ACOUSTIC SCENE CLASSIFICATION BY FUSING LIGHTGBM AND VGG-NET MULTICHANNEL PREDICTIONS

Rong Gong, Eduardo Fonseca, Dmitry Bogdanov, Olga Slizovskaia, Emilia Gomez, Xavier Serra

Music Technology Group, Universitat Pompeu Fabra, Barcelona

name.surname@upf.edu

ABSTRACT

This report provides a solution for the task 1 of DCASE 2017 challenge. We build two parallel audio scene classification systems – LightGBM and VGG-net. Their prediction scores are output respectively from the multichannel version of the *TUT Acoustic Scenes 2017* dataset. We perform a *linear logistic regression* method to fuse the (1) LightGBM, (2) VGG-net and (3) LightGBM+VGG-net multichannel scores. Finally, three outputs from the fused systems are submitted for the challenge. The evaluation is done on the development set.

***Index Terms*—** Acoustic scene classification, LightGBM, VGG-net, linear logistic regression late fusion

1. INTRODUCTION

This report provides a solution for the task 1 of DCASE 2017 challenge [1] - acoustic scene classification, of which the goal is to classify a test recording into one of the predefined classes that characterize the environment in which it was recorded.

This submission (sub2) differs from "Acoustic scene classification by ensembling gradient boosting machine and convolutional neural networks" (sub1) regarding the following points:

- The multichannel version dataset is used in sub2; the mono down-mixed version dataset is used in sub1.
- The VGG-net experiment in sub2 is carried out by using a "homemade" setup; a two-layers multiple filter shapes convolutional neural networks is implemented by using the official experiment framework in sub1.
- The *linear logistic regression* fusion model is used in sub2; an arithmetic mean fusion is used in sub1.

This solution trains two parallel classification models - gradient boosting machine (LightGBM[2], section 3.2) and convolutional neural networks (VGG-net [1], section 3.3) on the four channels version (section 3.1) of the *TUT Acoustic Scenes 2017* dataset (section 2). To yield the predictions on the evaluation dataset, we late-fuse the prediction probabilities of two models by using a *linear logistic regression model* (section 3.4).

2. DATASET

We use the *TUT Acoustic Scenes 2017* dataset, which is split into a development dataset and an evaluation dataset, of 4680 and 1620 audio recordings respectively. The development dataset is provided at the beginning of the challenge, together with ground truth. It includes 15 acoustic scenes[3] each of them containing 312 recordings of 10s. A four-fold cross-validation setup is provided so as to make results reported strictly comparable.

3. SYSTEM DESCRIPTION

3.1. Multichannel audio preprocessing

The audios in the dataset were recorded by using a binaural microphone. To take advantage the multichannel information embedded in this dataset, we extract four monoaural versions from each audio file:

- Left channel
- Right channel
- Average channel: (left channel + right channel) / 2
- Difference channel: (left channel - right channel) / 2

3.2. LightGBM system

Gradient boosting machine [2] is a powerful technique for building predictive models. It selects a loss as the objective function, and uses the addictive model of many weak learners—typically regression trees—to minimize the loss. The parameters of added trees are tuned by a gradient descent algorithm. There are two GBM frameworks which are used widely in the data science community: XGBoost [3] and LightGBM. The former is very popular among Kaggle community where it has been used for many competitions. The latter is a newcomer, which includes several improved features:

- It uses histogram based algorithms, which aggregates continuous features into discrete bins, to speed up training and reduce memory usage.
- It grows the tree by leaf-wise, which can reduce more loss than the level-wise algorithm.

In our experiments, we also found that LightGBM is faster than XGboost on training and achieves a slightly better overall classification accuracy. In consequence, we choose LightGBM as the GBM framework for the experiment.

[1] http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification

[2] https://github.com/Microsoft/LightGBM

[3] A list of the scenes together with more details about the dataset can be found in http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification.

3.2.1. Feature Extraction and Pre-processing

To consider the temporal characteristics, we segment each multi-channel recording into 10 equal length non-overlapped sequences. We then extract features on each sequence using *FreesoundExtractor*[4], a feature extractor from Essentia open-source library for audio analysis [4]. This extractor is originally used by Freesound[5] in order to provide sound analysis API and search by similar sounds functionality. It allows calculating hundreds of sound and music features. However, we discard some music-related features in rhythm, key, chords and tonal categories since we do not observe much musical trait in the development dataset. We further discard some feature statistics such as histogram and covariance matrix due to their high dimensionality and sparsity. The selected features and their dimensionality are listed in Table 1. The features are calculated on frame-level by using a 4096 samples frame size and a 2048 samples hop size. All other parameters are set to FreesoundExtractor default values. We then perform four statistical aggregations—mean, variance, mean of the derivative and variance of the derivative—to the frame-level feature vector of each sequence. Finally, a $\mathbb{R}^{820x1}$ (205×4) feature vector is output for each sequence. In the cross-validation experiment, we fit a mean and variance standardization scaler for each fold by using the features of the training set, which is then used for scaling the training and test set. In the final prediction step, we fit a standardization scaler for the whole development dataset and then apply it to the evaluation dataset.

Table 1: Selected features extracted by *FreesoundExtractor*. Dim: dimensionality.

Feature name	Dim	Feature name	Dim
Bark bands energy	32	Tonal features	3
ERB bands energy	23	Pitch features	3
Mel bands energy	45	Silence rate	3
MFCC	13	Spectral features	32
HPCP	38	GFCC	13

3.2.2. LightGBM Parameters

Since ASC is a multiclass classification problem, we use logarithmic loss as the objective function, which yields a $\mathbb{R}^{15x1}$ prediction probability for each sequence. The three most important parameters are set as *i)* Learning rate: 0.05, *ii)* Number of trees: 500 *iii)* Number of leaves: 255. All other parameters are default values. All parameters are held unchanged through the 4-fold cross-validation experiment and the model for the prediction of the evaluation dataset.

We keep all 820 dimensions features because according to our pilot experiment, to remove irrelevant features only affected the training speed rather than improved the prediction accuracy. The parameter tuning process has also been simplified because several techniques [5] in LightGBM such as *weak learner*, *Taylor approximation of the loss function* and *bagging*, make the system robust to over-fitting.

3.3. VGG-net system

In this section, we describe the convolutional neural network system – VGG-net. This architecture is taken from the paper [6]. However, some minor changes have been done to adapt it to our input representation.

3.3.1. Input representation and preprocessing

We use the 96 bands *log-mel* energies as the network input representation, which is calculated by Essentia open-source library [4] as follows:

1. We calculate the short-time Fourier transform (STFT) with 2048 frame size and 1024 hop size on the multichannel recordings.
2. The STFT spectrogram is then used to calculate the 96 bands Mel energies with an 11 kHz high frequency bound.
3. The Mel energies are converted finally to the log scale by using the formula $\log(100000 \cdot \mathrm{Mel} + 1)$.

We then segment the *log-mel* spectrogram of each recording into 10 equal length non-overlapped sequence, each contains 43 frames. In the cross-validation experiment, we fit a mean and variance standardization scaler for each fold by using the *log-mel* representations of the training set, which is then used for scaling the training, validation and test set. In the final prediction step, we fit a standardization scaler for the whole development dataset and then apply it to the evaluation dataset.

3.3.2. Architecture

The VGG-net architecture is depicted in table 2. All convolutional layers are initialized by He uniform initializer and regularized by *l2* regularizer with a parameter of $1e-5$. The loss function is categorical cross-entropy and the optimizer is ADAM. The system is implemented using the Keras library[6].

Table 2: VGG-net architecture; BN: Batch Normalization; ELU: Exponential Linear Unit.

Input 1×43×96
5×5 Conv(padding:*same*)-32-BN-ELU
3×3 Conv(padding:*same*)-32-BN-ELU
2×2 Max-pooling + Drop-out(0.3)
3×3 Conv(padding:*same*)-64-BN-ELU
3×3 Conv(padding:*same*)-64-BN-ELU
2×2 Max-pooling + Drop-out(0.3)
3×3 Conv(padding:*same*)-128-BN-ELU
3×3 Conv(padding:*same*)-128-BN-ELU
3×3 Conv(padding:*same*)-128-BN-ELU
3×3 Conv(padding:*same*)-128-BN-ELU
2×2 Max-pooling + Drop-out(0.3)
3×3 Conv(padding:*valid*)-512-BN-ELU
Drop-out(0.5)
1×1 Conv(padding:*valid*)-512-BN-ELU
Drop-out(0.5)
1×1 Conv(padding:*valid*)-15-BN-ELU
Global average pooling
15 way Soft-max

[4]http://essentia.upf.edu/documentation/extractors_out_of_box.html

[5]https://freesound.org/

[6]https://github.com/fchollet/keras

3.4. Late fusion

For each predicting sequence, both LightGBM and VGG-net systems output a $\mathbb{R}^{15x1}$ probability score. Thus, for the four channels, we obtain in total 8 scores: $proba^{\text{left}}_{\text{LightGBM}}$, $proba^{\text{right}}_{\text{LightGBM}}$, $proba^{\text{average}}_{\text{LightGBM}}$, $proba^{\text{difference}}_{\text{LightGBM}}$, $proba^{\text{left}}_{\text{VGG-net}}$, $proba^{\text{right}}_{\text{VGG-net}}$, $proba^{\text{average}}_{\text{VGG-net}}$, $proba^{\text{difference}}_{\text{VGG-net}}$. To fuse these scores on the evaluation dataset, we use *linear logistic regression* late fusion method, which has been implement in FoCal multi-class MATLAB toolkit[7]. This fusion method has been used also in paper [6].

We train three regression fusion models respectively for (1) LightGBM, (2) VGG-net and (3) LightGBM+VGG-net multichannel scores by using the development test set (dev) of each fold (*i*). We use the toolkit function *train_nary_llr_fusion* to train the fusion models and then apply them to predict the scores on the evaluation dataset (eval) by using the function *apply_nary_lin_fusion*:

Model 1 or 2 training on the development test set for fold i:

$$model^{\text{dev}}_{1or2,foldi} = train\_nary\_llr\_fusion(proba^{\text{left,dev}}_{1or2,foldi}, proba^{\text{right,dev}}_{1or2,foldi}, proba^{\text{average,dev}}_{1or2,foldi}, proba^{\text{difference,dev}}_{1or2,foldi}) \quad (1)$$

Applying model 1 or 2 to predict scores on the evaluation dataset for fold i:

$$proba^{\text{eval}}_{1or2,foldi} = apply\_nary\_llr\_fusion(proba^{\text{left,eval}}_{1or2,foldi}, proba^{\text{right,eval}}_{1or2,foldi}, proba^{\text{average,eval}}_{1or2,foldi}, proba^{\text{difference,eval}}_{1or2,foldi} | model^{\text{dev}}_{1or2,foldi}) \quad (2)$$

Model 3 training on the development test set for fold i:

$$model^{\text{dev}}_{3,foldi} = train\_nary\_llr\_fusion(proba^{\text{left,dev}}_{1,foldi}, proba^{\text{right,dev}}_{1,foldi}, proba^{\text{average,dev}}_{1,foldi}, proba^{\text{difference,dev}}_{1,foldi}, proba^{\text{left,dev}}_{2,foldi}, proba^{\text{right,dev}}_{2,foldi}, proba^{\text{average,dev}}_{2,foldi}, proba^{\text{difference,dev}}_{2,foldi}) \quad (3)$$

Applying model 3 to predict scores on the evaluation dataset for fold i:

$$proba^{\text{eval}}_{3,foldi} = apply\_nary\_llr\_fusion(proba^{\text{left,eval}}_{1,foldi}, proba^{\text{right,eval}}_{1,foldi}, proba^{\text{average,eval}}_{1,foldi}, proba^{\text{difference,eval}}_{1,foldi}, proba^{\text{left,eval}}_{2,foldi}, proba^{\text{right,eval}}_{2,foldi}, proba^{\text{average,eval}}_{2,foldi}, proba^{\text{difference,eval}}_{2,foldi} | model^{\text{dev}}_{3,foldi}) \quad (4)$$

The final submitted predicting labels are resulted from these scores across four folds:

$$pred^{\text{eval}}_{1or2or3} = \arg\max \sum_i proba^{\text{eval}}_{1or2or3,foldi} \quad (5)$$

4. EVALUATION SETUP

We use the development dataset for training and testing both LightGBM and VGG-net systems, according to the suggested four-fold cross-validation setup. Since no parameter tuning is performed for LightGBM, we use the entire training set in each fold to train the model. For the VGG-net, a 10% validation set is randomly split from the training data in each fold to early-stop the training process with a 10 epochs patience. To evaluate the fused system, we carry out the late fusion as explained in Section 3.4. The development test set in each fold is split to 50%, 50%. The former is used to train the fusion models, and the latter is used to perform the evaluation. The metric used is classification accuracy, i.e., the number of correctly classified audio recordings divided by the total amount of recordings and we report the average accuracy across the four folds. The evaluation dataset is used to predict acoustic scenes with our final proposed systems for the challenge submission.

5. RESULTS AND DISCUSSION

Table 3 shows that the LightGBM+VGG-net fusion system achieves the best classification accuracy (86.8%). However, this system doesn't outperform the LightGBM significantly. After a close inspection, we found that the VGG-net performs considerably bad on the development test set and might already overfit the training set[8]. Thus we believe that the VGG-net can't effectively provide complementary information to the LightGBM+VGG-net system. We can confirm this assumption by looking into the table 4, because, in 8 out of 15 classes, the best accuracies come from the LightGBM fusion system.

The prediction labels on the evaluation dataset of the three systems listed in table 3 are submitted to the challenge.

Table 3: Overall fusion model accuracies.

Fusion models	Accuracy (%)
LightGBM	84.0
VGG-net	86.1
LightGBM+VGG-net	**86.8**

Table 4: Class-wise fusion model accuracies (%).

Classes	LightGBM	VGG-net	LightGBM+VGG-net
Beach	**91.7**	91.6	91.0
Bus	98.1	97.4	**99.4**
Cafe/restaurant	73.1	76.9	76.9
Car	96.8	96.2	**98.1**
City center	90.4	**93.6**	92.0
Forest path	**97.6**	94.4	97.6
Grocery store	84.6	**87.2**	85.3
Home	**86.8**	77.4	86.2
Library	**81.4**	70.5	80.1
Metro station	**90.4**	82.7	90.4
Office	**92.9**	86.5	91.6
Park	**74.4**	69.9	73.7
Residential area	**82.2**	66.7	64.7
Train	84.6	82.7	**86.5**
Tram	88.5	89.1	**90.4**

6. CONCLUSION

This report provides a solution for the task 1 of DCASE 2017 challenge. Two audio scene classification systems – LightGBM

[7]https://sites.google.com/site/nikobrummer/focalmulticlass

[8]The accuracy of the non-fused VGG-net is not reported in this report.

and VGG-net are constructed. We output the prediction scores from the multichannel version of the dataset. We perform a *linear logistic regression* method to fuse the LightGBM, VGG-net and LightGBM+VGG-net scores respectively.

In the future, we plan to tune the VGG-net so that it can provide useful information for the fusion system.

7. ACKNOWLEDGMENT

This work is partially supported by the European Research Council under the European Union's Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583), and the European Union's Horizon 2020 research and innovation programme under grant agreement No 688382 "AudioCommons". We are grateful for the GPUs donated by NVidia.

8. REFERENCES

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

[2] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[3] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *CoRR*, vol. abs/1603.02754, 2016.

[4] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra, "Essentia: An open-source library for sound and music analysis," in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM '13. New York, NY, USA: ACM, 2013, pp. 855–858.

[5] H. Zhang, S. Si, and C.-J. Hsieh, "GPU-acceleration for Large-scale Tree Boosting," *ArXiv e-prints*, June 2017.

[6] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "Cp-jku submissions for dcase-2016: A hybrid approach using binaural i-vectors and deep convolutional neural networks," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2016.