WEAKLY-SUPERVISED AUDIO EVENT DETECTION USING EVENT-SPECIFIC GAUSSIAN FILTERS AND FULLY CONVOLUTIONAL NETWORKS

Ting-Wei Su[1], *Jen-Yu Liu*[1,2], *Yi-Hsuan Yang*[1]

[1]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
[2]Department of Electrical Engineering, National Taiwan University, Taiwan

ABSTRACT

Audio event detection aims at discovering the elements inside an audio clip. In addition to labeling the clips with the audio events, we want to find out the temporal locations of these events. However, creating clearly annotated training data can be time-consuming. Therefore, we provide a model based on convolutional neural networks that relies only on weakly-supervised data for training. These data can be directly obtained from online platforms, such as Freesound, with the clip-level labels assigned by the uploaders. The structure of our model is extended to a fully convolutional networks, and an event-specific Gaussian filter layer is designed to advance its learning ability. Besides, this model is able to detect frame-level information, e.g., the temporal position of sounds, even when it is trained merely with clip-level labels.

Index Terms— Weakly-supervised learning, audio events

1. INTRODUCTION

Audio event detection (AED), or sound event detection, can be used in many scenarios, and a great body of research work has been devoted to this topic. For example, when multimedia files are uploaded to a website, we can automatically recognize its content by analyzing the audio [1]. If AED is applied to surveillance cameras and home security devices, they can alert us when abnormal sound events such as screaming, shouting, or gun-shots occur [2, 3, 4]. Special purposes of sound events detection also include the work of Fischer *et al.* which classifies breath and snore [5]. Salamon *et al.* and Piczak apply AED to urban and environmental sound events classification. The formers focus their work on feature learning for urban sound classification [6], and the latter implements a model based on convolutional neural networks (CNN) to distinguish environmental sounds [7].

However, all these prior arts on AED relies on fully-supervised data for training. That is, an audio event is learned from clips that only and clearly contain this specific event [4, 6, 7, 8]. In real cases, such richly annotated data are generally hard to come by, and producing them can be very time-consuming. Additionally, in many cases, multiple audio events may occur at the same time, making it harder to collect pure and clean training data. The difficulty grows even higher when we try to scale up the number of sound classes.

On platforms like Freesound [1], there are a lot of audio clips tagged with specific events. However, instead of being omnipresent, the events may appear for a short period of time in a clip. Moreover, we do not even know the location of those events. These clips are called weakly-supervised data. If we can train a competitive model with these weakly-supervised files, our training data can be easily expanded.

Recently, a few people have also started to work on weakly-supervised learning (WSL). In the field of AED, Kumar *et al.* propose an approach of multiple instance learning [9], and suggest two frameworks respectively based on neural networks and support vector machine to solve the problem. The difference between our model and their work is that we use fully convolutional neural networks as the basic structure. We basically follow the model proposed by Liu and Yang for music auto-tagging [10] but with data augmentation on different volumes, which is essential in AED due to the uncontrolled recording environment.

In our model, we apply a deep CNN based model, which has performed successfully in visual object recognition and music auto-tagging [10, 11]. This model is able to not only detect the events appearing in an audio clip (i.e. making *clip-level prediction*) but also localize these events on the temporal axis (i.e. making *frame-level prediction*). It is important to have frame-level prediction. For example, in a several-hour surveillance video, we want to know not only the appearance, but also the exact time of gun-shots or screaming.

In this paper, we investigate whether a model based on WSL can detect the correct sound as accurate as state-of-the-art method trained with fully-supervised data. Thus, we firstly test the model on short clips, with each containing one clear sound event, to see if the model can perform clip-level classification of the events. Then, we do frame-level prediction. Our result of frame-level prediction will be compared with the ground-truth annotations. We build and evaluate such models with UrbanSound and UrbanSound8K datasets [12].

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research.

[1]Freesound.org: https://www.freesound.org/

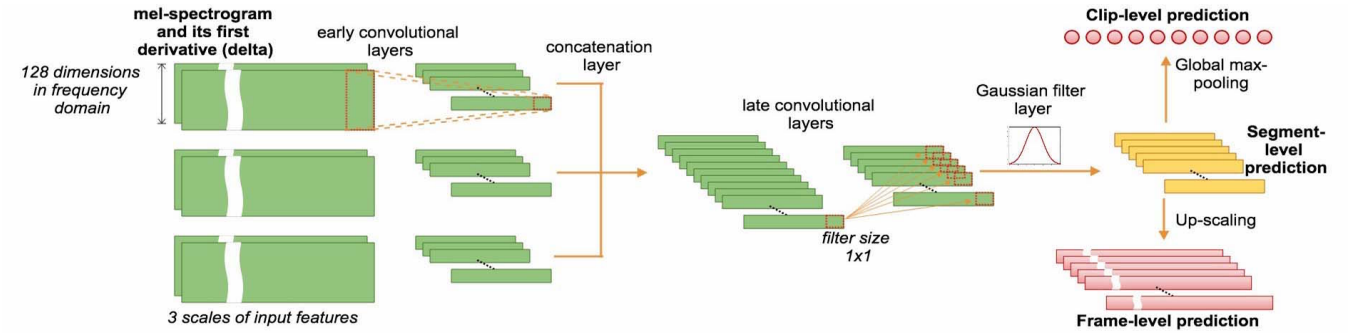


Fig. 1: The proposed FCN model. This figure briefly shows the status of data during every stage of our model.

2. MODEL

The primary difficulty of performing WSL is to locate the targets, so we create a model based on CNN [13], replace the fully-connected layers commonly used in CNN by convolutional layers, and apply a global max pooling [14] layer at the end of the model. Unlike fully-connected layers, which may mix all information on the temporal axis into the final result, global max pooling can select the most potential segment on the time domain to be a final clip-level prediction. Moreover, a model with this design, usually referred to as the fully convolutional networks (FCN), can easily take inputs of arbitrary length.

Audio events may have different temporal properties. Some events happen in merely a few frames, like dog bark, while others can last for several minutes, such as air conditioner. To deal with it, a Gaussian filter layer is inserted between the last later convolutional layer and the final max-pooling. This layer is implemented with convolution, but the filter weights are set to fit a centered Gaussian distribution:

$$g[t] = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{t^2}{2\sigma^2}}, \tag{1}$$

where σ is the standard deviation in the Gaussian distribution. g is applied to the signal s and scan through the signal:

$$(s \cdot g)[n] = \sum_{m=\frac{-M}{2}}^{\frac{M}{2}} s[m] \cdot g[n-m], \tag{2}$$

where M is a pre-determined filter size.

During training phase, the Gaussian filter will alter its standard deviation (σ) and learn a best value for each sound class. We expect that labels with longer min-necessary duration will receive larger σ to prevent from capturing relatively short activations, while labels with shorter min-necessary duration will get smaller σ to emphasize the short activations.

In addition, we investigate the importance of multi-scale features for AED, following the method used by Liu *et al.* [10]. Since our input feature is based on spectrogram, we will analyze features of multiple resolutions on time domain. Our approach is to extract log-scale mel-spectrograms of different window sizes by short-time Fourier Transform (STFT).

Overall, we propose a structure as shown in Fig. 1. The input features are N log-scale mel-spectrograms, obtained by performing STFT with N different window sizes, and their first derivatives (delta).

In this model, we separate the convolutional layers into two parts, early and late convolutional layers. The function of early layers is the same as conventional CNN model, and a max-pooling layer is provided after each convolutional layer. We do the early layers of every scale individually, and then join all of their results together at the concatenation layer. The distinguishing feature of fully-convolutional networks is performed at late convolutional layers. These layers are constructed of several filters with all their window sizes being 1, and there are no strides and no max-pooling after each layer. At the last layer, we set the number of filters to the total number of labels we are predicting. Afterwards, the output of late convolutional layers will go through a Gaussian filter layer as mentioned above, whose output is referred to as segment-level prediction. Then, the segment-level prediction will go through different processes for clip-level prediction and frame-level prediction.

2.1. Clip-level prediction

The clip-level output is generated by a global max-pooling layer after the late convolutional layers. This global max-pooling layer will choose the highest prediction in segment-level prediction as the final output of clip-level prediction.

Although the training and testing data we use in this paper are all single labeled, the task should be treated as a multi-label problem since most of the weakly-supervised data contain more than one kind of event. Furthermore, we do not know if a provided test clip contains the given events in a real-life scenario, and a multi-label approach can accommodate the possibility of detecting nothing.

Since the task is formulated as a multi-label problem, we use logistic function as the activation function in the last late convolution layer, and the mean of binary cross-entropies of each label as the cost function.

2.2. Frame-level prediction

To obtain frame-level predictions, we simply replace the global max-pooling layer in the trained model with a upscaling layer. This layer repeats every value on segment-level result for K times so that the final output has the same length as its input in temporal axis. K is the product of all stride sizes of early convolutional layers. For example, if the segment-level prediction of one label is [1, 0.8, 0], and we get two early convolutional layers with both their stride sizes being 2, the final output will become [1, 1, 1, 1, 0.8, 0.8, 0.8, 0.8, 0, 0, 0, 0]. Apart from this change, all the other layers are exactly the same as clip-level prediction.

3. DATASETS

We employ two datasets in our experiments: UrbanSound and UrbanSound8K. We use UrbanSound for training and frame-level evaluation, and use UrbanSound8K for clip-level evaluation. Both datasets are composed of ten different classes of sounds came from Freesound.org. The ten classes and the number of data can be seen in Table 1. Every clip in these datasets contains only one label.

UrbanSound dataset comprises 1302 recordings with their durations varying from from 1 second to over 30 seconds. All audio clips are annotated with the onset and offset of the sound events appearing in them. Because of these annotations, we are able to evaluate frame-level predictions. However, notice that these temporal annotations are used only for evaluating testing result. We do not use them in the training phase. Although the size of UrbanSound is not large enough, it is still the most suitable public dataset for our work.

UrbanSound8K is composed of 8732 short clips segmented from files in UrbanSound. Files in UrbanSound8K are less than or equal to 4 seconds and is labelled with one class. It is used for clip-level evaluation.

4. EVALUATION AND DISCUSSION

There are two major parts in our experiment. The first part shows the clip-level prediction of our model in different structures and compares the best result with a fully-supervised work which was also test on UrbanSound8K. In the second part, we will evaluate the result of frame-level prediction.

Our model is implemented with Theano [15] and Lasagne [2] and the features are extracted with Librosa [16]. The sampling rate of audio files is set to 44100 Hz. The input feature is composed of an 128-dimension mel-spectrogram and its first derivative, which is also 128 dimensions. Our model contains 2 early convolutional layers. Each layer comprises 60 filters with filter size 5 in time domain. We follow [17] and do convolution only on time domain. Therefore, only the first convolutional layer has its filter size 128 in frequency domain while the others have a filter size of 1. Each early convolutional layer is followed by a max-pooling layer with both pooling size and stride size being 4 in temporal axis (we are not doing pooling on the frequency axis). The late layers include 3 consecutive convolutional layers with their filter size being 1. We provide 128 filters for first two layers and 10 for the last one, which is same number of total labels in UrbanSound. As for the Gaussian filter layer, we set the filter size to 32, and the initial standard deviation σ of every class to 2. All dropout rates in this model are set to 0.5, and the learning rate is initialized to 0.006. We use Adaptive Gradient Algorithm (AdaGrad) as our update function [18], so the learning rate will be changed every time we update the parameters. As the training data consist of clips of varied duration, we set the batch size to 1 to simplify the training process. We run 300 epochs in every training set, and the model belong to the epoch with highest validation accuracy will be selected as the final model for an experimental setting.

[2]Lasagne: https://github.com/Lasagne/Lasagne

Table 1: This table provides the number of data in each dataset, the learned standard deviations σ of Gaussian filter layer after training, and the performance of our model. The US dataset evaluated with AUC score represents the performance of frame-level predictions, and the accuracy tested on US8K is our clip-level evaluation result.

	# of data			AUC	Acc.
Class	US	US8k	σ	US	US8k
Air conditioner	64	1000	2.93±0.30	0.612	76.7%
Car horn	125	429	1.32±0.05	0.807	69.0%
Children playing	158	1000	1.08±0.07	0.594	41.8%
Dog bark	337	1000	0.96±0.06	0.790	79.5%
Drilling	119	1000	2.05±0.13	0.764	52.3%
Engine idling	97	1000	2.97±0.19	0.688	51.8%
Gun shot	117	347	1.13±0.16	0.921	94.4%
Jackhammer	45	1000	1.93±0.10	0.704	39.8%
Siren	74	929	2.09±0.17	0.763	59.0%
Street music	166	1000	1.51±0.10	0.737	61.3%
Total	1302	8732		0.738	59.4%

Table 2: The comparison between the clip-level accuracy of basic setting and of adding one of multiple scales, data augmentation, and Gaussian filter to the model.

Basic	w/ Multiple scales	w/ Data augmentation	w/ Gaussian filter
25.93%	37.51%	39.78%	51.73%

4.1. Clip-level Evaluation

We begin with evaluating the following three modifications of the CNN model. First, in the basic structure, the window size of STFT is set to 1024. In the multi-scale setting, we instead use a structure of 3-scale input feature with window sizes being 1024, 4096, and 16384. As we see in the Table 2, multi-scale model outperforms the basic one with a large margin. Second, owing to the fact that weakly-supervised data may

vary quite a lot in the volume of audio events, and that the training data are scarce from UrbanSound, we augment our training data by adding and reducing 5db to every clip. Thus, we get triple training data and should make the model less sensitive to the effect of diverse volume. The result shows that it does improve the performance after data augmentation on volume.

Third, we add the Gaussian filter described in Section 2. We refer to the model with only single-scale input, no augmentation and no Gaussian filter as the 'basic' model. We than add one of these three modification to the model and investigate which one of them can more effectively improve the basic model. The result is shown in Table 2. We can see that the multi-scale feature is indeed helpful, improving the basic model by a margin. In addition, the use of data augmentation is also quite effective. More importantly, we found that Gaussian filter largely improves the performance. Furthermore, the final standard deviations of the Gaussian filters provide insights regarding the classes. From Table 1, we can see that sound classes with longer durations, such as "air conditioner" and "engine idling," obtain higher values, while classes with shorter durations, like "gun shot" and "dog bark," get lower values. Therefore, if "gun shot" and "engine idling" are both detected in a very short duration, gun shot is more likely to be highlighted by the Gaussian filter layer. On the contrary, the final prediction of engine idling will be reduced since this kind of sound is supposed to occur in a longer duration.

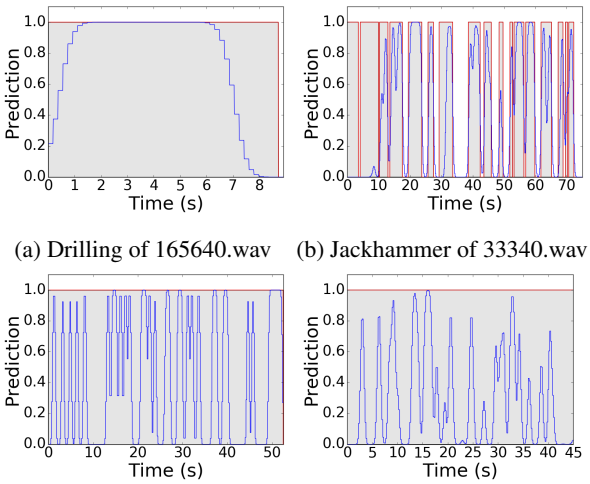
Finally, we turn on all three functions. As shown in Table 1, our model attains 59.4% accuracy, which is better than the result of using Guassian filter along. In a fully-supervised setting, Piczak achieved 73.1% accuracy by training on subsets of the UrbanSound8K itself. Although there is still a performance gap, we consider our result promising for it only uses weakly-supervised data from UrbanSound.

4.2. Frame-level Evaluation

We evaluate the frame-level result with average area under ROC curve (AUC) [19], and Table 1 shows the results of overall and each class. In addition, we provide some visualized frame-level results in Fig. 2. The ability of localizing events can be well seen. Generally, when an event is detected, their temporal locations are usually correct. Nevertheless, we still found some reasons that may impact on AUC score. These reasons may come from either the imperfection of ground-truth annotation or the shortage of our model.

First is the shortcoming of labels. Take '77927.wav' for example, in this clip, we detected a large amount of street music that is really inside the audio file. Nonetheless, the ground-truth annotation contains only dog bark, which appeared to be much smaller in volume and duration than street music. Another reason is related to the duration of an event. Some events are labeled in a long interval, but the sounds are not continuously happening in that interval. For example, in Fig. 2c, the dog was barking off and on throughout the clip, but the whole clip was labeled as one continuous event. On the other hand, one similar case occurs in Fig. 2d. However, this time is the fault of our model that it recognized only some components in street music instead of the whole region.

Like '77927.wav' we just mentioned, some clips may include more than one kind of sound. In '106905.aif', our model discovered not only engine idling and siren, but also street music in a very large scale. Furthermore, we can clearly hear a loud music in the audio file. Due to the limited space, we cannot show all these kinds of examples[3], but these results indicate that our model is able to detect polyphonic audio files even though no polyphonic training is done in this paper.



(a) Drilling of 165640.wav (b) Jackhammer of 33340.wav

(c) Dog bark of 105088.wav (d) Street music of 41364.wav

Fig. 2: Visualized frame-level results. Red line is the ground-truth, and blue denotes frame-level prediction.

5. CONCLUSION

We have used a model based on fully convolutional networks to demonstrate that weakly-supervised learning is able to discover and identify audio events. On the other hand, the result of frame-level evaluation shows that the model is able to perform sound event localization even when it is trained without temporal annotations.

In this work, we have very limited number of data for training for fair comparison with the state-of-the-art method. Nevertheless, the proposed model is able to train with weakly supervised data of arbitrary durations, without expensive human labors for data segmentation and cleaning. Besides, the proposed model could detect multiple sound events in an audio clip, which is not the case in UrbanSound8k. Therefore, a future work is to train with larger datasets and evaluate on multi-label datasets.

[3]For more demonstrations, please check out our project website: https://tweihaha.github.io/research/aed-by-cnn/

6. REFERENCES

[1] Y. Wang, S. Rawat, and F. Metze, "Exploring audio semantic concepts for event-based video retrieval," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1360–1364.

[2] J. C. Wang, C. H. Lin, B. W. Chen, and M. K. Tsai, "Gabor-based nonuniform scale-frequency map for environmental sound classification in home automation," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 607–613, April 2014.

[3] P. Foggia, N.Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: a system for detecting anomalous sounds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, 2015.

[4] P. Laffitte, D. Sodoyer, C. Tatkeu, and L. Girin, "Deep neural networks for automatic detection of screams and shouted speech in subway trains," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 6460–6464.

[5] T. Fischer, J. Schneider, and W. Stork, "Classification of breath and snore sounds using audio data recorded with smartphones in the home environment," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 226–230.

[6] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 171–175.

[7] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, Sept 2015, pp. 1–6.

[8] H. Lim, M. J. Kim, and H. Kim, "Cross-acoustic transfer learning for sound event classification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2504–2508.

[9] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," in *24th ACM International Conference on Multimedia (ACM MM)*. ACM, 2016.

[10] J.-Y. Liu and Y.-H. Yang, "Event localization in music auto-tagging," in *24th ACM International Conference on Multimedia (ACM MM)*. ACM, 2016.

[11] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free?-weakly-supervised learning with convolutional neural networks," in *Proc. CVPR*, 2015, pp. 685–694.

[12] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, New York, NY, USA, 2014, MM '14, pp. 1041–1044, ACM.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[14] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[15] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[16] B. McFee, M. McVicar, C. Raffel, D. Liang, O. Nieto, E. Battenberg, J. Moore, D. Ellis, R. Yamamoto, R. Bittner, D. Repetto, P. Viktorin, J. F. Santos, and A. Holovaty, "librosa: 0.4.1," Oct. 2015.

[17] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Proc. ICASSP*, May 2014, pp. 6964–6968.

[18] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[19] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.