DCASE 2019: CNN DEPTH ANALYSIS WITH DIFFERENT CHANNEL INPUTS FOR ACOUSTIC SCENE CLASSIFICATION

Technical Report

Javier Naranjo-Alcazar, Sergi Perez-Castanos, Pedro Zuccarello
Visualfy
Benisanó, València, Spain
{javier.naranjo, sergi.perez, pedro.zuccarello}@visualfy.com

Maximo Cobos
Universitat de València
València, Spain
maximo.cobos@uv.es

ABSTRACT

The objective of this technical report is to describe the framework used in Task 1, Acoustic scene classification (ASC), of the DCASE 2019 challenge. The presented approach is based on Log-Mel spectrogram representations and VGG-based Convolutional Neural Networks (CNNs). Three different CNNs, with very similar architectures, have been implemented. The main difference is the number of filters in their convolutional blocks. Experiments show that the depth of the network is not the most relevant factor for improving the accuracy of the results. The performance seems to be more sensitive to the input audio representation. This conclusion is important for the implementation of real-time audio recognition and classification on edge devices. In the presented experiments the best audio representation is the Log-Mel spectrogram of the harmonic and percussive sources plus the Log-Mel spectrogram of the difference between left and right stereo-channels ($L - R$). Also, in order to improve accuracy, ensemble methods combining different model predictions with different inputs are explored. Besides geometric and arithmetic means, ensembles aggregated with the Orness Weighted Averaged (OWA) operator have shown interesting and novel results. The proposed framework outperforms the baseline system by 14.34 percentage points. For Task 1a, the obtained development accuracy is 76.84%, being 62.5% the baseline, whereas the accuracy obtained in public leaderboard is 77.33%, being 64.33% the baseline.

Index Terms— Deep Learning, Convolutional Neural Networks, Acoustic Scene Classification, Mel-Spectrogram, HPSS, DCASE2019, Ensemble Methods, OWA

1. INTRODUCTION

Sounds carry a large amount of information about the everyday environment. Therefore, developing methods to automatically extract this information has a huge potential in relevant applications such as autonomous cars or home assistants. In [1], an audio scene is described as a collection of sound events on top of some ambient noise. Given a predefined set of tags where each tag describes a different audio scene (e.g. airport, public park, metro, etc.) and given an audio clip coming from a particular audio scene, Audio Scene Classification (ASC) consists in the automatic assignment of one single tag to describe the content of the audio clip.

The objective of the Task 1 of DCASE 2019 Challenge [2] is to encourage the participants to propose different solutions to tackle the ASC problem in a public tagged audio dataset. The first edition of the DCASE challenge took place in 2013 [3], showing the increasing interest on ASC among the scientific community. The following editions took place in 2016, 2017 and 2018. The challenge has been a backbone element for researches in the audio signal processing and machine learning area. While submissions in the first edition were mostly based on Gaussian mixture models (GMMs) [4] or Support Vector Machines (SVMs) [5], deep learning methods such as those based on Convolutional Neural Networks (CNNs) took the lead from 2016. The use of these techniques is directly correlated with the amount of data available. Task 1 of DCASE 2019 provides around 15000 audio clips per subtask, each one of 10 seconds duration, made available to the participants for algorithm development. This task contains 3 subtasks approaching different issues: (1) ASC, (2) ASC with mismatched recording devices and (3) Open-set ASC; this last one aimed at solving the problem of identifying audio clips that do not belong to any of the predefined classes used for training. Each subtask has its own training and test datasets: TAU Urban Acoustic Scenes 2019, TAU Urban Acoustic Scenes 2019 Mobile and TUT Urban Acoustic Scenes 2019 Openset respectively.

First approaches to ASC relied exclusively on feature-engineering [6, 7]. Most research efforts tried to develop meaningful features to feed classical classifiers, such as GMMs or SVMs [8]. Over the last years, Deep Neural Networks (DNNs) and, particularly CNNs, have shown remarkable results in many different areas [9, 10, 11], thus being the most popular choice among researchers and application engineers. CNNs allow to solve both problems, feature extraction and classification, into one single structure. In fact, deep features extracted from the internal layers of a pre-trained CNN can be successfully used for transfer learning in audio classi-
The way audio examples are presented to a neural network can be important in terms of system performance. In the system used in this challenge, a combination of the time-frequency representations detailed in Table 1 has been used as input to the CNN (see Table 2).

### 2. METHOD

This section describes the method proposed for this challenge. First, a brief background on CNNs is provided, explaining the most common layers used in their design. Then, the pre-processing applied over the audio clips before being fed to the network is explained. Finally, the submitted CNN is presented.

#### 2.1. Convolutional Neural Networks

CNNs were first presented by LeCun et al. in 1998 [20] to classify digits. Since then, a large amount of research has been carried out to improve this technique, resulting in multiple improvements such as new layers, generalization techniques or more sophisticated networks like ResNet [21], Xception [22] or VGG [10]. The main feature of a CNN is the presence of convolutional layers. These layers perform filtering operations by shifting a small window (receptive field) across the input signal, either 1D or 2D. These windows contain kernels (weights) that change their values during training according to a cost function. Common choices for the non-linear activation function in convolutional layers are the Rectified Linear Unit (ReLU) or the Leaky Rectified Linear Unit (Leaky ReLU) [23]. The activations computed by each kernel are known as feature maps, which represent the output of the convolutional layer. Other layers commonly employed in CNNs are Batch normalization and Dropout. These layers are interspersed between the convolutions to achieve a greater regularization and generalization of the network.

Traditional CNNs include one or two fully-connected layers before the final output prediction layer [24, 25]. Nevertheless, recent approaches [26] do not include fully-connected layers before the output layer. Under this approach, known as fully-convolutional CNN, the feature map is reshaped before the output layer by using global max or average pooling techniques. This procedure maps each filter with just one feature, either the maximum value or the average [26]. Most common approaches reshape all feature maps to a 1D vector (Flatten) before the final fully-connected layer used for prediction [27].

#### 2.2. Audio preprocessing

All of them are based on the Log Mel spectrogram [26, 24, 28] with 40 ms of analysis window, 50% of overlap between consecutive windows, Hamming asymmetric windowing, FFT of 2048 points and a 64-band normalized Mel-scaled filter bank. Each row, corresponding to a particular frequency band, of this Mel-spectrogram matrix is then normalized according to its mean and standard deviation.

For the case of harmonic and percussive features, a Short-Time Fourier Transform (STFT) of the time waveform with 40 ms of analysis window, 50% of overlap between consecutive windows, Hamming asymmetric windowing and FFT of 2048 points has been taken as starting point. This spectrogram was used to compute the harmonic and percussive sources using the median-filtering HPSS algorithm [29]. The resulting spectrogram was treated in 64 Log Mel frequency bands. Considering that the audio clips are 10 s long, the size of the final Log Mel feature matrix is $64 \times 500$ for all the cases. The audio preprocessing has been developed using the LibROSA library for Python [30].

#### 2.3. Proposed Network

The network proposed in this work is inspired in the architecture of the VGG [10], since this has shown successful results in ASC [17, 31, 26, 32]. The convolutional layers were configured with small $3 \times 3$ receptive fields. After each convolutional layer, batch normalization and exponential linear unit (ELU) activation layers [33] were stacked. Thus, two consecutive convolutional layers, including their respective batch normalization and activation layers, plus a max pooling and dropout layer correspond to a convolutional block. The final network (see Table 3) is composed of three convolutional blocks plus two fully-connected layers acting as classifiers.

Three different values for the number of filters for the first convolutional block have been implemented and tested: 16, 32, and 64 (Vfy-3L16, Vfy-3L16 and Vfy-3L64 in Table 2). The detailed architecture is given in Table 3. The developed network is intended to be used in a real-time embedded system, therefore a compromise has been achieved between the number of parameters and the final classification accuracy.

#### 2.4. Model ensemble

Combining predictions from different classifiers has become a popular technique to increase the accuracy [34]. This is known as ensemble models. For this work, different model probabilities have been aggregated using the arithmetic and geometric means as well as the Oreness Weighted Average (OWA) operator [35]. The following two weight vectors have been used for OWA ensembles: $w_1 = [0.05, 0.15, 0.8]^T$ and $w_2 = [0.1, 0.15, 0.75]^T$.

### 3. RESULTS

#### 3.1. Experimental details

The optimizer used was Adam [36] configured with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, decay = 0.0 and amsgrad = True. The models were trained with a maximum of 2000 epochs. Batch size was set to 32. The learning rate started with a value of 0.001 decreasing with a factor of 0.5 in case of no improvement in the validation accuracy after 50 epochs. If validation accuracy does not improve after 100 epochs, training is early stopped. Keras with Tensorflow backend was used to implement the models.
Detection and Classification of Acoustic Scenes and Events 2019 Challenge

M  Mono  Log Mel spectrogram (computed as detailed in Sect. 2.2) of the arithmetic mean of the left and right audio channels.
L  Left  Log Mel spectrogram (computed as detailed in Sect. 2.2) of the left audio channel.
R  Right Log Mel spectrogram (computed as detailed in Sect. 2.2) of the right audio channel.
D  Difference Log Mel spectrogram (computed as detailed in Sect. 2.2) of the difference of the left and right audio channels ($L - R$).
H  Harmonic Harmonic Log Mel matrix (computed as detailed in Sect. 2.2) using the Mono signal as input.
P  Percussive Percussive Log Mel matrix (computed as detailed in Sect. 2.2) using the Mono signal as input.

Table 1: Basic audio representations used in this work. Several combinations of the above alternatives have been used as input to the CNN (see Table 2 and Sect. 3).

<table>
<thead>
<tr>
<th>Audio preprocessing</th>
<th>Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mono (M)</td>
<td>Vfy-3L16 Vfy-3L32 Vfy-3L64</td>
</tr>
<tr>
<td>Left + Right + Difference (LRD)</td>
<td>72.69 -</td>
</tr>
<tr>
<td>Harmonic + Percussive (HP)</td>
<td>71.23 -</td>
</tr>
<tr>
<td>Harmonic + Percussive + Mono (HPM)</td>
<td>69.37 -</td>
</tr>
<tr>
<td>Harmonic + Percussive + Difference (HPD)</td>
<td>72.64 -</td>
</tr>
<tr>
<td>Harmonic + Percussive + Left + Right (HPLR)</td>
<td>71.57 -</td>
</tr>
</tbody>
</table>

PN (3D): 176,926  PN (3D): 495,150  PN (3D): 1,560,142

Table 2: Network accuracy (%) in development stage. The accuracy for the development set (Dev set) was calculated using the first evaluation setup of 4185 samples. Public leaderboard accuracy (Public LB) was extracted from Kaggle's public leaderboard composed of 1200 samples. The models labeled with an (*) were used for ensembles (see Table 4).

3.2. Results on the development dataset

Table 2 shows the results obtained for the development dataset with the three networks detailed in Table 3, combined with the different inputs specified in Table 1.

On the other hand, when ensembles are used (see subsection 2.4) the accuracy is improved. As shown in Table 4, the combination showing the highest accuracy is LRD + HPD + HPLR. Although Vfy-3L64 shows, in some cases, better accuracy in Dev set, this improvement is not correlated in Public Leaderboard. An interpretation for this could be that the network is more prone to overfitting due to the high number of parameters.

3.3. Subtask 1B

Although all the development has been focused on Task 1a, we have analyzed our networks using Task 1b data. The key of this task is that there is a mismatch among recording devices. These new devices are commonly customer devices such as smartphones or sport cameras. The main differences are the sample rate and the number of channels. Therefore, we decided to work with mono signals in this subtask, resampling them to 32 kHz before any pre-processing. The networks are fed with mono, harmonic and percussive spec-
In order to embed an ASC classifier into an edge system, a study of the network depth becomes a crucial stage. Real-time devices usually work under sharp constraints concerning the classification time. In this technical report, a study of 3 different CNNs has been carried out using sum technique. The first “E” is used as abbreviation of the models ensemble shown in Table 2. Ensemble programs. As explained in Subsection 3.2, the network used for ensemble with HP or HPM input is Vfy-3L32 and Vfy-3L16 in case of mono. This analysis has only been run in Public Leaderboard.

4. CONCLUSION

In order to embed an ASC classifier into an edge system, a study of the network depth becomes a crucial stage. Real-time devices usually work under sharp constraints concerning the classification time. In this technical report, a study of 3 different CNNs has been proposed. It turns out that deeper networks do not always present the best accuracy. In our study, we conclude that there is not a substantial difference in accuracy between a model of 0.5M parameters or 1.5M parameters. Therefore, taking into account that the majority of CNN architectures proposed in the literature follow a repeating structure that duplicates the number of filters in each repetition, it seems that an appropriate selection of the number of filters in the first layer is essential for real-time applications. Although our network is not very deep for preventing overfitting, this aspect should be considered more carefully when designing deeper networks. In addition, a brief study on the use of different model ensembles has been presented, showing that even simple averaging already improves the final accuracy. Finally, we have discussed the importance of the right input (audio representation) when training a CNN. The novel HPD representation shows the best accuracy in all the three networks, which suggests that testing alternative input representations might be more worthy than tuning complex and consuming networks.

Table 4: Ensemble accuracy (%). The name is composed using the abbreviation of the models ensemble shown in Table 2. Initial letter E stands for ensemble. Models labeled with an (*) have been submitted for challenge rank.

<table>
<thead>
<tr>
<th>Ensemble Network</th>
<th>Method</th>
<th>Dev set</th>
<th>Public LB</th>
</tr>
</thead>
<tbody>
<tr>
<td>E_M_LRDP_HP</td>
<td>arith. mean</td>
<td>74.74</td>
<td>-</td>
</tr>
<tr>
<td>E_M_LRDP</td>
<td>arith. mean</td>
<td>75.02</td>
<td>-</td>
</tr>
<tr>
<td>E_LRDP_HP</td>
<td>arith. mean</td>
<td>75.11</td>
<td>75.83</td>
</tr>
<tr>
<td>E_M_HP</td>
<td>arith. mean</td>
<td>72.56</td>
<td>-</td>
</tr>
<tr>
<td>E_LRDP_HP</td>
<td>arith. mean</td>
<td>76.48</td>
<td>77.00</td>
</tr>
<tr>
<td>E_LRDP_HP</td>
<td>arith. mean</td>
<td>75.51</td>
<td>-</td>
</tr>
<tr>
<td>E_HP_HP</td>
<td>arith. mean</td>
<td>76.81</td>
<td>-</td>
</tr>
</tbody>
</table>
*E_LRDP_HP        | arith. mean | 76.64   | 77.33     |
*E_LRDP_HP        | geom. mean  | 77.06   | 77.00     |
*E_LRDP_HP        | OWA1      | 76.84   | 76.33     |
*E_LRDP_HP        | OWA2      | 76.94   | 76.50     |

Table 5: Network accuracies (%). The name is composed using the abbreviation of the models ensemble shown in Table 2. Ensemble has been carried out using sum technique. The first “E” is used as initial for ensemble.

<table>
<thead>
<tr>
<th>Network</th>
<th>Method</th>
<th>Public LB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>43.83</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>53.16</td>
<td></td>
</tr>
<tr>
<td>E_M_HP</td>
<td>arith. mean</td>
<td>58.33</td>
</tr>
<tr>
<td>E_HP_HPM</td>
<td>arith. mean</td>
<td>60.33</td>
</tr>
<tr>
<td>E_M_HP_HPM</td>
<td>arith. mean</td>
<td>60.66</td>
</tr>
<tr>
<td>E_M_HP_HPM</td>
<td>geom. mean</td>
<td>59.50</td>
</tr>
<tr>
<td>E_M_HP_HPM</td>
<td>OWA1</td>
<td>60.33</td>
</tr>
<tr>
<td>E_M_HP_HPM</td>
<td>OWA2</td>
<td>60.50</td>
</tr>
</tbody>
</table>

5. ACKNOWLEDGMENT

This work has received funding from the European Unions Horizon 2020 research and innovation program under grant agreement No 779158, as well as from the Spanish Government through project RTI2018-097045-B-C21. The participation of Dr. Pedro Zuccarello in this work is partially supported by Torres Quevedo fellowship PTQ-17-09106 from the Spanish Ministry of Science, Innovation and Universities.

6. REFERENCES


