\mathcal{G}_t$ and $\mathcal{G}_s$). The use of uni-directional edges from all other nodes to the stack node helps to preserve information in both $\mathcal{G}_t$ and $\mathcal{G}_s$. It does not transmit information to other nodes. The output stack node of the first HS-GAL layer is used to initialise the stack node of the following layer. The behaviour of the stack node is similar to that of classification tokens [16], except that connections to other nodes are uni-directional.

3.3. Max graph operation and readout

The new "*max graph operation*" (MGO), highlighted with a large grey box in Fig. 1, is inspired by a number of works in the anti-spoofing literature that showed the benefit of element-wise maximum operations [11, 26]. MGO aims to combine and exploit different solutions to spoofing detection and the potentially different artefacts they detect. It utilises two parallel branches where an element-wise maximum is applied to the two branch outputs. Specifically, each branch involves two HS-GALs in sequence where a graph pooling layer is applied to the output of each HS-GAL. Thus, MGO comprises four HS-GALs and four graph pooling layers. The two HS-GALs in each branch share the same stack node; the stack node of the preceding HS-GAL is passed to the following HS-GAL. An element-wise maximum is also applied to the stack nodes of each branch.

The proposed readout scheme is illustrated in the right-most grey box of Fig. 1. First, we derive four nodes by applying a node-wise maximum and average to nodes originally belonging to $\mathcal{G}_s$ and $\mathcal{G}_t$, respectively. The last hidden layer is then formed from the concatenation of these four nodes with the stack node.

3.4. Lightweight variant of AASIST

We also report a lightweight variant, namely *AASIST-L*. Leaving the architecture identical, we tune the number of parameters to compose a model with 85k parameters using the population-based training algorithm [27]. This results in a 332kB model. Leveraging techniques such as half-precision training, the size can be further reduced. We demonstrate that AASIST-L still outperforms all models except AASIST.

4. EXPERIMENTS AND RESULTS

4.1. Dataset and metrics

All experiments were performed using the ASVspoof 2019 logical access (LA) dataset [3, 37]. It comprises three subsets: train, development, and evaluation. The train and the development sets contain attacks created from six spoofing attack algorithms (A01-A06), whereas the evaluation set contains attacks created from thirteen algorithms (A7-A19). Readers are referred to [37] for full details.

We use two metrics: the default minimum tandem detection cost function (min t-DCF) [18] and the equal error rate (EER). The min t-DCF shows the impact of spoofing and the spoofing detection system upon the performance of an automatic speaker verification system whereas the EER reflects purely standalone spoofing detection performance.

Wang et al. [33] showed that the performance of spoofing detection systems can vary significantly with different random seeds. We observed the same phenomenon; when trained with different random seeds, the EER of the baseline RawGAT-ST [11] system was found to vary between 1.19% and 2.06%. Thus, our baseline results are slightly different to those reported in [11]. All results reported in this paper are average results in addition to the best result from three runs with different random seeds.

4.2. Implementation details

AASIST was implemented using PyTorch, a deep learning toolkit in Python. Inputs in the form of raw waveforms of $64,600$ samples ($\approx$ 4 seconds) are fed to the RawNet2-based encoder. The

Table 2. A comparison to recently proposed, competing state-of-the-art systems. Results reported in terms of pooled min t-DCF and pooled EER (%). For the proposed AASIST and AASIST-L models, we report the best single result. All systems shown are single models without any kind of score-level fusion.

System	**# Param**	**Front-end**	**Architecture**	**min t-DCF**	**EER (%)**
Ours	**297k**	**Raw waveform**	**AASIST**	**0.0275**	**0.83**
Ours	**85k**	**Raw waveform**	**AASIST-L**	**0.0309**	**0.99**
Tak et al. [11]	437k	Raw waveform	RawGAT-ST	0.0333	1.19
Zhang et at. [28]	1,100k	FFT	SENet	0.0368	1.14
Hua et al. [29]	350k	Raw waveform	Res-TSSDNet	0.0481	1.64
Ge et al. [30]	24,480k	Raw waveform	Raw PC-DARTS	0.0517	1.77
Li et al. [31]	960k	CQT	MCG-Res2Net50	0.0520	1.78
Chen et al. [32]	-	LFB	ResNet18-LMCL-FM	0.0520	1.81
Wang et al. [33]	276k	LFCC	LCNN-LSTM-sum	0.0524	1.92
Luo et al. [34]	-	LFCC	Capsule network	0.0538	1.97
Zhang et al. [35]	-	LFCC	Resnet18-OC-softmax	0.0590	2.19
Li et al. [36]	-	CQT	SE-Res2Net50	0.0743	2.50

Table 3. Ablation experiments which demonstrate the impact of each of the listed techniques. Performance reported in terms of pooled min t-DCF and EER and for "average(best)" results from three experiments with different random seeds.

Configuration	min t-DCF	EER
AASIST	0.0347(0.0275)	1.13(0.83)
w/o heterogeneous attention	0.0415(0.0384)	1.44(1.37)
w/o stack node (conventional readout)	0.0380(0.0330)	1.21(1.03)
w/o MGO	0.0410(0.0378)	1.35(1.19)

first layer of the encoder, sinc-convolution [38], has 70 filters. The RawNet2-based encoder consists of six residual blocks. The first two have 32 filters while the remaining four have 64 filters. The first two graph attention layers have 64 filters. Graph pooling layers remove 30% and 50% of temporal and spectral nodes, respectively. All subsequent graph attention layers have 32 filters and are followed by graph pooling which further reduces the number of nodes by 50%. We used Adam optimiser [39] with a learning rate of 10^{-4} and cosine annealing learning rate decay. The AASIST-L system was tuned with a population-based training algorithm using 7 generations where each generation includes 30 experiments with different hyper-parameters [27].

4.3. Results

Table 1 shows the EER for individual attacks, the pooled min t-DCF, and the pooled EER. For pooled results, the best performance is shown in brackets. Results are shown for the proposed AASIST model and the state-of-the-art baseline RawGAT-ST [11] model. AASIST performs similarly or better than the baseline for 9 out of the 13 conditions. For the remaining 4 conditions for which the baseline performs better, the differences are modest. For conditions where AASIST performs better, improvements can be substantial, e.g. for the A15 condition where AASIST outperforms the baseline by over 35% relative (1.03% vs 0.65%). Pooled min t-DCF and EER results are shown in the two right-most columns of Table 1. AASIST outperforms the RawGAT-ST baseline in terms of both the pooled min t-DCF and the pooled EER. For AASIST, the min t-DCF drops by over 20% relative (0.0443 vs 0.0347). In the best case, AASIST delivers an EER of 0.83% and a min t-DCF of 0.0275. These results are better than the results for any competing single system reported in the literature.

Comparison to state-of-the-art systems. Table 2 presents a comparison of the proposed AASIST model to the performance of a number of recently proposed competing systems [11, 28–36]. The set of systems covers a broad range of different front-end representations and model architectures. Five of the top six systems operate upon raw waveform inputs while the top three systems are based upon graph attention networks. The proposed AASIST system is the best performing of all.

AASIST-L. Table 2 also shows a comparison in terms of complexity for the lightweight AASIST-L model and other models for which the number of parameters is openly available. In using only 85k parameters, AASIST-L is substantially less complex than all other systems. The min t-DCF and EER achieved by the AASIST-L model are better than those of all other systems except for the full AASIST model. If appropriately modified using techniques such as half-precision inference and parameter pruning, we believe that it would be small enough to be used in embedded systems.

Ablations. Table 3 shows results for ablation experiments for which one of the components in the AASIST model is removed. Results show that all three techniques are beneficial; without any one of them, results are worse than for the full AASIST model. The use of heterogeneous attention has the greatest impact on performance. While having less impact than heterogeneous attention and MGO, the stack node is also beneficial.

5. CONCLUSION

We propose AASIST, a new end-to-end spoofing detection system based upon graph neural networks. New contributions are three-fold: (i) a heterogeneous stacking graph attention layer (HS-GAL), used to model temporal and spectral sub-graphs, consisting of a heterogeneous attention mechanism and a stack node to accumulate heterogeneous information; (ii) a max graph operation (MGO) that involves the competitive selection of artefacts; (iii) a new readout scheme. AASIST improves the performance of a state-of-the-art baseline by over 20% relative in terms of pooled min t-DCF. Even the lightweight version, AASIST-L, with 85k parameters, outperforms all competing systems.

6. REFERENCES

[1] Z. Wu, T. Kinnunen, N. Evans et al., "Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge," in *Proc. Interspeech*, 2015.

[2] T. Kinnunen, M. Sahidullah, H. Delgado et al., "The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection," in *Proc. Interspeech*, 2017.

[3] M. Todisco, X. Wang, V. Vestman et al., "Asvspoof 2019: Future horizons in spoofed and fake audio detection," in *Proc. Interspeech*, 2019.

[4] J. Yamagishi, X. Wang, M. Todisco et al., "Asvspoof 2021: accelerating progress in spoofed and deepfake speech detection," *arXiv 2109.00537*, 2021.

[5] J. Yang, R. K. Das and H. Li, "Significance of subband features for synthetic speech detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, 2019.

[6] K. Sriskandaraja, V. Sethu, P. N. Le and E. Ambikairajah, "Investigation of sub-band discriminative information between spoofed and genuine speech.," in *Proc. Interspeech*, 2016.

[7] J. Jung, H. Shim, H. Heo and H. Yu, "Replay attack detection with complementary high-resolution information using end-to-end dnn for the asvspoof 2019 challenge," in *Proc. Interspeech*, 2019.

[8] H. Tak, J. Patino, A. Nautsch et al., "An explainability study of the constant Q cepstral coefficient spoofing countermeasure for automatic speaker verification," in *Proc. Odyssey*, 2020.

[9] H. Tak, J. Patino, A. Nautsch et al., "Spoofing Attack Detection using the Non-linear Fusion of Sub-band Classifiers," in *Proc. Interspeech*, 2020.

[10] H. Tak, J. Jung, J. Patino et al., "Graph attention networks for anti-spoofing," in *Proc. Interspeech*, 2021.

[11] H. Tak, J. Jung, J. Patino et al., "End-to-end spectro-temporal graph attention networks for speaker verification anti-spoofing and speech deepfake detection," in *Proc. ASVspoof workshop*, 2021.

[12] J. Jung, S. Kim, H. Shim et al., "Improved RawNet with Feature Map Scaling for Text-Independent Speaker Verification Using Raw Waveforms," in *Proc. Interspeech*, 2020.

[13] H. Tak, J. Patino, M. Todisco et al., "End-to-end anti-spoofing with rawnet2," in *Proc. ICASSP*, 2021.

[14] P. Veličković, G. Cucurull, A. Casanova et al., "Graph attention networks," in *Proc. ICLR*, 2018.

[15] X. Wang, H. Ji, C. Shi et al., "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019.

[16] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019.

[17] X. Wu, R. He, Z. Sun and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, 2018.

[18] T. Kinnunen, K. A. Lee, H. Delgado et al., "t-dcf: a detection cost function for the tandem assessment of spoofing countermeasures and automatic speaker verification," in *Proc. Odyssey*, 2018.

[19] H. Shim, J. Jung, J. Kim and H. Yu, "Integrated replay spoofing-aware text-independent speaker verification," *Applied Sciences*, vol. 10, no. 18, 2020.

[20] K. He, X. Zhang, S. Ren and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, 2016.

[21] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015.

[22] G. Klambauer, T. Unterthiner, A. Mayr and S. Hochreiter, "Self-normalizing neural networks," in *Proc NeuralIPS*, 2017.

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

[24] J. Lee, I. Lee and J. Kang, "Self-attention graph pooling," in *Proc. ICML*, 2019.

[25] H. Gao and S. Ji, "Graph u-nets," in *Proc. ICML*, 2019.

[26] G. Lavrentyeva, S. Novoselov, E. Malykh et al., "Audio replay attack detection with deep learning frameworks," in *Proc. Interspeech*, 2017.

[27] M. Jaderberg, V. Dalibard, S. Osindero et al., "Population based training of neural networks," *arXiv 1711.09846*, 2017.

[28] Y. Zhang, W. Wang and P. Zhang, "The effect of silence and dual-band fusion in anti-spoofing system," in *Proc. Interspeech*, 2021.

[29] G. Hua, A. Beng jin teoh and H. Zhang, "Towards end-to-end synthetic speech detection," *IEEE Signal Processing Letters*, 2021.

[30] W. Ge, J. Patino, M. Todisco and N. Evans, "Raw differentiable architecture search for speech deepfake and spoofing detection," in *Proc. ASVspoof workshop*, 2021.

[31] X. Li, X. Wu, H. Lu et al., "Channel-wise gated res2net: Towards robust detection of synthetic speech attacks," in *Proc. Interspeech*, 2021.

[32] T. Chen, A. Kumar, P. Nagarsheth et al., "Generalization of audio deepfake detection," in *Proc. Odyssey*, 2020.

[33] X. Wang and J. Yamagishi, "A Comparative Study on Recent Neural Spoofing Countermeasures for Synthetic Speech Detection," in *Proc. Interspeech*, 2021.

[34] A. Luo, E. Li, Y. Liu et al., "A capsule network based approach for detection of audio spoofing attacks," in *Proc.ICASSP*, 2021.

[35] Y. Zhang, F. Jiang and Z. Duan, "One-class learning towards synthetic voice spoofing detection," *IEEE Signal Processing Letters*, vol. 28, 2021.

[36] X. Li, N. Li, C. Weng et al., "Replay and synthetic speech detection with res2net architecture," in *Proc. ICASSP*, 2021.

[37] X. Wang, J. Yamagishi, M. Todisco et al., "Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech," *Computer Speech & Language*, vol. 64, 2020.

[38] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *Proc. IEEE SLT*, 2018.

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