AI4EDGEPT SUBMISSION TO DCASE 2022 LOW COMPLEXITY ACOUSTIC SCENE CLASSIFICATION TASK1

Technical Report

Ricardo Anastácio[1], *Luís Ferreira* [2], *Mónica Figueiredo*[1,3] *and Luís Conde Bento*[1,4]

[1] Polytechnic of Leiria, Leiria, Portugal; [2] University of Coimbra, Coimbra, Portugal;
[3] Instituto de Telecomunicações, Portugal; [4] Institute of Systems and Robotics, Coimbra, Portugal
Corresponding author: Ricardo Anastácio, 2192601@my.ipleiria.pt

ABSTRACT

This report details the submission to task1 of DCASE2022 competition. The task aims to classify acoustic scenes using devices with low computational power and memory. We propose two ensemble models for scene classification. The first model clusters classes into 2 groups, each of a two-network ensemble being responsible for intra-group discrimination, *i.e.* discriminating between the classes that are most related in the confusion matrix. The second model implements a canonical one-versus-all ten-network ensemble architecture followed by knowledge distillation, *i.e.* the ensemble model is used as the *teacher* network. The *student* is an optimised version of the DCASE2022 baseline architecture. In both models we resort to three different data pre-processing techniques: audio downsample; mel-spectrogram tuning; and data augmentation. We've used the DCASE2022 baseline for all networks - two-network ensemble, ten-network ensemble and student network - on which we have conducted an architecture's hyperparameter search to identify the best performing architecture, while being compliant with DCASE2022 performance metrics. Results revealed that data preprocessing and knowledge distillation techniques improve overall performance. Nevertheless, a simple two-network ensemble without knowledge distillation, maintains the MACS and parameters size low, while achieving similar results.

***Index Terms*—** DCASE2022, Architecture search, ensemble, knowledge distillation, data augmentation

1. INTRODUCTION

Acoustic scene classification aims to recognise different acoustic scenes based on their audio features. Extracting context information based on audio features has a lot of applications, many of which have complexity constraints due to the limited computational and memory resources available on edge devices.

Task 1 of the DCASE2022 Challenge[1] – Low-Complexity Acoustic Scene Classification – aims to promote the research around this subject by comparing different classification approaches using a publicly available dataset, TAU Urban Acoustic Scenes 2022 Mobile dataset[2][3]. To ensure a good performance across different recording devices, the dataset includes data recorded and simulated with a variety of devices. The challenge sets complexity limits modelled after Cortex-M4 devices constraints, imposing a maximum of 128K model parameters (including the zero-valued ones) and a maximum of 30 million MACs per inference.

Mel-spectrograms from the audio signals were used as input features. To increase the extracted features relevance, hypertuning was applied to the spectrogram's parameters. Furthermore, the models' parameters and architecture were also tuned to achieve the best performance given the complexity limits. Ensemble techniques were used to increase the models' class discrimination. Finally, knowledge distillation(KD) was applied to reduce the ensemble models' complexity. Data augmentation techniques were also used to improve generalisation.

This report is organised as follows: in Sections 2 and 3 the input data and model tuning is described; Section 4 presents the ensemble models and posterior KD techniques; at last, results and conclusions are presented in Sections 5 and 6.

2. DATA PREPROCESSING TECHNIQUES

The TAU Urban Acoustic Scenes 2022 Mobile dataset, henceforth designated as original dataset, contains recordings from 12 European cities in 10 different acoustic scenes using 4 different devices. Additionally, synthetic data for 11 mobile devices S1-S11 were simulated using the audio recorded with the first device. The audio is provided in a single-channel 44.1kHz 24-bit format. The original dataset contains data from 10 cities and 9 devices, while the evaluation dataset contains data from 12 cities and 11 devices – 5 new devices not available in the development set. As per suggestion of the challenge guidelines, the original dataset was split in two: 70% of the data for each device for training and 30% for testing. In order to create a perfectly balanced test set, a number of segments from various devices are not included in this split.

Preprocessing was done using the Librosa library[4]. The input data was downsampled to 8kHz and the log Mel spectrogram was calculated using the Short-Time Fourier Transform (STFT) with a window length of 2048 samples. Data was augmented to balance the difference of samples available between devices, improving the model's generalization. The augmentation techniques used were: pitch shift, time stretch, mixup[5], time and frequency mask[6].

Additionally, to allow the search for optimal hyperparameters and the implementation of audio signal processing layers – Mel spectrogram – in the model, we use the Kapre[7] tool with the objective of finding signal representations that best contribute to the final result. The search was made with the KerasTuner[8] structure using the Hyperband[9] method. Four sampling frequencies (FS) were tested, each one two times.

The search grid was set to find the optimal value for Mel bins and hop length:

- Sampling Frequency search options:
  $FS = [8, 16, 22.05, 44.1][kHz]$

Table 1: Mel spectrogram tuning: most significant results. These results were obtained with 2048 STFT.

FS	Hop length	nMels	Input Shape	ACC	LOSS
44.1 kHz	1024	40	40x44	41.99	1.740
44.1 kHz	1024	60	60x44	43.21	1.713
22.05 kHz	1536	60	60x15	43.90	1.610
22.05 kHz	1536	140	140x15	47.44	1.558
16 kHz	1024	180	180x16	49.47	1.474
16 kHz	1024	120	180x16	48.88	1.496
8 kHz	1024	260	260x8	50.70	1.428
8 kHz	1024	140	140x8	49.50	1.450

- SFFT window size search options:
  $SFFT_{Wsize} = [256, 512, 1024, 2048]$
- Mel bins search options:
  $MEL_{range} = [20, 300]$; $MEL_{Step} = 20$
- Hop length SFFT window ratio search options:
  $HOP = [25\%, 50\%, 75\%]$

The model used was the baseline model with the pool size of the max pooling block of the 3rd convolutional layer changed from 4×10 to 2×2, this was done to avoid problems with different input sizes and to give equal importance to both input dimensions. The most significant results, among all tested configurations, are presented in Table 1. We observed that a lower sampling frequency (lower FS) and higher number of Mel bins (higher nMels) leads to better results. This may imply that lower frequency features may be more relevant to the DCASE2022 challenge. The two input configurations with better results were:

- Input Configuration 1 (IC1): $FS = 8kHz$ and $nMels = 260$
- Input Configuration 2 (IC2): $FS = 8kHz$ and $nMels = 140$

3. MODIFIED BASELINE MODEL HYPERTUNING

Taking the Baseline Model as reference, given the parameters and MAC limits of the challenge[10], a model hyperparameters search was made.

The Baseline Model architecture hypertunning was made using the two best performant input configurations identified in section 2. Hence two models topologies were searched, one for each input configuration, IC1 and IC2. The search grid was set has follows:

-CNN layer #1:
$Filters = [4, 32]$ $FiltersStep = 4$

-CNN layer #2/#3:
$Filters = [4, 64]$ $FiltersStep = 4$
$Pooling\_type = [avg, max]$
$Pooling\_size = [([1, 2, 3, 4, 6], [1, 2])]$

-Additional CNN layers #$[1, 2]$:
$Filters = [4, 32]$ $FiltersStep = 4$
$Pooling\_type = [avg, max]$
$Pooling\_size = [([1, 2], [1, 2])]$

-Global Pooling $[GlobalMax, GlobalAvg, noPool]$

-$Dropout = [0, 0.7]$ $DropoutStep = 0.1$

-Dense $units = [32, 256]$ $unitsStep = 32$

- All CNN layers have:
  $Kernel\_size = [([3, 5, 7], [3, 5, 7])]$
  $Dropout = [0, 0.7]$ $DropoutStep = 0.1$

Table 2: Best result for each proposed model topology: details and metrics

Model	FS	Input Shape	ACC	LOSS	Parameter	MACs
TBM1	8 kHz	260x8	49.40	1.425	26.7k	23.6M
TBM2	8 kHz	140x8	49.72	1.386	52.9k	25.5M

This way, the model's performance can be improved by choosing the most relevant features in an autonomous way. Has in data prepossessing (see section 2) the search for optimal hyperparameters was made with the KerasTuner[8] structure using the Hyperband[9] method.

Table 2 presents the best result for each topology, Tuned Baseline Model 1 (TBM1) and Tuned Baseline Model 2 (TBM2). Based on the search stages the model with the best performance was the TBM2 model and its respective input configuration, *i.e.* IC2, with the following search output configuration:

-CNN layer #1:
$Filters = 20$
$Kernel\_size = (7, 5)$

-CNN layer #2:
$Filters = 28$
$Kernel\_size = (7, 3)$
$MaxPooling\_size = (1, 2)$
$Dropout = 0.1$

-CNN layer #3:
$Filters = 28$
$Kernel\_size = (3, 7)$
$MaxPooling\_size = (2, 2)$
$Dropout = 0.3$

-CNN layer #4:
$Filters = 16$
$Kernel\_size = (7, 5)$
$Dropout = 0.3$

-Global Average Pooling

-Dropout = 0.4

-Dense units = 256

The TBM2 model's architecture is shown in Fig. 1 along with the baseline architecture for comparison.

4. ARCHITECTURE MODEL SEARCH AND DEVELOPMENT

Ensemble[11] techniques are known to increase the system's performance and even reduce the model's complexity by combining trained features from different sample sets and/or models. Also, classification models typically present better performance when the number of labels to classify is small. Hence, two ensemble approaches were developed:

- Six-Class Two Ensemble network + Final Classification Dense Layer (6C2EN+FCDL)
- Teacher/Student Knowledge Distillation of Two-Class Ten Ensemble network + Final Classification Dense Layer(TSKD@2C10EN+FCDL)

These models are based on the TBM2 architecture, depicted in Fig. 1, and will be individually described in sections 4.1 and 4.2.

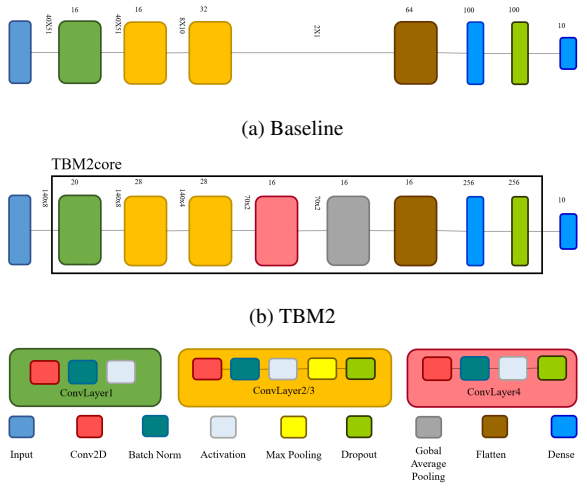


Figure 1: Baseline and Tuned Baseline Model 2(TBM2) architecture.

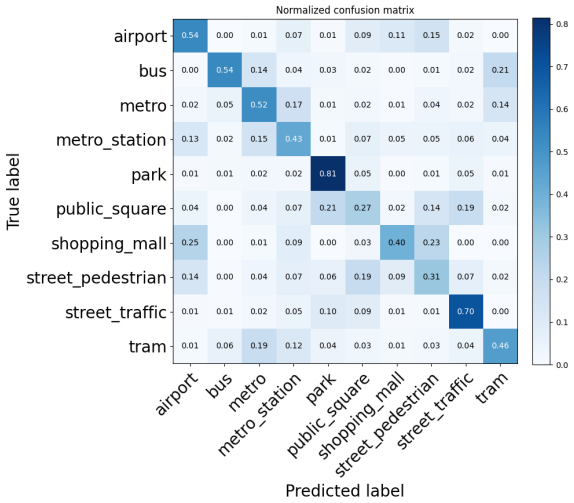


Figure 2: Confusion matrix (TBM2).

4.1. Six-Class Two Ensemble Network + Final Classification Dense Layer

The first approach uses a six-class ten ensemble network with a final classification dense layer. Two TBM2 models, TBM2core-1 and TBM2core-2, were used in an ensemble configuration, as shown in Fig. 3. In order to meet the complexity requirements of the DCASE2022 the number of filters from the first two convolution layers, of each TBM2core-1 and TBM2core-2, was reduced to half.

The training data was split in 2 sets of samples with 5 labels each, by joining the most related ones in the confusion matrix (Fig. 2), creating two groups of classes:

- Group 1: airport, public square, shopping mall, street pedestrian, park;
- Group 2: bus, metro, metro station, street traffic, tram.

One expects that each model learns to better discriminate among them. Furthermore a new label was added to contain the remainder labels that do not belong to the split.

The TBM2core-1 and TBM2core-2 models were trained individually and their weights were blocked afterwards, preventing the loss of trained features and overfitting during the last stage of the training of the 6C2EN+FCDL. After the learning stage only the core part (TBM2core) of the models were used. After concatenation a dense layer was added as the final classification layer. Finally, the added layer training was performed.

4.2. Teacher/Student Knowledge Distillation of a Two-Class Ten Ensemble network + Final Classification Dense Layer

The second approach use Teacher/Student Knowledge Distillation (KD) of Ten-Class Two Ten Ensemble network + Final Classification Dense Layer, as shown in Fig. 4. The TBM2 model is used to classify each label in a binary fashion (One vs. Rest), by teaching each model identify if a sample belongs to a specific class or not, resulting in 10 models in total, *i.e.*, TBM2core+FCDL-1 ... TBM2core+FCDL-10. This architecture is illustrated in Fig. 4. As in Section 4.1 only the TBM2core part of the model was used after the training stage.

Model	Ensemble	KD	ACC	LOSS	Parameter	MACs
Teacher_AI4EDGE_3	10	✗	42.6	2.707	529k	255M
AI4EDGE_3	10	✓	50.5	1.347	52.9k	25.5M
Teacher_AI4EDGE_4	10	✗	98.1	0.221	529k	255M
AI4EDGE_4	10	✓	60.5	1.103	52.9k	25.5M

Table 3: Knowledge Distillation models' results

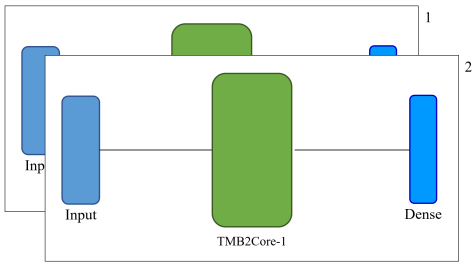
Although achieving a better performance, shown in Table 3, this second approach is computationally complex (about 10× the baseline model complexity) not fulfilling the DCASE2022 challenge complexity requirements. To tackle this, knowledge distillation techniques[12][13] were used to obtain a more compact model – a fairly complex model (teacher) is used to train a lower complexity model (student). This technique not only allows to reduce the model's dimensions but also fosters the optimisation of the student model, attaining a performance that is hardly achieved if trained in a traditional way.

5. RESULTS AND SUBMISSIONS

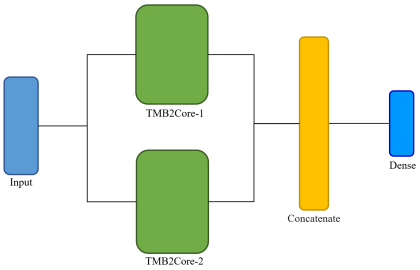
The models were trained for 200 epochs with a batch size of 64 using ADAM optimiser and early stopping. The objective evaluation metric was to minimise the categorical cross-entropy loss. The model was also quantized to INT8.

The submitted models' features and results are listed on Table 4. All models are based in the methods 6C2EN+FCDL (described in section 4.1) and TSKD@2C10EN+FCDL (described in section 4.2). The **AI4EDGE_1** & **AI4EDGE_2** submissions represent the first ensemble approach (6C2EN+FCDL), joining two models pre-trained in different label sets. The **AI4EDGE_3** & **AI4EDGE_4** submissions represent the second ensemble approach (TSKD@2C10EN+FCDL), *i.e.*, ensemble of 10 binary classification TBM2 models used as teacher model in the KD technique. The results of

In **AI4EDGE_1** and **AI4EDGE_3** the full development dataset was used for training phase, while the **AI4EDGE_2** and



(a) Learning



(b) Ensemble

Figure 3: 6C2EN+FCDL model's architecture.

Table 4: Submitted models' results for DCASE2022 Task 1

Model	Ensemble	KD	ACC	LOSS	Parameter	MACs
AI4EDGE_1	2	✗	75.6	0.742	70.6K	21.1M
AI4EDGE_2	2	✗	73.5	0.791	70.6K	21.1M
AI4EDGE_3	10	✓	50.5	1.347	52.9k	25.5M
AI4EDGE_4	10	✓	60.5	1.103	52.9k	25.5M

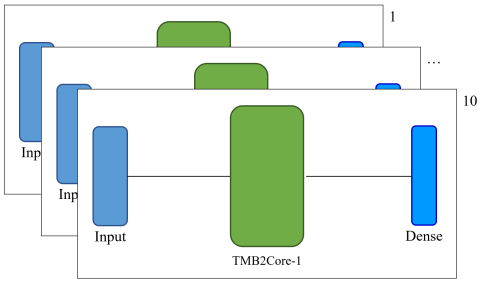
AI4EDGE_4 used only the training split (section 2) in the pre-training and the full development dataset in the final training.

6. CONCLUSIONS

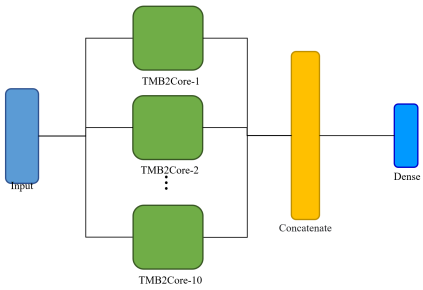
This work was developed with the objective of improving performance comparatively with the provided baseline model while fulfilling the complexity requirements of the DCASE2022 challenge, being able to improve the original ACC=42.9% and LOSS=1.575. This was achieved though the use of several techniques, namely input prep-processing, emsemble networks and knowledge distillation, meaning the best results were ACC=75.6 and LOSS=0.742.

Through input pre-processing it was testified the importance of adapting the dataset to the problem. The data augmentation allowed to balance the dataset providing a more robust model to the presence of new data. Downsampling to 8kHz enabled to change the model focus to a wider temporal window and consequently to a lower frequency range, reducing the number of features needed, which allowed to increase the number of Mel bins. The improvement of the results suggested that the most important features to distinguish between classes are found on lower frequencies.

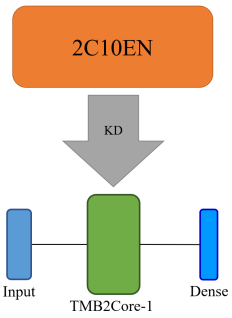
Hypertunning the model's architecture also proved to be important, achieving a more autonomous way of finding features that better improve performance within the complexity limits imposed. The ensemble methods showed an increased performance letting the model focus on a smaller number of classes.



(a) Learning



(b) Ensemble (Teacher)



(c) Knowledge Distillation (TBM2 Student)

Figure 4: TSKD@2C10EN+FCDL model's architecture.

The knowledge distillation technique granted a significant reduction of the model's size by taking advantage of the student's learning ability.

7. REFERENCES

[1] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in dcase 2022 challenge," 2022. [Online]. Available: https://arxiv.org/abs/2206.03835

[2] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events*, 2018. [Online]. Available: https://dcase.community/documents/workshop2018/proceedings/DCASE2018Workshop_Mesaros_8.pdf

[3] https://zenodo.org/record/6337421#.YrHTB_TMJD8.

[4] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," 2015. [Online]. Available: https://librosa.org/

[5] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2017. [Online]. Available: https://arxiv.org/abs/1710.09412

[6] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," 2019. [Online]. Available: https://arxiv.org/abs/1904.08779

[7] K. Choi, D. Joo, and J. Kim, "Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras," 2017. [Online]. Available: https://arxiv.org/abs/1706.05781

[8] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, *et al.*, "Kerastuner," 2019. [Online]. Available: https://github.com/keras-team/keras-tuner

[9] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," 2016. [Online]. Available: https://arxiv.org/abs/1603.06560

[10] https://dcase.community/challenge2022/task-low-complexity-acoustic-scene-classification#system-complexity-requirements.

[11] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan, "Ensemble deep learning: A review," 2021. [Online]. Available: https://arxiv.org/abs/2104.02395

[12] https://keras.io/examples/vision/knowledge_distillation/.

[13] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015. [Online]. Available: https://arxiv.org/abs/1503.02531