An Xception Residual Recurrent Neural Network for Audio Event Detection and Tagging

Gajarsky, Tomas; Purwins, Hendrik

Published in:
Proceedings of the 15th Sound and Music Computing Conference (SMC2018)

DOI (link to publication from Publisher):
10.5281/zenodo.1422563

Publication date:
2018

Document Version
Publisher's PDF, also known as Version of record

An Xception Residual Recurrent Neural Network for Audio Event Detection and Tagging

Tomas Gajarsky
Audio Analysis Lab
Aalborg University Copenhagen, Denmark
gajarsky.tomas@gmail.com

Hendrik Purwins
Audio Analysis Lab
Aalborg University Copenhagen, Denmark
hpu@create.aau.dk

T. Gajarsky and H. Purwins are with Sound and Music Computing and Audio Analysis Lab, Aalborg University Copenhagen.

ABSTRACT

Audio tagging (AT) refers to automatically identifying whether a particular sound event is contained in a given audio segment. Sound event detection (SED) requires a system to further determine the time, when exactly an audio event occurs within the audio segment. Task 4 in the DCASE 2017 competition required to solve both tasks automatically based on a set of 17 sounds (horn, siren, car, bicycle, etc.) relevant for smart cars, a subset of the weakly-labeled dataset called the AudioSet. We propose the Xception - Stacked Residual Recurrent Neural Network (XRRNN), based on modifications of the system CVSSP by Xu et al. (2017), that won the challenge for the AT task. The processing stages of the XRRNN consists of 1) an Xception module as front-end, 2) a 1×1 convolution, 3) a set of stacked residual recurrent neural networks, and 4) a feed-forward layer with attention. Using log-Mel spectra and MFCCs as input features and a fusion of the posteriors of trained networks with those input features, we yield the following results through a set of Bonferroni-corrected t-tests using 30 models for each configuration: For AT, XRRNN significantly outperforms the CVSSP system with a 1.3% improvement ($p = 0.0323$) in F-score (XRNN-logMel vs CVSSP-fusion). For SED, for all three input feature combinations, XRRNN significantly reduces the error rate by 4.5% on average (average $p = 1.06 \cdot 10^{-10}$).

1. INTRODUCTION

Deep neural networks are used for recognition and prediction of events or patterns. They have been successfully applied to image recognition and audio, e.g. in source separation [1], speech and music synthesis [2, 3] and acoustic scene analysis [4, 5]. AudioSet [6], a large-scale data set of nearly 2.1 million labeled sound clips, brings an opportunity to investigate sound detection further. AudioSet is created by taking 10 seconds long audio clips from YouTube videos. The data labels carry the information about the names of audio events located inside. The difficulty is that the clips can contain one or more audio events which can be overlapped and do not have to spread across the full length of the clip. The information about the start and end times of events is not available. Therefore, this kind of data is called weakly-labeled data [7]. This circumstance brings up another challenge of detecting the exact position of audio event within the clip.

The issue described in the previous paragraph was examined in Task 4 of DCASE 2017 competition, which evaluated systems for the large-scale detection of sound events within the AudioSet's subset of 17 classes from a traffic environment [8]. The assignment of the first sub-task was audio tagging (AT), where the audio events have to be recognized and labeled by the system. The second sub-task, sound event detection (SED), was dealing with prediction of the time stamps of the audio events.

The CVSSP system [9] ranked 1st and 2nd in these two sub-tasks. The team that won the SED sub-task achieved better error rate on the evaluation data with an ensemble of ConvNets with multiple analysis windows [10]. However, the input for SED was preprocessed in a different way than for the AT solution. The audio was segmented with duplicated labels to find the timestamps, therefore, the assumption of [10] was that every data segment contained all the labels. This approach has also a limitation to analyzing a small time window. The system that ranked 3rd [11] in SED is based on two deep neural network methods. One is training sample-level Deep Convolutional Neural Networks (DCNN) on raw waveforms. The other one makes predictions on aggregated features of multiscaled DCNN models. The team that ranked 3rd int the AT sub-task developed a DenseNet model trained on segmented log Mel filter banks [12]. Compared to the other DCASE participants, the method of [9] is unified without any assumption, which is together with the achieved score a reason why it was chosen to be the cornerstone for our project.

According to [9], the audio waveform is first transferred to a time-frequency (T-F) representation. Because the T-F representation is then treated as an image, some aspects of the current state-of-the-art CNNs used for image recognition can be introduced to the architecture. Therefore, we modify [9] inspired by the Xception [13], a successful CNN in image processing.

The paper is organized as follows. Section 2 introduces the architectures of neural networks, Section 3 describes the experiments, Section 4 shows the results, Sections 5 and

6 contain discussion and conclusion.

2. SYSTEM ARCHITECTURE

2.1 CVSSP System

The audio waveforms are transformed to log Mel spectrograms (logMel) and mel frequency cepstal coefficients (MFCC), which are then fed into the convolutional layers. Then a recurrent neural network (RNN) follows to capture the temporal context information followed by a fully connected neural network to determine the posteriors of each class at each frame. The probabilities of all frames are then averaged out to get the predicted probabilities for each tag. [9]

2.1.1 Preprocessing of data

The input audio data in wave format is first filtered using cutoff frequencies 0Hz and 8kHz, resampled to 16kHz and converted to logMels as well as MFCCs consisting of 64 Mel frequency bands and 240 frames using 1024 samples long Hamming windows and a hop size of 360 samples. The extracted features are then packed into HDF5 file together with corresponding names and labels. Mean and standard deviation is calculated on each frequency bin and stored to a scaler HDF5 file. HDF5 files in binary format help to accelerate the program [14].

2.1.2 Gated linear unit

The learnable gated linear unit (GLU) [15] is employed almost in every part of the original system, as a substitution for ReLU activation [16]. GLU is capable of regulating the information flow in between two consecutive layers. The gate value determines if the corresponding T-F unit is attended or ignored. The idea is that the network will be able to learn to focus on audio events and not on unrelated sounds. Thus, an attention mechanism is created. GLUs are defined as:

$$\mathbf{Y} = (\mathbf{W} * \mathbf{X} + \mathbf{b}) \odot \sigma(\mathbf{V} * \mathbf{X} + \mathbf{c}) \tag{1}$$

where $*$ expresses the convolution operator, $\odot$ the element-wise product and σ the sigmoid non-linearity. $\mathbf{W}$ and $\mathbf{V}$ are the convolutional filters, $\mathbf{b}$ and $\mathbf{c}$ are the biases. $\mathbf{X}$ stands for the input tensor. Equation 1 causes the output a linear mapping $(\mathbf{W} * \mathbf{X} + \mathbf{b})$ to be modulated by the gate $\sigma(\mathbf{V} * \mathbf{X} + \mathbf{c})$ [9]. Those GLUs occur almost in every part of the neural network (NN).

2.1.3 The gated convolutional block

Each convolutional layer in the CVSSP system is carried out in a block utilizing GLU as depicted in Figure (1).

2.1.4 The gated recurrent block

Following the convolutional layers, the utilization of temporal information is performed by recurrent layers. The final prediction in audio event detection depends on the whole input sequence. Therefore, a bidirectional RNN (Bi-RNN) which is a combination of an RNN that moves from the beginning of the sequence to the end and an RNN that moves backwards in the opposite direction [17] are applied.

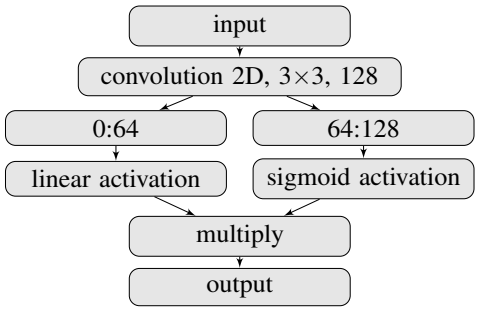

Figure 1: The gated convolutional block (GCB). The filters obtained from the convolutional layer are split into two equally sized parts of 64 units.

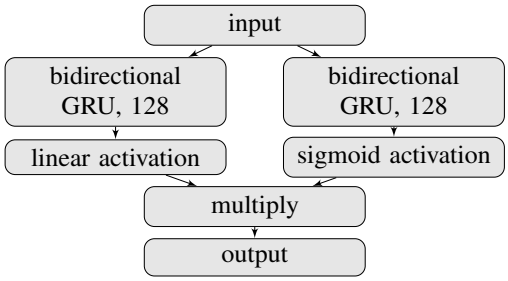

Figure 2: The gated recurrent block (GRB).

The RNN in [9] is created by two bidirectional layers employing the GLU concept. Each of the layers consists of 128 gated recurrent units (GRUs) [18].

2.1.5 Localization and classification layers

The last part of the system is devoted to localization and classification of the audio events. The temporal attention method is proposed to deal with this tasks. Two feed-forward (FF) layers consisting of 17 units are introduced to the architecture. These FF layers are time distributed which means that they produce predictions for every time-step of the sequence. The first FF with sigmoid activation function is used for classification at each frame and the FF with softmax activation function emphasizes the most prominent frames for each class. The outputs from both FF layers are then merged by element-wise multiplication

$$\mathbf{O}'(t) = \mathbf{O}(t) \odot \mathbf{Z}_{\mathbf{loc}}(t) \tag{2}$$

where the classification output of FF with sigmoid is defined as $\mathbf{O}(t)$ and the output of FF with softmax as the localization vector $\mathbf{Z}_{\mathbf{loc}}(t)$. Final predictions $\mathbf{O}''$ are obtained after $\mathbf{O}'(t)$ is averaged across the sequence

$$\mathbf{O}'' = \frac{\sum_{t=0}^{T-1} \mathbf{O}(t)}{\sum_{t=0}^{T-1} \mathbf{Z}_{\mathbf{loc}}(t)} \tag{3}$$

where T is frame-level resolution along the input data. The block for localization and classification (BLC) is depicted in Figure (3). [9]

2.1.6 CVSSP Architecture

The Figure (4) shows how the individual blocks are stacked from subsections 2.1.3, 2.1.4 and 2.1.5 to form the full Gated-CRNN-logMel.

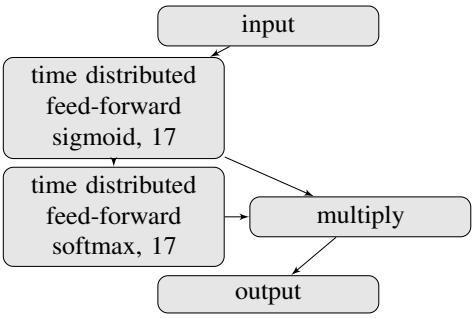

Figure 3: The block for localization and classification (BLC).

2.1.7 *Fusion of system results*

The final predictions submitted to the DCASE 2017 challenge were made by using two system fusion strategies. The first fusion of system results is conducted among the epochs in the same CVSSP system. This step improves the stability of the system [9]. The second one takes an average of the posteriors from different systems. Neither the paper [9] nor the published code specify neither the number of models nor the different architectures that were used for this method. It is only stated that part of this method was to train all the NNs on logMels and also on MFCCs separately and combine the predictions.

2.2 Xception-Recurrent Neural Network (XRRNN)

2.2.1 *Xception*

The Xception architecture introduced in [13] is a linear stack of depthwise separable convolutional layers with residual connections. The depthwise separable convolutions proposed by Sifre [19] perform a spatial convolution independently over each channel, followed by a pointwise convolution, i.e. a 1×1 convolution, projecting the channels output by the depthwise convolution onto a new channel space. They are similar to the Inception modules [20], but converge faster. The hypothesis is that the mapping of cross-channel correlations and spatial correlations in the feature maps of convolutional neural networks can be entirely decoupled. Therefore, richer feature representations are extracted. Chollet [21] showed that convolutional layers learn different shapes and patterns from the input data. Each class is represented by different set of these patterns, which enables the CNN to classify unseen data. The more varied shapes and patterns are created, the more robust and accurate model we obtain.

2.2.2 *Stacked bidirectional layers with residual connections*

The residual connections are introduced in [22] to ease the training of networks with deep architectures by making the input of lower layers available to the higher layers. Stacked bidirectional layers with residual connections were proven to produce more accurate results than simple stacked bidirectional layers in [23].

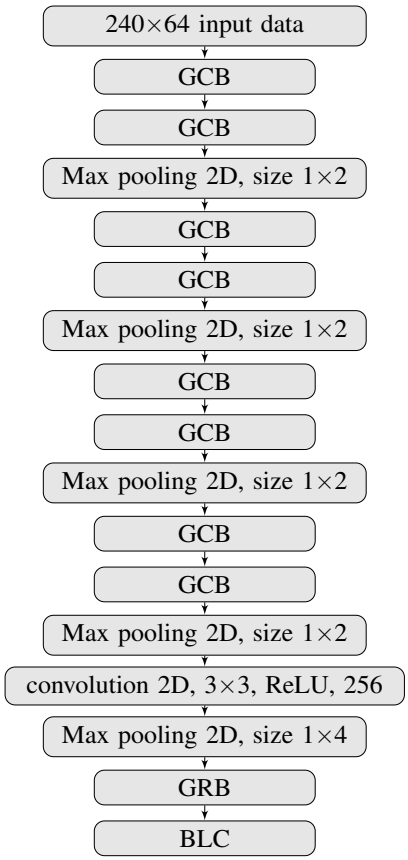

Figure 4: The full architecture of the CVSSP system [9]. (This diagram was created by analyzing the publicly available source code [1] .)

2.2.3 *Xception-Recurrent Neural Network*

We propose an architecture based on the CVSSP Gated-CRNN-logMel system [9]. The GCBs are replaced with the modified entry flow from Xception [13] where max pooling instead of longer strides takes care of the reduction of the frequency dimension. Every convolutional layer uses zero-padding,thus, the frequency dimension is reduced to 1 before entering the recurrent layers only by applying max pooling similarly as in the CVSSP system. The purpose of the very last convolutional layer is to extract the cross-channel correlations. The GRB is replaced with three stacked bidirectional layers with residual connections and the final part is the unchanged BLC from the CVSSP system. The full architecture is depicted in Figure (7) using the Xception separable convolutions block (XSCB) from Figure (5) and the Xception residual connection block (XRCB) from Figure (6).

3. EXPERIMENTS

For audio tagging (AT) and sound event detection (SED), we will compare the performance of our system (XRRNN) and the CVSSP system, using three different audio feature configurations as input: log Mel spectra, MFCC, and a fusion of both. We evaluate the performance using f-score (AT and SED) and error rate (SED).

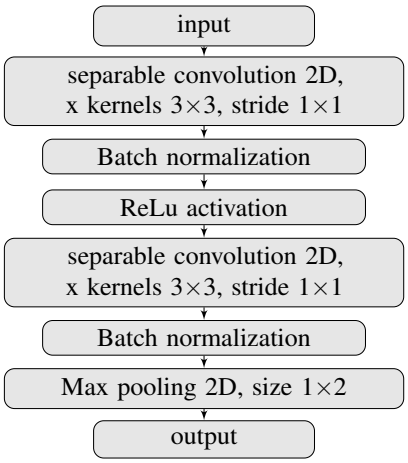

Figure 5: The Xception separable convolutions block (XSCB).

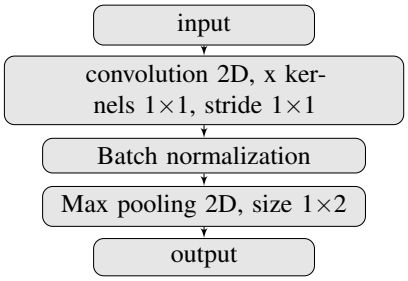

Figure 6: The Xception residual connection block (XRCB).

3.1 Input Data

All network architectures were trained on subset of Google AudioSet [6] formed by warning and vehicle sounds of 17 classes. We use the development dataset of the DCASE 2017 available at the website of Task 4 [2] consisting of 51172 sound clips in the training partition and 488 sound clips in the test partition. These sound clips were preprocessed as described in Subsection (2.1.1) to reduce the size of the data. The number of created T-F units from 160k samples is $240 \times 64 = 15,360$ per one clip.

3.2 Training Details

The data preprocessing is described in Subsection 2.1.1. The models are trained batch-by-batch using a batch generator of size 44 with data balancing [9] running in parallel with the model. The training lasts thirty epochs with 100 steps in each epoch.

For training, the binary cross-entropy loss function and the Adam stochastic optimization method [24] with learning rate 0.001 are used.

3.3 Evaluation Metric

The evaluation of the individual models was done using the saved states of the models from last ten epochs on the testing set. For AT, F-score is used as an evaluation metrics, and for SED, error rate is used. For each model architecture (XRRNN and CVSSP) and for both feature representations,

[2] `http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-large-scale-sound-event-detection`

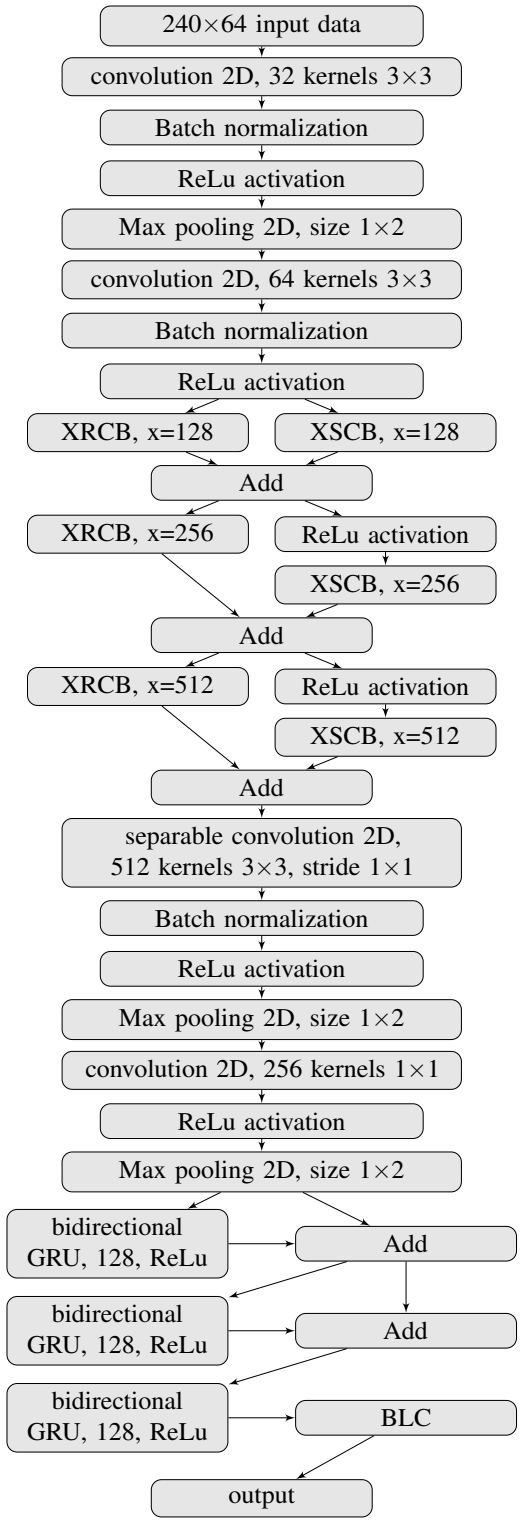

Figure 7: The architecture of the XRNN.

logMel spectra and MFCC, 30 identical models are trained. For the fusion models, 30 times, the posteriors for the same architecture trained with logMel spectra and with MFCCs are fused. Then for the AT sub-task, the F-scores and for the SED sub-task, the error rates between XRRNN and CVSSP are compared via a one-tailed independent t-test for each of the three input feature settings, resulting in $2 \cdot 3 \cdot 3 = 18$ independent t-tests. To correct for multiple testing, we use Bonferroni correction and multiply the resulting p-values of the individual tests by 18 to account for the performed 18 tests.

3.4 Implementation

Calculations were performed using an NVIDIA Titan X GPU. Keras 2 [25] with Tensorflow 1.4 [26] back-end was used for running the program. Also we used the Xu et al. (2017)'s implementation of their system [3].

4. RESULTS

In Table (1), performance means (F-score and error rate) across 30 runs for audio tagging (AT) and sound event detection (SED) are shown for our system (XRRNN) and the CVSSP system, using different audio features: log Mel spectra (first row), MFCC (second row), and a fusion of both (third row). In Table (2), the Bonferroni-corrected p-values for one-tailed independent t-tests comparing means across 30 runs for our system (XRRNN) and the CVSSP system are shown, using our three different audio feature configurations. As a result, for SED, our system (XRRNN) significantly outperforms the CVSSP system for all feature sets. The relative decrease of the error rate for the three feature sets is: 4.5% (logMel), 4.9% (MFCC), 4.2% (fusion). For AT, our system performs significantly better than CVSSP when using logMel features with a relative improvement of 3.1% in the F-score. XRRNN with logMel features performs significantly better than CVSSP with fusion ($p = 0.016$) with a relative improvement of the F-score of 1.3%. Tables (3, 4, 5) show the F-scores per class of the best XRRNN model trained on logMels. The best accuracy was achieved in the classes with loudest sounds like train horn, or civil defense siren and the worst in the car passing by class, which can easily get lost in the background noise.

5. DISCUSSION

The XRRNN was build step-by-step by replacing parts of the CVSSP architecture one-by-one. First we replaced the consecutive blocks of gated convolutional neural networks by an Xception front-end. Then we replaced their gated bidirectional RNN by a sequence of stacked residual bidirectional layers enhanced with residual connections. Later on, both of these components were integrated into the architecture together after they showed improved results individually. Finally we introduced the 1×1 - convolution placed in between the convolutional front-end and the recurrent layers. The only attention mechanism that remained in the architecture from the orginal CVSSP system is the BLC. It appears that one of the key ideas in Xu et al (2017)'s system, the gated convolutions, can be outperformed by using residual connections both in the front-end (within an Xception block) and in the RNN (by stacking recurrent layers with residual connections).

Looking at the F-score per class results, we see that not all loud sound classes achieved high accuracy. This can be caused by the fact that our dataset contains two groups of similar categories. First, in Table 3, we can see the f-scores of for vehicles with engine sounds. Those sounds contain a wide range of long-lasting static frequencies formed by car, bus, truck and motorcycle classes. Second, in Table 4, there is the group of sirens with constant modulating frequencies formed by ambulance (siren), fire truck (siren) and police car (siren). The XRRNN model achieved a high F-score within classes that have a unique sound opposed to the other ones, e. g. reversing beeps with periodic behavior, or skateboard with specific mid frequencies and a short burst when the wheels make an impact with the ground after a jump.

Our results for the CVSSP system differ a bit from the results reported in [9]. This might be due to the following reasons. From the paper by Xu et al. [9] and their accompanying implementation on github that we used to reproduce those results, some details in their system remained unclear. E.g. whereas in Fig.1 [9], they refer to 3 gated convolutional neural network blocks, in their github, they use 4 of those blocks. Their second strategy of system fusion is explained (in their Section 2.4) as "to average the posteriors from different systems with different configurations", where it remains unclear which configurations had been used exactly. For our XRRNN architecture, the concept of system fusion seems to be effective for solving the SED task but not for the AT task.

6. CONCLUSION

In this paper, we present the XRRNN system, based on the following modifications with respect to the CVSSP system that won the DCASE 2017 challenge for AT: an Xception module as front-end, followed by a 1×1 convolution, followed by stacked residual recurrent neural networks. For AT, XRRNN significantly outperforms the CVSSP system with a 1.3% improvement in F-score (XRNN-logMel vs CVSSP-fusion). For SED, XRRNN reduces the error rate by 4.5% on average.

Acknowledgments

The authors would like to thank Jose Luis Diez Antich for his help and NVIDIA for donating a Titan X GPU.

[3] https://github.com/yongxuUSTC/dcase2017_task4_cvssp

	F-score (AT)	Error rate (SED)	F-score (SED)
XRRNN-logMel	0.542	0.597	0.447
Gated-CRNN-logMel	0.526	0.625	0.434
XRRNN-MFCC	0.511	0.620	0.419
Gated-CRNN-MFCC	0.505	0.652	0.407
XRRNN-Fusion	0.540	0.579	0.447
Gated-CRNN-Fusion	0.535	0.604	0.446

Table 1: Performance means (f-score and error rate) across 30 simulations for audio tagging (AT) and sound event detection (SED) for our system (XRRNN) and the CVSSP system, using different audio features: log Mel spectra (first row), MFCC (second row), and a fusion of both (third row).

	F-score (AT)	Error Rate (SED)
XRRNN-logMel Gated-CRNN-logMel	$\mathbf{1.74 \cdot 10^{-6}}$	$\mathbf{1.01 \cdot 10^{-11}}$
XRRNN-MFCC Gated-CRNN-MFCC	0.24	$\mathbf{3.10 \cdot 10^{-10}}$
XRRNN-Fusion Gated-CRNN-Fusion	0.14	$\mathbf{2.96 \cdot 10^{-16}}$

Table 2: P-values with for one-sided independent t-tests comparing means of 30 runs (cf. Table (1)) for our system (XRRNN) and the CVSSP system, using different audio features. (see Table (1)) The p-values account the Bonferroni correction with respect to the total of 18 tests performed. Significance w.r.t. $\alpha = 0.05$ is indicated in boldface.

Bicycle	Skateboard	Car	Car passing by	Bus	Truck	Motorcycle	Train
0.483	0.712	0.462	0.178	0.400	0.460	0.476	0.688

Table 3: F-scores (AT) of individual vehicle classes for the best XRRNN-logMel model.

Car alarm	Ambulance (siren)	Fire engine, fire truck (siren)	Civil defense siren	Police car (siren)
0.706	0.455	0.568	0.744	0.586

Table 4: F-scores (AT) of individual warning classes with siren-like sounds for the best XRRNN-logMel model.

Train horn	Air horn, truck horn	Reversing beeps	Screaming
0.784	0.522	0.727	0.654

Table 5: F-scores (AT) of individual warning classes for the best XRRNN-logMel model.

7. REFERENCES

[1] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 31–35.

[2] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[3] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, and others, "Tacotron: Towards end-to-end speech synthesis," *arXiv preprint arXiv:1703.10135*, 2017.

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

[5] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *CoRR*, vol. abs/1608.04363, 2016. [Online]. Available: http://arxiv.org/abs/1608.04363

[6] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[7] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," *CoRR*, vol. abs/1605.02401, 2016. [Online]. Available: http://arxiv.org/abs/1605.02401

[8] T. Virtanen, A. Mesaros, T. Heittola, A. Diment, E. Vincent, E. Benetos, and B. M. Elizalde, *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*. Tampere University of Technology. Laboratory of Signal Processing, 2017.

[9] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Surrey-CVSSP system for DCASE2017 challenge task4," DCASE2017 Challenge, Tech. Rep., September 2017.

[10] D. Lee, S. Lee, Y. Han, and K. Lee, "Ensemble of convolutional neural networks for weakly-supervised sound event detection using multiple scale input," DCASE2017 Challenge, Tech. Rep., September 2017.

[11] J. Lee, J. Park, and J. Nam, "Combining multi-scale features using sample-level deep convolutional neural networks for weakly supervised sound event detection," DCASE2017 Challenge, Tech. Rep., September 2017.

[12] T. Vu, A. Dang, and J.-C. Wang, "Deep learning for DCASE2017 challenge," DCASE2017 Challenge, Tech. Rep., September 2017.

[13] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *CoRR*, vol. abs/1610.02357, 2016. [Online]. Available: http://arxiv.org/abs/1610.02357

[14] W. McKinney, *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. O'Reilly Media, 2012.

[15] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," *CoRR*, vol. abs/1612.08083, 2016. [Online]. Available: http://arxiv.org/abs/1612.08083

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814. [Online]. Available: http://dl.acm.org/citation.cfm?id=3104322.3104425

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[18] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[19] L. Sifre and S. Mallat, "Rigid-motion scattering for texture classification," *CoRR*, vol. abs/1403.1687, 2014. [Online]. Available: http://arxiv.org/abs/1403.1687

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[21] "How convolutional neural networks see the world," https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html, accessed: 2018-01-06.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[23] Y. Nie and M. Bansal, "Shortcut-stacked sentence encoders for multi-domain inference," *CoRR*, vol. abs/1708.02312, 2017. [Online]. Available: http://arxiv.org/abs/1708.02312

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[25] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[26] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/