HCDA13: Combining hand-crafted and deep audio descriptors for making sense of generic sounds

Aggelina Chatziagapi[1] and Theodoros Giannakopoulos[1]

{*aggelina, thodoris*}*@behavioralsignals.com*
[1]Behavioral Signals Technologies, INC

Abstract

The human ability to identify the source of a hitherto unheard sound is outstanding. Machine learning systems that will manage to categorize such sounds would contribute a lot to applications focusing on domains such as environmental monitoring, surveillance, home assistance and e-health. As part of the *Making Sense of Sounds Data Challenge*, we propose a method that employs a combination of deep learning and hand-crafted audio features to categorize audio data to the 5 broad categories (Effects, Human, Music, Nature, Urban). Our system consists of 3 main parts:

- a pre-trained Convolutional Neural Network (CNN)
- a pipeline that extracts hand-crafted audio features from audio signals
- a late-fusion approach that combines both deep and hand-crafted audio descriptors in a common SVM-based classification approach

For the training of the CNN, the ESC-50, TUT, Canal9, UrbanSound8K, LibriSpeech, TEDLIUM, Voxforge, GTZAN and FMA datasets were used, which are all publicly available and include audio recordings for various audio and music classification tasks. The mel-scaled spectrogram was extracted for each recording and fed as input to the CNN. The CNN architecture is based on VGG models [1], but with only one convolutional layer between successive max-pooling layers. In total, there are 5 convolutional layers with 64, 128, 256, 512 and 512 number of channels respectively, with stride fixed to 1 pixel and filter size 3x3 for the 4 last layers and 5x5 for the first one. Each convolutional layer is followed by a two-dimensional max-pooling layer with 2x2 filter size. The CNN ends up with 2 dense layers of 4096 and 1000 nodes, before the softmax output. Batch normalization and dropout are used after each layer and ReLU as the activation function. The CNN was developed using Keras [2]. For the evaluation of the network during training, the 20% of the data was kept after a random split.

Having the above pre-trained CNN, we extract the output of the last dense layer to use it as features for the main task of the challenge, in the context of transfer learning. The concatenation of these 1000 features and other 633 hand-crafted features composes the input of a SVM model with RBF kernel. This fusion aims at the best exploitation of the information gathered both from the development set and the pre-trained neural network on similar data, since the given amount of data is limited. As far as the hand-crafted features are concerned, we apply 9 statistics to the extracted audio features and their deltas for each segment of 5 seconds. The audio features consist of MFCCs, MFBs, spectral centroid and pitch and have been extracted with frame size 0.1 seconds and without overlap. Three mid-features (speech rate, arousal and pitch variation) are also added to the above statistics.

Cross-validation was performed on the development set to choose the best system among different CNN architectures, features combinations and SVM parameters. For this purpose, we grouped

the files of each sound type together and we splitted the data into 100 random folds (80% for training and 20% for testing), making sure each sound type exists only in train or test data. In this way, we hope to handle the possibility of a missing sound type on the development set and avoid overfitting. Furthermore, to approach the human ability to recognize the category of a sound (e.g. Music) even if they have not heard it before (e.g. a rare instrument). The SVM model was developed using Scikit-learn [3] framework and was optimized for the C parameter using Hyperopt [4]. This cross validation experimental procedure demonstrated a 65% accuracy using this "sound-type-independent" setup, while for the case of sound-type-dependent setup the accuracy was found to be equal to 85%.

References

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

[2] François Chollet et al. Keras. https://keras.io, 2015.

[3] 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, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[4] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, 8(1):014008, 2015.