POLYPHONIC SOUND EVENT DETECTION USING CONVOLUTIONAL BIDIRECTIONAL LSTM AND SYNTHETIC DATA-BASED TRANSFER LEARNING

Seokwon Jung[1,2], Jungbae Park[1,2], Sangwan Lee[1,2,3]**

[1] Humelo Inc.
[2] Korea Advanced Institute of Science and Technology (KAIST)
[3] KAIST Institute for Artificial Intelligence
*Corresponding authors
jay@humelo.com, jb@humelo.com, sangwan@kaist.ac.kr

ABSTRACT

This paper presents a novel approach to improve the performance of polyphonic sound event detection that combines a convolutional bidirectional recurrent neural network (CBRNN) with transfer learning. The ordinary convolutional recurrent neural network (CRNN) is known to suffer from a vanishing gradient problem, which significantly reduces the efficiency of information transfer to past events. To resolve this issue, we combine forward and backward long short-term memory (LSTM) modules and demonstrate that they complement each other. To effectively deal with the issue of overfitting that arises from increased model complexity, we apply transfer learning with a dataset that contains synthesized artifacts. We show that the model achieves faster and better performance with less data. Simulations with the 2016 TUT dataset show that the performance of the CBRNN with transfer learning is dramatically improved compared to the ordinary CRNN; the F1 score was 28.4% higher, and the error rate was 0.42 lower.

Index Terms— polyphonic sound event detection, convolutional recurrent neural network, bidirectional LSTM, transfer learning

1. INTRODUCTION

Sounds in the world come from various sources, such as cars, people, and buildings. Sound event detection (SED) can be performed to label the datasets derived from these sounds by predicting sound events in the given audio. SED techniques have been applied to audio surveillance [1], acoustic monitoring for social care [2], urban sound analysis [3], multimedia event detection [4] and bird call detection [5].

The specific type of SED one performs is based on the number of audio channels and the number of simultaneous sound events. Stereo SED [6,7] uses both audio channels with spatial information, while mono SED uses only one channel. Additionally, monophonic detection [8] recognizes one sound event, and polyphonic detection considers more than one sound event and overlapping sounds. Early research employing polyphonic SED transformed audio into mel-frequency cepstral coefficients (MFCC) and used machine learning techniques, such as support vector machines [1], the hidden Markov model (HMM), and the Gaussian mixture model (GMM) [9].

This paper proposes a new framework for polyphonic detection of monaural audio, which is a difficult task due to the limited amount of available information. Specifically, we combine a fully-connected neural network (FNN) [10], convolutional neural networks (CNN) [11, 12, 13], and recurrent neural networks (RNN) [14] for SED. Unlike recent studies on the capsule network-based model [15], bidirectional long short-term memory-hidden Markov model (LSTM-HMM) hybrid model [16], and convolutional recurrent neural networks (CRNNs) [17], this study presents an approach to SED that employs a convolutional bidirectional recurrent neural network (CBRNN), showing how this model solves the vanishing gradient problem and why the two opposing directional models complement each other.

Training this model is tricky due to the high complexity of the model. To resolve this issue, we propose a transfer learning strategy based on a synthetic dataset, which we call Humelo synthetic. Although raw data is easy to obtain, it is very difficult to manually label. Therefore, instead of labeling real-life audio, we added synthesized artifacts to the data. Most prior studies are based on four datasets: TUT-SED 2009 [18], TUT-SED 2016 [19], TUT-SED synthetic 2016 [20], and CHiME-Home [21]. However, TUT-SED 2009 is a private dataset, and CHiME-Home is chunk-based, not frame-based. Therefore, we used the TUT-SED 2016 and TUT-SED synthetic 2016 datasets, which are public and frame-based. Note that the public real-life audio dataset, TUT-SED 2016, has a very small quantity of data (78 minutes), while the artificial audio dataset, TUT-SED synthetic 2016, has about 9 hours of data.

We pretrained the proposed model, CBRNN, with our artificial dataset (Humelo synthetic) and transferred some

pre-trained weights from the pre-trained version to the main model.

2. PROPOSED METHOD

We transformed the raw audio into a mel spectrogram through preprocessing and used it as an input for the model. To solve the vanishing gradient problem of the CRNN model, we used a CBRNN. We also devised a synthetic data-based transfer learning scheme to improve the learning speed and solve the data shortage.

2.1. Preprocessing

Our system uses an input with a fixed length. So, we divided the raw audio into five-second chunks. If the remaining audio was shorter than five seconds, the rest was padded with 0. Then, we set the sampling rate and channel number to 44100 and 1, respectively. The five-second chunks were short-time Fourier transformed (STFT) with 50 ms frames and 50% overlap, and the results were given in the log magnitude of 40 mel per frame. After changing the mel spectrogram's amplitude to decibels, we clipped below -100 db and normalized them to fit [-1,1]. We also performed frame-based labeling, assigning a label vector to every frame. The k-th value of the vector is indicated as a binary variable depending on whether the k-th class exists in the frame.

2.2. Forward-backward learning architecture

As shown in Fig 1, the proposed CBRNN model combines bidirectional LSTM [16] and a CRNN [18]. The preprocessed mel spectrogram, which is 40 * 200 in size, was taken as an input and passed through the three layers of CNN. The number of filters was set to (256, 256, 256), and the kernel size was kept at 3. A sigmoid activation function was used, and the weights were randomly initialized each time.

After each convolutional layer operation, max pool, dropout, and batch normalization were performed. The max pool proceeded only on the decibel axis by (5, 4, 2). The dropout rate was fixed at 0.3. After all of the CNN layers, the output format was set to 1 * 200 * 256. The filter axis was concatenated to the decibel axis, resulting in 256 * 200, and then passed through the bi-directional RNN (BRNN) of the three layers. Each RNN layer consisted of 100 LSTM cells, and each cell had 100 units. Instead of batch normalization, we applied layer normalization and used the same sigmoid for the activation function.

Forward and backward LSTM differ in terms of input order; backward LSTM inputs information in the opposite order as forward LSTM. We merged the output of forward and backward LSTM into a simple concatenate and used them as input for the next layer. Finally, the results of the BRNN were derived from the dense layer, which converted the BRNN output to a class label output vector.

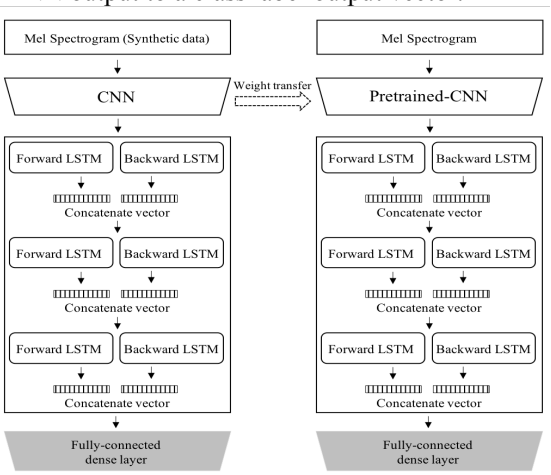

Fig 1. Proposed method: a CBRNN architecture with transfer learning and a synthetic dataset.

2.3. Transfer learning with a synthetic dataset

We performed transfer learning to manage the increased model complexity. Transfer learning is usually used to pre-train a model that performs the same or similar tasks as the main model, and then the pre-trained weights are used to improve the learning speed and performance of the main model. To apply transfer learning, we trained the CBRNN model with an artificially synthesized dataset and then created the CNN in the new CBRNN model using the pre-trained model's CNN weights.

To create an artificial dataset (Humelo synthetic), we chose 20 classes based on [8], downloaded three different kinds of public audio for each class from Freesound and YouTube, and obtained the overall mean and standard deviation of all the downloaded audio. After randomly selecting one to three classes to be synthesized, the length and position of audio were randomly selected and synthesized to produce five-second empty audio clips. In the process of synthesis, the amplitude was normalized by multiplying the Gaussian random value obtained from the overall mean and the standard deviation.

3. EVALUATION

3.1. Dataset

We used a total of three datasets: an artificial dataset to pre-train the model and TUT-SED 2016 and TUT-SED synthetic 2016 for evaluation. TUT-SED 2016 was recorded in two acoustic scenes in a home and a residential area and includes the dataset used in the DCASE 2016 challenge. The home scene produced 7 classes of sound events, which are 42 minutes long, and the residential area

scene produced 11 classes of sound events, which are 36 minutes long. We used the same classifier for both acoustic scenes. TUT-SED synthetic 2016 includes synthesized data and consists of 16 classes and 566 minutes of audio.

We used 60% for training, 20% for testing, and 20% for validation for all data sets.

3.2. Evaluation Metrics

We used segment-based evaluation metrics to measure performance. The length of the segment was a single frame (50 ms). We used the F1 score and the error rate (ER) to analyze the performance of the model and compare it with other models. The F1 score for precision (P) and recall (R) was defined as follows:

$$F1 = \frac{2 \cdot P \cdot R}{P + R}$$

Also, the error rate was defined as follows using insertions (I), deletions (D), the substitution (S) term, and active classes (N):

$$ER = \frac{\sum_{k=1}^{K} S(k) + \sum_{k=1}^{K} I(k) + \sum_{k=1}^{K} D(k)}{\sum_{k=1}^{K} N(k)}$$

where k represents the segment index and K is the total number of segments. Further definitions and explanations are given in [22].

In addition, the performance of each class was compared using AUC, the area of ROC curve, drawn by true positive ratio (TPR) and false positive ratio (FPR).

3.3. Experimental Setup

As described in detail above, the proposed CBRNN consists of three CNN layers, three RNN layers, and one dense layer. All activation functions were sigmoid functions, and the following cross entropy function was used as a loss function instead of sigmoid cross entropy to avoid duplication of the sigmoid after the dense layer:

$$loss = -\sum_{x} P(x) \log Q(x)$$

where P and Q are the true and predicted labels of the xth value, respectively. The optimizer proceeded with Adam and the learning rate from 0.001 to exponential decay.

4. RESULT

In this section, we compare the performance of the base line CRNN [18] and CRNN (pre-CRNN), CBRNN and the proposed method, transfer learned CBRNN (tl-CBRNN).

4.1. Performance comparison based on benchmark datasets

The results for the TUT-SED 2016 and TUT-SED synthetic 2016 datasets (i.e., the mean and standard deviation over 20 simulations) are shown in Table 1. The best case is indicated by a bold font. In the case of TUT-SED 2016, pre-CRNN was 0.6 better than the baseline but 0.15 better in terms of the ER. In the case of TUT-SED synthetic 2016, the pre-CRNN performed much worse than the baseline. For both datasets, CBRNN significantly outperformed CRNN.

Table 1. F1 score and error rate results for single frame segments.

	TUT-SED 2016		TUT-SED Synthetic 2016	
Method	F1	ER	F1	ER
CRNN [17]	27.5±2.6	0.98±0.04	66.4±0.4	0.45±0.0
pre-CRNN	26.9±3.9	0.83±0.03	39.2±2.1	0.69±0.14
CBRNN	49.9±5.8	0.61±0.06	70.7±0.6	0.40±0.01
tl-CBRNN	**55.9±1.9**	**0.56±0.03**	**74.0±0.5**	**0.36±0.01**

In the case of TUT-SED 2016, the F1 score and ER were improved by 22.4 and 0.37, respectively, and in the case of TUT-SED synthetic 2016, they were improved by 4.3 and 0.05, respectively. TUT-SED 2016, the real audio dataset, showed a tremendous improvement in results. Also, in the case of tl-CBRNN, the F1 score was improved by about 5 ~ 4 and the ER was improved by about 0.05 ~ 0.04 compared with CBRNN.

4.2. Post hoc analysis : forward-backward learning

The most notable difference between the CBRNN and baseline CRNN is that backward LSTM was added to the CBRNN. As mentioned earlier, the disadvantage of CRNN is that the gradient vanishes when RNN becomes longer. Fig 2. shows the gradient variation trends of forward and backward LSTM in CBRNN. In forward LSTM, the forward gradient was updated at less than half the speed of the backward gradient, and the results were reversed for backward LSTM. Thus, it appears that combining forward and backward LSTM in CBRNN solves the gradient vanishing problem.

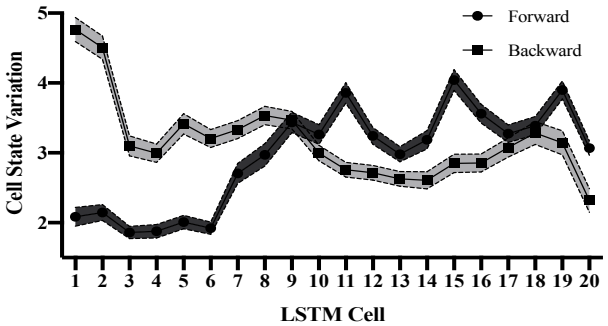

Fig 2. Cell state variation of forward and backward LSTM cells.

We used t-SNE [23] to show that forward LSTM and backward LSTM encode complementary information. One-dimensional t-SNE was performed for each of the 100 outputs of forward and backward LSTM. We plotted a scatter diagram using the results of forward and backward t-SNE for the y and x axes. Fig 3. shows the results of t-SNE analysis of the five best and five worst classes based on AUC. Figs. 3a and 3c show the clustering results for true label, and Figs. 3b and 3d are graphs comparing the label predicted by the model and the true label. The wrong labels are denoted by red. Comparison of the true label (Fig. 3a) and the predicted label (Fig. 3b) reveals that the top classes lead to better clustering results than the bottom classes. Also, comparison of Fig. 3a with Figs. 3c, 3b, and 3d shows that Figs. 3b and 3d have better and more predictable clustering results. This indicates that the value of each axis calculated by t-SNE is very important for classification. Finally, the most important implication of the results, shown in Fig. 3, is that clustering cannot be performed with the same class for forward and backward projection. In particular, it is evident that, in the case of the top classes, a combination of forward and backward learning is necessary as the two types of learning are complementary.

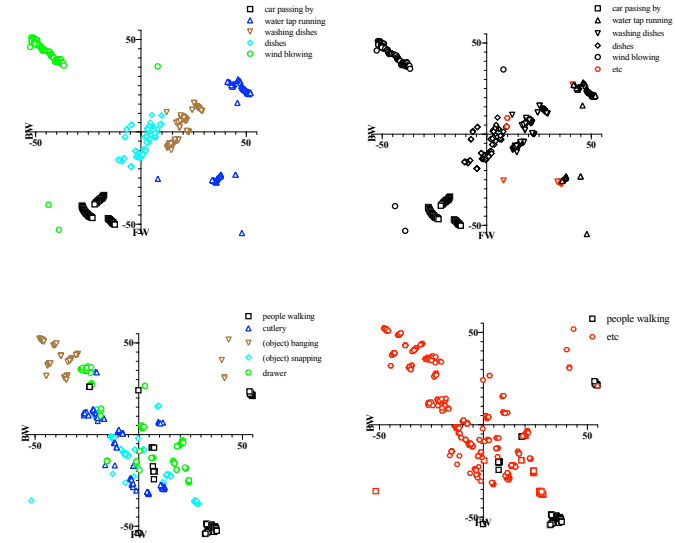

Fig 3. t-SNE results concerning forward and backward LSTM output. Predicted labels that were wrong are denoted by red. (a) True labels of the top five classes according to t-SNE. (b) Predicted labels of the top five classes according to t-SNE. (c) True labels of the bottom five classes according to t-SNE. (d) Predicted labels of the bottom five classes according to t-SNE.

4.3. Post hoc analysis : transfer learning

We show that transfer learning significantly improved the speed of training. Fig. 4a shows the convergence speed of each model for TUT-SED 2016. The average learning speed of pre-CRNN was slowest (1,670 steps), followed by CBRNN (635 steps) and tl-CBRNN (465 steps). The convergence speed was measured from the convergence step at which test loss occurred. As shown in Fig. 4b, an early stopping regime was imposed on each model to prevent overfitting.

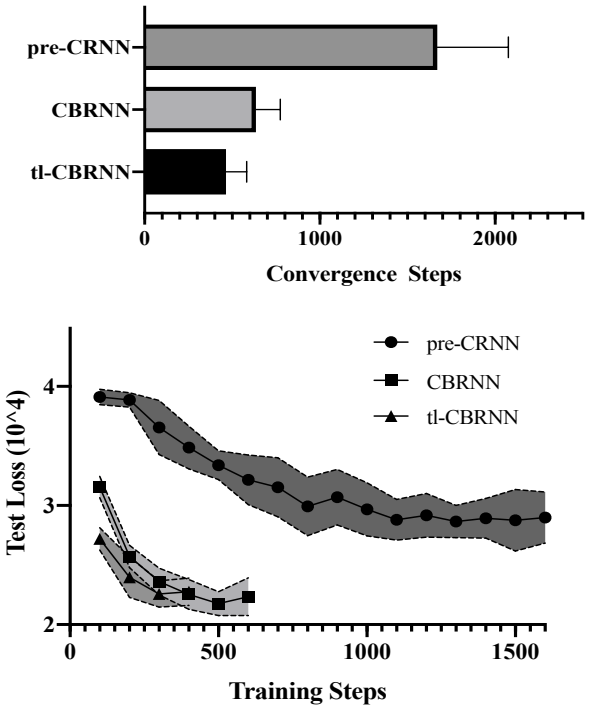

Fig 4. (a) Convergence steps by model. (b) Change of test loss according to the training steps for pre-CRNN, CBRNN, and tl-CBRNN.

5. CONCLUSION

This paper presents a novel framework for SED that employs transfer learning and is based on a synthetic dataset. The performance of the proposed method is demonstrated to be significantly better than that of other methods. Specifically, the F1 score and error rate were increased by 28.4 and 0.42, respectively, for the TUT-SED 2016 dataset and by 7.6 and 0.09, respectively, for the TUT-SED synthetic 2016 dataset. The learning speed was improved about fourfold. We also show that a combination of backward and forward LSTM and integration of complementary information can effectively solve the vanishing gradient problem, and using transfer learning with a synthetic dataset avoids overfitting issues. Further studies should focus on implementing a dynamic interaction between the backward and forward LSTM modules as well as designing a general scheme for artifact-based data synthesis for transfer learning.

6. REFERENCES

[1] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio and M. Vento, "Reliable detection of audio events in highly noisy environments," *Pattern Recognition Letters*, vol. 65, pp. 22-28, 2015.

[2] S. Goetze, J. Schroder, S. Gerlach, D. Hollosi, J.-E. Appell, and F. Wallhoff, "Acoustic monitoring and localization for social care," *Journal of Computing Science and Engineering*, vol. 6, no. 1, pp. 40–50, 2012.

[3] J. Salamon and J. P. Bello, "Feature learning with deep scattering for urban sound analysis," *European Signal Processing Conference (EUSIPCO). IEEE,* pp. 724–728, 2015.

[4] Y. Wang, L. Neves, and F. Metze, "Audio-based multimedia event detection using deep recurrent neural networks," *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2742–2746, 2016.

[5] D. Stowell and D. Clayton, "Acoustic event detection for multiple overlapping similar sources," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 1–5, 2015.

[6] S. Adavanne, G. Parascandolo, P. Pertila, T. Heittola, T. Virtanen, " Sound Event Detection in Multichennel Audio Using Spatial and Harmonic Features," *Detection and Classification of Acoustic Scenes and Events(DCASE) Workshop*, 2016.

[7] S. Advanne, P. Pertila, and T. Virtanen, "Sound Event Detection Using Spatial Features and Convolutional Recurrent Neural Network," *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[8] T. Heittola, A. Mesaros, A. Eronen and T. Virtanen, "Context-dependent sound event detection", *EURASIP Journal on Audio, Speech, and Music Processing*, pp. 1-13, 2013.

[9] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings", *European Signal Processing Conference*, pp. 1267-1271, 2010.

[10] E. Cakir, T. heittola, H. Huttunen, and T, Virtanen, "Polyphonic sound event detection using multi label deep neural networks," *in IEEE International Joint Conference on Neural Networks (IJCNN)*, 2015.

[11] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks", *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 559–563, 2015.

[12] H. Phan, L. Hertel, M. Maass, and A. Mertins, "Robust audio event recognition with 1-max pooling convolutional neural networks," *Interspeech*, 2016.

[13] K. J. Piczak, "Environmental sound classification with convolutional neural networks," *in Int. Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2015.

[14] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6440–6444, 2016.

[15] Y. Liu, J. Tang, Y. Song, L. Dai, "A Capsule based Approach for Polyphonic Sound Event Detection," *APSIPA-ASC,* 2018

[16] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J.L. Roux, K. Takeda, "Bidirectional LSTM-HMM Hybrid System for Polyphonic Sound Event Detection," *Detection and Classification of Acoustic Scenes and Events(DCASE) Workshop*, 2016.

[17] E. Cakir, G. parascandolo, T. Heittola, H Huttunen, and T. Virtanen, "Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, pp. 1291-1303, 2017.

[18] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Audio context recognition using audio event histograms," *Proc. of the 18th European Signal Processing Conference (EUSIPCO)*, pp. 1272–1276, 2010.

[19] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," *European Signal Processing Conference (EUSIPCO)*, 2016.

[20] http://www.cs.tut.fi/sgn/arg/taslp2017-crnn-sed/#datasets

[21] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "Chime-home: A dataset for sound source recognition in a domestic environment," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE*, pp. 1–5, 2015.

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

[23] L. Matten, G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research(JMLR),* pp. 2579-2605, 2008.