BIDIRECTIONAL LSTM-HMM HYBRID SYSTEM FOR POLYPHONIC SOUND EVENT DETECTION

Tomoki Hayashi[1], *Shinji Watanabe*[2], *Tomoki Toda*[1], *Takaaki Hori*[2], *Jonathan Le Roux*[2], *Kazuya Takeda*[1]

[1]Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi 464-8603, Japan
[2]Mitsubishi Electric Research Laboratories (MERL), 201 Broadway, Cambridge, MA 02139, USA

hayashi.tomoki@g.sp.m.is.nagoya-u.ac.jp,
{takeda,tomoki}@is.nagoya-u.ac.jp, {watanabe,thori,leroux}@merl.com

ABSTRACT

In this study, we propose a new method of polyphonic sound event detection based on a Bidirectional Long Short-Term Memory Hidden Markov Model hybrid system (BLSTM-HMM). We extend the hybrid model of neural network and HMM, which achieved state-of-the-art performance in the field of speech recognition, to the multi-label classification problem. This extension provides an explicit duration model for output labels, unlike the straightforward application of BLSTM-RNN. We compare the performance of our proposed method to conventional methods such as non-negative matrix factorization (NMF) and standard BLSTM-RNN, using the DCASE2016 task 2 dataset. Our proposed method outperformed conventional approaches in both monophonic and polyphonic tasks, and finally achieved an average F1 score of 67.1 % (error rate of 64.5 %) on the event-based evaluation, and an average F1-score of 76.0 % (error rate of 50.0 %) on the segment-based evaluation.

Index Terms— Polyphonic Sound Event Detection, Bidirectional Long Short-Term Memory, Hidden Markov Model, multi-label classification

1. INTRODUCTION

Sounds include important information for various applications such as life-log, environmental context understanding, and monitoring system. To realize these applications, It is necessary to extract internal information automatically from not only speech and music, which have been studied for long time, but also other various types of sounds.

Recently, studies related to sound event detection (SED) attracted much interest to aim for understanding various sounds. The objective of SED systems is to identify the beginning and end of sound events and to identify and label these sounds. SED is divided into two scenarios, monophonic and polyphonic. Monophonic sound event detection is under the restricted condition that the number of simultaneous active events is only one. On the other hand, in polyphonic sound event detection, the number of simultaneous active events is unknown. We can say that polyphonic SED is a more realistic task than monophonic SED because in real situations, it is more likely that several sound events may happen simultaneously, or multiple sound events are overlapped.

The most typical approach to SED is to use a Hidden Markov Model (HMM), where the emission probability distribution is represented by Gaussian Mixture Models (GMM-HMM), with Mel Frequency Cepstral Coefficients (MFCCs) as features [1, 2]. Another approach is to utilize Non-negative Matrix Factorization (NMF) [3, 4, 5]. In the NMF approaches, a dictionary of basis vectors is learned by decomposing the spectrum of each single sound event into the product of a basis matrix and an activation matrix, then combining the basis matrices. The activation matrix at test time is estimated using the basis vector dictionary. More recently, methods based on neural networks have achieved good performance for sound event classification and detection using acoustic signals [7, 8, 9, 10, 11, 12]. In the first two of these studies [7, 8], the network was trained to be able to deal with a multi-label classification problem for polyphonic sound event detection. Although these networks provide good performance, they do not have an explicit duration model for the output label sequence, and the actual output needs to be smoothed with careful thresholding to achieve the best performance.

In this paper, we propose a new polyphonic sound event detection method based on a hybrid system of bidirectional long short-term memory recurrent neural network and HMM (BLSTM-HMM). The proposed hybrid system is inspired by the BLSTM-HMM hybrid system used in speech recognition [13, 14, 15, 16], where the output duration is controlled by an HMM on top of a BLSTM network. We extend the hybrid system to polyphonic SED, and more generally to the multi-label classification problem. Our approach allows the smoothing of the frame-wise outputs without post-processing and does not require thresholding.

The rest of this paper is organized as follows: Section 2 presents various types of recurrent neural networks and the concept of long short term memory. Section 3 describes our proposed method in detail. Section 4 describes the design of our experiment and evaluates the performance of the proposed method and conventional methods. Finally, we conclude this paper and discuss future work in Section 5.

2. RECURRENT NEURAL NETWORKS

2.1. Recurrent Neural Network

A Recurrent Neural Network (RNN) is a layered neural network which has a feedback structure. The structure of a simple RNN is shown in Fig. 1. In comparison to feed-forward layered neural networks, RNNs can propagate prior time information forward to the current time, enabling them to understand context information in a sequence of feature vectors. In other words, the hidden layer of an RNN serves as a memory function.

An RNN can be described mathematically as follows. Let us denote a sequence of feature vectors as $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$. An RNN with a hidden layer output vector $\mathbf{h}_t$ and output layer one $\mathbf{y}_t$ are

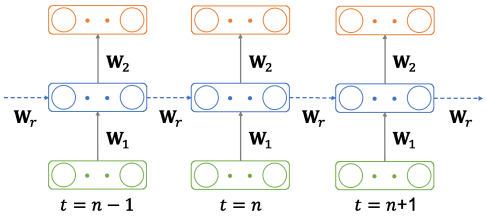

Figure 1: Recurrent Neural Network

calculated as follows:

$$\mathbf{h}_t = f(\mathbf{W}_1\mathbf{x}_t + \mathbf{W}_r\mathbf{h}_{t-1} + \mathbf{b}_1), \quad (1)$$

$$\mathbf{y}_t = g(\mathbf{W}_2\mathbf{h}_t + \mathbf{b}_2), \quad (2)$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ represent the input weight matrix and bias vector of the i-th layer, respectively, $\mathbf{W}_r$ represents a recurrent weight matrix, and f and g represent activation functions of the hidden layer and output layer, respectively.

2.2. Bidirectional Recurrent Neural Network

A Bidirectional Recurrent Neural Network (BRNN) [13, 17] is a layered neural network which not only has feedback from the previous time period, but also from the following time period. The structure of a BRNN is shown in Fig. 2. The hidden layer which connects to the following time period is called the *forward layer*, while the layer which connects to the previous time period is called the *backward layer*. Compared with conventional RNNs, BRNNs can propagate information not only from the past but also from the future, and therefore have the ability to understand and exploit the full context in an input sequence.

2.3. Long Short-Term Memory RNNs

One major problem with RNNs is that they cannot learn context information over long stretches of time because of the so-called *vanishing gradient* problem [19]. One effective solution to this problem is to use Long Short-Term Memory (LSTM) architectures [20, 21]. LSTM architectures prevent vanishing gradient issues and allow the memorization of long term context information. As illustrated in Fig. 3, LSTM layers are characterized by a *memory cell* $\mathbf{s}_t$, and three gates: 1) an *input gate* $\mathbf{g}_t^I$, 2) a *forget gate* $\mathbf{g}_t^F$, and 3) an *output gate* $\mathbf{g}_t^O$. Each gate $\mathbf{g}^*$ has a value between 0 and 1. The value 0 means that the gate is closed, while the value 1 means that the gate is open. In an LSTM layer, the hidden layer output $\mathbf{h}_t$ in

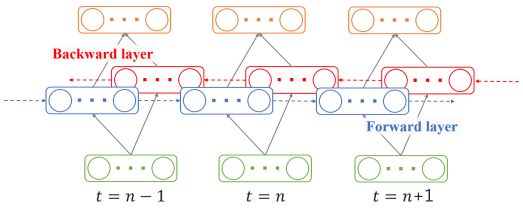

Figure 2: Bidirectional Recurrent Neural Network

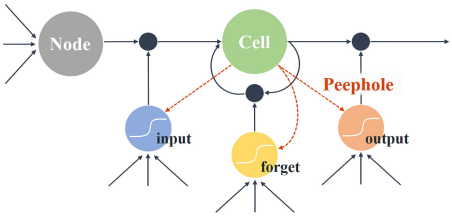

Figure 3: Long Short-Term Memory

Eq. 1 is replaced by the following equations:

$$\mathbf{g}_t^I = \sigma(\mathbf{W}^I\mathbf{x}_t + \mathbf{W}_r^I\mathbf{h}_{t-1} + \mathbf{s}_{t-1}), \quad (3)$$

$$\mathbf{g}_t^F = \sigma(\mathbf{W}^F\mathbf{x}_t + \mathbf{W}_r^F\mathbf{h}_{t-1} + \mathbf{s}_{t-1}), \quad (4)$$

$$\mathbf{s}_t = \mathbf{g}_t^I \odot f(\mathbf{W}_1\mathbf{x}_t + \mathbf{W}_r\mathbf{h}_{t-1} + \mathbf{b}_1) + \mathbf{g}_t^F \odot \mathbf{s}_{t-1}, \quad (5)$$

$$\mathbf{g}_t^O = \sigma(\mathbf{W}^O\mathbf{x}_t + \mathbf{W}_r^O\mathbf{h}_{t-1} + \mathbf{s}_{t-1}), \quad (6)$$

$$\mathbf{h}_t = \mathbf{g}_t^O \odot \tanh(\mathbf{s}_t), \quad (7)$$

where $\mathbf{W}$ and $\mathbf{W}_r$ denote input weight matrices and recurrent weight matrices, respectively, subscripts I, F, and O represent the input, forget, and output gates, respectively, $\odot$ represents point-wise multiplication, and σ represents a logistic sigmoid function.

2.4. Projection Layer

Use of a projection layer is a technique which reduces the computational complexity of deep recurrent network structures, which allows the creation of very deep LSTM networks [14, 15]. The architecture of an LSTM-RNN with a projection layer (LSTMP-RNN) is shown in Fig. 4. The projection layer, which is a linear transformation layer, is inserted after an LSTM layer, and the projection layer outputs feedback to the LSTM layer. With the insertion of a projection layer, the hidden layer output $\mathbf{h}_{t-1}$ in Eqs. 3-6 is replaced with $\mathbf{p}_{t-1}$ and the following equation is added:

$$\mathbf{p}_t = \mathbf{W}_I\mathbf{h}_t, \quad (8)$$

where $\mathbf{W}_l$ represents a projection weight matrix, and $\mathbf{p}_t$ represents a projection layer output.

3. PROPOSED METHOD

3.1. Data generation

There are only 20 clean samples per sound event in the DCASE2016 task 2 training dataset. Since this is not enough data to train a deeply structured recurrent neural network, we synthetically generated our own training data from the provided data. The training data generation procedure is as follows: 1) generate a silence signal of a

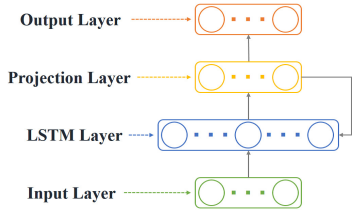

Figure 4: Long Short-Term Memory Recurrent Neural Network with Projection Layer

predetermined length, 2) randomly select a sound event from the training dataset, 3) add the selected sound event to the generated silence signal at a predetermined location, 4) repeat Steps 2 and 3 a predetermined number of time, 5) add a background noise signal extracted from the development set at a predetermined signal to noise ratio (SNR).

In this data generation operation, there are four hyper-parameters; signal length, number of events in a signal, number of overlaps, and SNR between sound events and background noise. We set signal length to 4 seconds, number of events to a value from 3 to 5, number of overlaps to a value from 1 to 5, and SNR to a value from -9 dB to 9 dB. We then generated 100,000 training samples of 4 seconds length, hence, about 111 hours of training data.

3.2. Feature extraction

First, we modified the amplitude of the input sound signals to adjust for the differences in recording conditions by normalizing the signals using the maximum amplitude of the input sound signals. Second, the input signal was divided into 25 ms windows with a 40 % overlap, and we calculated a log filterbank feature for each window in 100 Mel bands (more bands than usual since high frequency components are more important than low frequency ones for SED). Finally, we conducted cepstral mean normalization (CMN) for each piece of training data. Feature vectors were calculated using HTK [22].

3.3. Model

We extended the hybrid HMM/neural network model in order to handle a multi-label classification problem. To do this, we built a three state left-to-right HMM with a non-active state for each sound event. The structure of our HMM is shown in Fig. 5, where $n = 0$, $n = 5$ and $n = 4$ represent the *initial state*, *final state*, and *non-active state*, respectively. Notice that the non-active state represents not only the case where there is no active event, but also the case where other events are active. Therefore, the non-active state of each sound event HMM has a different meaning from the silence. In this study, we fix all transition probabilities to a constant value of 0.5.

Using Bayes' theorem, HMM state emission probability $P(\mathbf{x}_t|s_{c,t} = n)$ can be approximated as follows

$$P(\mathbf{x}_t|s_{c,t} = n) = \frac{P(s_{c,t} = n|\mathbf{x}_t)P(\mathbf{x}_t)}{P(s_{c,t} = n)} \simeq \frac{P(s_{c,t} = n|\mathbf{x}_t)}{P(s_{c,t} = n)} \quad (9)$$

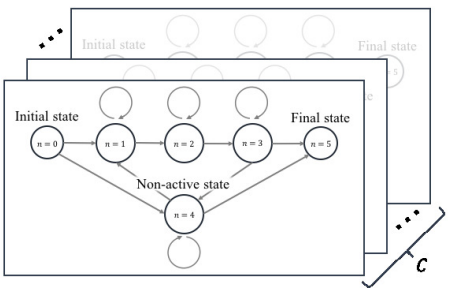

Figure 5: Hidden Markov Model of each sound event

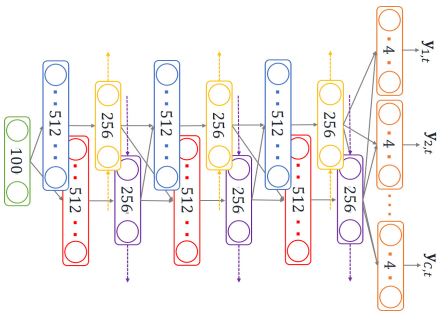

Figure 6: Proposed model structure

where $c \in \{1, 2, \ldots C\}$ represents the index of sound events, and $n \in \{1, 2, \ldots, N\}$ represents the index of HMM states, hence, $P(s_{c,t} = n|\mathbf{x}_t)$ satisfies the sum-to-one condition of $\sum_n P(s_{c,t} = n|\mathbf{x}_t) = 1$. In the BLSTM-HMM Hybrid model, HMM state posterior $P(s_{c,t} = n|\mathbf{x}_t)$ is calculated using a BLSTM-RNN. The structure of the network is shown in Fig. 6. This network has three hidden layers which consist of an LSTM layer, a projection layer, and the number of output layer nodes is $C \times N$. All values of the posterior $P(s_{c,t}|\mathbf{x}_t)$ have the sum-to-one condition for each sound event c at frame t, it is obtained by the following softmax operations

$$P(s_{c,t} = n|\mathbf{x}_t) = \frac{\exp(a_{c,n,t})}{\sum_{n'=1}^{N} \exp(a_{c,n',t})}, \quad (10)$$

where a represents the activation of output layer node. The network was optimized using back-propagation through time (BPTT) with Stochastic Gradient Descent (SGD) and dropout under the cross-entropy for *multi-class multi-label* objective function

$$E(\mathbf{\Theta}) = \sum_{c=1}^{C} \sum_{n=1}^{N} \sum_{t=1}^{T} y_{c,n,t} \ln(P(s_{c,t} = n|\mathbf{x}_t)), \quad (11)$$

where $\mathbf{\Theta}$ represents the set of network parameters, and $y_{c,n,t}$ is the HMM state label obtained from the maximum likelihood path at frame t. (Note that this is not the same as the multi-class objective function in conventional DNN-HMM.) HMM state prior $P(s_{c,t})$ is calculated by counting the number of occurrence of each HMM state. However, in this study, because our synthetic training data does not represent the actual sound event occurrences, the prior obtained from occurrences of HMM states has to be made less sensitive. Therefore, we smoothed $P(s_{c,t})$ as follows

$$\hat{P}(s_{c,t}) = P(s_{c,t})^{\alpha}, \quad (12)$$

where α is a smoothing coefficient. In this study, we set $\alpha = 0.01$. Finally, we calculated the HMM state emission probability using Eq. 9 and obtained the maximum likelihood path using the Viterbi algorithm.

4. EXPERIMENTS

4.1. Experimental condition

We evaluated our proposed method by using the DCASE2016 task 2 dataset [18, 6]. In this study, we randomly selected 5 samples per event from training data, and generated 18 samples which have 120 sec length just like DCASE2016 task 2 development set using selected samples. These generated samples are used as development set for open condition evaluation, and remaining 15 samples

Table 1: Experimental conditions

Sampling rate	44,100 Hz
Frame size	25 ms
Shift size	10 ms
Learning rate	0.0005
Initial scale	0.001
Gradient clipping norm	5
Batch size	64
Time steps	400
Epoch	20

per class are used for training. Evaluation is conducted by using two metrics: *event-based* evaluation, and *segment-based* evaluation, where F1-score (F1) and error rate (ER) are utilized as evaluation criteria (see [24] for more details).

We built our proposed model using the following procedure: 1) divide an active event into three segments with equal intervals in order to assign left-to-right HMM state labels, 2) train the BLSTM-RNN using these HMM state labels as supervised data, 3) calculate the maximum likelihood path with the Viterbi algorithm using RNN output posterior, 4) train the BLSTM-RNN by using the obtained maximum likelihood path as supervised data, 5) repeat step 3 and step 4. In this study, when calculating the maximum likelihood path, we fixed the alignment of non-active states, i.e., we just aligned event active HMM states. When training the networks, we checked the error for development data every epoch, and if the error became bigger than in the previous epoch, we restored the parameters of the previous epoch and re-trained the network with a halved learning rate. All networks were trained using the open source toolkit TensorFlow [23] with a single GPU (Nvidia Titan X). Details of the experimental conditions are shown in Table 1.

4.2. Comparison with conventional methods

To confirm the performance of our proposed method, we compared it with the following four methods: 1) NMF (DCASE2016 task2 baseline), 2) BLSTM-RNN, 3) BLSTM-RNN disregarding a few missing frames, 4) BLSTM-RNN with median filter smoothing. NMF is trained using the remaining 15 samples per class by the DCASE2016 task2 baseline script. In this study, **we do not change any settings** except for the number of training samples. BLSTM-RNN has the same network structure as BLSTM-HMM with the exception that the number of output layer nodes which have a sigmoid function as an activation function corresponds to the number of sound events C. Each node conducts a binary classification, hence, each output node y_c is between 0 and 1. We set the threshold as 0.5, i.e., $y_c > 0.5$ represents sound event c being active, and $y_c \leq 0.5$ non-active. For post-processing, we applied two methods: median filtering, and disregarding a few missing frames. In this step, we set the degree of median filtering to 9, and the number of disregarded frames to 10.

Experimental results are shown in Table 2. Note that the results on the test set are provided by DCASE2016 organizers [18]. From the results, we can see that the methods based on BLSTM are significantly better than the NMF-based method in polyphonic sound event detection. As regards post-processing, in study [8], the authors reported that they did not require post-processing since RNN outputs have already been smoothed. However, we confirmed that post-processing is still effective, especially for event-based evaluation. In addition, although RNN outputs are smoother than the outputs of neural networks without a recurrent structure, there is still room for improvement by smoothing RNN outputs. Our proposed method achieved the best performance on the development set for event-based evaluation, which supports this assertion.

Table 2: Experimental results

	Event-based (dev / test)		Segment-based (dev / test)	
	F1 [%]	ER [%]	F1 [%]	ER [%]
NMF (Baseline)	14.6 / 24.2	665.4 / 168.5	35.9 / 37.0	183.4 / 89.3
BLSTM	66.5 / -	85.3 / -	87.0 / -	25.9 / -
BLSTM (w/ disregard)	75.9 / -	52.5 / -	87.0 / -	25.9 / -
BLSTM (w/ median)	75.8 / **68.2**	53.2 / **60.0**	**87.7** / **78.1**	**24.2** / **40.8**
BLSTM-HMM	**76.6** / 67.1	**51.1** / 64.4	87.2 / 76.0	25.9 / 50.0

Table 3: Effect of background noise

	Event-based		Segment-based	
EBR [dB]	F1 [%]	ER [%]	F1 [%]	ER [%]
-6	73.7	58.0	86.0	28.0
0	76.7	51.3	87.4	25.9
6	79.6	44.1	88.1	23.9

4.3. Analysis

In this section, we focus on the factors which influenced the performance of our proposed method using the development set. The first factor is SNR between background noise and events. The performance of our proposed method on the development set for each SNR condition is shown in Table 3. From these results, there are clear differences in performance between the different SNR conditions. This is because the loud background noise caused more insertion errors, especially small loudness events such as *doorslam* and *pageturn*.

The second factor is the difference in performance between the monophonic and polyphonic tasks. The performance of our proposed method on each type of tasks is shown in Table 4. In general, the polyphonic task is more difficult than the monophonic task. However, we observed a strange behavior with better scores in the polyphonic task than in the monophonic task, while the opposite is normally to be expected. We will investigate the reason as a future work.

5. CONCLUSION

We proposed a new method of polyphonic sound event detection based on a Bidirectional Long Short-Term Memory Hidden Markov Model hybrid system (BLSTM-HMM), and applied it to the DCACE2016 challenge task 2. We compared our proposed method to baseline non-negative matrix factorization (NMF) and standard BLSTM-RNN methods. Our proposed method outperformed them in both monophonic and polyphonic tasks, and finally achieved an average F1-score of 67.1% (error rate of 64.5%) on the event-based evaluation, and an average F1-score 76.0% (error rate of 50.0%) on the segment-based evaluation.

In future work, we will investigate the reason for the counter-intuitive results in the difference between monophonic and polyphonic task, the use of sequence discriminative training for BLSTM-HMM, and we will apply our proposed method to a real-recording dataset.

Table 4: Difference in the performance between monophonic and polyphonic task

	Event-based		Segment-based	
	F1 [%]	ER [%]	F1 [%]	ER [%]
Monophonic	76.2	54.0	84.3	32.4
Polyphonic	76.9	49.6	88.7	22.5

6. REFERENCES

[1] J. Schrder, B. Cauchi, M. R. Schdler,N. Moritz, K. Adiloglu, J. Anemller, and S. Goetze, "Acoustic event detection using signal enhancement and spectro-temporal feature extraction," in *Proc. WASPAA*, 2013.

[2] T. Heittola, A. Mesaros, A. Eronen, and T. Virtunen, "Context-dependent Sound Event Detection," *EURASIP Journal on Audio, Speech, and Music Processing*, Vol. 2013, No.1, 2013, pp. 1–13.

[3] T. Heittola, A. Mesaros, T. Virtanen, and A. Eronen, "Sound event detection in multisource environments using source separation," in *Workshop on machine listening in Multisource Environments*, 2011, pp. 36–40.

[4] S. Innami, and H. Kasai, "NMF-based environmental sound source separation using time-variant gain features," *Computers & Mathematics with Applications*, Vol. 64, No. 5, 2012, pp.1333–1342.

[5] A. Dessein, A. Cont, and G. Lemaitre, "Real-time detection of overlapping sound events with non-negative matrix factorization," *Springer Matrix Information Geometry*, 2013, pp. 341–371.

[6] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *Proc. EUSIPCO*, 2016.

[7] E. Cakir, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic sound event detection using multi label deep neural networks," in *Proc. IEEE IJCNN*, 2015, pp. 1–7.

[8] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE ICASSP*, 2016, pp. 6440–6444.

[9] T. Hayashi, M. Nishida, N. Kitaoka, and K. Takeda, "Daily activity recognition based on DNN using environmental sound and acceleration signals, " in *Proc. EUSIPCO*, 2015, pp. 2306–2310.

[10] M. Espi, M. Fujimoto, K. Kinoshita, T. Nakatani, "Exploiting spectro-temporal locality in deep learning based acoustic event detection," *EURASIP Journal on Audio, Speech, and Music Processing*, Vol.1, No.1, 2015.

[11] Y. Wang, L. Neves, and F. Metze, "Audio-based multimedia event detection using deep recurrent neural networks," in *IEEE ICASSP*, 2016, pp. 2742–2746.

[12] F. Eyben, S. Bck, B. Schuller, and A. Graves, "Universal Onset Detection with Bidirectional Long Short-Term Memory Neural Networks," in *ISMIR*, 2010, pp. 589–594.

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

[14] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition, *ArXiv e-prints arXiv:1402.1128*, 2014.

[15] H. Sak *et. al.,* "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. IEEE INTERSPEECH*, 2014.

[16] Z. Chen, S. Watanabe, H. Erdogan, and J. Hershey, "Integration of speech enhancement and recognition using long Short-term memory recurrent neural network, in *Proc. IEEE INTERSPEECH*, 2015, pp. 3274–3278.

[17] M. Schuster, and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, Vol. 45, No. 11, 1997, pp. 2673–2681.

[18] `http://www.cs.tut.fi/sgn/arg/dcase2016/`.

[19] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, Vol. 5, No. 2, 1994, pp.157–166.

[20] S. Hochreiter, and J. Schmidhuber, "Long short-term memory," *Neural Computation*, No. 9, Vol. 8 ,1997, pp. 1735–1780.

[21] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Artificial Neural Networks*, Vol. 12, No. 10, 1999, pp. 2451–2471.

[22] `http://htk.eng.cam.ac.uk`

[23] `https://www.tensorflow.org`

[24] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection", *Applied Sciences*, Vol. 6, No. 6, 2016, 162.