INCORPORATING AUXILIARY DATA FOR URBAN SOUND TAGGING

Technical Report

Turab Iqbal, Yin Cao, Mark D. Plumbley, Wenwu Wang

Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey
{t.iqbal,yin.cao,m.plumbley,w.wang}@surrey.ac.uk

ABSTRACT

DCASE 2020 Task 5 presents a multi-label sound tagging problem for the detection of urban noises in acoustic scenes. The main theme is the use of auxiliary data to facilitate sound tagging. In this report, we provide a detailed description of our submission for Task 5 and present experimental results for the development set. Two different network choices are described: a pre-trained convolutional neural network (CNN) and a randomly-initialised gated CNN. To make use of the auxiliary information, we construct a feature vector based on the spatiotemporal metadata and use it in parallel with log-mel spectrogram features. Moreover, we address the presence of multiple annotations per recording by using a pseudo-labelling technique to estimate the true labels. Mean ensembling is also used with one of the proposed systems to combine several models.

***Index Terms*—** Sound tagging, multi-label, classification, spatiotemporal, convolutional neural network, pre-trained

1. INTRODUCTION

Monitoring noise pollution is of great concern in urban areas, where noise is abundant and often in need of regulation. In an effort to develop machine listening systems to automatically classify such urban noises, the Sounds of New York City Urban Sound Tagging (SONYC-UST) dataset [1] was released as part of the Sounds of New York City (SONYC) project [2]. The first iteration of this dataset appeared in Task 5 of the Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 challenge. Following this, the second iteration, SONYC-UST v2, was released for DCASE 2020 Task 5. Both datasets use crowdsourced annotations with multiple annotations per recording and include additional metadata pertaining to the recordings. The inclusion of spatiotemporal metadata is new in SONYC-UST v2, and is intended to be used as prior information for the sound tagging system to make use of.

This report details the four systems we submitted to participate in DCASE 2020 Task 5. These systems make use of the spatiotemporal context (STC) that is given with the annotations. All of the systems are convolutional neural networks that take as input a log-mel spectrogram and a feature vector representing a subset of the STC metadata. Three of the systems use networks pre-trained on the AudioSet dataset [3], while the remaining system uses a randomly-initialised network. Random time-frequency masking is used as data augmentation. To address the presence of multiple annotations per recording, we look at ways to estimate the 'true' label, which includes a pseudo-labelling approach inspired by the DCASE 2019 Task 5 submission from Adapa [4]. Finally, mean ensembling is also used with one of the systems.

The rest of the report is organised as follows. Section 2 describes the SONYC-UST v2 dataset, including the types of labels and the associated metadata. Section 3 describes the submitted systems[1], including how the features were extracted, how the training labels were estimated, and the network architectures. Section 4 presents the results for the development set of Task 5. Finally, Section 5 gives a summary of the work.

2. SONYC-UST DATASET

The SONYC-UST v2 dataset contains over $18\,000$ audio clips in total across a number of urban sound classes. The dataset is comprised of a training set, a validation set, and a test set. The training set and validation set are collectively known as the development set for the purpose of Task 5. Each audio clip is a 10-second clip recorded somewhere in New York City. In a given clip, several urban noises may be present, which makes this a *multi-label* sound tagging task. There are two types of classes in this dataset: coarse-level classes and fine-level classes. There are 8 coarse-level classes, which are further divided into one or more fine-level classes, giving a total of 29 fine-level classes. The sound tagging system can therefore output coarse-level predictions and/or fine-level predictions.

In this dataset, the annotations have been crowdsourced using the Zooniverse citizen science platform [5]. For the development set, at least three annotations have been provided per recording. Given the nature of crowdsourcing, the annotations for a given recording are generally different to each other. The exception to this is that 538 of the clips (out of 4308) in the validation set also include annotations that have been agreed upon by the SONYC team to be correct.

In addition to the class labels, each clip or specific annotation is associated with metadata identifying the annotator, the recording sensor, the perceived proximity, the time, and the location. The time includes the year, week, day, and hour. The location includes the borough, block, latitude, and longitude. Collectively, we will refer to the time and location as the spatiotemporal context (STC).

3. SYSTEM DESCRIPTION

3.1. Features

In this section, we describe the feature extraction stage. Two types of features were extracted: logarithmic mel-scaled (log-mel) spectrograms and STC feature vectors. The log-mel spectrograms were extracted from the audio clips, while the STC feature vectors were constructed from the STC metadata.

[1] https://github.com/tqbl/dcase2020_task5

3.1.1. *Log-mel spectrograms*

As a preprocessing step, the audio clips were first downsampled from 48 kHz to 32 kHz in order to reduce the size of the features. After this, the log-mel spectrograms were extracted by taking the short-time Fourier transform (STFT), squaring the result, scaling the frequency axis using mel filter banks, and scaling the magnitude using a logarithmic function. The spectrograms were extracted with 64 mel bins. Two different configurations were used for the window length and hop length of the STFT:

- **1024_512**: A window length of 1024 samples and a hop length of 512 samples, resulting in a 626×64 log-mel spectrogram.
- **2048_1002**: A window length of 2048 samples and a hop length of 1002 samples, resulting in a 320×64 log-mel spectrogram.

3.1.2. *STC features*

To construct the STC feature vector, we used the week, day, and hour i.e. the temporal metadata only. In our preliminary experiments, use of the spatial metadata did not improve the performance, so it was omitted. The week, day, and hour are given as integers in the dataset and will be denoted as w, d, t, respectively. Instead of simply creating a vector using these integers, the proposed feature vector is a combination of categorical and continuous data and is given by

$$\alpha = [\eta(\lfloor w/4 \rfloor), \eta(\lfloor t/3 \rfloor), \delta, \theta_{26}(w), \theta_{12}(t), d], \qquad (1)$$

where $\eta(z)$ is a one-hot encoding of some bounded integer z, $\delta \in \{0, 1\}$ indicates whether the day d is a weekend, and $\theta(x)$ is a triangular function defined as

$$\theta_B(x) = B - |x - B|$$

for some $x \in [0, 2B]$. The reason w and t are quantised is that we believe weekly or hourly precision is overly-specific; a more compact representation should help with learning. δ is present to emphasise that weekends are significantly different to weekdays. The purpose of using a triangular function is to 'wrap' the input to reflect its circular nature. For example, $t = 1$ and $t = 23$ are relatively far apart as values, but close in the context of time (1 AM and 11 PM). The triangular function resolves this. The final feature vector defined in (1) is a 26-element vector.

3.2. Label estimation

As explained in Section 2, each audio clip in the training set and validation set is associated with at least three annotations. The multi-label annotations, which are each of the form $y \in \{0, 1\}^K$, generally differ due to disagreements in labelling. To map them to a single annotation, we initially tried a number of basic methods, including (1) taking the mean, (2) taking the mean and then rounding, (3) doing a majority vote, and (4) taking the element-wise maximum. Out of these approaches, taking the maximum gave the best results.

Another approach, inspired by a submission from last year [4], is to estimate the labels using a learning algorithm. The SONYC team have agreed upon the labels of 538 of the recordings in the validation set. We shall call this subset the *verified subset*. As there are also crowdsourced annotations for these clips, the crowdsourced annotations can be used as the input for the learning algorithm, while the agreed-upon annotations are the ground truth. We used a two-layer neural network that takes the mean of the noisy annotations as input. The first fully-connected (FC) layer maps the input to a 128-dimensional embedding using an affine transformation followed by the ReLU function. This embedding is then mapped to the estimated annotation using the second FC layer followed by the sigmoid function. We used this model to map fine-level labels, so the input and output dimension is 29. This model was applied to the training set annotations of the SONYC-UST v2 dataset.

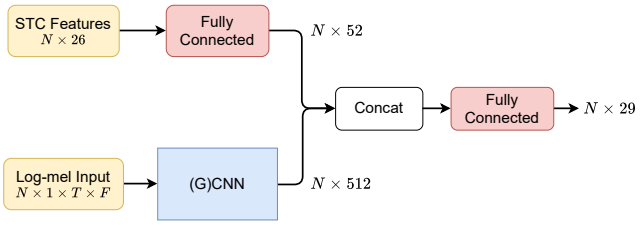

Figure 1: An illustration of the general network architecture, where N is the batch size. There are two branches: a branch for the STC features and a branch for the log-mel features. The outputs of these branches are concatenated and passed to a final fully-connected layer.

A serious concern with this pseudo-labelling approach is that the verified subset is very unbalanced; there are no examples for some of the fine-level classes. When training on this subset only, it overfits to the distribution of the subset. To improve upon this, some examples from the training set of SONYC-UST v2 were included. More specifically, let N_k denote the number of examples in the verified subset in which class k is present. If $N_k < 30$, $30 - N_k$ examples belonging to class k were sampled from the training set. This was done for each $k \in \{1, ..., 29\}$.

3.3. Models

In this section, we describe the models used for urban sound tagging. Although we used two different networks, the general design follows that of Figure 1. In this figure, it can be seen that there are two branches: a branch for the log-mel spectrogram and a branch for the STC features. The log-mel input is passed through a number of convolutional layers, the details of which will be given below. Parallel to this, the STC features are passed through a fully-connected (FC) layer with 52 output features, after which batch normalisation [6] is applied followed by the ReLU function. The output of the STC branch is a 52-feature embedding while the output of the log-mel branch is a 512-feature embedding. These embeddings are then concatenated to give 564 features. An FC layer is finally applied with a sigmoid non-linearity to give the class probabilities.

The log-mel branch was implemented using two different types of networks: a pre-trained CNN and a randomly-initialised gated CNN (GCNN) [7]. The pre-trained CNN is the 'CNN10' pre-trained audio neural network (PANN) proposed by Kong et al. [8, 9], which was pre-trained on AudioSet. It is comprised of eight convolutional layers and two FC layers. The final FC layer was omitted in our model as it is for mapping to class probabilities.

The GCNN is comprised of ten gated convolutional (GC) layers [7] with the following number of output feature maps: 64, 64, 128, 128, 256, 256, 512, 512, 512, 512. After every GC layer, batch normalisation is applied. After every two GC layers, the feature maps are reduced in size using 2×2 max pooling. Following the GC layers, the time and frequency dimensions are reduced to a scalar using average pooling. This results in a 512-feature embedding.

3.3.1. Training

The models were trained using the binary cross-entropy loss function with the Adam stochastic gradient descent algorithm [10]. We trained each model for 25 epochs with a batch size of 64 and a learning rate of 0.0005, which was decayed by 10 % after every two epochs. The exception is that the convolutional layers of the PANN were trained using a reduced learning rate of 0.000 25; this is because they only needed to be fine-tuned, and it led to better performance in our experiments. During inference, the top three epochs based on the primary performance metric (cf. Section 4) were selected and their predictions were averaged[2].

We used SpecAugment [11] on the log-mel inputs as a form of data augmentation. SpecAugment uses two types of transformations: time-frequency masking and time warping. We only used masking, as warping tended to produce worse results. For the frequency axis, up to $F = 8$ consecutive mel bins were masked and up to $m_F = 2$ frequency masks were applied. The number of bins, the location of the bins, and the number of masks were randomly chosen. Similarly, up to $T = 8$ consecutive frames in the time axis were masked and up to $m_T = 8$ time masks were applied.

3.3.2. Systems

Here, we describe the four systems we submitted for Task 5.

- **GCNN-Pseudo**: A single-model system based on the GCNN model that uses relabelled training data using the pseudo-labelling method (cf. Section 3.2). The 2048_1002 configuration was used to extract the log-mel spectrograms.
- **PANN-Pseudo**: A single-model system based on the PANN model that uses the pseudo-labelling method. The 1024_512 log-mel configuration was used.
- **PANN-Max**: A single-model system based on the PANN model that takes the element-wise maximum to estimate the labels. The 1024_512 log-mel configuration was used.
- **PANN-Ensemble**: A mean ensemble of four PANN models. Two of these models are the *PANN-Pseudo* and *PANN-Max* systems, while the other two are variants of these that were trained with the 2048_1002 log-mel configuration.

4. RESULTS

In this section, we evaluate the performance of the systems on the development set of Task 5. More specifically, we used the verified subset of the validation set for testing; this is the set of instances with verified annotations. To score the systems, we used the metrics proposed by the task organisers. The primary metric used for ranking the participants is the macro-averaged area under the precision-recall curve (AUPRC). The other metrics are the micro-averaged AUPRC and the micro-averaged F1 score. The AUPRC metrics consider the precision and recall for a wide range of thresholds and return the area under the precision-recall curve. The F1 score is a function of the precision and recall for a threshold of 0.5 specifically. For a detailed description of these metrics and the overall evaluation procedure, we refer the reader to Cartwright et al. [1].

As explained in Section 2, there are coarse labels and fine labels. The performance of the four systems for both label types is presented. To compare our systems to a baseline, we also include the scores for the official Task 5 baseline. The results for the fine-level predictions are presented in Table 1, while the results for the coarse-level predictions are presented in Table 2. In both tables, it can be seen that all of our systems significantly outperform the baseline across all three metrics. One can also see that the pre-trained (PANN) systems outperform the randomly-initialised (GCNN) system; this gap is especially pronounced for the primary macro AUPRC metric. Comparing *PANN-Pseudo* to *PANN-Max*, the results suggest that pseudo-labelling has a slightly positive effect on the micro AUPRC metric, but the effect is less conclusive for the other metrics. With the ensembled system, *PANN-Ensemble*, the performance is further improved in almost all cases. The best macro AUPRC score of 0.6548 gives a 24 % relative improvement and a 13 % absolute improvement over the official baseline system.

Table 1: Evaluation results of the fine-level predictions.

System	Micro AUPRC	Macro AUPRC	Micro F1
Baseline	0.7329	0.5278	0.6149
GCNN-Pseudo	0.7881	0.5645	0.6845
PANN-Pseudo	0.8111	0.6380	0.7067
PANN-Max	0.8045	0.6324	0.7127
PANN-Ensemble	**0.8214**	**0.6548**	**0.7174**

Table 2: Evaluation results of the coarse-level predictions.

System	Micro AUPRC	Macro AUPRC	Micro F1
Baseline	0.8391	0.6370	0.6736
GCNN-Pseudo	0.8740	0.6761	0.7506
PANN-Pseudo	0.8918	0.7514	**0.7795**
PANN-Max	0.8891	0.7515	0.7716
PANN-Ensemble	**0.8984**	**0.7667**	0.7776

5. CONCLUSION

In this technical report, we have given a detailed description of our submission for DCASE 2020 Task 5, in which the aim was to develop an urban sound tagging system that took advantage of the spatiotemporal metadata. Feature vectors were constructed using this auxiliary data and used in parallel with log-mel features to train one or more convolutional networks – both pre-trained and randomly-initialised. To address the presence of multiple annotations per audio recording, a pseudo-labelling approach was used. This labelling method was found to improve the performance marginally. Experiments on the development set showed that the best system achieved a relative improvement of 24% over the baseline.

6. ACKNOWLEDGEMENT

We would like to thank Qiuqiang Kong for providing the pre-trained neural network models. This research was supported by EPSRC grant EP/N014111/1, “Making Sense of Sounds”, and EPSRC grant EP/N509772/1, “DTP 2016-2017 University of Surrey”, which is part of the Doctoral Training Partnership (DTP) scheme.

[2] In this sense, all of our systems are ‘ensembles’, but we use this term specifically for ensembles of different models in this paper.

7. REFERENCES

[1] M. Cartwright, A. E. M. Mendez, J. Cramer, V. Lostanlen, G. Dove, H. Wu, J. Salamon, O. Nov, and J. Bello, "SONYC Urban Sound Tagging (SONYC-UST): A multilabel dataset from an urban acoustic sensor network," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE2019)*, New York University, NY, USA, 2019, pp. 35–39.

[2] J. P. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "SONYC: A system for monitoring, analyzing, and mitigating urban noise pollution," *Communications of the ACM*, vol. 62, no. 2, pp. 68–77, Jan. 2019.

[3] 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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, 2017, pp. 776–780.

[4] S. Adapa, "Urban sound tagging using convolutional neural networks," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE2019)*, New York University, NY, USA, 2019, pp. 5–9.

[5] R. Simpson, K. R. Page, and D. De Roure, "Zooniverse: Observing the world's largest citizen science platform," in *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*, New York, NY, USA, 2014, pp. 1049–1054.

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, vol. 37, Lille, France, 2015, pp. 448–456.

[7] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *International Conference on Machine Learning (ICML)*, vol. 70, Sydney, Australia, 2017, pp. 933–941.

[8] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *arXiv preprint arXiv:1912.10211*, Dec. 2019.

[9] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition (pretrained models)," Dec. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3576403

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICML)*, San Diego, CA, 2015.

[11] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, Apr. 2019.