MLP-BASED FEATURE LEARNING FOR AUTOMATIC ACOUSTIC SCENE CLASSIFICATION

Juliano Henrique Foleiss

Universidade Tecnológica Federal do Paraná
Computing Department
Campo Mourão, PR – Brasil
julianofoleiss@utfpr.edu.br

Tiago Fernandes Tavares

University of Campinas
School of Electrical and Computer Engineering
Campinas, SP – Brasil
tavares@dca.fee.unicamp.br

ABSTRACT

This paper presents an experimental setup for feature learning in the context of Automatic Acoustic Scene Classification. The setup presented in this paper has been successfully used for Automatic Music Genre Classification in [1]. First a MLP is trained with audio frames calculated from a 2048-sample STFT and one-shot encoding. Then, the activations of each hidden layer of the MLP are stored as learned features for the entire dataset. Such features are then used to train Random Forests in order to increase classification performance. Our results on the DCASE 2017 development dataset reaches 80% accuracy across supplied folds.

Index Terms— Feature Learning, Random Forests, MLP

1. INTRODUCTION

Automatic Acoustic Scene Classification (AASC) is an important audio signal processing application that has several potential uses in scenarios such as security, surveillance and context-aware consumer applications. In a sense this task is an instance of a broader audio classification problem, in which a particular sound signal is associated with a semantic label. Usually for this problem a set of features is calculated for each instance in a given dataset. It is expected that features calculated from semantically-related instances yield similar values, as long as the features chosen to represent the audio signal are correlated with the classes they represent.

Traditionally features are hand-crafted functions that describe certain aspects of audio signals [2]. Such functions are chosen based on previous domain knowledge about the problem being solved, thus require domain specialists to engineer them. Feature engineering can be costly and time-demanding, since finding a set of features that can satisfactorily solve the problem at hand may be tricky. Although there are some features that describe audio quite satisfactorily for a wide range of classification problems [3], they are usually sub-optimal and require fine-tuning both the feature extractor and the classifier.

Another approach to feature engineering is Feature Learning [1]. In this context, features are learned directly from the data, thus eliminating the need to manually engineer them based on domain knowledge. On the other hand, learning meaningful features and preventing over-fitting requires much more data as the number of input dimensions increases. In order to mitigate this problem regularization techniques may be used, such as Dropout [4]. In this work we use the neural network presented in [1] to learn features suitable for AASC. Learned features are then used to train Random Forests, which in turn yields class predictions.

2. OUR SYSTEM

Learned features are the activations of hidden layers of the neural network. Thus, it is necessary to train the network before extracting learned features for both training and left-out data. Once the network is trained, features are calculated for each audio frame by feeding them to the network and extracting the activations of every hidden layer. The neural network used is one of the networks presented in [1]:

1. Input layer with 1025 neurons, one for each absolute value of positive frequency bins of a 2048-sample STFT with 50% overlap;
2. Fully connected hidden layer with 500 neurons and `ReLU` activation function;
3. Dropout Layer with 25% chance of dropout;
4. Fully connected hidden layer with 500 neurons and `ReLU` activation function;
5. Dropout Layer with 25% chance of dropout;
6. Fully connected hidden layer with 500 neurons and `ReLU` activation function;
7. Dropout Layer with 25% chance of dropout;
8. Fully connected output layer with k neurons and `softmax` activation function. K is the number of output classes. For the DCASE 2017 Task 1 Dataset, $k = 15$.

Training was done through SGD optimization with 0.01 learning rate. 1000 epochs were run and the model corresponding to the smallest validation error was used for feature extraction. Audio frames were standardized to zero mean and unit variance. Each test set was standardized with the parameters obtained from standardizing the corresponding training set. All model parameter values correspond to the ones used in [1].

Once all features were extracted, a Random Forest classifier was trained with the features extracted from each of the three fully connected hidden layers. A classifier with features from all three hidden layers was also trained. Each Random Forest was tuned with a grid-search on the parameters presented in Table 1. Instead of using every frame for training and testing, successive frames were aggregated into 5s frames with 2.5s overlap.

The parameters were tested according to the guidelines presented in [5]. ANOVA % is the percentage of the top-scoring features to be used for model training. Optimizing over the top-scoring ANOVA features keeps the most complimentary set of features, leaving out uninformative ones, which can degrade Random Forest performance.

Parameter	Values
ANOVA %	{30, 53, 76, 100}
Min. Samples Leaf	{1, 2}
# of Estimators	{500, 1000, 1500}
Max. Features	{8, 16, 23, 31}

Table 1: Parameters tested

Fold	Accuracy	F1-Score
1	0.78	0.77
2	0.82	0.81
3	0.80	0.79
4	0.73	0.73
Average	**0.78**	**0.78**

Table 2: Results From The Neural Network Output (ProbSum & ProbMax)

Scene	Accuracy	F1-Score
beach	85.33	76.00
bus	85.33	85.00
cafe/restaurant	70.33	60.00
car	87.33	92.00
city_center	65.67	74.00
forest_path	94.00	88.33
grocery_store	86.67	75.33
home	82.00	72.33
library	78.33	82.67
metro_station	81.67	89.67
office	85.67	91.33
park	76.00	61.33
residential_area	55.33	60.67
train	98.67	79.00
tram	76.33	79.67

Table 3: Results per class across all 4 folds (in %, ProbSum)

3. EXPERIMENTS AND RESULTS

The neural network was implemented using the `Lasagne` deep learning framework [6]. We used the Random Forest implementation from the `scikit-learn` library [7].

We evaluated the system using the DCASE 2017 Task 1 development dataset [8]. In order to be able to compare results directly to other participants, we used the provided 4-fold cross-validation scheme. Notice that our system allows predictions directly from both the neural network and the Random Forest. Since more than one frame is used for each track, voting schemes may be used to yield a final prediction. Since the network output layer is a softmax layer, the activation of each neuron may be seen as the probability of that frame belonging to a certain class. Thus, for combining neural network outputs each a track we used two strategies: probability sum (ProbSum) and max probability (ProbMax). ProbSum sums the probabilities of each class through all examples for the track: the class corresponding to the largest sum is chosen. ProbMax, on the other hand, counts the number of times each class was the most probable through all examples for the track: thus, the class with the highest count is chosen. Table 2 shows the results using both ProbMax and ProbSum. For this dataset it seems that either strategy yields similar, but not exactly the same results. Although the accuracy and f1-score metrics are the same, confusion matrices expose a couple of differences (not shown).

Table 3 presents the average results for each acoustic scene across the 4 folds for the ProbSum fusion strategy. When comparing to the baseline system provided, we achieved better results for 9 scenes: beach, bus, cafe, forest path, grocery store, home, library, park and train.

Table 4 shows the results of the Random Forest predictions. Since multiple predictions are made for the same file, one for each aggregated frame chunk, we chose to use majority voting to determine the final prediction. In contrast to the results in [1], which was in the music genre classification problem, using further hidden layers from the input layer did not yield better results. Furthermore, selecting features from all layers did not increase performance either. When compared to the results in Table 2, using random forest did not improve results, although it performed more consistently than the output from the neural network across folds.

Lastly, Table 5 presents the average results for each acoustic scene across the 4 folds for Random Forest. When comparing to the baseline system provided, we achieved better results for 9 scenes: beach, bus, cafe, forest path, grocery store, home, library, park and train.

We have also calculated class predictions for the evaluation dataset. For this, we used the entire development dataset to train and tune our neural network and Random Forests. Two system outputs were computer. The first one, `Foleiss_UTFPR_task1_1` is the output from the neural network itself with ProbSum voting. The second one, `Foleiss_UTFPR_task1_2`, is the output from the Random Forest trained with features from all hidden layers and majority voting for deciding the final label.

4. REFERENCES

[1] S. Sigtia and S. Dixon, "Improved music feature learning with deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, 2014, pp. 6959–6963.

[2] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, Jul 2002.

[3] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, April 2011.

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[6] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri, *et al.*, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau,

Fold	**Acc (L1)**	**F1 (L1)**	**Acc (L2)**	**F1 (L2)**	**Acc (L3)**	**F1 (L3)**	**Acc (All)**	**F1 (All)**
1	0.80	0.80	0.79	0.79	0.78	0.77	0.80	0.80
2	0.81	0.80	0.82	0.81	0.80	0.80	0.81	0.81
3	0.78	0.78	0.78	0.78	0.78	0.78	0.79	0.78
4	0.79	0.79	0.78	0.78	0.78	0.79	0.79	0.78
Average	**0.80**	**0.79**	**0.79**	**0.79**	**0.79**	**0.79**	**0.80**	**0.79**

Table 4: Results From The Random Forest Output

Scene	**Accuracy**	**F1-Score**
beach	79.75	77.00
bus	90.25	92.00
cafe/restaurant	64.25	63.75
car	94.50	94.75
city_center	77.00	79.75
forest_path	91.25	89.50
grocery_store	83.50	71.00
home	75.75	74.75
library	85.50	81.00
metro_station	85.75	89.25
office	85.50	89.75
park	79.50	68.75
residential_area	64.00	65.50
train	88.75	73.50
tram	76.50	80.25

Table 5: Results per class across all 4 folds (in %, Random Forest – All Layers)

M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] A. Mesaros, T. Heittola, and T. Virtanen, “Tut acoustic scenes 2017, development dataset,” Mar. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.400515