Spatio-Temporal Attention Pooling for Audio Scene Classification

Huy Phan[*1], *Oliver Y. Chén*[2], *Lam Pham*[1], *Philipp Koch*[3], *Maarten De Vos*[2], *Ian McLoughlin*[1], *Alfred Mertins*[3]

[1]School of Computing, University of Kent, UK
[2]Department of Engineering Science, University of Oxford, UK
[3]Institute for Signal Processing, University of Lübeck, Germany

[*]Corresponding email: h.phan@kent.ac.uk

Abstract

Acoustic scenes are rich and redundant in their content. In this work, we present a spatio-temporal attention pooling layer coupled with a convolutional recurrent neural network to learn from patterns that are discriminative while suppressing those that are irrelevant for acoustic scene classification. The convolutional layers in this network learn invariant features from time-frequency input. The bidirectional recurrent layers are then able to encode the temporal dynamics of the resulting convolutional features. Afterwards, a two-dimensional attention mask is formed via the outer product of the spatial and temporal attention vectors learned from two designated attention layers to weigh and pool the recurrent output into a final feature vector for classification. The network is trained with *between-class* examples generated from between-class data augmentation. Experiments demonstrate that the proposed method not only outperforms a strong convolutional neural network baseline but also sets new state-of-the-art performance on the LITIS Rouen dataset.

Index Terms: audio scene classification, attention pooling, convolutional neural network, recurrent neural network

1. Introduction

Audio scene classification (ASC) is one of the main tasks in environmental sound analysis. It allows a machine to recognize a surrounding environment based on its acoustic sounds [1]. One way to look at an audio scene is to consider its foreground events mixed with its background noise. Due to the complex content of audio scenes, it is challenging to classify them correctly, as classification models tend to overfit the training data. A good practice in audio scene classification is to split a long recording (e.g. 30 seconds) into short segments (a few seconds long) [2, 3, 4, 5]. By this, we increase the data variation and a classification model can be trained more efficiently with a large set of small segments rather than a small set of the whole long recordings. Classification of long recordings is then achieved by aggregating classification results across the decomposed short segments.

Similar to many other research fields, using deep learning for the ASC task has become a norm. Convolutional neural network (CNNs) [6, 7, 8, 5, 4, 9, 10, 11] are most commonly used deep learning techniques, thanks to their feature learning capability. Leveraging the nature of audio signals, sequential modelling with recurrent neural networks (RNNs) [2, 12, 13] and temporal transformer networks [10] have also demonstrated results on par with the convolutional counterparts. The deep learning models were trained either on time-frequency representations, such as log Mel-scale spectrograms [4, 11], or high-level features like label tree embedding features [6, 2]. Mitigation of overfitting effects via feature fusion [4, 6, 2] and model fusion [11, 14, 1, 9] has also been extensively explored.

Given the rich content of acoustic scenes, they typically contain a lot of irrelevant and redundant information. This fact naturally gives rise to the question of how to encourage a deep learning model to automatically discover and focus on discriminative patterns and suppress irrelevant ones from the acoustic scenes for better classification. We seek to address that question in this work using an attention mechanism [15]. To this end, we propose a spatio-temporal attention pooling layer in combination with a convolutional recurrent neural network (CRNN), inspired by their success in the audio event detection task [16, 17]. The convolutional layers of the CRNN network are used to learn invariant features from time-frequency input, whose temporal dynamics are subsequently modelled by the upper bidirectional recurrent layers. Temporal soft attention [15] has usually been coupled with a recurrent layer to learn a weighting vector to combine its recurrent output vectors at different time steps into a single feature vector. However, spatial attention (i.e. attention on the feature dimension), and hence, joint spatio-temporal attention, have been left uncharted. With the proposed spatio-temporal attention layer, we aim to learn a two-dimensional attention mask for spatio-temporal pooling purpose. The rationale is that those entries of the recurrent output that are more informative should be assigned with strong weights and vice versa. We expect discriminative features to be accentuated in the induced feature vector, while irrelevant ones to be suppressed and blocked after the spatio-temporal attention pooling. In addition, we harness between-class data augmentation [18] to generate between-class examples to better train the network to minimize the Kullback-Leibler (KL) divergence loss.

2. The proposed CRNN with spatio-temporal attention pooling

2.1. Input

Following the common practice in ASC [2, 3, 4, 5], we decompose an audio snippet, which is 30 seconds long in the experimental LITIS Rouen dataset [19] (cf. Section 4.1), into non-overlapping 2-second segments. It has been shown in previous works that an auxiliary channel which is created by excluding background noise from an audio recording is also helpful for the classification task, as the prominent foreground events of the scene are exposed more clearly to a network [4, 2, 6]. Hence, we create such an auxiliary channel by subtracting background noise using the minimum statistics estimation and subtraction method [20].

A 2-channel short audio segment is then transformed into the time-frequency domain, e.g. using log Mel spectrogram, to

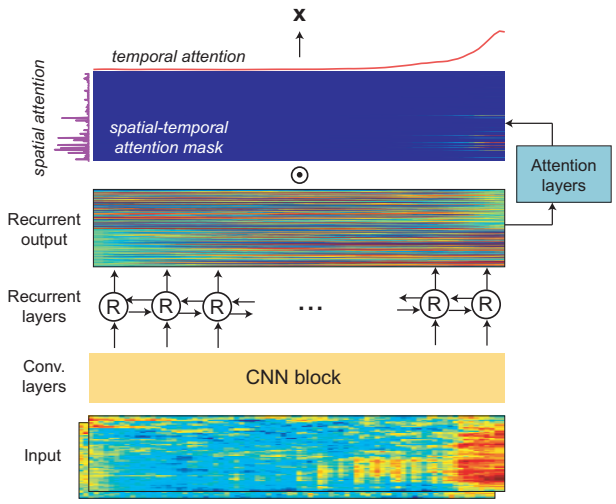

Figure 1: *Overview of the proposed CRNN with spatio-temporal attention pooling.*

obtain a multi-channel image $\mathbf{S} \in \mathbb{R}^{M \times T \times K}$, where M, T, and $K = 2$ denote the number of frequency bins, the number of time indices, and the number of channels, respectively (cf. Section 4.2 for further detail).

2.2. Convolutional layers

The convolutional block of the network consists of three convolutional layers followed by three max-pooling layers. For clarity, we show the configuration of the convolutional and max-pooling layers in Table 1 alongside their corresponding resulting feature maps.

For each convolutional layer, after the convolution operation, Rectified Linear Unit (ReLU) activation [21] and batch normalization [22] are exercised on the feature map. The number of convolutional filters of a convolutional layer is designed to be the double of its preceding layer, i.e. $64 \rightarrow 128 \rightarrow 256$, in order to gain the representation power when the spectral size gets smaller and smaller after the pooling layers. Note that zero-padding (also known as *SAME* padding) is used during convolution to keep the temporal size unchanged (i.e. always equal to T).

With the pooling kernel size 4×1 and a stride 1×1, the max pooling layers are only effective on the frequency dimension to gain spectral invariance. As a consequence, the spectral dimension is reduced from M of the original input to $\frac{M}{4} \rightarrow \frac{M}{16} \rightarrow \frac{M}{64}$ after the three pooling layers, respectively. The last resulting feature map of size $\frac{M}{64} \times T \times 256$ is reshaped to $\mathbf{O} \in \mathbb{R}^{F \times T}$, where $F = \frac{M}{64} \times 256$, to present to the upper recurrent layers of the network which will be elaborated in the following section.

2.3. Bidirectional recurrent layers with spatio-temporal attention pooling

In the context of sequential modelling with recurrent layers, the convolutional output $\mathbf{O} \in \mathbb{R}^{F \times T}$ is interpreted as a sequence of T feature vectors $(\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T)$ where each $\mathbf{o}_t \in \mathbb{R}^F$, $1 \leq t \leq T$. A bidirectional recurrent layer then reads the sequence of convolutional feature vectors into a sequence of recurrent output vectors $\mathbf{Z} \equiv (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_T)$, where

Table 1: *Configuration of the convolutional layers.*

Layer	Input size	Filter size	Stride	Num. filter	Feat. map
Conv. 1	$M \times T \times K$	5×5	1×1	64	$M \times T \times 64$
Max pool. 1	$M \times T \times 64$	4×1	4×1	–	$\frac{M}{4} \times T \times 64$
Conv. 2	$\frac{M}{4} \times T \times 64$	3×3	1×1	128	$\frac{M}{4} \times T \times 128$
Max pool. 2	$\frac{M}{4} \times T \times 128$	4×1	4×1	–	$\frac{M}{16} \times T \times 128$
Conv. 3	$\frac{M}{16} \times T \times 128$	2×2	1×1	256	$\frac{M}{16} \times T \times 256$
Max pool. 3	$\frac{M}{16} \times T \times 256$	4×1	4×1	–	$\frac{M}{64} \times T \times 256$

$$\mathbf{z}_t = [\mathbf{h}_t^{\mathrm{b}} \oplus \mathbf{h}_t^{\mathrm{f}}]\mathbf{W} + \mathbf{b}, \tag{1}$$

$$\mathbf{h}_t^{\mathrm{f}} = \mathcal{H}(\mathbf{o}_t, \mathbf{h}_{t-1}^{\mathrm{f}}), \tag{2}$$

$$\mathbf{h}_t^{\mathrm{b}} = \mathcal{H}(\mathbf{o}_t, \mathbf{h}_{t+1}^{\mathrm{b}}). \tag{3}$$

Here, $\mathbf{h}_t^{\mathrm{f}}, \mathbf{h}_t^{\mathrm{b}} \in \mathbb{R}^H$ represent the forward and backward hidden state vectors of size H at recurrent time step t, respectively, while $\oplus$ indicates vector concatenation. $\mathbf{W} \in \mathbb{R}^{2H \times 2H}$ denotes a weight matrix and $\mathbf{b} \in \mathbb{R}^{2H}$ denotes bias terms. $\mathcal{H}$ represents the hidden layer function of the recurrent layer and is realized by a Gated Recurrent Unit (GRU) [23] here. We further stack multiple bidirectional GRU cells onto one another to form a deep RNN for sequential modelling as in [2, 24].

The recurrent output $\mathbf{Z}$ is of size $2H \times T$. In order to learn a spatio-temporal attention mask to pool and reduce $\mathbf{Z}$ into a single feature vector, we learn two attention vectors, $\mathbf{a}^{\mathrm{tem}} \in \mathbb{R}^T$ for temporal attention and $\mathbf{a}^{\mathrm{spa}} \in \mathbb{R}^{2H}$ for spatial attention. Formally, the temporal attention weight a_t^{tem} at the time index t, $1 \leq t \leq T$, and the spatial attention weight a_s^{spa} at the spatial index s, $1 \leq s \leq 2H$ are computed as

$$a_t^{\mathrm{tem}} = \frac{\exp\left(f\left(\mathbf{z}_t\right)\right)}{\sum_{i=1}^{T} \exp\left(f(\mathbf{z}_i)\right)}, \tag{4}$$

$$a_s^{\mathrm{spa}} = \frac{\exp\left(\tilde{f}\left(\tilde{\mathbf{z}}_t\right)\right)}{\sum_{i=1}^{2H} \exp\left(\tilde{f}(\tilde{\mathbf{z}}_i)\right)}, \tag{5}$$

respectively. In (4) and (5), $\mathbf{z}_t$ represents the column of $\mathbf{Z}$ at the column (i.e. temporal) index t whereas $\tilde{\mathbf{z}}_s$ represents the row of $\mathbf{Z}$ at the row (i.e. spatial) index s. f and $\tilde{f}$ denote the scoring functions of the temporal and spatial attention layers and are given by

$$f(\mathbf{z}) = \tanh(\mathbf{z}^{\mathsf{T}}\mathbf{W}^{\mathrm{att}} + \mathbf{b}^{\mathrm{att}}), \tag{6}$$

$$\tilde{f}(\tilde{\mathbf{z}}) = \tanh(\tilde{\mathbf{z}}^{\mathsf{T}}\tilde{\mathbf{W}}^{\mathrm{att}} + \tilde{\mathbf{b}}^{\mathrm{att}}), \tag{7}$$

respectively, where $\mathbf{W}^{\mathrm{att}}$ and $\tilde{\mathbf{W}}^{\mathrm{att}}$ are the trainable weight matrices and $\mathbf{b}^{\mathrm{att}}$ and $\tilde{\mathbf{b}}^{\mathrm{att}}$ are the trainable biases. The spatio-temporal attention mask $\mathbf{A}$ is then obtained as

$$\mathbf{A} = \mathbf{a}^{\mathrm{spa}} \otimes \mathbf{a}^{\mathrm{tem}}, \tag{8}$$

where $\otimes$ denotes vector outer product operation.

The final feature vector $\mathbf{x} \in \mathbb{R}^{2H}$ is achieved via spatio-temporal attention pooling. Intuitively, element-wise multiplication between the recurrent output $\mathbf{Z}$ and the spatio-temporal attention mask is first carried out, followed by summation over

the time dimension. Formally, the s-th entry, $1 \leq s \leq 2H$, of $\mathbf{x}$ is given as

$$x_s = \sum_{t=1}^{T} \tanh(\mathbf{A}_{st}\mathbf{Z}_{st}). \quad (9)$$

Inspired by [25], a $\tanh$ activation is applied prior to the summation in (9). Due to its output range $(-1, 1)$, it is likely that $\tanh$ activation does not only suppress the irrelevant features but also enhances the informative ones in the resulting feature vector $\mathbf{x}$ [25].

Eventually, the obtained feature vector $\mathbf{x}$ is presented to a softmax layer to accomplish classification.

2.4. Calibration with Support Vector Machine

Compared to the standard softmax, Support Vector Machines (SVM) usually achieve better generalization due to their maximum margin property [26]. Similar to [6, 2], after training the network, we calibrate the final classifier by employing a linear SVM in replacement for the softmax layer. The trained network is used to extract feature vectors for the original training examples (without data augmentation) which are used to train the SVM classifier. During testing, the SVM classifier is subsequently used to classify those feature vectors extracted for the test examples. The raw SVM scores are also calibrated and converted into a proper posterior probability as in [27].

3. Between-class data augmentation and KL-divergence loss

In deep learning, data augmentation, which is to increase the data variation by altering the property of the genuine data, is an important method to improve performance of the task at hand. Techniques like adding background noise [28, 29], pitch shifting [29], and sample mixing [18, 30], have been proven to be useful for environmental sound recognition in general. Motivated by the work of Tokozume *et al.* [18], we pursue a *between-class* (BC) data augmentation approach that mixes two samples of different classes with a random factor to generate BC examples for network training.

Let $\mathbf{S}_1$ and $\mathbf{S}_2$ denote two samples of two different classes and let $\mathbf{y}_1$ and $\mathbf{y}_2$ denote their corresponding one-hot labels. A random factor $r \sim U(0, 1)$ is then generated and used to mix the two samples and their labels to create a new BC sample $\mathbf{S}^{\text{BC}}$ and its labels $\mathbf{y}^{\text{BC}}$:

$$\mathbf{S}^{\text{BC}} = \frac{r\mathbf{S}_1 + (1-r)\mathbf{S}_2}{\sqrt{r^2 + (1-r)^2}}, \quad (10)$$

$$\mathbf{y}^{\text{BC}} = r\mathbf{y}_1 + (1-r)\mathbf{y}_2. \quad (11)$$

Note that the BC label $\mathbf{y}^{\text{BC}}$ is no longer a one-hot label but still a proper probability distribution, which represents the amplitude of the constituents $\mathbf{S}_1$ and $\mathbf{S}_2$ in the between-class sample $\mathbf{S}^{\text{BC}}$. For example, mixing a *restaurant* scene and a *tubestation* scene with a factor $r = 0.3$ will result in a label {*restaurant*: 0.3, *tubestation*: 0.7}. Training a network with the BC sample $(\mathbf{S}^{\text{BC}}, \mathbf{y}^{\text{BC}})$, we expect the network's class probability distribution output $\hat{\mathbf{y}}$ to be as close to the $\mathbf{y}^{\text{BC}}$ as possible. Therefore, KL-divergence between $\mathbf{y}^{\text{BC}}$ and $\hat{y}$ is used as the network loss:

$$L = D_{\text{KL}}(\mathbf{y}^{\text{BC}} || \hat{\mathbf{y}}) = \sum_{c=1}^{C} y_i^{\text{BC}} \log \frac{y_c^{\text{BC}}}{\hat{y}_c}, \quad (12)$$

where C is the number of classes. Learning with between-class examples was shown to enlarge Fisher's criterion, i.e. the ratio of the between-class distance to the within-class variance [18].

4. Experiments

4.1. LITIS-Rouen dataset

The LITIS-Rouen dataset consists of 3026 examples of 19 scene categories [19]. Each class is specific to a location such as a train station or an open market. The audio recordings have a duration of 30 seconds and a sampling rate of 22050 Hz. The dataset has a total duration of 1500 minutes. We follow the training/testing splits in the seminal work [19] and report average performances over 20 splits.

4.2. Features

A 2-second audio segment, sampled at $fs = 22050$ Hz, was transformed into a log Mel-scale spectrogram with $M = 64$ Mel-scale filters in the frequency range from 50 Hz to Nyquist rate. A frame size of 50 ms with 50% overlap was used, resulting in $T = 80$ frames in total. Likewise, another image was produced for the auxiliary channel (cf. Section 2.1). All in all, we obtained a multi-channel image $\mathbf{S} \in \mathbb{R}^{M \times T \times K}$ where $K = 2$ denotes the number of channels.

Beside log Mel-scale spectrogram, we also studied log Gammatone spectrogram in this work. Repeating a similar feature extraction procedure using $M = 64$ Gammatone filters in replacement of the above-mentioned Mel-scale filters, we obtained a multi-channel log Gammatone spectrogram image for a 2-second audio segment.

4.3. Network parameters

The studied networks were implemented using *Tensorflow* framework [31]. We applied *dropout* [32] to the convolutional layers described in Section 2.2 with a dropout rate of 0.25. The GRU cells used to realize two bidirectional recurrent layers in Section 2.3 have their hidden size $H = 128$, and a dropout rate of 0.1 was commonly applied to their inputs and outputs. Both the spatial and temporal attention layers have the same size of 64. The networks were trained for 500 epochs with a minibatch size of 100. *Adam* optimizer [33] was used for network training with a learning rate of 10^{-4}.

Finally, the trade-off parameter C for the SVM classifier used for calibration was fixed at 0.1.

4.4. Baseline

In order to illustrate the efficiency of the recurrent layers with spatio-temporal attention pooling, we used the CNN block in Figure 1 as a deep CNN baseline. For this baseline, global max pooling was used after the last pooling layer to derive the final feature vector for classification. Other configuration settings were the same as for the proposed CRNN with spatio-temporal attention pooling.

4.5. Experimental results

Table 2 shows the classification accuracy obtained by the proposed network (referred to as Att-CRNN) and the CNN baseline. Note that the classification label of a 30-second recording was derived via aggregation of the classification results of its 2-second segments. To this end, probabilistic multiplicative fusion, followed by likelihood maximization were carried out similar to [2].

Overall, the proposed Att-CRNN outperforms the CNN baseline regardless of the features used, improving the accuracy on 2-second segment classification by 1.45% and 1.52% absolute using log Mel-scale and log Gammatone spectrograms, re-

Table 2: *Classification accuracy obtained by the proposed Att-CRNN and the CNN baseline.*

System	2s	30s
Att-CRNN (logMel)	93.78	98.53
Att-CRNN (logGam)	93.65	98.46
CNN baseline (logMel)	92.33	98.11
CNN baseline (logGam)	92.13	97.87

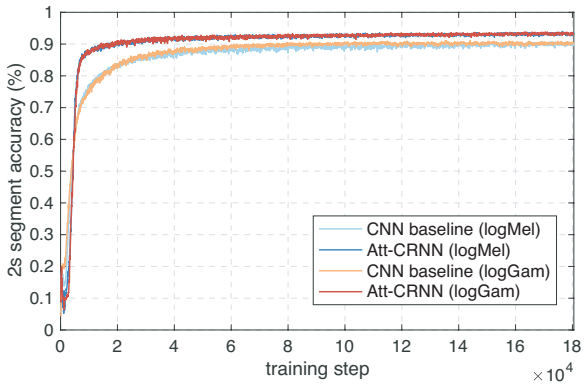

Figure 2: *Variation of the 2-second segment test accuracy during network training. The first cross-validation fold is shown as representative here.*

spectively. The better generalization of the proposed Att-CRNN over the CNN baseline can also be seen via patterns of their test accuracy curves during network training as shown in Figure 2 for the first cross-validation fold. In turn, the improvements on the 2-second segment classification led to 0.42% and 0.59% absolute gains on the 30-second recordings classification using log Mel-scale and log Gammatone spectrograms, respectively. These are equivalent to a relative classification error reduction of 22.22% and 27.70%, respectively.

4.6. Performance comparison with state-of-the-art

The experimental LITIS Rouen dataset has been extensively evaluated in literature. Table 3 provides a comprehensive comparison between the performance obtained by the proposed Att-CRNN and the CNN baseline to those reported in previous works in terms of overall accuracy, average F1-score, and average precision. Overall, this comparison shows that our presented systems obtain better performance than all other counterparts. On the one hand, despite being simple, the CNN baseline alone performs comparably well compared to the state-of-the-art system, i.e. Temporal Transformer CNN [10], likely due to the positive effect of the between-class data augmentation. On the other hand, the proposed Att-CRNN with log Mel-scale and log Gammatone spectrogram as features improves the accuracy by 0.47% and 0.40% over the state-of-the-art system, respectively. These accuracy gains are equivalent to a relative classification error reduction of 24.2% and 20.6%, respectively. Combining the classification results of the Att-CRNN on both types of feature using the probabilistic multiplicative aggregation [2] (i.e. Att-CRNN (fusion) in Table 3) further enlarges the margin up to 0.66% absolute gain on the overall accuracy, or 34.02% on relative classification error reduction.

Table 3: *Performance comparison on the LITIS Rouen dataset. We mark in bold where the performances achieved by our proposed systems are better than all those of previous works.*

System	Acc.	F1-score	Prec.
Att-CRNN (fusion)	**98.72**	**98.57**	**98.40**
Att-CRNN (logMel)	**98.53**	**98.39**	**98.20**
Att-CRNN (logGam)	**98.46**	**98.28**	**98.10**
CNN baseline (fusion)	**98.17**	**97.92**	**97.71**
CNN baseline (logMel)	**98.11**	**97.82**	**97.63**
CNN baseline (logGam)	97.87	97.59	97.39
Temp. Transformer CNN [10]	98.06	–	–
Temp. Transformer LSTM [10]	97.86	–	–
LSTM-SA [34]	97.92	–	–
LTE-RNN [2]	97.8	97.7	97.5
Temp. Transformer DNN [10]	97.40	–	–
CQT+HOG [35]	97.0	–	–
LTE-CNN [6]	96.6	96.5	96.3
Scene-LTE + Speech-LTE [36]	96.4	96.2	95.9
1D+2D+3D CNNs [9]	96.4	–	–
FisherHOG+ProbSVM [37]	96.0	–	–
Kernel PCA [38]	–	95.6	–
Convolutive NMF [38]	–	94.5	–
Sparse NMF [38]	–	94.1	–
HOG+SPD [39]	93.4	92.8	93.3
MFCC+DNN [40]	–	–	92.2
HOG [19]	–	–	91.7

5. Conclusions

This paper has presented an approach for audio scene classification using spatio-temporal attention pooling in combination with convolutional recurrent neural networks. The convolutional layers in this network are expected to learn invariant features from the input whose temporal dynamics are further encoded by bidirectional recurrent layers. Attention layers then learn attention weight vectors in the spatial and temporal dimensions from the recurrent output, which collectively construct a spatio-temporal attention mask able to weigh and pool the recurrent output into a single feature vector for classification. The proposed network was trained with between-class examples and KL-divergence loss. Evaluated on the LITIS Rouen dataset, the proposed method achieved good classification performance, outperforming a strong CNN baseline as well as the previously published state-of-the-art systems.

6. Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research and Specialist and High Performance Computing systems provided by Information Services at the University of Kent.

7. 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 Transactions on Audio, Speech, and Language Processing (TASLP)*, vol. 26, no. 2, pp. 379–393, 2018.

[2] H. Phan, P. Koch, F. Katzberg, M. Maass, R. Mazur, and A. Mertins, "Audio scene classification with deep recurrent neural networks," in *Proc. Interspeech*, 2017, pp. 3043–3047.

[3] S. Mun, S. Park, D. Han, and H. Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using SVM hyper-plane," in *Proc. DCASE Workshop*, 2017.

[4] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," in *Proc. DCASE Workshop*, 2017.

[5] Y. Han and K. Lee, "Convolutional neural network with multiple-width frequency-delta data augmentation for acoustic scene classification," DCASE2016 Challenge, Tech. Rep., September 2016.

[6] H. Phan, L. Hertel, M. Maass, P. Koch, R. Mazur, and A. Mertins, "Improved audio scene classification based on label-tree embeddings and convolutional neural networks," *IEEE/ACM Trans. on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 6, pp. 1278–1290, 2017.

[7] M. Valenti, A. Diment, G. Parascandolo, S. Squartini, and T. Virtanen, "DCASE 2016 acoustic scene classification using convolutional neural networks," Detection and Classification of Acoustic Scenes and Events 2016, Tech. Rep., 2016.

[8] H. Phan, P. Koch, L. Hertel, M. Maass, R. Mazur, and A. Mertins, "CNN-LTE: a class of 1-X pooling convolutional neural networks on label tree embeddings for audio scene classification," in *Proc. ICASSP*, 2017.

[9] Y. Yin, R. R. Shah, and R. Zimmermann, "Learning and fusing multimodal deep features for acoustic scene categorization," in *Proc. ACMMM*, 2018, pp. 1892–1900.

[10] T. Zhang, K. Zhang, and J. Wu, "Temporal transformer networks for acoustic scene classification," in *Proc. Interspeech*, 2018, pp. 1349–1353.

[11] T. Nguyen and F. Pernkopf, "Acoustic scene classification using a convolutional neural network ensemble and nearest neighbor filters," in *Proc. DCASE Workshop*, 2018.

[12] T. H. Vu and J.-C. Wang, "Acoustic scene and event recognition using recurrent neural networks," Detection and Classification of Acoustic Scenes and Events 2016, Tech. Rep., 2016.

[13] J. Guo, N. Xu, L.-J. Li, and A. Alwan, "Attention based CLDNNs for short-duration acoustic scene classification," in *Proc. Interspeech*, 2017.

[14] H. Phan, O. Y. Chén, P. Koch, L. Pham, I. McLoughlin, A. Mertins, and M. De Vos, "Beyond equal-length snippets: How long is sufficient to recognize an audio scene?" in *Proc. 2019 AES Conference on Audio Forensics*, 2019.

[15] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. EMNLP*, 2015, pp. 1412–1421.

[16] E. Çakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 5, no. 6, pp. 1291–1303, 2017.

[17] H. Phan, O. Y. Chén, P. Koch, L. Pham, I. McLoughlin, A. Mertins, and M. D. Vos, "Unifying isolated and overlapping audio event detection with multi-label multi-task convolutional recurrent neural networks," in *Proc. ICASSP*, 2019.

[18] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," in *Proc. ICLR*, 2018.

[19] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene classification," *IEEE/ACM Trans. Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 142–153, 2015.

[20] R. Martin, "Noise power spectral density estimation based on optimal smoothing and minimum statistics," *IEEE Trans. on Speech and Audio Processing*, vol. 9, no. 5, pp. 504–512, 2001.

[21] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[23] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. EMNLP*, 2014, pp. 1724–1734.

[24] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[25] C. Zhu, X. Tan, F. Zhou, X. Liu, K. Yue, E. Ding, and Y. Ma, "Fine-grained video categorization with redundancy reduction attention," in *Proc. ECCV*, 2018, pp. 139–155.

[26] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. COLT*, 1992, pp. 144–152.

[27] J. Platt, *Advances in Large Margin Classifiers*. MIT Press, 1999, ch. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods.

[28] Y. Tokozume and T. Harada, "Learning environmental sounds with end-to-end convolutional neural network," in *Proc. ICASSP*, 2017, pp. 2721–2725.

[29] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[30] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," *arXiv preprint arXiv:1805.07319*, 2018.

[31] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv:1603.04467*, 2016.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research (JMLR)*, vol. 15, pp. 1929–1958, 2014.

[33] D. P. Kingma and J. L. Ba, "Adam: a method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–13.

[34] T. Zhang, K. Zhang, and J. Wu, "Data independent sequence augmentation method for acoustic scene classification," in *Proc. Interspeech*, 2018.

[35] J. Ye, T. Kobayashi, N. Toyama, H. Tsuda, and M. Murakawa, "Acoustic scene classification using efficient summary statistics and multiple spectro-temporal descriptor fusion," *Applied Sciences*, vol. 8, p. 1363, 2018.

[36] H. Phan, L. Hertel, M. Maass, P. Koch, and A. Mertins, "Label tree embeddings for acoustic scene classification," in *Proc. ACMMM*, 2016, pp. 486–490.

[37] J. Ye, T. Kobayashi, M. Murakawa, and T. Higuchi, "Acoustic scene classification based on sound textures and events," in *Proc. ACM Multimedia*, 2015, pp. 1291–1294.

[38] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Acoustic scene classification with matrix factorization for unsupervised feature learning," in *Proc. ICASSP*, 2016, pp. 6445–6449.

[39] V. Bisot, S. Essid, and G. Richard, "HOG and subband power distribution image features for acoustic scene classification," in *Proc. EUSIPCO*, 2015, pp. 719–723.

[40] Y. Petetin, C. Laroche, and A. Mayoue, "Deep neural networks for audio scene recognition," in *Proc. EUSIPCO*, 2015, pp. 125–129.