GCT: GATED CONTEXTUAL TRANSFORMER FOR SEQUENTIAL AUDIO TAGGING

Yuanbo Hou[1]*, Yun Wang*[2]*, Wenwu Wang*[3]*, Dick Botteldooren*[1]

[1]WAVES, Ghent University, Belgium [2]Meta AI, USA [3]CVSSP, University of Surrey, UK

ABSTRACT

Audio tagging aims to assign predefined tags to audio clips to indicate the class information of audio events. Sequential audio tagging (SAT) means detecting both the class information of audio events, and the order in which they occur within the audio clip. Most existing methods for SAT are based on connectionist temporal classification (CTC). However, CTC cannot effectively capture event connections due to the conditional independence assumption between outputs at different times. The contextual Transformer (cTransformer) addresses this issue by exploiting contextual information in SAT. Nevertheless, cTransformer is also limited in exploiting contextual information as it only uses forward information in inference. This paper proposes a gated contextual Transformer (GCT) with forward-backward inference (FBI). In addition, a gated contextual multi-layer perceptron (GCMLP) block is proposed in GCT to improve the performance of cTransformer structurally. Experiments on the two real-life audio datasets with manually annotated sequential labels show that the proposed GCT with GCMLP and FBI performs better than the CTC-based methods and cTransformer.

Index Terms— Sequential audio tagging, connectionist temporal classification, gated contextual Transformer

1. INTRODUCTION

Audio tagging (AT) [1] is a fundamental task in audio classification, where events in audio clips are identified via multi-label classification. Sequential audio tagging (SAT) [2] aims to mine both the type information of events and the order information between events. With SAT, the number of events can be estimated as a byproduct. Sequence-level AT offers more information about the audio clip than temporal agnostic event-level AT. SAT can not only provide information that is useful for AT tasks [1], but also support tasks like audio captioning [3], event anticipation [4] and scene perception [5].

Prior works on SAT mostly use connectionist temporal classification (CTC) [6] as its core. To perform SAT on polyphonic audio clips, sequential labels are introduced in [7] to train CTC-based convolutional recurrent neural networks to tag diverse event sequences. The polyphony of audio makes it hard to define the order of events, therefore, the order of the beginning and end boundaries of events are used as sequential labels in [7]. The double-boundary sequential labels are also used to train a CTC-based recurrent neural network equipped with the long short-term memory (BLSTM-CTC) [8] to perform sound event detection (SED), which detects the type and temporal position of events in audio clips. Apart from these methods using double-boundary labels, single-boundary sequential labels (sequences of the start boundary of events) are exploited in a 2-stage CTC-based method [9] for SED. However, CTC-based methods have difficulty modelling the contextual information in event sequences because CTC implicitly assumes that the network outputs are conditionally independent at each time step [6]. To take advantage of context, a contextual Transformer (cTransformer) [2] is proposed to explore bidirectional information in event sequences to make more effective use of the contextual information in SAT.

To exploit both forward and backward information of events, cTransformer uses a bidirectional decoder to model correlations between preceding and following events in both directions in training, while only the forward direction of the decoder is used for inferring [2]. In inference, cTransformer does not utilize the event sequences information in the reverse sequence branch. To address this limitation, we propose a gated contextual Transformer (GCT) with a forward-backward inference (FBI) algorithm to infer the target event from both directions, where the event context is incorporated during inference. In addition, to enhance the decoder's power to capture the context implicit in event sequences, a gated contextual multi-layer perceptron (GCMLP) block is proposed to adapt the contextual information to estimate final predictions.

The contributions of this work are: 1) We propose GCT equipped with GCMLP and FBI to improve cTransformer's ability to capture contextual information of event sequences; 2) We explore the effect of pretrained weights on modules of GCT under two transfer learning modes to gain insight into the role of GCT modules; 3) We visualize the attention distribution in hidden layers of GCT to investigate how GCT connects acoustic representations with clip-level event tokens and bidirectionally infers the event sequences; 4) To evaluate the performance of GCT, we sequentially annotate a polyphonic audio dataset. We compare the performance of GCT, cTransformer, and CTC-based methods on two real-life datasets. This paper is organized as follows, Section 2 introduces the GCT. Section 3 describes the dataset, experimental setup, and analyzes the results. Section 4 gives conclusions.

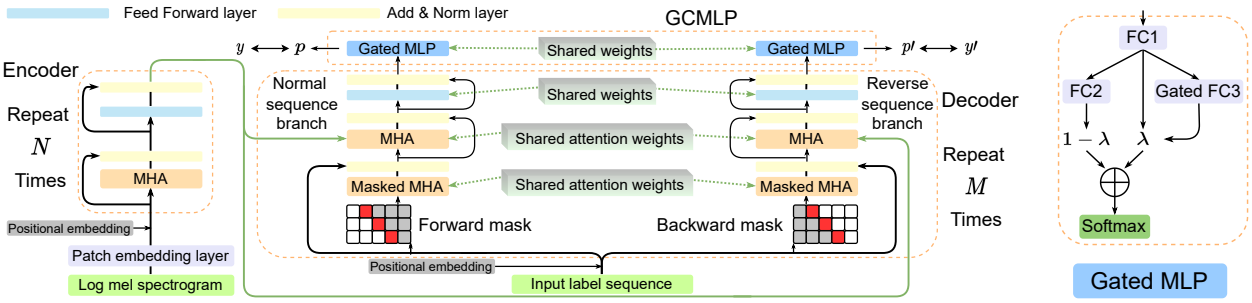

Fig. 1: The proposed gated contextual Transformer and gated MLP block. In the mask matrices, the red, gray, and white blocks present the positions corresponding to the target to be predicted, the positions of masked data, and the positions of available data.

2. GATED CONTEXTUAL TRANSFORMER (GCT)

Following the approach in cTransformer [2], the sequence of event start boundaries is used as the sequential label. Given the normal sequential label l is "<*S*>, *event*$_1$, ..., *event*$_k$, <*E*>", where k is the number of events, <*S*> and <*E*> are the tokens indicating the start and end of the sequence. The reverse sequential label l' is "<*S'*>, *event*$_k$, ..., *event*$_1$, <*E*>", where <*S'*> is the token indicating the start of the reverse sequence.

2.1. Encoder and Decoder of GCT

Encoder. There are two ways for the input: 1) the entire spectrogram of the audio clip, as in cTransformer [2]; 2) the patch sequence by dividing the spectrogram clip into patches, as in AST [10]. Inputting the entire clip enables the encoder to utilize the global audio information of events directly. However, the patch sequence may help the model to align acoustic patch sequences with the corresponding event label sequences. Fig. 1 shows the structure of GCT with input patches. Referring to AST, GCT uses a patch embedding layer to map the patches containing basic acoustic features into high-level representations, and uses updatable positional embedding (Pos_emb) to capture spatial information of each patch. When inputting clips, Pos_emb before the encoder is removed, and the patch embedding layer is replaced with a linear layer to keep the encoder input dimension consistent with input patches. The encoder consists of N identical blocks with a multi-head attention layer (MHA) and a feed-forward layer (FF) with layer normalization, which are analogous to the encoder in Transformer [11].

Decoder. The bidirectional decoder in GCT consists of a normal and a reverse sequence branch. To facilitate information exchange between branches, in Fig. 1, each branch consists of M identical blocks serially. To preserve the autoregressive property [11] of Transformer, forward and backward masks are used to block future and past information about the target in the normal and reverse branches, respectively. With the combined effect of forward and backward mask matrices, the normal and reverse sequence branches will infer the same target at each time step. Thus, the decoder can extract forward and backward information about the target.

2.2. Gated contextual multi-layer perceptron (GCMLP)

GCMLP aims to perform the final conditioning of the decoder output based on the gated MLP (gMLP) block and shared weights while considering the contextual information about the target to achieve more accurate predictions. In Fig. 1, gMLP consists of 3 fully-connected (FC) layers of the same size. Denote the input as X, the weight of FC1, FC2 and FC3 as W_1, W_2 and W_3, then the corresponding output is $F_1 = W_1X$, $F_2 = ReLU(W_2F_1)$, and $\lambda = \sigma(W_3F_1)$, where σ is logistic sigmoid: $\sigma(x) = 1/(1+e^{-x})$, and $ReLU$ is activation function [12]. Then, the output of gMLP is

$$gMLP = Softmax((1-\lambda) \odot F_2 + \lambda \odot F_1) \qquad (1)$$

Where $\odot$ is the element-wise product, F_2 is a higher-level representation of the target based on F_1, and can be viewed as the target's embedding from another perspective. FC3 evaluates the relative importance of each element in F_1, and then combines it with F_2 according to the estimated importance of each element. That is, gMLP is used to generate multi-view results and fuse them relying on the learnable gate unit.

At each time step, denote the output of the normal and reverse branches through GCMLP as p and $p\prime$, the corresponding labels are y and $y\prime$, respectively. Cross entropy (CE) [11] loss is used in GCT: $\mathcal{L}_{\text{normal}} = CE(p, y)$, $\mathcal{L}_{\text{reverse}} = CE(p\prime, y\prime)$. To further align the classification space of the two branches to allow the model to focus on the contextual information of the same target, the mean squared error (MSE) [13] is used as a context loss to measure the distance between p and $p\prime$ in the latent space: $\mathcal{L}_{\text{context}} = MSE(p\prime, p)$. Hence, the total loss of GCT is $\mathcal{L} = \mathcal{L}_{\text{normal}} + \mathcal{L}_{\text{reverse}} + \mathcal{L}_{\text{context}}$.

2.3. Forward-backward inference (FBI)

To utilize both the normal and reverse branches in inference, we propose FBI to make the two branches infer the same goal at each step, and fuse the prediction to form the final output. While preserving the autoregressive property [11], FBI integrates the forward and the backward sequence information implied in the normal and reverse branches during inference, which benefits the model to make fuller use of the contextual information of events in inference. Details of FBI are below.

Algorithm 1 PyTorch pseudo code for the proposed FBI

```
# X: input log mel spectrogram; X': X reversed along the time axis
E, E' = Encoder(X), Encoder(X')                    # output of encoder
I, I' = <S>, <S'>          # start token of the normal and reverse sequence
for k in range(L − 1):     # L: max length of event sequences; B: batch size
    D = Decoder_normal_branch(E, I)        # D: (B, L, number of tokens)
    p = GCMLP(D[:, -1, :])                 # pick the latest target probability vector
    D' = Decoder_reverse_branch(E', I')
    p' = GCMLP(D'[:, -1, :])    # p' and p are the same target's predictions
    p_ci = αp + (1 − α)p'  # p_ci: final prediction with contextual information; α: importance factor of the forward information, default to 0.5.
    p_et = torch.max(p_ci, dim=1).item()   # p_et: predicted event token
    if p_et == <E>: break                  # <E>: end token of event sequences
    I = torch.cat([I, torch.ones(1, 1).fill_(p_et)], dim=1)
    I' = torch.cat([I', torch.ones(1, 1).fill_(p_et)], dim=1)
```

3. EXPERIMENTS AND RESULTS

3.1. Dataset, Baseline, Experiments Setup, and Metrics

The *DCASE* [14] and *Noiseme* [15] real-life datasets are used. *DCASE* contains 10 classes of audio events, where the training and test sets consist of 1578 and 288 10-second (10s) audio clips, respectively. *Noiseme* contains 33 classes of audio events, where the training and test sets consist of 2312 and 505 10s audio clips, respectively. In training, 20% of training samples are randomly selected to form the validation set. The sequential labels of *DCASE* are here [2]. We manually annotated sequential labels of *Noiseme*. For more details, please visit the homepage (https://github.com/Yuanbo2020/GCT).

The BLSTM-CTC [8], convolutional bidirectional gated recurrent units with CTC (CBGRU-CTC) [16], CBGRU-CTC with GLU in convolutional layers (CGLU-BGRU-CTC) [7] and in convolutional-recurrent layers (CBGRU-GLU-CTC) [9] are used as baselines. Performance of cTransformer [2], the first Transformer-based model for SAT, is also presented.

For features, log mel-bank energy with 128 banks [17] is used with a Hamming window length of 25*ms* and a hop size of 10*ms* between windows. A batch size of 64 and stochastic gradient descent with a momentum 0.99 [18] with an initial learning rate of 1e-3 are used. Layer normalization [19] and dropout [20] are used to prevent overfitting. Systems are trained on a card Tesla V100 GPU for 500 epochs. Accuracy (*Acc*), *F-score* [21], and area under curve (*AUC*) [22] are used to quantify the classification ability of models on audio events. BLEU score [23] is used to evaluate the accuracy of predicted audio event sequences.

Table 1: AUC of different input modes of GCT with different numbers of encoder and decoder blocks on the *Noiseme* dataset.

#	N	M	*Patches*	*Clip*	#	N	M	*Patches*	*Clip*
1	1	2	0.575±0.010	0.647±0.012	7	8	6	0.534±0.033	0.557±0.058
2	2	4	0.584±0.009	0.661±0.016	8	8	8	0.614±0.020	0.518±0.063
3	4	4	0.600±0.018	**0.662±0.013**	9	9	5	0.609±0.026	0.512±0.017
4	5	5	0.599±0.046	0.660±0.071	10	9	7	0.604±0.066	0.511±0.013
5	6	6	0.609±0.017	0.596±0.075	11	9	9	0.608±0.027	0.511±0.007
6	7	7	**0.627±0.019**	0.543±0.024	12	10	10	0.606±0.052	0.508±0.032

3.2. Results and Analysis

Model structure. In Table 1, changes in the number of blocks do not significantly affect the performance of GCT when the input is patches, and shallow GCT ($N \leq 6$, $M \leq 6$) outperforms deep GCT when the input is clips, probably because the manually labeled dataset is not large enough to take advantage of large and deep models. Deep neural networks are hard to train [24]. System #12 in Table 1 has about 170 layers, and the vanishing gradient [25] we observed during training implies that such a deep model is challenging to optimize effectively, which may also be a reason for the poor performance of deep GCT. We will use $\{7, 7\}$ and $\{4, 4\}$ as default structures for GCT when the input is patches and clips, respectively.

Ablation study. Prior work [2] shows the benefits of context for SAT, so this paper focuses on the role of GCT components. GCT with patches has a more complex structure than GCT with spectrogram clips. With patches as input, GCT uses Pos_emb to capture the position information of patches. In Table 1, GCT does not degrade much as the number of N and M are increased when patches equipped with Pos_emb are input. Does this mean that Pos_emb is a crucial component of GCT? Table 2 conducts ablation experiments for this.

Table 2: Ablation study of GCT $\{7, 7\}$ component on *Noiseme*.

#	*Pos_emb*	*GCMLP*	***AT:*** *Acc (%)*	***AT:*** *AUC*	***SAT:*** *BLEU*
1	✗	✗	92.23±0.61	0.600±0.014	0.297±0.045
2	✓	✗	93.00±0.54	0.616±0.012	0.312±0.019
3	✗	✓	92.55±0.62	0.610±0.009	0.309±0.023
4	✓	✓	**93.21±0.27**	**0.627±0.019**	**0.338±0.012**

In Table 2, Pos_emb (shown as #2) slightly outperforms GCMLP (i.e. #3). This reveals that when the input is small patches, the position information is valuable for the model to effectively capture the local information of events. Combining Pos_emb and GCMLP offers better results, shown as #4 in Table 2, due to the use of local context information of events.

Table 3 illustrates that FBI plays a more powerful role when coarse-grained clips are input. The reason may be that after the spectrogram is split into patches, the time interval between forward and reverse information is shortened in each patch, equivalent to reducing the range of context that FBI can capture. Hence, FBI is more valuable for clips as input.

Pretrained weight. Table 2 indicates the role of Pos_emb for GCT with patches as input. In GCT, both Pos_emb and encoder designs are inspired by AST [10]. Compared with datasets used in this paper, the 5800-hour Audioset [26] used to train AST is super large-scale. If the knowledge learned from Audioset by AST is transferred to GCT, what impact will it have on GCT? Table 4 shows the results of transferring parameters of AST to the corresponding parts of GCT

Table 3: Ablation study of the inference method on *Noiseme*.

	FBI	*Patches*	*Clip*		*FBI*	*Patches*	*Clip*
Acc (%)	✗	93.21±0.27	93.49±0.39	AUC	✗	0.627±0.019	0.662±0.013
	✓	93.57±0.46	**94.01±0.31**		✓	0.635±0.014	**0.685±0.022**

in the fixed and fine-tuning modes [27]. In training, the rest of GCT is randomly initialized. To make the result of GCT a benchmark for Transformer-based models on *DCASE* dataset, in Table 4, $\{N, M\} = \{6, 6\}$ is adopted in GCT.

Compared with #1 in Table 4, which is randomly initialized and trained, the remaining models using pretrained weights achieve larger improvements, which proves the benefit of knowledge from large datasets in improving models' performance. Specifically, #5 outperforms #4, indicating that the encoder with the ability in acoustic feature extraction is more important than Pos_emb in providing the position information of patches. The fixed mode (i.e. #2) is better than fine-tuning the transferred parameters (i.e. #3). The reason may be that the part (Pos_emb and encoder) containing pretrained weights and the remaining randomly initialized part (decoder and GCMLP) differ greatly in the latent space, fine-tuning these two disparate parts using the same learning rate will inevitably affect the performance of (Pos_emb and encoder) with audio events expertise. Freezing the parameters containing audio knowledge in #2 will reduce the learning burden of GCT, which helps GCT to focus more on optimizing the remaining parts, thus achieving better results.

Comparison with prior methods. In this paper, the weights of the sub-losses of GCT default to 1. To compare fairly, the weights of cTransformer losses are also set to 1. For single event recognition (i.e. AT), CTC-based models perform similarly to Transformer-based models. But for modelling of event sequences (i.e. SAT), Transformer-based models work better. That means CTC, which assumes that networks' outputs at different times are conditionally independent, is inferior to Transformer's ability to model long-term dependences on sequences. Compared to Transformer, cTransformer performs better because it captures contextual information of events in sequences. The proposed GCT improves cTransformer in structure and inference and achieves the best results on AT and SAT tasks.

Case study. There is often overlap among various events in polyphonic audio datasets. If one event is almost covered by another, can GCT identify it? By visualizing the attention distribution, Fig. 2 illustrates how GCT recognizes event sequences based on start tokens combined with frame-level acoustic representations from the encoder. In Fig. 2 (a) and (b), the x-axis represents the number of audio frames in the encoder, and the y-axis is the event token for each input step in the decoder. GCT's attention to the audio clip varies after different start tokens are given as input. For the normal branch of input $<S>$, GCT's attention goes forward from the beginning of the audio clip, focusing on acoustic representations and cues of the next event in turn. For the reverse order branch of input $<S'>$, GCT analyses from the end of the audio clip backward. Even though most of the first *speech* (sp) event overlaps with the continuous *running water* (wa) event, GCT can still use the contextual information to effectively identify the covered *speech*. Fig. 2 (c) clearly shows how much GCT pays attention to different existing cues in each inference step. After combining the fine-grained audio representations in Fig. 2 (a) and (b), under the guidance of $<S>$ and $<S'>$, the normal and reverse branches of GCT infer event sequences in turn. The inferred sequences match the corresponding labels consistently, which means that GCT is good at exploiting event context to identify event sequences.

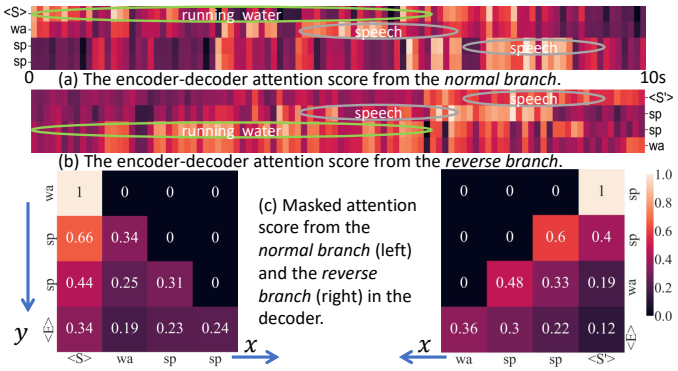

Fig. 2: Attention in GCT. In subgraph (c), the x-axis is each event predicted in an autoregressive way, the y-axis is the reference event.

Table 4: Effect of transfer learning on GCT on *DCASE*.

#	***Pos_emb***	***Encoder***	***AT:** Acc (%)*	***SAT:** BLEU*
1	No Transfer		89.13±0.58	0.435±0.037
2	Fixed	Fixed	**97.68±0.18**	**0.677±0.014**
3	Fine-tuned	Fine-tuned	96.27±0.36	0.645±0.019
4	Fixed	Fine-tuned	93.84±0.85	0.639±0.016
5	Fine-tuned	Fixed	96.45±0.47	0.662±0.015

4. CONCLUSION

To improve cTransformer in structure and inference, we propose a gated contextual Transformer (GCT) with GCMLP and FBI for SAT. To gain insight into the role of GCT modules, we study the contribution of different modules to GCT with the help of pretrained weights. GCT performs well in event classification and sequence modelling compared to CTC and Transformer based models. Finally, we illustrate the potential of GCT by visualising a typical polyphonic audio sample.

5. ACKNOWLEDGEMENTS

The WAVES Research Group received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

Table 5: Comparison of other methods on *Noiseme*.

Method	***AT:** F-score (%)*	***AT:** AUC*	***SAT:** BLEU*
BLSTM-CTC [8]	44.06±1.92	0.549±0.021	0.252±0.047
CBGRU-CTC [16]	48.55±1.74	0.583±0.009	0.290±0.043
CGLU-BGRU-CTC [7]	50.03±1.66	0.579±0.011	0.297±0.015
CBGRU-GLU-CTC [9]	48.79±1.87	0.572±0.007	0.275±0.011
Transformer [11]	45.98±0.75	0.563±0.027	0.310±0.019
cTransformer [2]	46.44±1.10	0.577±0.032	0.323±0.009
Proposed GCT (Clip)	**56.03±1.53**	**0.685±0.022**	**0.403±0.068**

6. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE/ACM TASLP*, vol. 26, no. 2, pp. 379–393, 2017.

[2] Y. Hou, Z. Liu, B. Kang, Y. Wang, and D. Botteldooren, "CT-SAT: Contextual transformer for sequential audio tagging," in *INTERSPEECH*, 2022, pp. 4147–4151.

[3] Y. Koizumi, R. Masumura, K. Nishida, M. Yasuda, and S. Saito, "A transformer-based audio captioning model with keyword estimation," in *INTERSPEECH*, 2020, pp. 1977–1981.

[4] D. P. Ellis, *Prediction-driven computational auditory scene analysis*, Ph.D. thesis, Columbia university, 1996.

[5] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*, Springer, 2018.

[6] A. Graves and F. Gomez, "Connectionist temporal classification:labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006, pp. 369–376.

[7] Y. Hou, Q. Kong, J. Wang, and S. Li, "Polyphonic audio tagging with sequentially labelled data using crnn with learnable gated linear units," in *DCASE Workshop*, 2018, pp. 78–82.

[8] Y. Wang and F. Metze, "A first attempt at polyphonic sound event detection using connectionist temporal classification," in *ICASSP*, 2017, pp. 2986–2990.

[9] Y. Hou, Q. Kong, S. Li, and M. D. Plumbley, "Sound event detection with sequentially labelled data based on connectionist temporal classification and unsupervised clustering," in *ICASSP*, 2019, pp. 46–50.

[10] Y. Gong, Y. A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *INTERSPEECH*, 2021, pp. 571–575.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., "Attention is all you need," in *NIPS*, 2017, pp. 6000–6010.

[12] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with relu activation function and linear spline-type methods," *Neural Networks*, vol. 110, pp. 232–242, 2019.

[13] T. Kim, J. Oh, N. Kim, et al., "Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation," in *IJCAI*, 2021, pp. 2628–2635.

[14] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *DCASE Workshop*, 2018, pp. 19–23.

[15] S. Burger, Q. Jin, P. Schulam, and F. Metze, "Noisemes: Manual annotation of environmental noise in audio streams," technical report CMU-LTI-12-07, Carnegie Mellon University, 2012.

[16] Y. Hou, Q. Kong, and S. Li, "Audio tagging with connectionist temporal classification model using sequentially labelled data," in *CSPS*, 2018, pp. 955–964.

[17] A. Bala, A. Kumar, and N. Birla, "Voice command recognition system based on MFCC and DTW," *IJEST*, vol. 2, no. 12, pp. 7335–7342, 2010.

[18] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *ICML*, 2013, pp. 1139–1147.

[19] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, et al., "On layer normalization in the transformer architecture," in *ICML*, 2020, pp. 10524–10533.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, pp. 162, 2016.

[22] J. Huang and C. Ling, "Using auc and accuracy in evaluating learning algorithms," *IEEE TKDE*, vol. 17, no. 3, pp. 299–310, 2005.

[23] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BlEU: a method for automatic evaluation of machine translation," in *ACL*, 2002, pp. 311–318.

[24] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks," in *ICML*, 2019, pp. 1675–1685.

[25] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *IJUFKS*, vol. 6, no. 02, pp. 107–116, 1998.

[26] J. Gemmeke, D. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *ICASSP*, 2017, pp. 776–780.

[27] Y. Hou, F. K. Soong, J. Luan, and S. Li, "Transfer learning for improving singing-voice detection in polyphonic instrumental music," in *INTERSPEECH*, 2020, pp. 1236–1240.