BOSCH RARE SOUND EVENTS DETECTION SYSTEMS FOR DCASE2017 CHALLENGE

Shabnam Ghaffarzadegan[1], *Asif Salekin*[2], *Anirudh Ravichandran*[3], *Samarjit Das*[1], *Zhe Feng*[1]

[1] Robert Bosch Research and Technology Center, USA,
{shabnam.ghaffarzadegan, zhe.feng2, samarjit.das}@us.bosch.com
[2] University of Virginia, USA, as3df@virginia.edu
[3] University of California, San Diego, USA, anravich@eng.ucsd.edu

ABSTRACT

In this report, we describe three systems designed at BOSCH for rare audio sound events detection task of DCASE 2017 challenge. The first system is an end-to-end audio event segmentation using embeddings based on deep convolutional neural network (DCNN) and deep recurrent neural network (DRNN) trained on Mel-filter banks and spectogram features. Both system 2 and 3 contain two parts: audio event tagging and audio event segmentation. Audio event tagging selects the positive audio recordings (containing audio events), which are later processed by the audio segmentation part. Feature selection method has been deployed to select a subset of features in both systems. System 2 employs Dilated convolutional neural network on the selected features for audio tagging, and an audio-codebook approach to convert audio features to audio vectors (Audio2vec system) which are then passed to an LSTM network for audio events boundary prediction. System 3 is based on multiple instance learning problem using variational auto encoder (VAE) to perform audio event tagging and segmentation. Similar to system 2, here a LSTM network is used for audio segmentation. Finally, we have utilized models based on score-fusion among different systems to improve the final results.

Index Terms— Audio event detection, DCNN, DRNN, VAE, Audio2vec

1. SYSTEMS DESCRIPTION

In this technical report, we have introduced three systems proposed for rare sound event detection task of DCASE 2017 challenge [1]. The performance of the proposed systems are evaluated on the provided development set in the challenge website.

1.1. System1: CNN+RNN embeddings based system

Convolutional Neural Network (CNN) architectures for audio classification using the log-Mel spectrogram as features are being increasingly used [2, 3]. CNN's are naturally suited to exploit *image-like* log-Mel spectrograms to learn and identify both high-level and low-level features. In [3], CNN architectures were applied for large scale classification of audio. It was shown that classifiers that used the embeddings learned from the pre-final layer of the CNN produced impressive results for audio classification. Recurrent Neural Networks (RNN's) are another network choice that are useful for exploiting temporal dependencies in audio. Conditioning on previous and future frames, as in a bi-directional Gated Recurrent Unit (GRU) which can help to better localize the event in the clip. Based on this approach, we train CNN's, RNN's and plain Deep Neural Nets (DNN) in various capacities for audio classification at the frame level. However, instead of dumping embeddings learned for each sample and using them as input features to another separate classifier, we attach a few fully connected layers to the end of the final convolutional layer (or GRU layer, in case of an RNN), and train the system end-to-end. Next, we predict audio class at a frame level using the Deep CNN (DCNN) and Deep RNN (DRNN) network ensemble. Our DCNN and DRNN structures are detailed in figures 1 and 2.

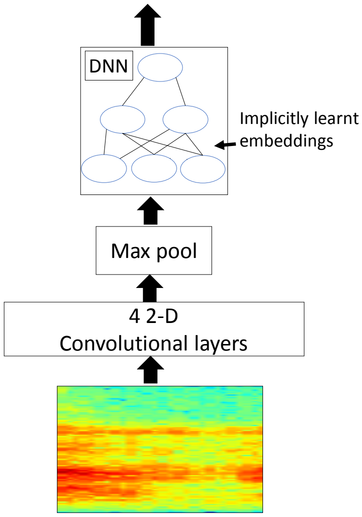

Figure 1: Deep CNN with a single-frame spectrogram input.

Based on the experiments, in case of the events *babycry* and *glassbreak*, the DNN gives only slightly inferior performance to the deep CNN and deep RNN networks, but falls well short in the case of *gunshot* detection.

1.2. System2: Audio2Vec based system

Acoustic features

System 2 and system 3 both employ 30-dimensional features selected from a pool of 272-dimensional features summarized in table 2. We have used random forest based feature selection method to chose the best subset for each class.

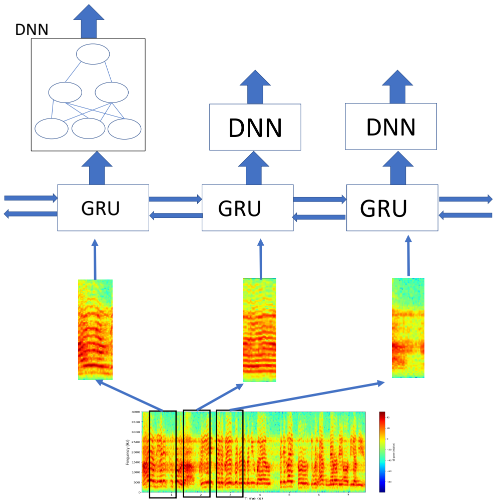

Figure 2: Deep RNN with spectrogram inputs.

DCNN based audio tagging

In this approach Dilated convolution neural network [4, 5] is applied for audio tagging. The goal is to identify if the target events are present in a 30 second audio clip. Compared to regular convolution layer (in CNN) with larger filters, Atrous convolution layer [5] (dilated filter) allows to effectively enlarge the field, covered by filters without increasing the number of parameters or the amount of computation. Since, for audio tagging we want to use larger filters, but have limited train samples, Atrous convolutions performs better compare to CNN. To progressively reduce the amount of features and the computational complexity of the network, Max pooling is used. Size of the Max pooling windows and strides are determined through iteration. Two fully connected dense layers are attached with the Atrous convolutions, which make binary event tagging decision, taking embeddings learned by Atrous convolution layers.

Audio2vec+LSTM based audio segmentation

In Audio2vec system, 30 second audio clips from the datasets are segmented into small frames (iterated between 100 to 500ms). Features from Table 2 are extracted for each of these small frames. Next, in the feature modeling stage a representation of the speech is developed that reflects the information for the specific task. Similar to [6], the state of audio in the small frames, audio words, are represented via K-means based Audio-Codebook model.

Table 1: Low level descriptive features and functionals computed on audio data; *min*: minimum; *max*: maximum; *std*: standard deviation; *var*: variance.

Features	Functionals
	Min
	Max
Zero crossing rate & Δ	std
Energy & Δ	var
Spectral centroid & Δ	skew
Pitch & Δ	kurtosis
MFCC & Δ	mean
	median

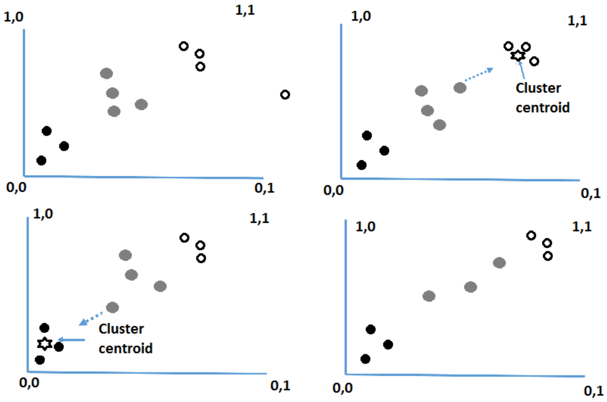

Figure 3: Audio2vec approach

Since, audio signals from targeted audio events are different from other events, audio states representing these events should be different from others. Also, some states occurs more frequently in the targeted event signals compare to other events. To identify and exploit these frequently or uniquely occurring audio states as features in our approach, we developed a novel audio word to vector conversion (Audio2vec approach), that generates audio word to vector dictionary.

Audio2vec dictionary generation approach, assigns an N-dimensional vector for each of the audio words representing the small frames of audio. In the initialization stage of vector generation, similar vector representations are assigned to audio words which uniquely occur on our targeted event, and similar vector representations are assigned to audio words which never occur on our targeted event. Audio words which are common, are assigned random vector representations. Figure 3 shows an example of our approach, where vector dimension is N=2. In Figure 3 (a), black points are the vectors (audio words) unique for targeted events, white points are ones never occurs in the targeted events, and the grey ones are common between two classes. Later, in the iterative stage of Audio2vec, every time an audio word (representing audio stage) occurs in the targeted event in training set, a small fragment of the cluster centroid of black points are added with the vector representation of that audio word, which moves that audio word closer to the targeted event clusters in the vector space, as shown in Figure 3 (c). Similarly, if an audio word occurs for non-targeted events, centroid of the white points are added with its vector representation, hence, moved farther from the targeted event clusters. As shown in Figure 3 (d), Audio2vec approach bring words (audio stages) occuring in our targeted events uniquely or more frequently closer in the vector space compare to others.

An overview of our audio segmentation system is shown in Figure 4. We segment 30 second audio into small frames and extract acoustic features (From Table 2). Feature representation from a small frame is converted to audio word using Audio-Codebook approach. Our Audio2vec dictionary (generated), converts audio words to vector representations. Generated Audio2vec vectors from small audio segments are then used as features in a many-to-many LSTM, which predicts the location of targeted event.

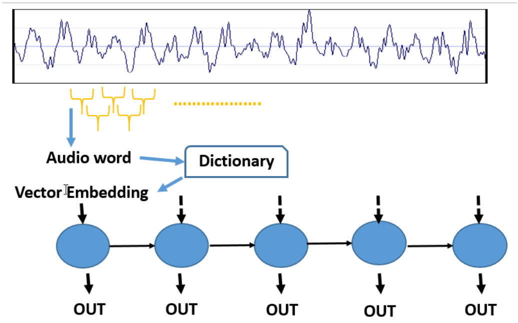

Figure 4: Overview of Audio2vec audio segmentation

1.3. System3: VAE based system

In this section, a novel weakly supervised deep learning structure is proposed for audio tagging to detect if an audio event exists in a given input and estimate its time boundary. This method is based on Variational Autoencoder (VAE) network for learning feature representation in the multiple instance learning (MIL) setting. In MIL framework, an input data (bag) is tagged as positive when at least one of the instances in the bag is a positive example. On the contrary, a bag is tagged as negative if all the instances in the bag are negative. This framework fits audio tagging task in which only a part of the audio contains the target event. To reformulate audio tagging problem into MIL framework, each audio recording is assumed as a bag and a fixed length window (e.g. 0.5 second) as instance.

Variational autoencoder (VAE) [7] is a directed graphical model consisting of encoder and decoder. In the encoder part, the input data is mapped to a latent representation $p(z|X)$, and in the decoder, the latent representation is mapped back to the data space $p(X|z)$. The VAE loss function is defined as:

$$\mathcal{L}_{VAE} = \mathrm{KL}(q(z|X) \parallel p(z)) - \mathbb{E}_{q(z|X)}[\log p(X|z)] \quad (1)$$

By regularizing the encoder with a prior over the latent distribution $p(z)$, $z \sim \mathcal{N}(0, \mathbf{I})$ where $\mathbf{I}$ is identity matrix, the VAE keep the representation z of different data sufficiently diverse.

By training two VAE networks, one on all instances from both positive and negative bag samples, $VAE_{\pm}$, and the other on negative bag samples only, VAE_{-}, we estimate the posterior of $p(z|X)$ and $p(z|X, Y = -1)$, noted as $VAE_{\pm}$ and VAE_{-}, respectively.

The proposed VAE structure is summarized in Figure 5, which consists of two VAEs sharing the same configuration, and a classifier network that take the latent layer in VAEs as inputs. The overall loss of the network consists of $\mathcal{L}_{VAE_{\pm}}$, $\mathcal{L}_{VAE_{-}}$ and the binary cross-entropy loss for classifier $\mathcal{L}_{clf}$.

During training, difference between the two posterior estimates $q_{\lambda_{\pm}} \| q_{\lambda_{-}}$ and $\mathcal{L}_{VAE_{\pm}}$, $\mathcal{L}_{VAE_{-}}$ is maximized using $\mathcal{L}_{clf}$. Training samples to $VAE_{\pm}$ network are randomly chosen from all the negative and positive, and the input to VAE_{-} is randomly chosen from negative instances only. During training, different positive and negative instance pairs are included. We also use the normalized reconstruction error from the VAE_{-} as sample weight to reduce the loss when the sample can be well reconstructed by the $VAE-$.

We have implemented our method in keras. RMSprop optimizer and a fixed learning rate of 0.001 and momentum of 0.9 are used throughout all the experiments. The networks wights are initialized to zero mean Gaussian noise with a standard deviation of 0.01.

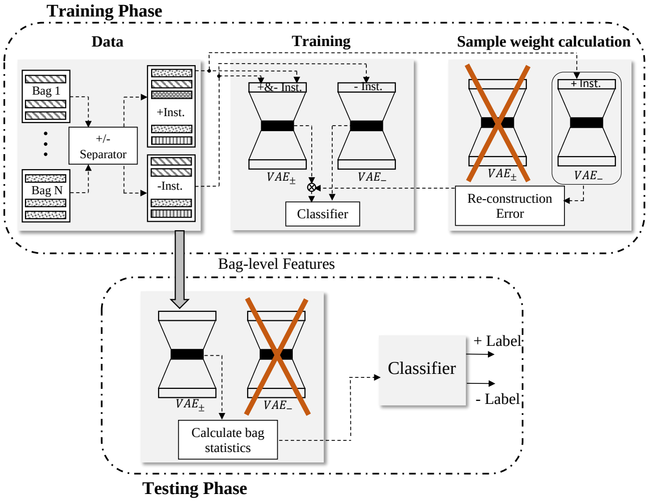

Figure 5: Variational autoencoder based feature representation diagram.

Two VAE network are trained to learn a better representation of the instances. However, bag-level features are required for classification task. As a result, maximum, minimum, standard deviation, mean and median encoding value along each latent dimension are extracted as bag-level representation.

As mentioned before, audio tagging problem is reformulated to MIL framework by assuming each audio recording as a bag and 0.1 or 0.5 second window as instance. The instance-level representations are fed to VAE networks to extract the bag-level features to train the classifier. In total, we have 500 audio bags with 148 or 599 instances in each bag (depending on the window length), and each instance is a 30 dimension feature vector.

Next, only positive outputs of the audio tagging pipeline will be processed for audio segmentation. To perform audio segmentation with VAE representation, the encoded features are fed to a many-to-many LSTM network to detect the time boundaries of audio signal.

1.4. Performance Evaluation

The evaluation results on the released development set for all the systems are summarized in Table 2. F-score and error rate (ER) are calculated based on the evaluation toolkit provided by the challenge.

For system1 evaluation, three different networks are compared: DNN, DCNN, and DRNN. For DNN architecture, 64 Mel-band features and their delta and acceleration coefficients with 3 context frame-concatenation from 40 ms windows with 50% overlap are used as input. A three layer network with [100,100,50] nodes in the hidden layers with drop-out and batch normalization [8] is utilized in the experiments. We trained this architecture for 5 epochs with a fixed learning rate of 0.003 and stochastic gradient descend (SGD) optimization. The network gives F-score of 84% for *babycry* and 91% for *glassbreak*. However, using this structure we could not get any reasonable performance for the *gunshot* class.

In DCNN structure, 128×16 rectangular spectrogram patches, 128 Mel-bands and 16 frame context are extracted from 100 ms windows with 80% overlap as input features. DCNN is trained in an

Table 2: F-score(%) and error rate (ER)(%) of audio event segmentation task; *tag*: audio tagging; *seg*: audio segmentation; system1:DCNN+DRNN; system2: CNN tag+Audio2vec seg; system3:VAE tag+VAE seg; dev: development set; eval: evaluation set

Task	**Audio event segmentation**									
Method	DCASE2017 Baseline		VAE tag+VAE seg		CNN tag+Audio2vec seg		DCNN+DRNN			
	dev		dev		dev		dev		eval	
	F-score	ER	F-score	ER	F-score	ER	F-score	ER	F-score	ER
Baby cry	72.0	0.67	84.7	0.30	92.2	0.16	89.5	0.17	75.9	0.5
Glass break	88.5	0.22	94.1	0.12	94.5	0.11	94	0.12	87.8	0.24
Gun shot	57.4	0.69	87.1	0.24	89.9	0.2	80	0.22	71.9	0.54
Average	72.7	0.53	88.6	0.22	92.2	0.16	87.83	0.17	78.6	0.43

Table 3: F-score(%) and accuracy(%) of fusion systems for audio event segmentation; dev: development set; eval: evaluation set

Task	**Audio event segmentation**							
Method	Binary fusion+VAE				Binary fusion+Audio2Vec			
	dev		eval		dev		eval	
	F-score	ER	F-score	ER	F-score	ER	F-score	ER
Baby cry	92.2	0.16	78.8	0.41	91.8	0.17	78.0	0.43
Glass break	94.3	0.12	91.5	0.16	91.5	0.11	87.5	0.24
Gun shot	87.1	0.24	52.3	0.93	89.9	0.2	49.8	0.98
Average	91.2	0.17	74.2	0.5	92.1	0.16	71.8	0.55

supervised setting. We tried to have a similar structure as *VGG* network [9]. However, due to the relatively small size of the dataset, we used only four 2D convolutional layers, with [16,16,32,32] learned square 3×3 kernels and Max pooling layers. We experimented with rectangular kernels, with higher strides along the frequency domain, but observed no improvement. Similar to DNN, this network is also optimized with SGD method. The DCNN network results F-score of 87% for *babycry* and 93% for *glassbreak*, and 79% for *gunshot*.

For the DRNN network, 64 Mel-band features from 120 ms window with 50% overlap and a 3 frame context is used, with the sequences of length ∼475 for a 30s audio clip. A bidirectional GRU's with SGD is trained, primarily avoiding LSTM's to control the number of parameters. 1e-4 regularization, and batch normalization with drop out are also utilized. The two stacked GRU layers output [256,128] dimensional hidden layers followed by [100,50] dimensional DNN hidden layers. We use sigmoid outputs for the output layers on all our networks, and *relu* activations on all other layers. This network gives F-score of 88% for *babycry* and 93% for *glassbreak*, and 75% for *gunshot*.

For Dilated CNN based audio tagging system, a three layer network with [50,50,50] nodes in the hidden layers with 20% drop-out rate and batch normalization is utilized in the experiments. Max pooling with pool size 2×2 strides is used. Two dense layer has been attached (with [20,1] nodes) with the Dilated CNN. We train this architecture with 20 epochs, with mean squared error loss function and RMSprop optimization. This network achieves 92%, 96% and 88% audio tagging F-score for events: *baby cry*, *glass break* and *gunshot*, respectively.

For Audio2vec vector generation, we generated vector of size 30, for each of the events, and perform 30 iteration of the Audio2vec feature separation approach.

For Audio2vec audio segmentation, two two-layer LSTM network with [50,50] nodes in hidden layers with 20% drop-out rate and batch normalization is utilized in the experiments. One two-layer LSTM network takes raw audio features and another takes generated Audio2vec vectors as input. These two LSTM networks are merged together, and a TimeDistributed dense layer is attached to generate many-to-many output for each of the small segments of the audio. We train this architecture with 30 epochs, with mean squared error loss function and RMSprop optimization. This network achieves 92.1%, 94.5% and 89.9% audio segmentation F-score for events: *baby cry*, *glass break* and *gunshot*, respectively (shown in table 2).

For VAE based audio tagging, we have trained two VAE networks with [512,256,512] hidden units and a 2-layer classifier with [64,64] hidden units shown in Fig. 5 in the ”Training” box. For the final classifier, radial basis function (RBF) kernel SVM has shown superior performance in audio tagging task compare to other classifiers such as: k-nearest-neighbor (KNN), linear kernel SVM, neural net, etc. We use the RMSprop optimizer, a fixed learning rate of 0.001, momentum of 0.9, *relu* activation function, 512 batch size and drop out rate of 0.25 throughout our experiments. The VAE network results 89%, 96% and 85% F-scores for *baby cry*, *glass break* and *gunshot* event, respectively.

Next, only the positive audio recordings (containing the target audio events) are processed via the VAE segmentation system. The VAE segmentation system structure is similar to the VAE tagging one with the difference that the latent representation from $VAE_{\pm}$ is used as input to a many-to-many LSTM network to detect the target events boundaries. The many-to-many LSTM structure here is similar to the one used in system 2. System 3 leads to final results of 84.7%, 94.1% and 87.1% F-scores for *baby cry*, *glass break* and *gunshot* event, respectively.

Finally, to further improve the results linear SVM based fusion system is used to combine the binary predictions of system 2 and 3. Given the results shown in Table 3, fusion system improves the final event detection systems with 91.2% and 92.1% average F-score for VAE-based and Audio2vec-based segmentation systems, respectively.

2. REFERENCES

[1] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, “DCASE 2017 challenge

setup: Tasks, datasets and baseline system," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, November 2017, submitted.

[2] J. Li, W. Dai, F. Metze, S. Qu, and S. Das, "A comparison of deep learning methods for environmental sound detection," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 126–130.

[3] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 131–135.

[4] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv preprint arXiv:1606.00915*, 2016.

[6] S. Pancoast and M. Akbacak, "Bag-of-audio-words approach for multimedia event classification," in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[7] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[8] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv:1502.03167*, pp. 1–11, 2015.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.