INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Rare Sound Event Detection Using Deep Learning and Data Augmentation

Yanping Chen, Hongxia Jin

Samsung Research America
{yanping.c, hongxia.jin}@samsung.com

Abstract

There is an increasing interest in smart environment and a growing adoption of smart devices. Smart assistants such as Google Home and Amazon Alexa, although focus on speech, could be extended to identify domestic events in real-time to provide more and better smart functions. Sound event detection aims to detect multiple target sound events that may happen simultaneously. The task is challenging due to the overlapping of sound events, the highly imbalanced nature of target and non-target data, and the complicated real-world background noise. In this paper, we proposed a unified approach that takes advantages of both the deep learning and data augmentation. A convolutional neural network (CNN) was combined with a feed-forward neural network (FNN) to improve the detection performance, and a dynamic time warping based data augmentation (DA) method was proposed to address the data imbalance problem. Experiments on several datasets showed a more than 7% increase in accuracy compared to the state-of-the-art approaches.

Index Terms: sound event detection, data augmentation, model ensemble, CNN, FNN

1. Introduction

The adoption of smart home devices is steadily increasing [15]. Many of the devices, from security cameras to smart speakers, are equipped with microphones, enabling us to collect sounds continuously with little costs. Given the ready access to audios, sound intelligence becomes an important component for smart devices. Automatic sound event detection (SED) allows devices to understand the context around them, provide contextual AI to intelligent machines, and enables devices to make a better impact in our lives.

Real life sound recordings typically have overlapping sound events. For example, a domestic audio clip may contain the simultaneous sounds of TV playing, people talking, air conditioner running and the random background noise. The overlapping nature of sound events makes the task of sound event detection challenging. In addition, a common problem in all target detection tasks is the level of data imbalance. Generally, only a small percentage of audio data contains the target events. Take the sound of security alarm for example. In a typical 1,000 hours' recording of domestic sound, less than one hour of the recordings contains the target event of the alarm going off. This is especially true for the target events that happen rarely, such as window breaking. The highly skewed data distribution of rare target events makes the detection task even more challenging.

Traditional approaches that have been successfully used for speech recognition [5][6] do not work well for overlapping sound events detection due to the additive nature of event sounds. Recently, deep learning approaches have shown to achieve good performance for SED [3]. State-of-the-art SED systems use Mel-Frequency Cepstral Coefficients (MFCC) to represent the audio signals and use neural networks as classifiers [3][12][14]. In [3], the feed forward neural network (FNN) is used as the classifier. The authors of [14][24] proposed using CNN with the spectrogram images as features. For tasks where the target events are complicated with a dynamic process, the long short-term memory (LSTM) recurrent neural networks are often used [20]. A CRNN network that combined the strength of CNN and RNN was proposed in [7]. The authors in [21][22] proposed using connectionist temporal classifier to deals with weakly labeled training data.

Different from the existing approaches that focused on a single classifier, in this paper, we propose a novel model ensemble method that takes advantage of existing approaches to improve the detection accuracy. Specifically, FNN and CNN are combined using a simple yet effective method to reduce misclassifications when the two classifiers do not agree. A new DA method is proposed to solve the data imbalance to better train the networks. Experiments on several datasets based on DCASE Challenges [19] demonstrate the effectiveness of the proposed approach.

2. Data Augmentation

In most target detection tasks, the amount of target data is significantly smaller than the amount of the non-target data. Training a classifier with a highly imbalanced class distribution using conventional machine learning algorithms often results in biased and inaccurate classifiers.

Traditional methods to deal with the imbalance problem focused on changing the training algorithm without modifying the original training dataset, such as using class weights [13] and using ensemble-based algorithms [9]. Although these techniques help to reduce classifier bias to some extent, the improvements are limited. This is especially true when the amount of original data is far from sufficient.

Data augmentation is a technique commonly adopted to expand the size of the original dataset. In this paper, we use it to increase the quantities of the training examples for the target events to address the data imbalance problem. Traditional methods for acoustic data augmentation includes pitch shifting, random erasing and range compression that expands the dataset with perturbed examples [16]. However, these simple techniques have a less than satisfactory impact on the modeling performance. In this paper, we propose a new technique for audio data augmentation that simulates the real-world sound variations based on dynamic time warping.

2.1. Rescaling and merging based data augmentation

In real world, the durations of the occurrences of an event may vary, depending on how fast the event progresses. Sound

http://dx.doi.org/10.21437/Interspeech.2019-1985

instances of an event can be perceived as the time warped versions of some platonic ideal (prototypical shape), with other types of noise/distortions. To simulate this variation pattern, we began by using an intuitive data augmentation method that synthetizes examples by first rescaling the samples and then merging multiple samples.

Rescaling: The rescaling step is similar to the elastic distortion [16] technique that is widely used in image augmentation. It stretches or shrinks the samples to simulate the real-world sound variations of progressing slower or faster. Specifically, given a randomly chosen training sample X of length n and a scaling factor α, a new sample is synthetized using the formula below:

$$x'_j = x_{\lceil j*\frac{n}{m} \rceil} \quad (\text{for } 1 \le j \le m) \qquad (1)$$

where $m = \lceil n * \alpha \rceil$ is the length of the scaled example. x'_j is the j^{th} point of the scaled example, and $\lceil\rceil$ denotes rounding up the number.

Merging: The merging step constructs new examples by combining multiple rescaled samples from a same event. This step provides better generalization in data generation and supplements the simple time stretching functionality of re-scaling. Specifically, given multiple rescaled training samples X'_i that belong to a same target event, we merge the examples to create a new training sample $\overline{X}$ as follows:

$$\overline{X} = \sum_i \lambda_i X'_i \qquad (2)$$

where λ_i is the weight for sequence X_i' , and $\sum_i \lambda_i = 1$.

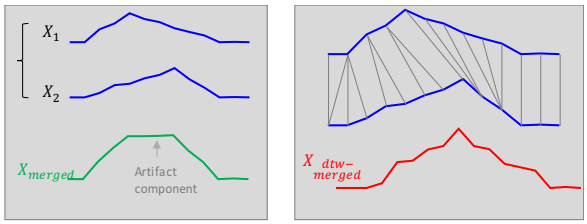

Figure 1: *Left-top) Two examples of an event that characterizes an upward component immediately followed by a downward component. Left-bottom) a generated sample by merging X_1 and X_2 using regular addition. Right-top) The optimal alignment of the two examples using DTW. Right-bottom) a generated sample by merging X_1 and X_2 using proposed DTW-based average.*

This rescaling and merging based method simulates the real world sound variations, and thus, helps to improve the modeling performance. However, there is still room for improvements.

To explain our observations and our key insights, we consider a concrete example. Imagine we have a target sound event that characterizes an upward component immediately followed by a downward component. Two instances of the event are shown in Figure 1, with the first example X_1 faster in the upward and slower in the downward, and the second example X_2 the opposite. Suppose we merge the two examples using equation (2), we get a new example X_merged as shown at the *left-bottom* of Figure 1. As we can see, the new training example has an artifact component that lies between the upward and the downward component, distorting the concept of the original event.

In retrospect, this is not surprising. Merging two acoustic data using regular addition is equivalent to overlapping two sound instances. When the two instances are not well aligned, the event components get mixed up and the generated example is too distorted that it does not retain the original shape of the class. The key solution to this problem is to keep the major components of the examples well aligned when performing the merging step. Fortunately, dynamic time warping technique is particularly suited for this purpose.

2.2. Dynamic Time Warping based merging

Dynamic time warping (DTW) is widely used for measuring distances between two time series, which may vary in speed [1]. It finds the optimal alignment between two sequences that minimizes the distance. In our task, this alignment can be used in the merging step to compute the weighted average.

For merging two instances, the computation of weighted DTW average is straightforward. We take each index-pair from the optimal alignment returned by the DTW calculation, and perform the weighted sum over each pair of points accordingly. An example is shown at the *right* of Figure 1, where the grey lines showed the optimal alignment. The resulting sequence $X_{dtw-merged}$ is shown at the *right-bottom.* As we can see, the DTW merged subsequence retains the shape of the original prototype, and thus, is better at simulating the sound variations.

When there are more than two instances, computing the DTW average becomes much more complicated as it is related to multiple alignment. Recently, a method called DBA uses an expectation-maximization technique to compute the average. It has been shown to outperform most of other existing techniques [14] and is the state-of-art method. However, since DBA does not use weights, we need to modify the algorithm to compute the *weighted* average. Incorporating weights in the merging step allows better generalization in synthetizing the variations.

Table 1: *Pseudo code of the modified DBA method to compute the DTW weighted average*

def dtw_weighted_average_modified (λ, X):	
""" Inputs: @ λ is the weight vector @ X is a set of subsequences, each corresponds to an occurrence of the event Output: @res: the weighted DTW average """	
1	ref_ave = DBA(X)
2	L = len(ref_ave)
3	res = [0] * L
4	for k, x in enumerate(X):
5	path = dtw (x, ref_ave)
6	for i in range(L):
7	align-indexes = [p[0] for p in path if p[1] == i]
8	res[i] += λ_k * sum([x[j] for j in align-indexes])
9	return res

The pseudo code for the modified DBA method to compute the weighted DTW average is shown in Table 1. In Line 1, we called the original DBA method to generate a reference average. Then for each raw subsequence x to be merged, we find the optimal alignment between x and the reference average obtained from DBA. In lines 6-8, for each point i in the reference average, we find all the points in x that

are aligned to that point. These are the points in x that contribute to the i^{th} point in the final result. We then add these points to the resulting sequence using weighted sum. The resulting sequence is the weighted average using DTW.

Table 2: *Pseudo code for the proposed data augmentation method.*

def data_augmentation():	
1	X = randomly pick a set of examples
2	for each x in X:
3	α = random.uniform (0.5, 2)
4	x' = rescale(x)
5	X'. append(x')
6	λ = [random.randint(1,100) for i in range(len(X))]
7	$\lambda = \lambda/\text{sum}(\lambda)$
8	res = dtw_weighted_average_modified (λ, X)
9	return res

To summarize, the proposed data augmentation includes three steps. The Pseudo code of synthesizing an example is shown in Table 2.

1) Randomly choose multiple instances from a same event. (Line 1)
2) Rescale each instance (Line 2-5).
3) Randomly generate the weight-vector and compute the weighted DTW average (Line 6-8). Return the weighted average as a new example.

3. Model Ensemble

For our task, the input is a continuous audio stream. At any time instance, there are overlapping sound events, with some/none target events. The goal is, for each data frame, identify the target events that are present in that frame.

Figure 2 shows the framework of our proposed method. The system consists two neural network classifiers with different spectral features, and a fusion algorithm that combines the two networks.

3.1. Audio representation

The input data stream is first divided into data frames using a sliding window, with 64ms duration and 50% overlap. Two sets of spectral features are extracted from each frame, the Mel frequency cepstral coefficients (MFCCs) and the short term Fourier transform (STFT) coefficients.

The use of MFCCs has been a standard for audio processing for decades. Following the convention, we extracted 39 dimensions of MFCC, composed of 13 MFCCs and their first and second temporal derivatives.

We then extracted the raw STFT coefficients. Different from MFCC, which is a *compressed* representation of the audio clip, STFT coefficients retain most of the information carried in the original audio clips, and thus, provides more information to the system for better detection. In our method, only coefficients in the frequency range of [0, 4000Hz] are used, as little information is contained in the frequencies beyond this range.

In order to utilize the dynamic property of the signal, we followed the audio representation method in [3] and used context windowing. Adjacent frames that precede and follow the central frame are concatenated to form a training instance. For each data frame, the output y is represented using multi-hot-encoding. For a total of M target events, y is a M dimensional array. Each element $y(i)$ is a binary variable indicating the presence of a target event i.

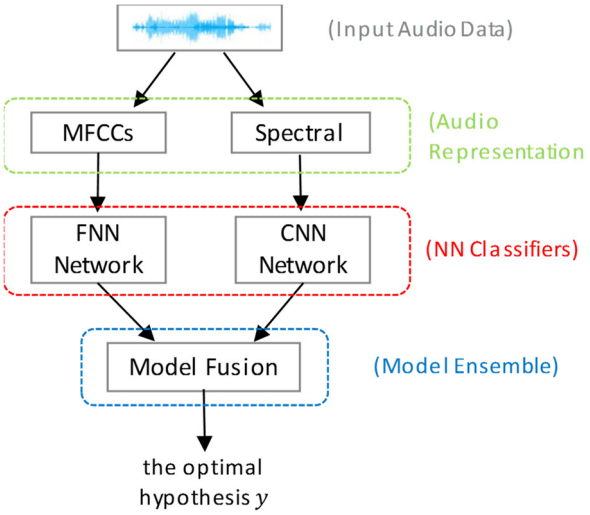

Figure 2: *The overview of the proposed method. Our method consists of three main blocks, audio feature extraction, neural network classifiers and classifier ensemble.*

3.2. Feedforward neural network

Using MFCC features with a feedforward neural network has been shown to achieve high accuracy for sound event detection [3][4]. In this work, we follow the same idea and feed MFCC to a FNN network. The configuration of FNN contains two hidden layers. The number of neurons in each hidden layer is learned using cross-validation. The *tanh* function is used as the activation function for the hidden layers, and *sigmoid* function is used in the output layer to output values in the range from 0 to 1.

3.3. Convolutional neural network

Because of the high dimension of the STFT coefficients, we used CNN instead of FNN to reduce the size of the model. In our configuration, we used two sets of convolutional and pooling layers, with each set containing a one-dimensional convolutional layer and a maximum pooling layer. Passing through each set of layers, the complexity of features declines gradually. Then, the output from the second max pooling layer is passed to a fully connected layers with neurons using *tanh* activation. The final output layer has M neurons where M is the total number of target events.

3.4. Model fusion

To combine the two networks, we began with the straightforward idea of adding a fully connected layer at the output layer to fuse the outputs from the two networks. However, as we will show later in Sec. 4, the performance of this method is inferior to that of using model ensemble.

The idea of using model ensemble came from our experiments, where we observed that, for many misclassifications, the outputs of the two networks do not agree. Assume there is a smart combination algorithm that is capable of choosing the right output, the performance would have been improved significantly.

We consider the ensemble problem as finding a mapping function. The input are probabilities output from the networks, $x = [p_{cnn}, p_{dnn}] \in \mathbb{R}^2$, and the output is a binary variable $y = \overline{y} \in \{0,1\}^C$. The goal is to find the function f that best maps x to y: $y = f(x)$.

This is a typical machine learning problem, and a simple binary classifier would work. After trying several classifiers, the SVM classifier with RBF kernel performs the best, and thus, is used in this work for model fusion.

Note that, we fuse the model in an element-wise ensemble manner. A SVM is trained for each target event. With a total number of M target events, we trained M SVM classifiers.

4. Experiments

We evaluated our proposed method on three datasets based on DCASE challenges, including the rare sound event dataset (DCASE 2017), the domestic activity dataset (DCASE 2019) and the domestic audio tagging dataset (DCASE 2016). For each dataset, we used a single channel that is randomly picked.

To better serve our experiment purpose, we modified the DCASE 2016 dataset by mixing it with some rare target events, including *glass breaking*, *smoke alarm ringing* and *doorbell ringing*. The rare event data were downloaded online and each event had no more than 5 minutes of recordings in total. For DCASE 2019 dataset, we chose the three events that has the least amount of data as the target events.

For all datasets, we randomly chose 3-hours' recordings for training and 1-hour's recording for testing. All models were trained using 5-fold cross validation, and the performance was evaluated on the testing data using the equal error rate (EER) as the measuring metric. The learned hyper parameters for FNN and CNN are shown in Table 3.

Table 3: *Hyper-parameters for FNN and CNN*

FNN	
#units in hidden layer 1	1024
#units in hidden layer 2	512
Learning rate	0.0001
context frame	2

CNN	
size of convolution filter (for each layer)	2 filters, 16 dims
size of max pooling layer	4
#units in fc layer	256
Learning rate	0.002

To address the data imbalance problem, we used data augmentation to increase the amount of target event data. For each target event, we generated examples that add up to approximately 30 minutes. Table 4 showed the impact of using data augmentation and not using data augmentation. As we can see, using data augmentation significantly reduced the EER. The improvement is more than 5% for both DCASE2017 and DCASE 2016, and 1.67% for DCASE 2019 dataset. The improvement of the latter is less significant. This is because the amount of target event data in this dataset is sufficient and the data distribution is more balanced.

We also compared our data augmentation method to the state-of-art algorithm called mix-up [8][23], which synthetizes new examples by mixing samples from different labels, and the results are also shown in Table 4. We can see that, the proposed method consistently outperforms the baseline algorithm for all datasets. Also, modifying the merging step from regular addition to DTW averaging greatly improves the impact of the proposed method, as shown in the last two rows.

Table 4: *Comparison of the effect of different data augmentation methods on the modeling performance*

	EER		
	DCASE2017	DCASE2019	DCASE2016
Without DA	19.67%	19.99%	21.55%
Mix-up DA	15.71%	18.27%	18.36%
Rescaling and merging based DA	16.43%	19.75%	18.07%
DTW based DA	13.46%	18.32%	15.38%

Table 5 showed the comparison results of our system to the methods. As we can see, our method outperforms the base line [3] by more than 6% for all datasets, with a more than 7% improvements for both the DCASE 2017 and the DCASE 2016 datasets. In addition, we investigated the effect of model ensemble by comparing our system to each single network. We can see that, while the FNN and the CNN have similar performance separately, the model has more than 3% improvements when combined. This confirms our hypothesis that the two sets of spectral features supplement each other with addition information and improve the detection accuracy.

We also compared two different ensemble methods: the SVM-based ensemble and the fully-connected layer fusion. The results are shown at the bottom of Table 5. We can see that, using SVM ensemble method is slightly better than the fully-connected layer method. This is possibly because a fully connected layer uses the sigmoid function to combine the outputs, and thus, it has the similar effect as using a logistic regression classifier; whereas, the RBF kernel based SVM maps the input probabilities to a higher dimension and obtains better performance for fusion.

Table 5: *Performance comparison of different methods*

	EER		
	DCASE2017	DCASE2019	DCASE2016
Baseline method	20.78%	25.13%	23.22%
FNN only	18.01%	22.41%	18.76%
CNN only	16.70%	23.15%	19.43%
Fusion using fully connected layer	15.33%	19.58%	15.90%
Fusion using SVM	13.46%	18.32%	15.38%

5. Conclusions

In this work, we proposed a unified approach that takes advantages of both data augmentation and model ensemble for rare target sound event detection. The data augmentation technique is a new method based on linear re-scaling and DTW merging. Two neural networks with two different sets of spectral features are combined using a simple yet very effective ensemble method to generate the final prediction. Experiments on extensive datasets showed the superiority of the proposed method over the state-of-the-art methods.

6. References

[1] D. J. Berndt, J. Clifford, "Using dynamic time warping to find patterns in time series". *In KDD workshop*, Vol. 10, No. 16, pp. 359-370, July, 1994.

[2] A. Aditya, et al. "Minimally supervised sound event detection using a neural network." *Advances in Computing, Communications and Informatics (ICACCI),* International Conference on. IEEE, 2016.

[3] E. Cakir, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic sound event detection using multi label deep neural networks," *in Neural Networks (IJCNN), International Joint Conference on. IEEE*, pp. 1–7, 2015.

[4] E. Cakir, T. Heittola, H. Huttunen, & T. Virtanen. "Multi-label vs. combined single-label sound event detection with deep neural networks". *In Signal Processing Conference (EUSIPCO), IEEE.* pp. 2551-2555, 2015.

[5] A. Dario, et al. "Deep speech 2: End-to-end speech recognition in English and Mandarin." *International Conference on Machine Learning*. 2016.

[6] Y. Dong, and L. Deng. "Automatic speech recognition." *Springer London Limited*, 2016.

[7] C. Emre, and T. Virtanen. "End-to-End Polyphonic Sound Event Detection Using Convolutional Recurrent Neural Networks with Learned Time-Frequency Representation Input." *arXiv preprint arXiv*:1805.03647, 2018.

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

[9] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, & Herrera, F., "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4), 463-484, 2012.

[10] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Context-dependent sound event detection," *EURASIP Journal on Audio, Speech, and Music.* Processing, vol. 2013, no. 1, p. 1, 2013.

[11] T. Heittola, A. Mesaros, T. Virtanen, and A. Eronen, "Sound event detection in multisource environments using source separation," *in Machine Listening in Multisource Environments,* 2011.

[12] Y. Liu, et al. "A Capsule based Approach for Polyphonic Sound Event Detection." *arXiv preprint arXiv*:1807.07436, 2018.

[13] B. Mirza, Z. Lin, & K. A. Toh, "Weighted online sequential extreme learning machine for class imbalance learning". *Neural processing letters,* 38(3), 465-486. 2013

[14] I. McLoughlin, H. Zhang, Z. Xie et al., "Robust sound event classification using deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 540–552, 2015.

[15] C. McNeill, July 2018, https://www.gutcheckit.com/blog/smart-home-adoption-au-blog/

[16] F. Petitjean, G. Forestier, G.I. Webb, A.E. Nicholson, Y. Chen, and E. Keogh. "Dynamic time warping averaging of time series allows faster and more accurate classification". *In 2014 IEEE international conference on data mining*, pp. 470-479. 2014.

[17] J. Salamon and J. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–1, 2017

[18] P. Simard, D. Steinkraus, and J. Platt. "Best practices for convolutional neural networks applied to visual document analysis". *In ICDAR*, 2003.

[19] J. A. Stork, J. Silva, J.L. Spinello, and K. O. Arras, "Audio-Based Human Activity Recognition with Robots". *In International Conference on Social Robotics (ICSR'11), Amsterdam, The Netherlands,* 2011.

[20] H. Tomoki, et al. "Duration-controlled LSTM for polyphonic sound event detection." *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 25.11: 2059-2070. 2015.

[21] Y. Wang, "Polyphonic Sound Event Detection with Weak Labeling". *Diss. Google Inc*, 2017.

[22] Y. Wang and M. Florian, "A first attempt at polyphonic sound event detection using connectionist temporal classification." *Acoustics, Speech and Signal Processing (ICASSP),* 2017 IEEE International Conference on. IEEE, 2017.

[23] S.Wei, K.Xu, D.Wang, F.Liao, Wang, H., and Q.Kong, "Sample mixed-based data augmentation for domestic audio tagging". *arXiv preprint arXiv:1808.03883*. 2018.

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