Interspeech 2018
2-6 September 2018, Hyderabad

Data independent sequence augmentation method for acoustic scene classification

Teng Zhang[1], *Kailai Zhang*[2], *Ji Wu*[3]

[123]Multimedia Signal and Intelligent Information Processing Lab
Department of Electronic Engineering, Tsinghua University, Beijing, P.R.China
zhangteng1887@gmail.com, zhang-kl13@tsinghua.org.cn, wuji_ee@mail.tsinghua.edu.cn

Abstract

Augmenting datasets by transforming inputs in a way such as vocal tract length perturbation (VTLP) is a crucial ingredient of the state of the art methods for speech recognition tasks. In contrast to speech, sounds coming from realistic environments have no speaker to speaker variations. Thus VTLP is invalid for acoustic scene classification tasks. This paper investigates a novel sequence augmentation method for long short-term memory (LSTM) acoustic modeling to deal with data sparsity in acoustic scene classification tasks. The audio sequences are randomly rearranged and concatenated during training, but at test time, a prediction is made by the original audio sequence. The rearrangement is well-designed to adapt to the long short-term dependency in LSTM models. Experiments on acoustic scene classification task show performance improvements of the proposed methods. The classification errors in LITIS ROUEN dataset and DCASE2016 dataset are reduced by 18.1% and 6.4% relatively.

Index Terms: acoustic scene classification, sequence augmentation, long short-term memory

1. Introduction

Acoustic modeling based on neural networks (NN) has established excellent performance for acoustic scene classification in recent years [1][2][3][4][5][6]. However, despite the strong modeling capabilities of these NN structures, the performance is severely limited by the sparse training data. For instance, LITIS Rouen dataset [7], which is the largest dataset publicly available for the task, contains only 1500 minutes of acoustic scene recordings.

In supervised classification problems, a classifier can only be learned using observed training data and their labels. When the training data is sparse, the classifier will have poor classification invariance and encounter severe over-fitting problems. Unobserved data can be introduced using data augmentation methods. Ideally, the combination of observed and unobserved data should denote a smooth distribution to facilitate the training of classifiers [8]. Under this condition, data augmentation based on label-preserving transformations can help to alleviate this problem [9]. Label-preserving transformations generate samples that preserve the class labels.

In computer vision, data augmentation methods are often utilized to generate additional data, such as generating image translations and horizontal reflections [10], extending image crops with extra pixels [11], color casting, vignetting, lens distortion [12], etc [13][14]. Although some studies treated audio spectrograms as natural images in acoustic scene classification tasks, the time-frequency structure of spectrograms has a clear physical meaning which is completely different from natural images. Thus data augmentation methods in computer vision are not applicable for environmental sounds. In speech recognition, a strategy based on VTLP [15] was proposed, and experiments on the TIMIT database showed decent improvements. A statistical voice conversion named stochastic feature mapping (SFM) [9] was also investigated as an approach. However, both VTLP and SFM are speaker-adaptive approaches and rely on speaker to speaker variations in speech signals. Sounds coming from realistic environments have no speaker to speaker variations. Thus these speaker-adaptive approaches do not work in this case. In general, compared to computer vision and speech recognition, data augmentation for acoustic scene classification is less known and needs more studies.

Environmental sounds can be affected by a wide variety of factors, these variabilities are difficult to generate via simple transformations. Thus we turn our attention to classification models. The state-of-the-art result on LITIS Rouen dataset was obtained using an LSTM model in [5], where audio spectrograms were processed as a sequence of feature vectors. In this paper, we proceed from LSTM models and propose a data independent sequence augmentation method for acoustic scene classification tasks. LSTM allows information to be stored across arbitrary time lags, and error signals to be carried far back in time. However, when the back-propagation through time (BPTT) [16] is conducted for LSTM with sigmoid functions, the problem of gradient vanishing or exploding appears since an audio recording can have a large temporal depth. Previous work [17] use truncated BPTT to avoid this problem. The truncated BPTT stops the BPTT after k time steps, where k is a hand-defined hyper-parameter. For our case, environmental sound such as an audio recording in "restaurant" scene, consists of different acoustic patterns including the voice, birds, the collision of dishes, etc. When LSTM encounters a new pattern, the forget gate in LSTM will reset its memory blocks and forget all previously encountered patterns. This is a problem when the training data is sparse. If a pattern appears in the starting position of an audio sequence and occurs only once in the dataset, this pattern will never be learned by LSTM models. Our solution to this problem is a two-stage sequence augmentation method containing a random segmentation and rearrangement of input sequences, and the following concatenation of these subsequences. The segmentation and rearrangement are designed here to guarantee that all acoustic patterns appear in the tail of audio sequences and can be fully learned. By the concatenation operation, we aim to maintain the possibly existing connection between patterns.

The rest of the paper is organized as follows. In Section 2, we describe implementation details of the proposed sequence augmentation method. Next, we discuss how to use this method to train an LSTM model in Section 3. Then we conduct several experiments and evaluate the performance of the proposed method in Section 4. At last, we conclude this paper and present

10.21437/Interspeech.2018-1250

our future work in Section 5.

2. Sequence Augmentation Method

In this section, we describe implementation details of the sequence augmentation method. The input audio signal is first transformed to a sequence of feature vectors using Short-time Fourier Transform (STFT) [18], the output spectrogram can be represented as $\boldsymbol{X}_{1...T} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$. T is the sequence length, and N is the dimension of each vector $\boldsymbol{x}$. New samples are arbitrarily generated using specified permutations of $\boldsymbol{X}$ as Fig.1.

The sequence augmentation mechanism is split into three parts. In order of computation, first, the input audio sequence is split into subsequences with equal lengths. Then, the subsequences are rearranged to get new subsequences with an arbitrary turn. Finally, the rearranged subsequences are concatenated together to generate a new sequence. More details will be discussed in the following sections.

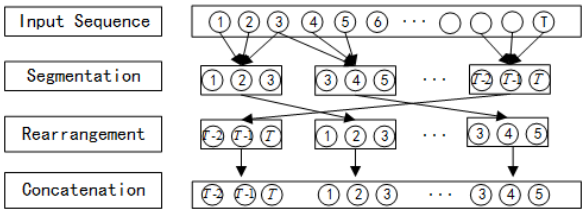


Figure 1: *Sequence augmentation procedure. The segmentation length here is 3.*

2.1. Sequence Segmentation

This step takes an input audio sequence $\boldsymbol{X} \in \mathbb{R}^{N \times T}$ with length T and N channels, and outputs a new set of subsequences $\mathcal{Y} = \{\boldsymbol{Y}_1, \boldsymbol{Y}_2, ..., \boldsymbol{Y}_{T/L}\}$. L is a hand-defined hyper-parameter, representing the length of subsequences. $\boldsymbol{Y}_i$ is a subsequence split from $\boldsymbol{X}$ and can be represented as $\{\boldsymbol{x}_{t_i}, \boldsymbol{x}_{t_i+1}, ..., \boldsymbol{x}_{t_i+L-1}\}$, where t_i is the starting position of the ith subsequence.

To perform meaningful segmentation of different acoustic patterns which have been described in Section 1, L should be well-designed. Larger L maintains more pattern information but gives fewer variations. The best configuration is the tradeoff between information integrity and diversity. Experiments are carried out in Section 4.4 to show the influence of L.

A direct segmentation method is to split the input sequence into T/L subsequences without overlap. For some situations, such as when $L = T/2$, this method can only generate twice as many new samples, which increases limited variations. At the same time, t_i can be selected uniformly from 1 and $T - L$. Theoretically, we can get $T - L$ new subsequences using this method. In this case, we randomly preserve T/L subsequences of them during each generation, to maintain consistency of sequence length after the following concatenation step.

2.2. Sequence Rearrangement

Rearrangement operation is designed to move the previous pattern to the tail. Thus all acoustic patterns can be fully learned by LSTM models, as described in Section 1. A set of subsequences $\mathcal{Y}$ is rearranged with a random order in this step, the output can be represented in a form $\hat{\mathcal{Y}} = \{\boldsymbol{Y}_2, \boldsymbol{Y}_{T/L}, ..., \boldsymbol{Y}_1\}$. The order is different during each generation. Thus we can generate an infinite number of new samples, regardless of the value of L defined in Section 2.1.

2.3. Sequence Concatenation

Instead of selecting one subsequence from $\hat{\mathcal{Y}}$ directly, this step concatenates T/L subsequences in $\hat{\mathcal{Y}}$ together to generate a new sequence, which can be represented as $\hat{\boldsymbol{X}} = \{\boldsymbol{x}_{t_2}, \boldsymbol{x}_{t_2+1}, ..., \boldsymbol{x}_{t_2+L-1}, \boldsymbol{x}_{t_{T/L}}, \boldsymbol{x}_{t_{T/L}+1}, ..., \boldsymbol{x}_{t_1+L-1}\}$.

The concatenation operation is designed to maintain the possibly existing connection between subsequences, which seems not essential especially when L is large enough. However, as discussed in Section 2.1, L should not be too small or too large. In this case, the concatenation step should be involved in the procedure to improve the augmentation performance.

The combination of the segmentation, rearrangement and concatenation steps form a complete sequence augmentation method, which can generate new sequences to increase the data variations and improve the following classification invariance.

3. Sequence Augmentation for LSTM

In this section, we first briefly describe the LSTM structure in [19]. When $\boldsymbol{X}$ is given to an LSTM model, the processing procedure inside a cell can be described as following equations:

$$
\begin{aligned}
&LSTM : \boldsymbol{x}_t, \boldsymbol{h}_{t-1}, \boldsymbol{c}_{t-1} \rightarrow \boldsymbol{h}_t, \boldsymbol{c}_t \\
&\begin{pmatrix} \boldsymbol{i}_t \\ \boldsymbol{f}_t \\ \boldsymbol{o}_t \\ \boldsymbol{c}_{\_} \end{pmatrix} = \begin{pmatrix} sigm \\ sigm \\ sigm \\ tanh \end{pmatrix} \boldsymbol{L}_{M+N,4N} \begin{pmatrix} \boldsymbol{x}_t \\ \boldsymbol{h}_{t-1} \end{pmatrix} \\
&\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{c}_{\_} \\
&\boldsymbol{h}_t = \boldsymbol{o}_t \odot tanh(\boldsymbol{c}_t)
\end{aligned} \tag{1}
$$

where $\boldsymbol{h}_t$ is an N-dimensional hidden state in timestep t, $\boldsymbol{L}_{n,m} : R^n \rightarrow R^m$ is a biased linear mapping $x \rightarrow Wx + b$ for some W and b, the symbol $\odot$ represents element-wise multiplication.

As a practice, the final hidden state is then fed to several fully connected layers and a softmax layer as shown in Fig.2. The classification loss of this model is given by Eq.2, where n is the number of audios, k is the number of categories, $\boldsymbol{W}_{lstm}$ is the LSTM parameters, $\boldsymbol{W}_{fc}$ is the fully connected parameters, $\boldsymbol{o}$ is the category labels and $\boldsymbol{p}$ is the probability distribution produced by Fig.2.

$$
\epsilon = \sum_{i=1}^{n} \sum_{j=1}^{k} \boldsymbol{o}_{ij} \cdot log(\boldsymbol{p}_{ij}) + \lambda(\| \boldsymbol{W}_{lstm} \|^2 + \| \boldsymbol{W}_{fc} \|^2) \tag{2}
$$

With these notations, the sequence augmentation algorithm for training an LSTM classification model can be summarized as follows.

The main difference with standard LSTM is in step 6 that new training data is generated using sequence augmentation method for every epoch during the training procedure. However, during the test phase, the original test data is used to get a prediction.

4. Experimental Evaluation

In this section, we employ LITIS ROUEN dataset [7] and DCASE2016 dataset [20] to conduct acoustic scene classification experiments.

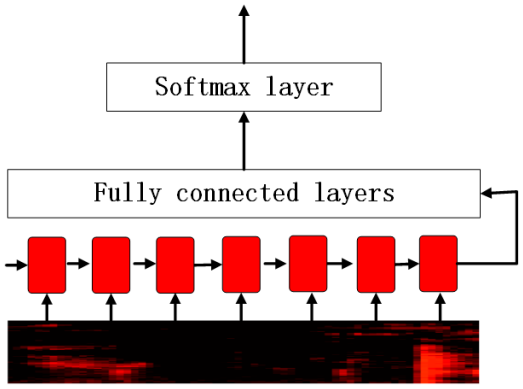


Figure 2: *LSTM classification model.*

Algorithm 1 Sequence augmentation for LSTM classification model

Input: training data $\mathcal{X}$, testing data $\mathcal{Z}$, initial weights $\boldsymbol{W}_{lstm}$, $\boldsymbol{W}_{fc}$
During train:
for each $epoch \in [1, 100]$ **do**
  for each $i \in [1, n]$ **do**
    select $\boldsymbol{X} \in \mathcal{X}$ randomly
    generate $\hat{\boldsymbol{X}}$ as Fig.1
    forward propagate $\hat{\boldsymbol{X}}$ as Fig.2 and get ϵ using Eq.2
    backward propagate and update $\boldsymbol{W}_{lstm}$, $\boldsymbol{W}_{fc}$
During test:
for each $\boldsymbol{Z} \in \mathcal{Z}$ **do**
  forward propagate $\boldsymbol{Z}$ as Fig.2
  get the prediction $\boldsymbol{p}$ using Eq.2

Details of these datasets are listed as follows.

- *LITIS ROUEN dataset*: This is the largest publicly available dataset for ASC to the best of our knowledge. The dataset contains about 1500 minutes of acoustic scene recordings belonging to 19 classes. Each audio recording is divided into 30-second examples without overlapping, thus obtain 3026 examples in total. The sampling frequency of the audio is 22050 Hz. The dataset is provided with 20 training/testing splits. In each split, 80% of the examples are kept for training and the other 20% for testing. We use the mean average accuracy over the 20 splits as the evaluation criterion.
- *DCASE2016 dataset*: The dataset is released as Task 1 of the DCASE2016 challenge. We use the development data in this paper. The development data contains about 585 minutes of acoustic scene recordings belonging to 15 classes. Each audio recording is divided into 30-second examples without overlapping, thus obtain 1170 examples in total. The sampling frequency of the audio is 44100 Hz. The dataset is divided into 4 folds. Our experiments obey this setting, and the average performance will be reported.

4.1. Audio Pre-procession

For both datasets, the audio signal is first transformed using Short-time Fourier Transform with a frame length of 1024 and a frameshift of 220, the number of frequency filters is set to 64. For both datasets, the examples are 30 seconds long. In the data preprocessing step, we first divide the 30-second examples into 1-second clips with 50% overlap. Then each clip is processed using LSTM in Fig.2. The classification results of all these clips will be averaged to get an ensemble result for the 30-second examples.

4.2. Hyper-parameters and Evaluation

The size of input audio sequences is 64×128, where the sequence length is 128 and the dimension of feature vectors is 64. For LSTM models, we use the number of LSTM cells as 128, LSTM layers as 1, the fully connected layers can be summarized as $128 \times 128 \times 19(15)$. For DCASE2016 dataset, we use the dropout rate of 0.5. For all these models, the learning rate is 0.001, l_2 weight is $1e^{-4}$, training is done using the Adam [21] update method and is stopped after 100 training epochs.

In order to compute the results for each training-test split, we use the classification error over all classes. The final classification error is its average value over all splits.

4.3. Results of Sequence Augmentation Method

According to Section 2, there are three variables in the proposed sequence augmentation method, which can be listed as follows:

- *Segmentation Length*: L is a hand-defined hyper-parameter representing the length of subsequences.
- *Segmentation Method*: For the segmentation method, we have two options. The first is to split the input sequence without overlap, which is named as Const Segmentation. The other is to select the starting position of subsequences uniformly from 1 and $T - L$, which is named as Random Segmentation.
- *Concatenation*: Concatenation is also an optional step in the method.

We begin with experiments where the variables above are pre-set empirically. L is set to 64, which is half of the sequence length. For the segmentation method, we use the random version. The concatenation step is involved in the method.

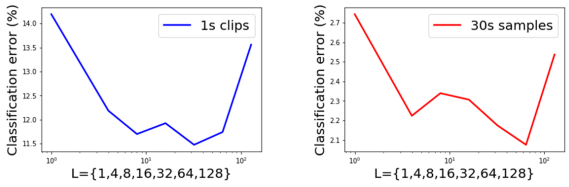
The results of these experiments are shown in Table 1. On LITIS Rouen dataset, our approach of vanilla LSTM model performs much better than other models such as CNN, DNN and Nonnegative Matrix Factorization (NMF) [22] and results in state-of-the-art performance. However, on DCASE2016 dataset, LSTM model is the worst model when compared with CNN, DNN and NMF, this can be attributed to the lack of training data for the DCASE2016 dataset. To test and verify the superior performance of our sequence augmentation method, two more LSTM experiments are conducted on both datasets. After applying the sequence augmentation method, our LSTM model achieves performance gains on both datasets because of the variations introduced by generated samples. For 1-second clips and 30-second samples on LITIS Rouen dataset, as described in Section 4.1, our approach obtains relatively 14.0% and 18.1% reductions on classification error when compared with vanilla LSTM model. And on DCASE2016 dataset, the relative error reductions are 2.7% and 6.4%.

4.4. Influence of Segmentation Length

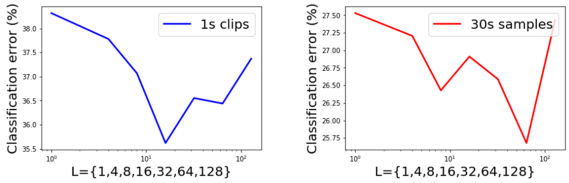
We now test the influence of segmentation length. As discussed in Section 2.1, L is closely related to the information integrity and diversity of generated samples. In this section, L varies in the set $\{1, 4, 8, 16, 32, 64, 128\}$. When $L = 1$, the order of input sequences is completely upset. And when $L = 128$, no

Table 1: *Acoustic scene classification errors using sequence augmentation and LSTM method. SA represents the sequence augmentation option.*

Model	LITIS Rouen (%)		DCASE2016 (%)	
	1s clips	30s samples	1s clips	30s samples
vanilla LSTM	13.6	2.54	37.4	27.4
LSTM+SA	**11.7**	**2.08**	**36.4**	**25.7**
CNN-Mel [1]	-	-	-	24.0
MFCC-GMM [20]	-	-	-	27.5
DNN-CQT [2]	-	3.4	-	21.9
Sparse-NMF [2]	-	5.4	-	**17.3**
DNN-Mel [3]	-	-	-	23.6
RNN-Gam [5]	-	3.4	-	-
CNN-Gam [6]	-	4.2	-	-



(a) *LITIS ROUEN*



(b) *DCASE2016*

Figure 3: *Experiments of different segmentation lengths.*

new samples can be generated, which degrades into the vanilla LSTM situation.

The results of these experiments can be seen in Fig.3. On both datasets, performance improves with the increase in segmentation length, except for $L = 128$. When $L = 1$, sequence augmentation method even reduces the performance, compared to vanilla LSTM model. As discussed in Section 2.1, Larger L tends to maintain more pattern information but give fewer variations. From the experimental results, the difference of variations introduced by different L is small, but the pattern information suffers a severe loss with a small L. Thus the best results of 30-second samples on both datasets are obtained when $L = 64$. This is the largest L in our experiments, except for $L = 128$ which is the vanilla LSTM situation.

4.5. Influence of Segmentation Method and Concatenation

In this section, we test the influence of the remaining two variables: segmentation method and concatenation. Experiments using random segmentation and const segmentation methods are first conducted respectively, these two methods have been defined in Section 4.3. Then the necessity of concatenation step is tested using two additional experiments. During the experiments about segmentation methods, L is set to be 64, and the concatenation step is involved. During the concatenation related experiments, L is also set to be 64, and the random segmentation method is used.

Table 2: *Acoustic scene classification errors with different segmentation methods and concatenation options. Seg is the shortening of Segmentation, Concat is the shortening of Concatenation.*

Model	LITIS Rouen (%)		DCASE2016 (%)	
	1s clips	30s samples	1s clips	30s samples
Const Seg	13.4	2.36	36.8	26.2
Random Seg	**11.7**	**2.08**	**36.4**	**25.7**
no Concat	13.3	2.27	36.7	26.6
with Concat	**11.7**	**2.08**	**36.4**	**25.7**

The results of these experiments are shown in Table 2. In the experiments using different segmentation methods, the results of random segmentation method are better than const segmentation method on both datasets. As discussed in Section 2.1, random segmentation method allows us to generate an infinite number of new samples, which can increase more pattern variations than const segmentation method and our experiments verify this. In the concatenation related experiments, augmentation method with concatenation step performs better than the method without concatenation step on both datasets. Concatenation step is involved to maintain the long-term pattern information and the connection between different patterns. Our experiments show the necessity of this step.

5. Conclusions

In this paper we introduce a new sequence augmentation method for LSTM classification models. Unlike data augmentation in computer vision and speech recognition, this method is able to rearrange different patterns appearing in a sequence and overcome the forgetting mechanism in LSTM when the training data is sparse. In our experiments, we see significant performance improvements using sequence augmentation method on two acoustic scene classification datasets. On LITIS ROUEN dataset, our approach of sequence augmentation based LSTM model is able to perform significantly better than the state-of-the-art result and obtains 2.08% on classification error. We also conduct several experiments to show the influence of the segmentation length, segmentation method and concatenation option during the augmentation procedure, and get the best configuration of these variables. This method is data-independent and useful for other sequence modeling tasks, this is possible to be extended to text and video classification tasks.

6. Acknowledgment

This work is partly funded by National Natural Science Foundation of China (Grant No: 61571266)

7. References

[1] D. Battaglino, L. Lepauloux, N. Evans, F. Mougins, and F. Biot, "Acoustic scene classification using convolutional neural networks," *DCASE2016 Challenge, Tech. Rep.*, 2016.

[2] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Feature learning with matrix factorization applied to acoustic scene classification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1216–1229, 2017.

[3] Q. Kong, I. Sobieraj, W. Wang, and M. Plumbley, "Deep neural network baseline for dcase challenge 2016," *Proceedings of DCASE 2016*, 2016.

[4] Y. Petetin, C. Laroche, and A. Mayoue, "Deep neural networks for audio scene recognition," in *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE, 2015, pp. 125–129.

[5] H. Phan, P. Koch, F. Katzberg, M. Maass, R. Mazur, and A. Mertins, "Audio scene classification with deep recurrent neural networks," *arXiv preprint arXiv:1703.04770*, 2017.

[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 Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1278–1290, 2017.

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

[8] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov, "Good semi-supervised learning that requires a bad gan," in *Advances in Neural Information Processing Systems*, 2017, pp. 6513–6523.

[9] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1469–1477, 2015.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[11] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," *arXiv preprint arXiv:1312.5402*, 2013.

[12] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, "Deep image: Scaling up image recognition," *arXiv preprint arXiv:1501.02876*, vol. 7, no. 8, 2015.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *european conference on computer vision*. Springer, 2014, pp. 346–361.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich *et al.*, "Going deeper with convolutions." Cvpr, 2015.

[15] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013.

[16] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[17] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[18] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[19] W. Zaremba and I. Sutskever, "Learning to execute," *arXiv preprint arXiv:1410.4615*, 2014.

[20] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference*, vol. 2016, 2016.

[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.