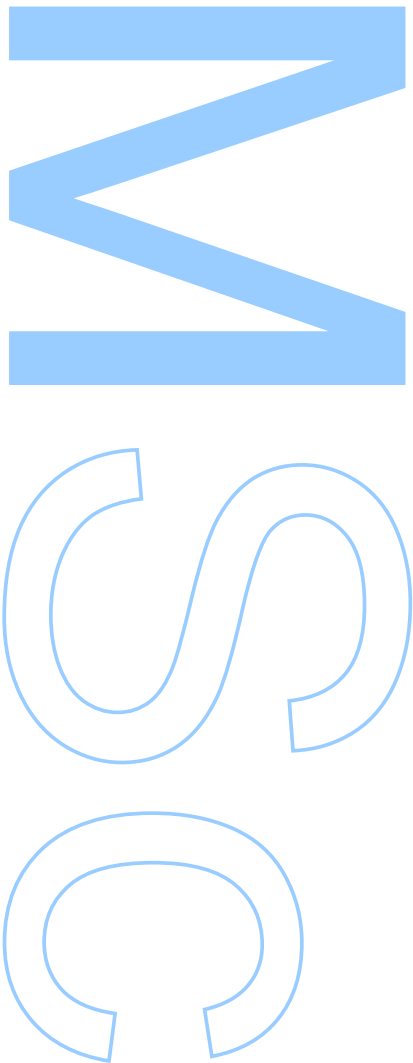
Sound Characterization of Urban Environments by Computational Methods

Ana Filipa Rodrigues Nogueira
Mestrado em Engenharia Física
Departamento de Física e Astronomia
2022

Orientador
Professor Doutor João Manuel R. S. Tavares, Faculdade de Engenharia

Coorientador
Mestre Hugo Oliveira, Faculdade de Engenharia



U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, _____/_____/_____

UNIVERSIDADE DO PORTO

MASTERS THESIS

Sound Characterization of Urban Environments by Computational Methods

Author:
Ana Filipa R. NOGUEIRA

Supervisor:
João Manuel R. S. TAVARES

Co-supervisor:
Hugo OLIVEIRA

A thesis submitted in fulfilment of the requirements
for the degree of MSc. Engineering Physics

at the

Faculdade de Ciências da Universidade do Porto
Departamento de Física e Astronomia

September 30, 2022

" Never say never because limits, like fears, are often just an illusion. "

Michael Jordan

Sworn Statement

I, Ana Filipa Rodrigues Nogueira, enrolled in the Master Degree of Engineering Physics at the Faculty of Sciences of the University of Porto hereby declare, in accordance with the provisions of paragraph a) of Article 14 of the Code of Ethical Conduct of the University of Porto, that the content of this dissertation reflects perspectives, research work and my own interpretations at the time of its submission.

By submitting this dissertation, I also declare that it contains the results of my own research work and contributions that have not been previously submitted to this or any other institution.

I further declare that all references to other authors fully comply with the rules of attribution and are referenced in the text by citation and identified in the bibliographic references section. This dissertation does not include any content whose reproduction is protected by copyright laws.

I am aware that the practice of plagiarism and self-plagiarism constitute a form of academic offense.

Ana Filipa Rodrigues Nogueira

September 30, 2022.

Acknowledgements

I would like to thank Ph.D. João Manuel R. S. Tavares, for the opportunity to work on this project, for the recommendations and orientation during the development of this dissertation.

To M.Sc. Hugo Oliveira, for all the shared knowledge, support, availability and help throughout the project.

To Ph.D. José Machado, for the careful review of the dissertation and tips to improve.

Also, to my parents and brother for all the support to successfully complete this stage.

To all my friends who accompanied me throughout the years with whom I shared very good times.

Finally, this Thesis is related to the project Safe Cities - "Inovação para Construir Cidades Seguras", with reference POCI-01-0247-FEDER-041435, co-funded by the European Regional Development Fund (ERDF), through the Operational Programme for Competitiveness and Internationalization (COMPETE 2020), under the PORTUGAL 2020 Partnership Agreement; which I am also thankful.

UNIVERSIDADE DO PORTO

Abstract

Faculdade de Ciências da Universidade do Porto
Departamento de Física e Astronomia

MSc. Engineering Physics

Sound Characterization of Urban Environments by Computational Methods

by Ana Filipa R. NOGUEIRA

Smart Cities are emerging to improve their citizens' lives, in which Environmental Sound Classification (ESC) has a crucial role due to the different areas in which it can be applied, such as security, surveillance, manufacturing, autonomous vehicles, and noise mitigation, among others. Since Urban Sounds, for the significant part, are composed of audio events occurring daily, which presents unstructured characteristics containing a lot of noise and some sounds unrelated to the sound event making ESC a very challenging problem; thus, several computational algorithms have been proposed to solve this problem.

In this thesis, various model architectures were implemented to overcome this problem, being its efficiency assessed, and several datasets were used: the UrbanSound8K, the ESC-50 and the ESC-10 dataset.

Therefore, firstly, a baseline model that consists of several dense and dropout layers using handcrafted features as input and different dropout rates were employed, and the influence of increasing the models' depth was tested. Several experiments showed that a combination of features provides more information to the model due to higher representation capabilities; the dropout rate should be used to avoid models' overfitting but not a value too high to allow the model to have enough information to learn, lastly, the increase in depth did not show significant benefits being detrimental in most cases.

Some end-to-end models were explored in the following step: DenseNet, ResNet and Inception. For these models, it was considered image domain pre-training that gave a considerable performance boost allowing better results and faster convergences; furthermore, it was tested the influence of data augmentation, which only shown to be beneficial in some situations.

Finally, a Transformer model was used, trained with image domain pre-training once again, which gave a huge performance boost compared to the no pre-trained version. Nonetheless, pre-training from the image and audio domain was also used, showing even better results, and highlighting the benefits of having an in-task domain pre-training. Moreover, using data augmentation techniques such as SpecAugment, noise and mixup were employed; however, the best results were obtained when only SpecAugment was considered.

All models were tested with different optimization functions making it possible to conclude that SGD, Adagrad and Adadelta present a poor performance regardless of the model or dataset, showing their inability to produce robust models. On the other hand, for the best optimizer, there is no consensus between the models being the best for the baseline models, the Nadam optimizer, for the end-to-end models, the Adam optimizer, and for the Transformer, AdamW was the best for the ESC-50, Adam for the ESC-10 dataset and for the UrbanSound8K, Adamax was considered the best, however, for this last dataset, AdamW gave a pretty similar performance. Therefore, it can be concluded that the choice of the best optimization function depends on the model and the chosen dataset.

Therefore, out of all these models, the best performance was obtained for the Transformer model with an accuracy score of 89.8% for UrbanSound8K, 95.8% for ESC-50 and 99% for ESC-10.

Keywords - Convolutional Neural Networks, Transformers, Data augmentation, Feature extraction, Optimization function, Classification.

UNIVERSIDADE DO PORTO

Resumo

Faculdade de Ciências da Universidade do Porto
Departamento de Física e Astronomia

Mestrado em Engenharia Física

Caracterização Sonora de Ambientes Urbanos por Métodos Computacionais

por Ana Filipa R. NOGUEIRA

Com o intuito de melhorar a vida dos cidadãos estão a emergir cidades inteligentes nas quais a classificação de ambientes sonoros tem um papel crucial devido às diferentes áreas nas quais pode ser aplicada, tal como segurança, vigilância, fabricação, veículos autónomos, mitigação de ruídos, entre outros. Uma vez que os sons urbanos, na sua maioria, são compostos por eventos de áudio que ocorrem diariamente, os quais apresentam características não estruturadas, contendo muito ruído e alguns sons não relacionados com o evento sonoro tornando a classificação de ambientes sonoros um problema bastante desafiador. Assim, para resolver este problema têm sido propostos vários algoritmos computacionais.

Nesta tese, foram implementadas várias arquiteturas de modelos para ultrapassar este problema e para testar a sua eficiência foram usados diversos conjuntos de dados: UrbanSound8K, ESC-50 e ESC-10.

Inicialmente foi criado um modelo de base que consiste em várias camadas densas e de abandono que utilizam recursos artesanais como entrada. Utilizaram-se diferentes taxas de abandono e testou-se também, a influência do aumento da profundidade dos modelos. As diversas experiências permitiram entender que uma combinação de recursos fornece mais informações ao modelo devido à maior capacidade de representação; a taxa de abandono deve ser usada para evitar o sobreajuste dos modelos, mas, não deve ser muito elevada para permitir que o modelo tenha informações suficientes para aprender; por fim, o aumento da profundidade não mostrou benefícios significativos sendo prejudicial na maioria dos casos.

De seguida, foram explorados alguns modelos ponta-a-ponta, nomeadamente, DenseNet, ResNet e Inception. Para estes modelos, foi considerado um pré-treinamento do

domínio de imagem o que originou um grande aumento do desempenho, permitindo melhores resultados e convergências mais rápidas. Testou-se também a influência do aumento de dados e verificou-se que só em algumas situações era benéfico.

Por fim, foi utilizado um modelo Transformer treinado com pré-treinamento do domínio de imagem que, mais uma vez, permitiu obter um grande aumento do desempenho quando comparado com a versão não pré-treinada. Porém, os resultados foram ainda melhores quando foi utilizado o pré-treinamento do domínio de imagem e áudio, destacando-se os benefícios de ter pré-treinamento do mesmo domínio da tarefa. Além disso, foram empregues técnicas de aumento de dados como SpecAugment, ruído e mistura, contudo, os melhores resultados foram obtidos quando apenas SpecAugment foi utilizado.

Todos os modelos foram testados com diferentes funções de otimização permitindo concluir que SGD, Adagrad e Adadelta apresentam um mau desempenho independente do modelo ou conjunto de dados, mostrando a sua incapacidade de produzir modelos robustos. Por outro lado, para o melhor otimizador não há consenso entre os modelos. Para os modelos de base o melhor é o otimizador Nadam; para os modelos ponta-a-ponta, o otimizador Adam; já para o Transformer, o AdamW foi o melhor para o conjunto de dados ESC-50, o Adam para o ESC-10 e para o UrbanSound8K o Adamax, porém, neste último conjunto de dados o AdamW teve um desempenho bastante semelhante. Por conseguinte, pode-se concluir que a escolha da melhor função de otimização depende do modelo e do conjunto de dados escolhido.

Assim, de todos estes modelos o melhor desempenho foi obtido para o modelo Transformer com um resultado de precisão de 89,8% para UrbanSound8K, 95,8% para ESC-50 e 99% para ESC-10.

Palavras-chave **-** Redes Neuronais Convolucionais, Transformers, Aumento de dados, Extração de caraterísticas, Função de optimização, Classificação.

Contents

List of Figures

List of Tables

Acronyms

1D one-dimensional. 23, 107

2D two-dimensional. 17

6D six-dimensional. 40

AAML Additive Angular Margin Loss. 9

ANN Artificial Neural Networks. 22

APNet Audio Prototype Network. 9, 11, 14

AST Audio Spectrogram Transformer. xix, 13, 17, 18, 107, 108

AUC area under the receiver operating characteristic (ROC) curve. xvii, xviii, 37, 49, 52, 53, 55, 57, 59, 61, 62, 63, 64, 65, 67, 68, 69, 70, 71, 72, 73, 75, 76, 77, 78, 96

B-GRU Bidirectional Gated Recurrent Unit. 25

BERT Bidirectional Encoder Representations from Transformers. 13, 15, 16, 21

BLSTM Bidirectional Long-Short Term Memory. 23, 24, 26

CENS Chroma Energy Normalized Statistics. 10, 41, 42, 44, 51, 58, 60, 62, 65, 83, 126

CNN Convolutional Neural Networks. 2, 9, 10, 11, 13, 14, 15, 16, 21, 22, 24, 25, 26, 27, 29, 30, 31, 35

CNN-Transformer Convolutional Neural Network Transformer. 13, 14, 15, 21

CQT Constant Q-transform. 10, 16, 41, 42, 44, 51, 58, 60, 62, 65, 74, 78, 83, 126

CRNN Convolutional Recurrent Neural Networks. 25, 27, 29, 35

dB decibel. 20

DCNN Deep Convolutional Neural Networks. 9, 14, 22, 26, 35, 87

DeiT Data efficiency image Transformer. 18, 19

DenseNet Dense Convolutional Network. xiv, xviii, xxii, 3, 32, 33, 35, 87, 88, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 104, 119, 122

DL Deep Learning. 2, 4, 14, 30, 35

DNN Deep Neural Network. 85, 89

ESC Environmental Sound Classification. ix, 9, 15, 16, 21, 31, 35, 37

FN False Negative. 49, 50

FP False Positive. 49, 50

GFCC Gammatone Frequency Cepstral Coefficient. 16

GMM Guassian Mixture Model. 22

HMM Hidden Markov Model. 22, 23, 26

HPSS Harmonic Percussive Source Separation. 16, 31

k-NN k-Nearest Neighbor. 22

LSTM Long-Short Term Memory. 2, 9, 10, 14, 35

M2M-AST Many-to-Many Audio Spectrogram Transformer. 13, 18, 19

MFCC Mel Frequency Cepstral Coefficients. 9, 10, 11, 14, 16, 24, 41, 44, 51, 53, 55, 57, 58, 59, 60, 62, 65, 67, 71, 74, 78, 82, 83, 125

ML Machine Learning. 3, 8

MSE Mean Squared Error. 19

NCE Noise Contrastive Estimation. 20

pp percentage points. 65, 92, 96, 99, 100, 104, 109, 110, 112, 113, 118, 120

PRISMA Preferred Reporting Items for Systematic Reviews and Meta-Analyses. xvii, 8

RBF Radial Basis Function. 24

ResNet Residual Neural Network. xiv, xxii, 3, 27, 32, 33, 35, 85, 86, 87, 91, 92, 93, 94, 95, 96, 97, 99, 100, 104, 119, 122

RGB Red-Green-Blue. 32

RNN Recurrent Neural Networks. 2, 23, 26

SGD Stochastic Gradient Descent. 46, 47, 50, 51, 53, 54, 55, 57, 58, 60, 61, 62, 67, 68, 70, 71, 74, 75, 78, 81, 82, 91, 92, 96, 97, 120

SIF Spectrogram Image Features. 32

SOTA State-of-the-Art. 4, 7, 18, 20, 85, 120

SSLS Sound Source Localization and Separation. 28, 30

SSSC Sound Source Separation and Classification. 28, 30

STFT Short-Term Fourier Transformation. 10, 11, 14, 24, 41, 42, 44, 51, 58, 59, 60, 62, 65, 74, 78, 83, 126

SVM Support Vector Machine. 22, 24

TFCNN Temporal-frequency attention based Convolutional Neural Network. 9, 12, 14

TP True Positive. 50

VATT Video-Audio-Text Transformer. 13, 19

ViT Vision Transformer. 17, 18, 21

VQ-VAE Vector-quantized varitional autoencoders. 16

YOHO You Only Hear Once. 30, 31

Chapter 1

Introduction

This chapter presents a small overview of the dissertation. Starting with a brief contextualisation of the importance of distinguishing sounds in urban environments and the challenges this task presents, followed by some of the implemented classification algorithms developed to solve this problem. Afterwards, it is described the employed methodology, the research objectives and a summary of this thesis's structure with a short description of each chapter.

1.1 Urban Sound

As a result of the growth of the urban population worldwide (Syed et al. [32]), cities are consolidating their position as one of the central structures in human organisations. This concentration of resources around cities offers new opportunities to be exploited. Smart Cities are emerging as a paradigm to take advantage of these opportunities to improve their citizens' lives. Smart Cities use sensing architecture deployed in the city to provide new and disruptive city-wide services to the citizens and policy-makers. One of the main requirements concerns Urban Sound characterisation, which still poses different problems (Das et al. [3], Mushtaq and Su [22]). It is estimated that major cities must handle thousands of co-occurring events, with rapid, occurring events that require immediate action passing unnoticed by authorities (Das et al. [3], Mushtaq and Su [22]).

Urban Sound characterisation is a problem that has been subjected to studies by the scientific community. It consists of analysing and detecting relevant sound class events that can arise from various occurrences and locations to reason about abnormal occurrences and actions for a given location.

Efforts have been made to develop computational algorithms to automatically classify Urban Sounds acquired at different instants in the same location or different ones, extract sound features and classify a set of particular sounds. However, limitations are still present regarding the combination of multiple classes, abnormal noise conditions and a wide range of events co-occurring (Das et al. [2, 3], Mushtaq and Su [22]).

Earlier sound classification algorithms are traditionally based on handcrafted features (Giannakopoulos et al. [9], Gong et al. [11], Luz et al. [18], Mu et al. [21]). Recently, the proposed algorithms are based on Deep Learning (DL) approaches, with the most successful DL architecture being the Convolutional Neural Networks (CNN) (Luz et al. [18], Mu et al. [21]) and recent Transformers (Akbari et al. [1], Elliott et al. [8], Koutini et al. [16], Park et al. [23], Wyatt et al. [41]). In the CNN, the data is propagated through layers via convolutions and other operations, such as pooling, flattening and dropout, giving the network the ability to learn both local and high-level on the image space (Giannakopoulos et al. [9], Luz et al. [18]). Sound Classification based on CNNs has already been proposed, with most of the current approaches exploring the use of pre-trained CNNs, by redefining the last layers to address the Sound Classification problem (Mushtaq and Su [22], İlker Türker and Aksu [46]), and recently using attention models (Akbari et al. [1], Kong et al. [15]) and novel augmentation techniques (Mushtaq and Su [22], Salamon and Bello [29]).

As to Urban Sound classification, which has as the main objective the extraction of sound events of relevance from urban scenarios, most of the proposed solutions are based on CNNs, with some works being supported by Recurrent Neural Networks (RNN) (Gimeno et al. [10], Kong et al. [15]) particularly for sound events that occur in sequence, but should be understood as only one sound event, e.g. footsteps (Kong et al. [15]), also by exploring Long-Short Term Memory (LSTM) models (Das et al. [2], Gimeno et al. [10]), and recently attention mechanisms (Qiao et al. [25], Ristea et al. [26], Tripathi and Mishra [38], Zhang et al. [43, 44]). However, the optimal architecture for each application has not yet been established, and many opportunities are still possible. Urban Sound understanding has not been addressed properly to operate in real urban scenarios and distributed environments. There is a bright future for DL applied to Urban Sound systems, with its huge potential to complement other forms of sensing such as an image, and enable multimodal sound and image understanding, solutions still not fully explored (Das et al. [3]).

1.2 Methodology to be employed

For the development of this work, first, the representation capabilities of different features and their combinations using a model composed of several dense layers and dropout layers will be explored. Besides, the influence of the dropout rate and the increase of models' depth will be tested.

Afterwards, different end-to-end models will be employed, like Dense Convolutional Network (DenseNet), Residual Neural Network (ResNet) and Inception. These models will be used with no pre-training and pre-training from the image domain to see the variations in terms of performance, and the difference data augmentation can cause.

Ultimately, a Transformer model will be implemented with no pre-training, with pre-training only from the image domain and pre-training from the image and audio domain. Furthermore, the influence of using distinct batch sizes and data augmentation techniques will be studied.

Moreover, for all of the mentioned models, it will be applied several optimization functions to discover which one is the most advantageous.

These procedures will be followed to find the best model for sound classification in urban environments that is robust in different scenarios.

1.3 Objectives

The principal objective of this project is to develop and apply Machine Learning (ML) algorithms to automatically detect and classify Urban Sound events in different urban environments and obtain the reasoning about abnormal sound occurrences, which are vital tasks in modern urban surveillance systems. Specifically, the idea consists of characterising and exploring a wide set of ML models to perform multi-class sound classification from urban scenarios, where the diversity of classes that can be automatically identified ranges from dog bark, traffic, horns and others, while simultaneously being capable of dealing with abnormal sound events such as gun fires, explosions or other unusual sound events. The input data will be sounds from publicly available datasets, allowing the extraction of useful information from the sound events and automatically identifying and classifying relevant sound events.

The underlying purpose is to decrease the response time of the authorities by providing relevant alerts to sound occurrences that require their prompt response while, at the

same time, freeing them to mitigate other issues that concern the urban scenario operation.

The Urban Sound understanding underlies multiple variables that are hard to identify in useful time and require constant attention from a vast personnel group. The solutions to be developed will be useful to reduce the person's workload while providing information about city patterns that can be explored to improve the quality and safety of the citizens' life.

With this in mind, the objectives of this thesis rely on the accomplishment of the following steps:

- Deepening the knowledge in the field of sound features and models for Urban Sound Classification and their optimal combination to obtain the best performing model.
- Evaluation of the most promising State-of-the-Art (SOTA) techniques and formulation of model variations to address specific Urban Sound challenges to be applied to real-world scenarios, among auxiliary methods.
- Design and implementation of DL model to overcome some of the identified limitations in the main baselines and derived methods to better understand the limitations and potentialities of the end-to-end models.
- Publish the attained developments, starting with a survey that encompasses the relevant SOTA in the field and a specific application of Transformers in the Urban Sound scenarios, supported by extensive comparison with other SOTA approaches.

1.4 Outline of the Thesis

This thesis is structured as follows:

- Chapter 2 concisely presents a literature review regarding Urban Sound models and applications, focusing on strategies and architecture variations to enable multi-class classification.
- Chapter 3 presents a detailed analysis of baseline models, focusing on feature combinations, optimization and model depth to identify the most prominent set of models.

- Chapter 4 presents a detailed analysis of end-to-end models, focusing on the multi-sound classification tasks, to identify the ideal set of models and hyperparameters to be employed in real-world scenarios.
- Chapter 5 presents a detailed analysis of the end-to-end model based on the attention mechanism, with a focus on the multi-sound classification tasks, to identify the impact of model architecture, transfer learning approaches from image and sound domains, and data augmentation and corresponding hyperparameters to determine the optimal model.
- Chapter 6 discusses the overall findings in models for Urban Sound classification and points for new lines of research.

Chapter 2

Literature Review

In this chapter, the main methods employed in sound classification and segmentation are assessed by identifying and discussing State-of-the-Art (SOTA) works relevant to the development of this thesis.

2.1 Methodology of Systematic Review

This section addresses the methodology used to search and select the SOTA works under study. The main goal was to sort out the important recent works on Environmental/Urban Sound classification and processing. The following complementary questions were considered:

- Which dataset was used?
- Which architecture was utilized (developed or adapted)?
- What metrics were used for evaluation?

2.1.1 Search Method

A systematic literature search was conducted from June to September 2022 using Scopus, Science Direct, and Semantic Scholar with the following keywords in various combinations: ”environmental sound”, ”urban sound”, ”classification”, ”processing”, ”segmentation”, ”machine learning”, ”deep learning” and ”transformers”. After removing duplicated results, 1215 unique results were produced. Based on an analysis of the title and abstract, 826 studies were excluded for being utterly unrelated to the subject of study. Of the remaining studies, 301 works were excluded by applying the following criteria:

including only studies written in English, peer-reviewed studies (research articles, literature reviews, and book chapters), and finally, by carefully reviewing the body of text, only 26 studies were maintained. Figure 2.1 presents a Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) diagram of the systematic search process performed.

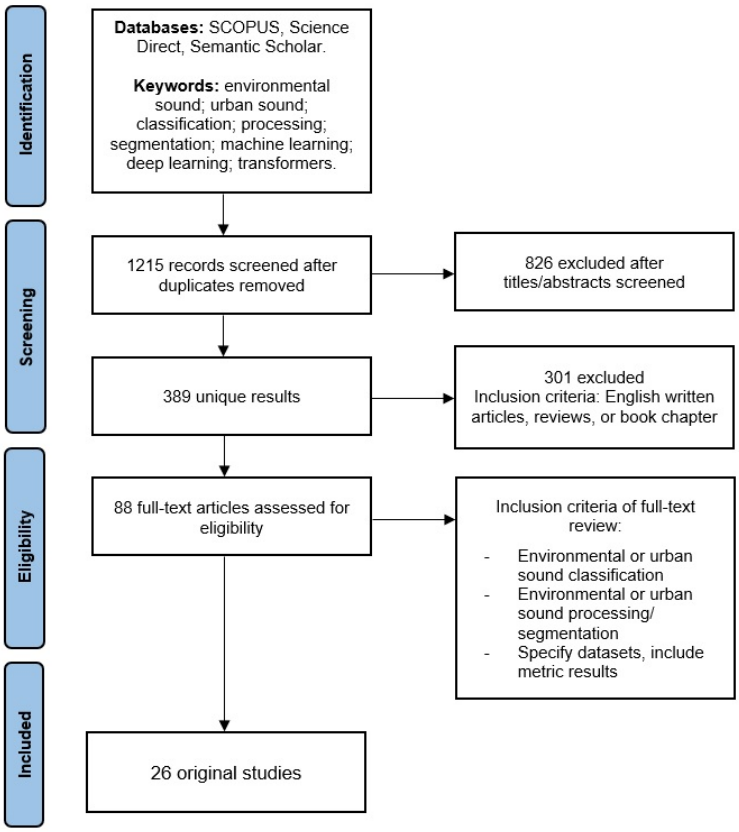


FIGURE 2.1: PRISMA diagram of the performed literature search process.

2.2 Sound Classification Methods

Sound classification methods can be applied in several areas, ranging from surveillance and noise mitigation, or context-aware computing. Therefore, to most accurately attribute a class to a specific sound, several Machine Learning (ML) models were developed capable of extracting nuclear characteristics of audio samples during training and then classifying unseen audios with a certain degree of confidence.

2.2.1 Neural Networks

Researchers have identified some limitations that prevented them from obtaining good results on the sound classification task. Therefore, Salamon and Bello [29] employed a Deep Convolutional Neural Networks (DCNN) in combination with data augmenting techniques (noise injection, shifting time, changing pitch and speed) among the training set to solve the scarcity of labelled data. Das et al. [2] used a Long-Short Term Memory (LSTM) in combination with spectral features obtained from the audio training segments. Das et al. [3] explored the use of a Convolutional Neural Networks (CNN) model with a specific Additive Angular Margin Loss (AAML) and more commonly explored the use of stacked features such as Mel Frequency Cepstral Coefficients (MFCC) and Chromagram in combination with a CNN. Zinemanas et al. [45] used an Audio Prototype Network (AP-Net) model which is composed of two components: an autoencoder and a classifier. Mu et al. [21] introduced a CNN-based model associated with attention mechanisms, called Temporal-frequency attention based Convolutional Neural Network (TFCNN).

The goal of the models is to provide a good generalization performance for unseen data, commonly requiring large quantities of data to effectively train the models. To address the scarcity of labelled data for Environmental Sound Classification (ESC), Salamon and Bello [29] proposed four different augmenting deformations to apply to the training set:

- **Time stretching**: slows down or speeds up the audio sample, but the pitch remains unchanged.
- **Pitch shifting**: the audio sample's pitch is raised or lowered while keeping the duration unchanged.
- **Dynamic range compression**: compress the dynamic range of the audio using parameterizations from the Dolby E standard* and the Icecast online radio streaming server†.
- **Background noise addition**: mix background sounds' recordings from different scenes with the audio sample.

Furthermore, a detailed analysis of the different techniques is performed to determine the impact of the various data augmentation in the final accuracy, enabling quantification

*Standards and Practices for Authoring Dolby Digital and Dolby E Bitstreams. Dolby E Bitstreams [5]
†https://icecast.org/ (accessed 29 August 2022)

of the contributions of each of the data transformations employed on the training data, suggesting that a class-conditional augmentation during training would be beneficial.

Moreover, it is necessary to understand which features and models can achieve better accuracy. Das et al. [2] presented a comparative study between a CNN and a LSTM model using different combinations of spectral features. First, a pre-processing of the audio signal is performed to reduce the amount of redundant information; the Nyquist-Shannon theorem states that the sample rates should be at least twice the value of the frequency of a continuous waveform. However, to reduce the training time, the down-sampling was achieved using the librosa library (McFee et al. [20]) default sampling rate. The next step corresponded to the extraction of spectral features such as MFCC, Mel-spectrogram, Chroma Short-Term Fourier Transformation (STFT), Chroma Constant Q-transform (CQT), Chroma Energy Normalized Statistics (CENS), Spectral Contrast, and Tonnetz, combined with augmentation of the training data (pitch shift, time stretch and pitch shift with time stretch), with the final models employed in the classification of the sound event and with a detailed evaluation of the respective accuracy, made it possible to reach the following conclusions:

- An increase in the number of epochs lead to an exponential decrease in the validation error for training and testing data. Still, after a certain number of epochs, the improvement in the validation error is negligible.
- LSTM model has better performance, in most cases than the CNN, which becomes more pronounced with the data augmentation since the LSTM memory cell includes constant error backpropagation, which allows dealing better with data noise.
- Focusing on the influence of the different features, the one which gives the best accuracy result is the MFCC. However, it is possible to outperform that by using a stack of different features such as MFCC and Chroma STFT.

Hence, the best accuracy performance was achieved by the LSTM model with the stacked features of MFCC and Chroma STFT.

Besides the concerns with the type of model and the features that are the best performing ones, it is also necessary to consider the loss function, limiting the model's classification accuracy potential. Das et al. [3] evaluated different loss functions like Softmax loss, angular Softmax loss, large margin cosine loss, and additive angular margin loss among the model's final accuracy; as input, the MFCC features used alone, and the

stacked features version that combines MFCC and Chromagram are compared in terms of final model accuracy results. Detailed analysis of the results showed a significant improvement in the performance of the models when the features were stacked together. Besides, it was possible to conclude that there is a boost in the accuracy when a modified Softmax loss function is used in comparison with the Softmax loss function, resulting in the best model to be the CNN-based model with additive angular margin loss, with the input data to be the stacked features of MFCC and Chroma STFT.

In addition, it is essential to make the model predictable to identify which input parameter drives the model's decisions and reduce future malfunctions. To achieve this, Zinemanas et al. [45] proposed an APNet composed of two main components: an autoencoder and a classifier. The autoencoder is composed of the encoder, which is constituted by three convolutional layers, where after the first two convolutional layers, a max-pooling layer is applied to capture features at different time-frequency resolutions, and the decoder, formed by three transpose convolutional layers that allow obtaining good audio quality in the reconstruction by minimizing the reconstruction error given by the Euclidean mean square loss function over its inputs and output. Then, the classifier consists of three layers: a prototype layer, a weighted sum layer and a fully connected layer. The prototype layer is responsible for storing a set of prototypes, which are learned in the latent space, that are representatives of each class. In order to learn the prototypes in the latent space, it is necessary to minimize the loss function, which happens when there is at least one similar training example for each learned prototype. Therefore, the training examples will cluster around prototypes in the latent space. The output of this layer is a similarity measure based on the distance from each encoded data instance to each prototype. The similarity measure has two steps: calculating a frequency-dependent similarity and integrating the frequency dimension using a learnable weighted sum. The frequency-dependent similarity assigns a different weight to each frequency bin in the latent space, allowing the prototypes to be based on the most relevant frequency bins, calculated using the square Euclidean distance, followed by a Gaussian function. Subsequently, the frequency dimension is integrated to obtain $\hat{S}$ while using the following weighted sum:

$$\hat{S}_{ij} = \sum_{f=1}^{F} H_j[f] S_{ij}[f] \tag{2.1}$$

where H is the trainable kernel and F the length of the vector for each prototype.

This allows the network to learn the best way to weigh each frequency bin for each prototype. The kernel is important in discriminating between overlapping sound classes by focusing on each prototype's most relevant frequency bins. Finally, the fully connected layer learns the decisions to transform the similarity measure into the predictions; to be able to perform classification, the activation function of this layer is a Softmax. Due to the expectation that the network gives more weight to the prototypes related to the class and obtains more interpretable kernel weights, the bias term is not used. It is also important to refer that since the prototypes can be converted from the latent space to the time-frequency input representation by applying the decoder function, it is possible to illustrate the prediction process while using the Melspectograms of data instances and prototypes, even though this is performed in the latent space. Therefore, the network has as input a time-frequency representation of the audio signal, from where the autoencoder transforms into its representation in the latent space of valuable features. Then, the classifier reuses the encoded input to make a prediction based on the similarity between the encoded input and a set of prototype representatives of each class. Accordingly to the previous description is possible to understand that this model provides an insight into the decision-making process, eliminating redundant prototypes and channels and determining the prototypes that are more representative of each class. This enables an understanding of which operation is more beneficial for identifying a specific sound, leading to an immediate improvement in the results. However, even though the model provides a good baseline, the accuracy of a non-interpretative model is much higher, despite not being as transparent about what drives its decisions.

As different mechanisms can identify sounds, Mu et al. [21] proposed a TFCNN that, due to the frequency and temporal attention mechanisms, can reduce the influence of background noise and irrelevant frequency bands. The authors also concluded that transient sounds enhance their classification performance when using temporal attention mechanisms. In contrast, continuous sounds benefit more from a frequency attention mechanism. So, the weight combination of both attention mechanisms gives more attention to the useful information and makes the feature distribution of sound events clearer and distinguishable. The model's architecture consists of two parts, the generation attention and the backbone network. The generation attention part aims that the calculations used to represent learning to be concentrated in specific areas by giving different degrees of attention to the frequency band and time frame parts of the extracted Log-Melspectrogram

from the original data. Accordingly, the temporal attention mechanism focuses on the semantically related time frame part and suppresses noise or silent frames. The frequency attention mechanism assigns more weight to the active frequency bands with distinguishable information. Then, the backbone network part of the model consists of a convolutional layer, a pooling layer and a fully connected layer, making it possible to extract time-frequency features from the spectrogram processed by the attention mechanism and predict sound events. The results obtained for this implementation were inferior to some CNN-based models. Nevertheless, the authors find their model still advantageous because it can ensure high accuracy with low network structure complexity and simple feature processing.

Table 2.1 summarizes the works found on audio classification using Neural Networks and their focus, limitations, and performances.

2.2.2 Transformers

Other researchers also based their models on attention mechanisms, particularly in a transduction model called Transformer, due to its several advantages, such as the total computational complexity per layer, the amount of computation that can be parallelized and the path length between long-range dependencies in the network. This section presents some models based on the Transformer's architecture.

Some researchers created models with a hybrid architecture combining Transformers with CNN like Kong et al. [15], which presents a Convolutional Neural Network Transformer (CNN-Transformer) and an automatic threshold optimization method. Others focus on models based only on Transformers, such as the ones developed by Elliott et al. [8] and Wyatt et al. [41] which present Bidirectional Encoder Representations from Transformers (BERT) based models capable of performing sound classification at the edge. In the case of Gong et al. [11], the authors developed an Audio Spectrogram Transformer (AST), which is a convolutional-free, purely attention-based model able to provide one output for a single channel audio input. Park et al. [23] introduced Many-to-Many Audio Spectrogram Transformer (M2M-AST), a model based on AST, but that allows different resolution outputs sequences for multi-channel audio inputs. Akbari et al. [1] presented a Video-Audio-Text Transformer (VATT) and a technique to reduce the training complexity

TABLE 2.1: Summary of the found works on audio classification using Neural Networks.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Salamon and Bello (2017) [29]	DCNN combined with data augmentation techniques (Time stretching, pitch shifting, dynamic range compression, background noise).	Overcomes the problem of data scarcity; Shows that Deep Learning (DL) models produce better results due to their representational power and capacity combined with data augmentation.	Some augmentation techniques have a negative impact on some classes' accuracy results.	UrbanSound8K; Accuracy (73% without data augmentation and 79% with data augmentation).
Das et al. (2020) [2]	CNN and LSTM models with a stack of multiple features as input and data augmentation techniques (pitch shift, time stretch and pitch shift along with time stretch).	Increasing the number of epochs leads to a decrease in the validation error until reaching convergence; LSTM deals better with data noise; The single feature input that allows the best result is MFCC; however, the stack of features of MFCC and Chroma STFT gave the best results out of all input features.	Needs large datasets; Execution time of 37.14 min with a GeForce RTX200 GPU with 6 Gigabytes of VRAM and boost clock of 1.68 GHz and consumes around 8 Gigabytes of RAM to train both models.	UrbanSound8K; Accuracy (98.81% for LSTM model using data augmentation and stack of MFCC and Chroma STFT as input).
Das et al. (2021) [3]	CNN model with a single feature input (MFCC) and with a stack of features using as loss function a modified Softmax loss function.	The stack of MFCC and Chroma STFT as input provided the best results; A modified Softmax loss function shows to be more beneficial than the Softmax loss function; Additive angular margin loss is the loss function that gave the best results.	The sophisticated loss functions create an intelligible space to separate the different classes due to a compactness increase within classes.	UrbanSound8K; Accuracy (99.60% of CNN model with an additive angular margin loss function without data augmentation).
Zinemanas et al. (2021) [45]	APNet with a time-frequency representation of the audio signal as input; prediction based on the similarity between the encoded input and a set of prototypes.	Provides insights into the decision-making process, helping the design of better models; Model more explicit, giving the possibility to understand which prototypes are more representative of each class and which operation is more beneficial for identifying a specific sound.	The obtained results aren't competitive with a non-interpretative DL model.	Medley-Solos-DB, Google Speech Commands, UrbanSound8K; Accuracy (65.8% Medley-Solos-DB; 89% Google Speech Commands; 76.2% for UrbanSound8K).
Mu et al. (2021) [21]	TFCNN model which is a CNN model with temporal and frequency attention mechanisms.	Attention mechanisms reduce the background noise and irrelevant frequency bands influence; Low network structure complexity and simple feature processing.	Doesn't show a similar improvement for all classes, even negatively impacting the classification for some classes.	UrbanSound8K, ESC-50; Accuracy (84.4% for ESC-50; 93.1% for UrbanSound8K).

with a minor reduction of the end Transformer's performance, DropToken. Also, to reduce the computational and memory complexity, Koutini et al. [16] introduced a method designated as Patchout.

Motivated by the fact that CNN does not capture the long-time dependencies in an audio clip well, audio recordings are usually weakly labelled and need the right thresholds to detect sound events, Kong et al. [15] developed a CNN-Transformer and an automatic threshold optimization method.

Hence, the proposed model has as input a time-frequency representation such as Log-Melspectrogram to which a CNN is applied to extract high-level features used to obtain embedding vectors along the time axis. Then, these embedding vectors serve as input to the encoder part of the Transformer, which allows the modelling of dependencies without

regard to their distance in the input sequence and more parallel computing. Finally, a fully connected layer followed by a sigmoid non-linearity is applied to the output of the encoder part of the Transformer to predict the presence probabilities of sound classes over time steps.

Consequently, to solve the scarcity of strongly labelled data, the use of weakly labelled datasets to train the model was proposed, which can be categorized into two types of training: segment-wise training and clip-wise training. The difference between them is that the audio clip is divided into several segments for segment-wise training. Each segment inherits the tags of the audio clip, which can lead to incorrect tags of the segments because they may not contain the sound event. Clip-wise training addresses the previous issue by learning the tags from the hidden layer of a Neural Network. The prediction on the clip can be obtained by aggregating the segment-wise predictions, so it can be trained in an end-to-end way with weakly labelled data and outputs directly from the clip-level prediction, in contrast with segment-wise whose outputs are latent representations learnt by the Neural Network.

Lastly, applying thresholds to the system's output is necessary to obtain the presence or absence and the onset and offset times of sound events. The researchers have proposed an automatic threshold optimization method to select the optimal thresholds. This method consists primarily of optimizing and evaluating the systems based on the metrics that do not depend on the thresholds, such as mean average precision. Then, for a trained system, an optimization of the thresholds was made over a specific metric such as F1-score or error rate. This optimization method was tested in several CNN-based models, including the CNN-Transformer, and proved to be effective and beneficial by improving the results of the models.

However, effective CNN-based models require many parameters, which poses difficulties in operating in small edge devices, being unsuitable for real-life situations. To carry out a sound application in real-life situations, Elliott et al. [8] have evaluated various audio features extraction techniques on BERT-based Transformers, then Wyatt et al. [41] have employed a trained BERT-based tiny Transformer on a resource-constrained device and deployed it in noisy environments to perform ESC. Both works are based on BERT architecture introduced by Devlin et al. [4] and consist of a multi-layer bidirectional Transformer encoder based on the original implementation described by Vaswani et al. [39] having as input a given token summed with the position embeddings.

In the research work of Elliott et al. [8], the main contribution is evaluating Transformers' performance using several feature extraction methods and their convenience when applied at the edge. So, besides introducing several feature extraction techniques: amplitude reshaping, curve tokenization, Vector-quantized varitional autoencoders (VQ-VAE), MFCC, Melspectrogram and the combination of MFCC, Gammatone Frequency Cepstral Coefficient (GFCC), CQT and Chromagram, the authors also used eleven different augmentation techniques: amplitude clipping, volume amplification, echo, lowpass filter, pitch, partial erase, speed adjust, noise, Harmonic Percussive Source Separation (HPSS), bitwise downsample and sampling rate downsample, on the raw audio files. By analysing the results obtained, it was possible to understand that, in general, the various data augmentations techniques lead to better accuracy results and that the best feature extraction method was the Melspectrogram, which outperformed all the others. This method is advantageous because it's a reasonable inexpensive computational operation. Regarding MFCC, they performed slightly better than raw amplitudes, and adding additional feature extraction methods improved the accuracy. However, the cost of computing features using all four feature extraction methods becomes prohibitive, leading to a prolonged training and inference time, which would unlikely be helpful at the edge. The authors also found that models trained in traditional frameworks have relatively little support for models that can be run at edge devices. The accuracy results obtained with the Transformer based model to datasets with a small number of examples per class leads to an inferior performance compared to a CNN-based model.

For the work presented by Wyatt et al. [41], the objective was to develop a robust ESC model capable of working in operational resource-constrained settings. The model's architecture is based on the design implemented by Elliott et al. [8], which is divided into three parts:

- an input transformation base which allows choosing the embedding dimension and is composed of a batch normalization layer, followed by a linear layer. This linear layer allows one to scale up or down one of the dimensions of audio features to a chosen dimension. After passing through the linear layer, a positional embedding to the feature vector is added to incorporate positional information.

- a classic Transformer body which is a scaled-down version of BERT.

- a prediction head, after which three layers: a mean, a linear and a Softmax are used. The mean layer does a global average pooling of the output. The other two layers enable the mapping of the features to output classes before training using cross-entropy loss.

The methodology implemented to train the model was to use a data loader that takes a random slice of audio from each audio file and then augments it with a random amount of noise between a chosen noise threshold. The amount of noise added varies to make the model robust to high and low signal-to-noise ratios, reducing overfitting. Therefore, the training with noise can generalize to audio without noise, leading to better results when compared to its non-noisy counterpart. Integrating noise resiliency directly into the model prevents having to construct specific environmental/device noise acoustic filters to handle noise. It can be scaled to be used in cases where thousands of low-power devices are deployed in various environments.

Due to the need to have a Transformer model that is capable of having competitive results with datasets that have few examples per class, that can support variable-length inputs and can be applied to different tasks without change of architecture, Gong et al. [11] proposed the AST model. The model is a convolutional-free, purely attention-based model directly applied to an audio spectrogram and can capture long-range global context even in the lowest layers. Its architecture consists only of the encoder part of the standard Transformer's architecture which is simple to implement and reproduce and makes it easier to perform transfer learning. Since images and audio have similar formats, it is possible to apply cross-modality transfer learning. To do that, it was used an off-the-shelf pre-trained Vision Transformer (ViT) since it has an architecture similar to AST. However, some modifications were still needed because ViT input is a 3-channel image. In contrast, the AST's input is a single-channel spectrogram; so to solve this, it is necessary for ViT patch embedding layer's weights for each of its three input channels to be averaged. Then, they serve as the weights of the AST patch embedding layer. The input audio spectrogram is normalized so that the datasets mean and standard deviation are 0 and 0.5, respectively. Another concern to have is with the positional embeddings because it learns to encode the spatial information during the training of the ViT, so to adapt the positional embeddings, it's proposed a cut and bi-linear interpolate method that enables the transference of the two-dimensional (2D) spatial knowledge from a pre-trained ViT to the AST even when

the input shapes are different. Finally, the last classification layer of the ViT is abandoned and reinitializes a new one for AST.

These modifications make it possible for AST to use various pre-trained ViT weights for initialization, which leads to better results than a randomly initialized AST, becoming more significant when the training volume is smaller, demonstrating the reduction in the demand for in-domain audio data. The authors also found that Data efficiency image Transformer (DeiT), because it uses data augmentation and a knowledge distillation token, improves data efficiency and generates better results. Then, regarding the impact of positional embedding adaptation, it demonstrated the importance of transferring spatial knowledge. As for the impact of patch split overlap, it was noticeable that enlarging the overlap length increases the model's performance and the computational overhead, which grows quadratically. Lastly, regarding patch shape and size, splitting the audio spectrogram into rectangular patches in temporal order achieves better results than splitting it into square patches, which cannot be in the temporal order. However, researchers have used squares patches because no pre-trained model was available that used the same dataset as ViT and rectangular patches.

Ultimately, the AST model was tested for various datasets achieving SOTA results while maintaining the same architecture regardless of the input audio length.

Although the previous model has achieved good results, it can only give one audio classification output for single channel input. Thus, to handle a multichannel audio input and have different resolution output sequences, Park et al. [23] proposed the M2M-AST, which is capable of doing sound event localization and detection that consists of two tasks: sound event detection and direction of arrival estimation.

The proposed model has a similar architecture to AST. The only differences are the input feature and the classification token configuration. M2M-AST uses as input features multichannel feature images extracted from 4-channel audio recordings, which will be segmented into a patch sequence. Afterwards, patch tokens will be extracted through a linear projection for each patch. Since the goal is to do sound event localization and detection, the model should output a series of outputs. Therefore, patch embedding consists of appending a classification token sequence with a length equal to the length of the output sequence at the beginning of the patch token sequence. Then, a learnable positional embedding was added to the patch embedding to take advantage of the patch tokens' position information. The output of the classification token sequence learns the

audio spectrogram representation by computing self-attention between each patch token. Finally, it used a denser layer with an activation layer for each task.

Regarding transfer learning, M2M-AST transfers the weights learned by DeiT. However, some changes were necessary because of the layer learning patch embeddings, which vary in size. Therefore, since DeiT uses 3-channel input images, for M2M-AST, the weight corresponding to each channel in the linear projection layer uses the average weight of the three channels in DeiT. Another change is in the positional embeddings for the patch tokens, which are transferred as scale values through cut and bi-linear interpolation to map relative positions of the positional embeddings in DeiT to the input feature.

Lastly, some experiments were performed, and the results showed that longer inputs improved both precision and recall, configuring dense patch segmentation with significant overlap helped improve performance, and a smaller resolution resulted in slight performance gains due to median filtering for sound event detection. However, for the other task, the results did not vary significantly with changes in output resolution, finally, for sound event detection, Soft F-loss performed slightly better than binary cross-entropy, and for the direction of arrival estimation, masked Mean Squared Error (MSE) improved performance over binary cross-entropy.

Another concern is reducing the training time while maintaining competitive results. To address that, Akbari et al. [1] introduced VATT and a technique to reduce the training complexity with a minor reduction of the end Transformer's performance, DropToken.

The VATT model is suitable for different downstream tasks in audio, text and video fields. Its architecture is the same as the encoder part of the standard Transformer's architecture, except for the layer of tokenization and linear projection reserved for each modality separately. Therefore, for each modality, the raw input is projected into an embedding vector in the tokenization layer and fed into a Transformer. However, for the video or audio modality, before feeding the token sequence into the Transformer, the DropToken technique has been applied, which randomly samples a portion of the tokens and then only feeds the sampled sequence to the Transformer. This approach reduces the computational costs, consequently reducing the training time and enabling hosts of large models on hardware with limited memory. The model also presents two major settings: the backbone Transformers separated and with specific weights for each modality, and the single backbone Transformer applied to any modality with shared weights. In both, the backbone extracts modality-specific representations, which are then mapped to common

spaces, by multi-level projections to be compared with each other by contrastive losses, so the model is optimized based on the back-propagation of the average loss calculated over a batch of samples, the loss objective used to align video-audio pairs was Noise Contrastive Estimation (NCE) and to align video-text pairs, Multiple Instance Learning NCE.

Regarding the results of several experiments with this model, researchers concluded that Transformers are effective for learning semantic video, audio and text representations, even with a model that shares across modalities. Also, multi-modal self-supervised pre-training is promising for reducing the dependency on large-scaled labelled data. DropToken proved to significantly reduce the pre-training complexity and to have accuracy and training costs comparable to or better than low-resolution inputs for audio and video modalities with little impact on the model's generalization.

Koutini et al. [16] introduced Patchout, which is a method to reduce the computational and memory complexity for the Transformers' training and, in addition, improves the generalization of the trained Transformers by acting as a regularizer. Therefore, its function is to drop parts of the Transformer's input sequence during training. First, small overlapping patches are extracted from the input spectrograms to form the Transformer input sequence and projected linearly to vectors. Then, the patches are augmented with both frequency and time encoding. Lastly, to reduce the sequence length and regularize the training process, parts of the sequence are randomly dropped during training. However, the whole input sequence is given to the Transformer during inference.

Two types of Patchout methods were introduced the unstructured Patchout, which chooses the patches randomly regardless of their position and the structured Patchout, which randomly picks some frequency bins or time frames and removes a whole column or row of extracted patches.

The researchers also enhance the models' performance and prevent overfitting by making use of ImageNet pre-training and some data augmentation techniques such as two-level mix-up: mix the final spectrograms with random raw waveforms from the dataset; SpecAugment: masks up to a certain number of frequency bins and time frames; rolling: rolls the waveforms randomly over time; random gain: multiplies the audio waveform to change the gain by $\pm$7decibel (dB). Thus, with the development of Patchout was possible to effectively train Transformers on audio spectrograms and achieve SOTA results.

Table 2.2 summarizes the found works on audio classification using Transformers and their focus, limitations, and performances.

TABLE 2.2: Summary of the found works on audio classification using Transformers.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Kong et al. (2020) [15]	CNN-Transformer model and an automatic threshold optimization method.	Computations are done in parallel; Use weakly labelled datasets to train the model and outputs directly clip-level predictions; Automatic threshold optimization.	CNN-based models need a lot of parameters.	DCASE2017 Task4; F1-score (AT - 64.6%; SED - 57.3%); Precision (AT - 69.1%); Recall (AT - 60.7%); Error rate (SED - 68%). AT - audio tagging; SED - sound event detection.
Elliott et al. (2021) [8]	BERT-based Transformer for ESC at the edge.	Evaluation of Transformers' performance using several feature extraction techniques and data augmentation; Enables ESC on edge devices.	Models trained in traditional frameworks have little support to be converted to models that run at the edge; Cannot have competitive results when trained with small datasets.	ESC-50, Office Sounds; Accuracy (67.71% for ESC-50, 95.31% for Office Sounds).
Wyatt et al. (2021) [41]	BERT-based Transformer for ESC on a resource-constrained device applied in noisy environments.	The model trained with noise augmented data can generalize to audio without noise and prevents having to construct custom acoustic filters to be able to apply the model in real-life environments.	Needs large datasets; Only employed on small form factor edge devices.	Office Sounds; Accuracy (non-noisy dataset: 75.4%, noisy dataset: 81.2%), Precision (non-noisy dataset: 76.5%, noisy dataset: 79.7%), Recall (non-noisy dataset: 75.6%, noisy dataset: 80.6%), F1-score (non-noisy dataset: 75%, noisy dataset: 80%).
Gong et al. (2021) [11]	Audio Spectogram Transformer, a purely attention-based audio classification model.	Capture long-range global context even in the lowest layers; Able to handle different input audio lengths without changing the architecture; Few parameters and fast convergence.	Can't use rectangular patches due to the inexistence of a pre-trained model that used the same dataset as ViT; Unable to use only an AudioSet pre-trained model.	AudioSet, ESC-50, Speech Commands V2; mAP (AudioSet: 0.485), Accuracy (ESC-50: 95.6% and Speech Commands V2: 98.11%).
Park et al. (2021) [23]	Audio Spectrogram Transformer that can handle various output resolutions.	Shows that Soft F-loss performs better than binary cross-entropy; Design to have a variety of output resolutions.	Large model size; Evaluates sound event localization and detection using only one dataset.	TAU-NIGENS Spatial Sound Events 2021; Error rate (0.50), F1-score (65.7%), Recall dominant score (74.7%).
Akbari et al. (2021) [1]	Transformers for multimodal self-supervised learning from raw video, audio and text.	Learns effectively semantic video, audio and text representations; DropToken technique reduces the pre-training complexity and training time, making it possible to host large models in limited hardware.	Needs large datasets due to the large size of the network.	Only 2 out of 10 datasets were from the audio domain: ESC-50, AudioSet. mAP (AudioSet: 39.4%), AUC (AudioSet: 97.1%), d-prime (AudioSet: 2.895), Accuracy (ESC-50: 84.9%).
Koutini et al. (2021) [16]	Audio Transformer with Patchout which optimizes and regularizes Transformers on audio spectrograms.	Patchout improves the generalization and reduces the computation and memory complexity.	Increases the training time.	AudioSet, OpenMIC, ESC-50, DCASE20; mAP (AudioSet: 0.496, OpenMIC: 0.843), Accuracy (ESC-50: 96.8%, DCASE20: 76.3%).

mAP - mean average precision, AUC - area under the receiver operating characteristic curve.

2.3 Sound Segmentation Methods

Proper audio segmentation is a pre-processing step in audio analysis. Its purpose is to separate the digital audio signal into segments containing audio information from a specific acoustic type. To be able to perform audio segmentation, some steps should be followed:

- **Feature extraction:** The audio input is cut into overlapping frames to allow extraction of the parametric feature vector from each frame.
- **Initial detection:** It is an optional step in which the objective is to remove the silent parts and discard the parts of the signal that are out of interest.
- **Segmentation:** The vector sequence of features is segmented into sub-sequences with common acoustic characteristics. Two main approaches can be employed: distance-based and model-based techniques.
- **Post-processing or smoothing:** It is also an optional step in which the goal is to correct the errors related to detecting segments with a duration shorter than the defined threshold.

Several metrics and algorithms can be used for the two mentioned main techniques in the segmentation step. Some examples of distance metrics are the Euclidean distance, the Bayesian Information Criterion, Kullback Leibler KL2 distance, the generalized likelihood ratio, and the Hotelling T2 statistic; in terms of models, they can range from Guassian Mixture Model (GMM), Hidden Markov Model (HMM), Support Vector Machine (SVM), Artificial Neural Networks (ANN), Boosting Technology, k-Nearest Neighbor (k-NN), Decision Trees and Fuzzy Logic (Theodorou et al. [37]).

Next, some examples of segmentation based on models are introduced; Tax et al. [36] presented a DCNN model that is capable of learning the log-scaled Melspectrogram transformation from raw waveform, providing a spectrum visually similar but slightly smoothed, showing that upon initializing the first layers of an end-to-end Neural Network classifier with the learned transformation can give comparable results to a model trained on the highly processed Melspectrograms. Besides, due to the capacity of CNNs to approximate complex mappings, it is possible to force the network to learn such transformation implicitly and limit the need for ad-hoc architectural choices. Therefore, these

findings showed that the performance of Neural Network-based models could be improved by incorporating knowledge from established audio signal processing methods.

Martín-Morató et al. [19] discussed two problems arising from the temporal uncertainty of audio events: the generation of errors at the decision level for models trained with carefully annotated strong labels or perfectly segmented audio events when implemented in a realistic scenario, where weakly segmented audio events and different levels of background noise exists; and systems trained with weakly labelled datasets that face the temporal uncertainty problem directly during training. Therefore, to solve these problems, the authors have proposed a pooling layer aimed at compensating for non-relevant information of audio events by applying a non-linear transformation of the learned convolutional feature maps on the temporal axis, which follows a uniform distance subsampling criterion in the learned feature space that allows propagating better the information of the actual event through the network. The proposed pooling layer shows to be an advantageous method to learn from weakly labelled data without adding more parameters and to increase the robustness under adverse scenarios with severe training and test mismatches.

As for Gimeno et al. [10], the authors introduced a Bidirectional Long-Short Term Memory (BLSTM) network with a new block incorporated onto the Neural Network, named Combination and Pooling block, that aims to reduce the redundant temporal information through a time pooling mechanism while learning an appropriate representation through an one-dimensional (1D) convolution layer. Then, the system consists of a Recurrent Neural Networks (RNN) based classifier and an HMM re-segmentation module with the combination of Mel log filter bank, chroma features, and also, 1^{st} and 2^{nd} derivatives, as input. Adding the chroma features improved the accuracy in the ground truth boundary experiments. On the other hand, the 1^{st} and 2^{nd} derivatives incorporate the audio signal's dynamic information, which is more relevant to generate the class boundaries than for the classification task. The HMM re-segmentation significantly reduces the system error by forcing a minimum segment length for the class labels and also proved that is beneficial for the segmentation system when the output's temporal resolution is reduced. The introduction of the Combination and Pooling block allows the downsampling to be implemented inside of the Neural Network. Then, to configure the temporal pooling layers, a pooling factor regulates the length of the output sequences in relation to the input length. The pooling layers separate an input sequence into different sub-sequences with

the same length and no overlapping. After some experiences with multiple configurations changing the layers and the position of the Combination and Pooling block compared to the BLSTM, allowed the authors to conclude that having only a pooling layer between the two BLSTM layers makes it possible to achieve better results without adding more parameters while decreasing computational complexity. Then, to improve the model's results, it was used a data-agnostic data augmentation routine, Mixup, that generates new virtual training examples $(\tilde{x}, \tilde{y})$ according to the following equations:

$$\begin{cases} \tilde{x} = \lambda x_i + (1 - \lambda) x_j \\ \tilde{y} = \lambda y_i + (1 - \lambda) y_j \end{cases} \tag{2.2}$$

where (x_i, x_j) are two feature vectors randomly drawn from the training dataset, (y_i, y_j) are their corresponding one hot encoding labels and $\lambda \in [0, 1]$.

Other methods like the one introduced by Giannakopoulos et al. [9] focus on the feature extraction step. The objective of Giannakopoulos et al. [9] was to use CNNs as a method to extract context-aware deep audio features that can offer additional feature representations to any soundscape analysis classification, which proved, when combined with handcrafted audio features, to give a boost in the classification accuracy without the need for CNN training. The two feature representation steps are combined in an early-fusion scheme and classified using SVM with a Radial Basis Function (RBF) kernel. The handcrafted audio features aim to represent the audio signal in a space able to discriminate an unknown sample between the involved audio classes. In order to do that, each signal is represented by a series of statistics computed over short-term audio features processed on a mid-term basis, which consists of first dividing the audio signal into mid-term overlapping or non-overlapping windows, then, each one of those is processed by short-term processing, and the feature sequence from each mid-term segment is used for computing feature statistics, resulting in each mid-term segment being represented by a set of statistics, either from the time or frequency domain. The examples of such features used in this work were the zero crossing rate, energy, entropy of energy, spectral centroid, spectral spread, spectral entropy, spectral flux, spectral roll-off, MFCC, chroma vector and chroma deviation. The context-aware deep features were extracted using a supervised CNN trained to discriminate between different audio urban context classes based on spectrograms using STFT of short-term segments. The output of the last fully connected layer is used as a feature extractor in the initial soundscape classification task.

By evaluating the model's performance on the datasets was possible to show that the combination of handcrafted features with the context-aware deep features culminates in a boost of the model's results.

Luz et al. [18] proposed a small parameter space CNN model to extract deep features combined with handcrafted features. In addition, a feature selection step was performed to reduce feature dimensionality, making the training process faster and less computationally expensive by identifying redundant and inconsistent features, attaining it suitable for mobile sound recognition applications or embedded systems. Also, identifying which group of handcrafted features can enrich deep features to better discriminate between Urban Sounds. The feature selection experiment results indicate that associating perceptual, static, and physical features from frequency and time domains with deep features significantly improves the classification performance.

Table 2.3 summarizes the found works on audio processing with segmentation based on models or/and handcrafted features and their focus, limitations, and performances.

Researchers have shown that deep features contain more relevant information than handcrafted features, which translates into better results. To further improve the models' performance, researchers have implemented attention mechanisms that allow focusing on the semantically relevant characteristics. Therefore, the following section focus on studies that implement different attention mechanisms.

2.3.1 Attention Mechanisms

Some studies focus on incorporating attention mechanisms to improve Convolutional Recurrent Neural Networks (CRNN) models' performance, such as the research works of Qiao et al. [25], Zhang et al. [43, 44]. The study presented by Zhang et al. [43] incorporates temporal attention and channel attention mechanisms, later, Zhang et al. [44] used a frame-level attention mechanism. Both proposals used a CRNN model constituted of eight convolution layers to learn high-level representations from the input log-gammatone spectrogram and the channel temporal attention mechanism enhanced the representational power of CNN. Therefore, to learn the temporal correlation information, it's used two layers of Bidirectional Gated Recurrent Unit (B-GRU) to which the CNN-learned features were given as input. Finally, the features are fed into a fully connected layer with Softmax as an activation function for the classification task. Also, some data augmentation techniques were used to avoid overfittings, such as time and frequency

TABLE 2.3: Summary of the found works on audio processing with segmentation based on models or/and handcrafted features.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Tax et al. (2017) [36]	End-to-end CNN model classifier with the first layers initialized.	Training the first layers of a DCNN model using unlabelled data allows us to learn high-level audio representations; Incorporating knowledge from audio processing methods can improve the performance of Neural Network-based models.	Not able to outperform the models trained on processed features.	ESC50; Accuracy (around 50%).
Martín-Morató et al. (2020) [19]	CNN-based models with an adaptive pooling layer based on a non-linear transformation of the learned convolutional feature maps on the temporal axis.	Distance-based pooling layer to improve CNN-based models for audio classification in adverse scenarios; Systems generalize better to mismatching test conditions; Learn more robustly from weakly labelled data; Allows to propagate better the information of the actual event through the network.	Only uses isolated events with a clear beginning and end.	UrbanSound8K, ESC-30, DCASE2017 T4; Macro-averaging accuracy (ESC-30: 77%, UrbanSound8K: 73.96%), F1-score (DCASE2017 T4: 48.3%), Precision (DCASE2017 T4: 68.2%), Recall (DCASE2017 T4: 46.7%).
Gimeno et al. (2020) [10]	BLSTM with a Combination and Pooling block.	A combination of BLSTM modelling capabilities with HMM backend smooths the results and significantly reduces system error; Combination and Pooling block reduce redundant temporal information.	Needs large datasets; The proposed block wasn't capable of outperforming the model with HMM re-segmentation.	3/24 TV, CARTV; Segmentation error rate (3/24 TV: 11.80%; CARTV: 24.93%), Average class error (3/24 TV: 19.25%), Accuracy (3/24 TV: 16.05%).
Giannakopoulos et al. (2019) [9]	CNN to extract context-aware deep audio features and combine them in an early-fusion scheme with handcrafted audio features.	Using CNN as a feature extractor can improve the performance of the audio classifier by the transference of audio contextual knowledge without the need for CNN training.	Low accuracy results.	TUT Acoustic Scene (used to train), UrbanSound8K, ESC-50; Accuracy (ESC50: 52.2%; UrbanSound8K: 73.1%).
Luz et al. (2021) [18]	CNN model to extract deep features that are combined with handcrafted features. As classifiers, it was used Support Vector Machine and Random Forest.	Feature selection steps to reduce feature dimensionality and understand which handcrafted features could enrich deep features to better discriminate between Urban Sounds; Deep features contain more relevant information than handcrafted features.	No application of data augmentation techniques; Only adopted one CNN model not too deep to extract features from Melspectrograms.	ESC-10, UrbanSound8K; Accuracy (ESC-10: 86.2%; UrbanSound8K: 96.8%).

masking and Mixup. These data augmentations allowed the models to focus on the semantically relevant frames, producing discriminative features, and arrive at the conclusion that applying the attention mechanism to lower layers since the attention mechanism can help preserve the lower-level features which normally carry basic and useful information, yields better results in comparison to applying to higher-level layers, and employing attention for RNN layers allows achieving the highest accuracy result and using sigmoid as scaling function generates better attention weights than Softmax when applying attention at CNN layers. Furthermore, it was possible to understand that temporal attention

reduces the impact of background noise since channel attention puts more attention on the filters, which can detect the essential characteristics of the sounds. Frame-level attention can focus on critical temporal events while reducing the impact of background noise.

Regarding the research of Qiao et al. [25], besides developing a CRNN model with a temporal-frequency attention mechanism, it has also developed a CRNN model using sub-spectrogram segmentation-based feature extraction and score level fusion to highlight the advantages of an attention mechanism. Therefore, the authors, on the one hand, show that the sub-spectrogram segmentation mechanism truncates the whole spectrogram into a certain number of parts instead of generating the log Gammatone spectrogram based on the entire frequency band. It uses a score level fusion to combine different classification results from different sub-spectrograms, considering the frequency domain characteristics but ignoring the temporal domain ones. The score level fusion improves the model accuracy compared to the uniform weights assignment, and that low-frequency bands contain a large proportion of the characteristics of Environmental Sounds. However, high-frequency bands contain a few characteristics that are still indispensable for the classification task. On the other hand, concerning the temporal-frequency attention mechanism, the following advantages were highlighted: the use of the CNN layers to extract temporal-frequency representations from the input log Gammatone spectrogram shows low complexity despite the ability to learn more valuable information from the input and gives higher accuracy results by focusing on the most critical frames and frequency bands. To conclude, SpecAugmented and Mixup data augmentation techniques were used in order to increase the diversity of the training.

Tripathi and Mishra [38] introduced an attention-based Residual Neural Network (ResNet) model that efficiently learns Spatio-temporal relationships in the spectrogram, skipping the irrelevant regions. Regarding the augmentation techniques, the authors utilized time shift, adding noise and SpecAugment. The proposed attention module allows capturing long-range contextual information between the spectrogram's local features, improving compactness and addressing intra-class inconsistency, which is the variations between spectrogram features extracted from the different signals belonging to the same class that can cause performance degradation. In addition, the study revealed that the attention module provided the best accuracy results when affixed after the last residual layer, so higher layers provided more valuable features to define the characteristics of a

sound, and the attention module conserves them.

Ristea et al. [26] developed an architecture that employs two Transformer blocks in sequence, the first block attends to tokens within the same frequency bin (vertical axis), and the second attends to tokens within the same time interval (horizontal axis) of the spectrogram. The model used noise perturbation, time shifting, speed perturbation, Mixup and SpecAugment as data augmentation techniques. This implementation linearly scales the number of trainable parameters with the input size, which reduces the memory footprint. It can handle high-resolution spectrograms and shows that performing attention only on one axis is insufficient. Thus, combining both attentions gives a considerable performance boost regardless of the chosen order for the vertical and horizontal block. The order only has a marginal influence on the results.

Table 2.4 summarizes the found works on audio processing with attention mechanisms and their focus, limitations, and performances.

The following subsection presents some autoencoder implementations.

2.3.2 Autoencoders

Different types of autoencoders allow a multiplicity of applications, such as the one described by Sudo et al. [31], which implements a multichannel Environmental Sound segmentation method which consists of four blocks: Feature Extraction, Sound Source Localization and Separation (SSLS), Sound Source Separation and Classification (SSSC) and reconstruction. In this approach, the feature extraction is constructed using the short-time Fourier transform of the initial input signal, the magnitude spectrograms as spectral features, and the sine and cosine interchannel phase difference as spatial features. The SSLS block uses Deeplav3+, which has an encoder-decoder structure, enabling it to improve the segmentation performance for Environmental Sounds with different durations. The Deeplav3+ allows the extraction of high-level features to predict a spectrogram for each azimuth angle regardless of the class. It creates a feature map much smaller than the original spectrogram, allowing it to extract an extensive range of contexts without increasing the number of parameters. The SSSC block also uses Deeplav3+, but here, the input corresponds to each spectrogram of the output of the SSLS block, inserted one by one. This block surpasses the influence of the spatial features, preventing the network from overfitting to the relationship between the direction of arrival and the given class. In

TABLE 2.4: Summary of the found works on audio processing with attention mechanisms.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Zhang et al. (2019) [43]	CRNN model with temporal and channel attention mechanisms.	The two attention mechanisms enhance the representation capabilities of CNN and lead it to focus on the semantically relevant parts of the sounds; Applying the attention mechanism to lower-layers yields better results than applying it to higher-level layers.	Doesn't quantify the robustness to noise.	ESC10, ESC50, DCASE2016; Accuracy (ESC10: 94.2%, ESC50: 86.5%, DCASE2016: 88.9%).
Zhang et al. (2020) [44]	Frame-level attention mechanism based on CRNN.	The attention model automatically focuses on the semantically relevant frames and produces discriminative features; Low computational complexity.	Doesn't quantify the robustness to noise.	ESC-50, ESC-10; Accuracy (ESC-10: 93.7%, ESC-50: 86.1%).
Qiao et al. (2021) [25]	CRNN model using sub-spectrogram segmentation based feature extraction and score level fusion; CRNN model using temporal-frequency attention.	Score level fusion improves the accuracy in comparison with the uniform weights assignment; Low complexity when generating the temporal-frequency attention map when using the attention mechanism; High accuracy results when using temporal-frequency mechanisms.	Sub-spectrogram segmentation mechanism only considers frequency domain characteristics; Multi-dimensional search spaces are needed to optimize segmentation boundaries and the number of segments which are, in general, computationally prohibitive.	ESC-50; Accuracy (ESC-50: sub-spectrogram segmentation: 82.1%, temporal-frequency attention: 86.4%).
Tripathi and Mishra (2021) [38]	Attention-guided residual network that efficiently learns Spatio-temporal relationships of a signal's spectrogram.	The attention module resolves the intra-class inconsistency; Identifies more semantically relevant parts of the spectrogram, and correctly highlights them while providing a visual description.	Doesn't quantify the robustness to noise.	ESC-10, DCASE 2019 Task-1(A); Accuracy (ESC-10: 92.16% (augmented), 92%, DCASE 2019: 82.21% (augmented), 82%), Precision (ESC-10: 88.70%, DCASE 2019: 83.47%), Recall (ESC-10: 89.80%, DCASE 2019: 82.28%), F1-score (ESC-10: 87.93%, DCASE 2019: 82.39%).
Ristea et al. (2022) [26]	Separable Transformer: separates the attention for the horizontal axis (time) from the vertical axis (frequency) of spectrograms.	Reduces the number of learnable parameters, which reduces the memory footprint; Able to handle high-resolution spectrograms.	Doesn't quantify the robustness to noise.	ESC-50, Speech Commands V2, CREMA-D; Accuracy (CREMA-D: 70.47%, Speech Commands V2: 98.51%, ESC-50: 91.13%).

conclusion, this approach allows performing a multichannel Environmental Sound segmentation without the need to set beforehand the number of sound sources, preventing the overfit between the direction of arrival and the class by explicitly separating the SSLS block and the SSSC block and, finally, the SSSC block can separate sound sources arriving from a close direction that the SSLS block was not able to separate.

Other approaches do not rely totally on autoencoders. Still, just a part of it, for example, only the encoder part, like the model presented by Venkatesh et al. [40] which illustrates a system called You Only Hear Once (YOHO) that predicts the boundaries of acoustic classes through regression. This model is a CNN whose architecture is constituted by the MobileNet architecture that allows the reduction of the time and frequency dimension by presenting a decoder-like architecture with some extra layers to flatten the last two dimensions and a final layer to perform a binary classification that detects the presence, the start, and endpoints of an acoustic class segment. The model's input feature is the Log-Melspectrograms, and the dimension of the input depends not only on the duration of the audio example but also on the specifications of the Log-Melspectrogram. For the post-processing, to smooth the output and eliminate spurious audio events, threshold-dependent smoothing is used that allows the removal of audio events whose duration was too short and the silence segments between consecutive events of the same acoustic class if they are also too short. In conclusion, this model leads to a fast post-processing and smoothing process due to YOHO's ability to directly predict the acoustic class boundaries, resulting in a more end-to-end setup. However, it is limited by the time resolution of the input; nevertheless, if the input were raw audio instead of Log-Melspectrogram, the model would be entirely an end-to-end DL approach.

Table 2.5 summarizes the found works on autoencoder-like architecture for Environmental Sound processing and their focus, limitations, and performances.

Some techniques and steps of audio segmentation had already been mentioned in the previous section, such as feature extraction widely used by all the earlier models to predict the class. Das et al. [2] used the librosa library due to the default sampling rate that allows a reduction in training time. Zinemanas et al. [45] mentioned the use of an autoencoder scheme which allows the extraction of features and the prediction of the class by

TABLE 2.5: Summary of the found works on audio processing with autoencoder-like architecture.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Sudo et al. (2021) [31]	Multichannel Environmental Sound segmentation method constituted by a sound source localization block and a sound source separation and classification block.	Not necessary to set the number of sound sources; No overfitting between the direction of arrival and the class relationship; Sine and cosine of inter-channel phase difference are optimum for sound source localization and separation.	Lack of sufficiently large datasets with separated sound source signals and direction of arrival labels.	The dataset is a combination of 10 datasets resulting in a dataset with 75 classes; Root Mean Square Error (18.59).
Venkatesh et al. (2021) [40]	YOHO: end-to-end model with a CNN architecture adapted from the MobileNet architecture.	Converts the detection of acoustic boundaries into a regression problem; Due to fast inference, YOHO is suitable for real-time applications; Directly outputs the time boundaries.	Limited by the time resolution of the input.	BBC Radio Devon and MuSpeak, MIREX music-speech detection, TUT Sound Event Detection, Urban-SED; F1-score (BBC Radio Devon and MuSpeak: 97.22%, MIREX: 90.20%, TUT Sound Event Detection: 44%, Urban-SED: $\approx$ 60%); Error rate (TUT Sound Event Detection: 0.7517).

constructing a latent space more capable of expressing the audio features. Mu et al. [21] introduced self-attention mechanisms combining a temporal and frequency attention mechanism which allows reducing the influence of background noise and irrelevant frequencies and focusing on the most important parts of the signal; besides that, the researchers used HPSS to separated between harmonic and percussive components. For the models based on Transformers, all of them also took advantage of feature extraction. However, this step is especially explored by Elliott et al. [8], which introduces several techniques, as mentioned in the previous chapter. Kong et al. [15] presented a post-processing method capable of automatically optimizing the values for the thresholds. Other researchers such as Akbari et al. [1] and Koutini et al. [16] introduced DropToken and Patchout, respectively, which are techniques that randomly drop part of the input sequence before feeding it to the Transformer in order to reduce the training complexity.

To end, the chapter has presented some new feature extraction techniques developed envisioning the enhancement of the ESC task.

2.3.3 New Feature Extraction Techniques

In this section, both works have presented new feature extraction techniques and show their efficiency by employing them in transfer learning models using several data augmentation techniques.

Therefore, the novel features presented by Mushtaq and Su [22] are based on the logarithmic scale of the Melspectrogram named L2M, corresponding to the Log(Log-Melspectrogram) and with L3M corresponding to Log(Log(Log-Melspectrogram)), which are particularly useful for two new data augmentation techniques, NA-1 and NA-2, based on Spectrogram Image Features (SIF). Both NA-1 and NA-2 use a single image as a feature at a time, but NA-1 consists of the enhancement of SIF data by combining various spectrogram-based audio features. At the same time, NA-2 is a vertical combination of various accumulated features in the form of spectral images in pairs. Besides, trim silence was used as a pre-processing technique due to the silent parts of the audio clips. Regarding the transfer learning model, Dense Convolutional Network (DenseNet)-161 with ImageNet weights was the chosen classifier which was further fine-tuned by using individual optimal learning rates in combination with discriminative learning.

İlker Türker and Aksu [46] proposed a novel time-convexity representation based on graph representations of consecutive frames after segmentation with constant window and hop-length parameters of the original sound signal. This representation, named Connectogram, is a colourful graph-generator approach that includes three layers, each derived with different undersampling rates in which the horizontal axis stands for time, and the vertical axis for signal fluctuations is a Red-Green-Blue (RGB) image that can serve as input for the models. This approach seems to carry frequency-related info pairs with amplitude information by having amplitude information of the original sound as vertical fluctuations. The intensity of these fluctuations corresponds to the colours. However, Connectogram is not a competitive representation but can significantly improve the representation capability of Melspectrograms if generated with the same segmentation parameters. The best accuracy result was obtained when a combination of two Melspectrogram with different parameters and a Connectogram was used as input for the ResNet50 model.

Concerning data augmentation, the input allows having two stages of augmentation, the first regarding deformation methods to the raw sounds and the second includes image distortion methods that are applied to the Connectogram such as rotation, horizontal and vertical shift, brightness, shear and zoom.

Table 2.6 summarizes the found works that employ new feature extraction techniques and their focus, limitations, and performances.

Table 2.7 summarizes the results of all of the articles considered in the literature review.

TABLE 2.6: Summary of the found works on audio processing that introduces new feature extraction techniques.

Authors/Year	Model features	Contributions/Benefits	Limitation(s)	Dataset/Metrics
Mushtaq and Su (2020) [22]	DenseNet-161 fine-tuned with optimal learning rates and discriminative learning.	Introduction of L2M and L3M features; New data augmentation techniques: NA-1 and NA-2; Can achieve high results with few training epochs and less amount of original data.	L2M and L3M are outperformed by other Mel filter-based features. Computationally heavy.	ESC-10, ESC-50, UrbanSound8K (US8K); Accuracy (ESC-10: 99.22%, ESC-50: 98.52%, US8K: 97.98%), Error rate (ESC-10: 0.777%, ESC-50: 1.476%, US8K: 2.018%), F1-score (ESC-10: 99.25%, ESC-50: 98.53%, US8K: 98.13%), Recall (ESC-10: 99.25%, ESC-50: 98.53%, US8K: 98.13%), Precision (ESC-10: 99.24%, ESC-50: 98.57%, US8K: 98.14%), Kappa score, Matthews Correlation Coefficient, False Discovery rate, Fowlkes-Mallows index, Miss rate.
İlker Türker and Aksu (2022) [46]	ResNet50 with a combination of two Melspectrogram with different parameters and a Connectogram as input	Introduces a time-convexity graph-based representation for sounds, Connectogram, capable of being fused with Melspectrograms to improve their representation capabilities.	Connectogram is not a powerful representation when used by itself.	ESC-10; Accuracy (ESC-10: 96.46%)

TABLE 2.7: Results summary of all models considered for the literature review.

Authors	Dataset	Acc	Other metrics
Salamon and Bello [29]	UrbanSound8k	79%	-
Das et al. [2]	UrbanSound8k (unofficial split)	98.81%	-
Das et al. [3]	UrbanSound8k (unofficial split)	99.60%	-
Zinemanas et al. [45]	UrbanSound8k Google Speech Commands Medley-Solos-DB	76.2% 89% 65.8%	-
Mu et al. [21]	UrbanSound8k ESC-50	93.1% 84.4%	-
Kong et al. [15]	DCASE2017 Task 4	-	AT - F1-score: 64.6%, Precision: 69.1%, Recall: 60.7% SED - F1-score: 57.3%, Error rate: 68%
Elliott et al. [8]	ESC-50 Office Sounds	67.71% 95.31%	-
Wyatt et al. [41]	Office Sounds	81.2%	Precision: 79.7%, Recall: 80.6%, F1-score: 80%
Gong et al. [11]	ESC-50 Speech Commands V2 AudioSet	95.6% 98.11% -	- - mAP: 0.485
Park et al. [23]	TAU-NIGENS Spatial Sound Events 2021	-	F1-score: 65.7%, Recall: 74.7%, Error rate: 0.50
Akbari et al. [1]	ESC-50 AudioSet	84.9% -	- mAP: 39.4%, AUC: 97.1%, d-prime: 2.895
Koutini et al. [16]	ESC-50 AudioSet OpenMIC DCASE20	96.8% - - 76.3%	- mAP: 0.496 mAP: 0.843 -
Tax et al. [36]	ESC-50	≈50%	-
Martín-Morató et al. [19]	UrbanSound8K ESC-30 DCASE2017 T4	73.96% 77% -	- - F1-score: 48.3%, Precision: 68.2%, Recall: 46.7%
Gimeno et al. [10]	3/24 TV CARTV	16.05% -	Segmentation error: 11.80%, ACE: 19.25% Segmentation error: 24.93%
Giannakopoulos et al. [9]	UrbanSound8K ESC-50	73.1% 52.2%	-
Luz et al. [18]	UrbanSound8K ESC-10	96.8% 86.2%	-
Zhang et al. [43]	ESC-50 ESC-10 DCASE2016	86.5% 94.2% 88.9%	-
Zhang et al. [44]	ESC-50 ESC-10	86.1% 93.7%	-
Qiao et al. [25]	ESC-50	86.4%	-
Tripathi and Mishra [38]	ESC-10 DCASE 2019 Task-1(A)	92.16% 82.21%	-
Ristea et al. [26]	ESC-50 Speech Commands V2 CREMA-D	91.13% 98.51% 70.47%	-
Sudo et al. [31]	75-classes dataset combining 10 datasets	-	Root Mean Square Error: 18.59
Venkatesh et al. [40]	Urban-SED TUT Sound Event Detection BBC Radio Devon and MuSpeak MIREX	-	F1-score: ≈ 60% F1-score: 44%, Error rate: 0.7517 F1-score: 97.22% F1-score: 90.20%
Mushtaq and Su [22]	UrbanSound8K ESC-50 ESC-10	**97.98%** **98.52%** **99.22%**	Error rate: 2.018%, F1-score: 98.13%, Recall: 98.13%, Precision: 98.14%, Kappa score: 97.09%, MCC: 97.73%, FDR: 1.854%, FM: 98.14%, Miss rate: 1.863% Error rate: 1.476%, F1-score: 98.53%, Recall: 98.53%, Precision: 98.57%, Kappa score: 98.95%, MCC: 98.49%, FDR: 1.469%, FM: 98.55%, Miss rate: 1.469% Error rate: 0.777%, F1-score: 99.25%, Recall: 99.25%, Precision: 99.24%, Kappa score: 98.93%, MCC: 99.13%, FDR: 0.758%, FM: 99.24%, Miss rate: 0.744%
İlker Türker and Aksu [46]	ESC-10	96.46%	-

Acc - Accuracy, AT - Audio Tagging, SED - Sound Event Detection, mAP - mean average precision, ACE - Average Class Error, AUC - area under the receiver operating characteristic curve, MCC - Matthews Correlation Coefficient, FDR - False Discovery rate, FM - Fowlkes-Mallows index.

2.4 Conclusion

There are three stages to performing the sound classification: pre-processing the audio signal, the acoustic feature extraction and the audio signal classification (Das et al. [2]).

Therefore, researchers have proposed several models to address this challenge, focusing on different stages of the task. Some try to develop or improve the model's architecture by using modified versions of the loss functions, methods to drop parts of the input sequence or by exploring various types of architectures such as DCNN, CRNN, LSTM, ResNet, DenseNet and more recently, Transformers. Most of the approaches were based on DL methods because even if it's more challenging to identify which parameters drive the model's decision, non-interpretative DL models show superior results (Salamon and Bello [29], Zinemanas et al. [45]). Others have concentrated more on the sound segmentation part of the task with approaches ranging from implementations of different features extraction techniques, however, the combination of handcrafted features allows typically to achieve better results (Das et al. [2, 3]), particularly when handcrafted features are combined with deep features (Giannakopoulos et al. [9], Luz et al. [18]); also, the use of model-based techniques to segment the sound like CNNs or autoencoders are capable of learning transformations from the raw waveforms and able to give comparable results to models trained on highly processed features (Tax et al. [36]) or by the introduction of new blocks or layers that allow reducing the redundant information (Gimeno et al. [10], Martín-Morató et al. [19]) and the employ of different attention mechanisms to focus on the semantically relevant characteristics showing greater improvements those that combine time and frequency attention.

Furthermore, a problem that ESC's researchers discuss is the scarcity of data, to which they used several audio data augmentation techniques to solve the problem and avoid overfitting. On the other hand, some researchers used cross-modality transfer learning by relying on pre-trained models in the image domain. Such fact allowed them to apply the weights feature knowledge to facilitate and accelerate the training process and use data augmentation techniques commonly used in the vision domain.

The objective of this literature review was to summarize the most recent works on the subject to understand the current approaches in this area, the problems or limitations and finally, the state-of-art approach. Out of the articles considered in this review, the work of Mushtaq and Su [22] was the approach that gave the best results for the most popular ESC datasets (97.98% for UrbanSound8k, 98.52% for ESC-50 and 99.22% for ESC-10),

evaluated according to the official splits by doing the k-fold cross-validation evaluation of the results.

Chapter 3

Baseline Models

This chapter describes the process used to obtain the baseline model. Starting with the feature extraction techniques, the definition of the overall architecture of the model, different optimization functions, complemented by an exhaustive evaluation of the performance of the various models, supported by six distinct metrics: Accuracy, area under the receiver operating characteristic (ROC) curve (AUC), precision, recall, micro and macro $F1$-score.

3.1 Datasets

This section describes the characteristics of some of the most used datasets in Environmental Sound Classification (ESC). The chosen datasets UrbanSound8K, ESC-50 and ESC-10 will allow not only to have more models from different published articles with which is possible to compare results, but also to see the behaviour of the models when the number of classes is maintained but the composition and name of the classes changes, and when there is an increase in the number of classes.

3.1.1 UrbanSound8K

The dataset consists of 8732 labelled audio slices with an audio length of 4 seconds or less, with the sampling rate varying from 16 to 44.1 kHz and is organized into 10 folds. All audio clips are taken from field recordings uploaded to the Freesound project. Figure 3.1 presents the data distribution per class (Salamon et al. [28]).

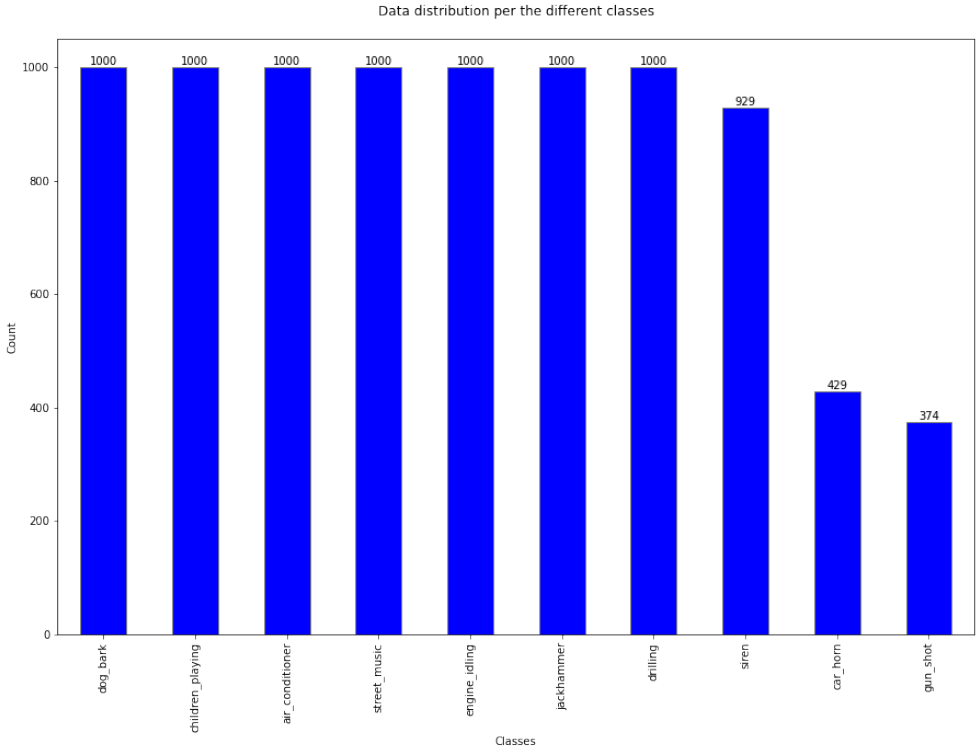


FIGURE 3.1: Distribution of data per class of UrbanSound8K dataset.

3.1.2 ESC-50

The dataset consists of 2000 labelled clips with a length of 5 seconds organized into 50 classes with 40 examples per class extracted from public field recordings available through the Freesound project. The extracted samples were converted to a unified format with a rate of 44.1 kHz, single-channel, and Ogg Vorbis* compressed at 192 kbit/s. The dataset has also been rearranged into 5 uniformly sized folds for comparable cross-validation, ensuring that clips from the same initial source file are always part of a single fold. The distribution of examples per class is shown in Figure 3.2 (Piczak [24]).

3.1.3 ESC-10

This dataset is a subset of 10 classes from the ESC-50 dataset, making it a more constrained set with the differences between classes much more pronounced. The distribution of examples per class is shown in Figure 3.3 (Piczak [24]).

*https://xiph.org/vorbis/ (accessed 29 August 2022)

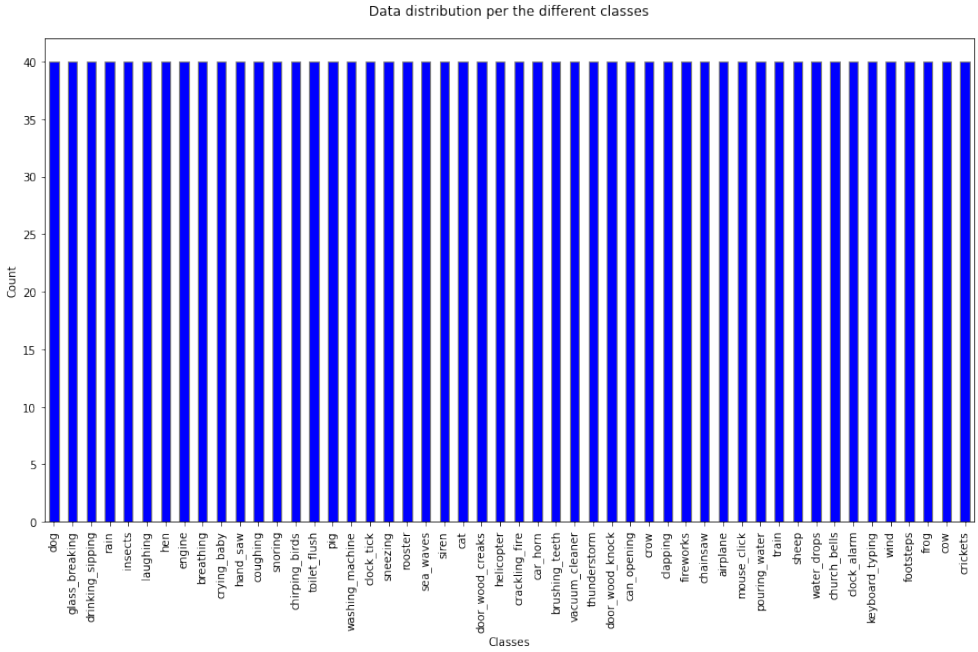


FIGURE 3.2: Distribution of data per class of ESC-50 dataset.

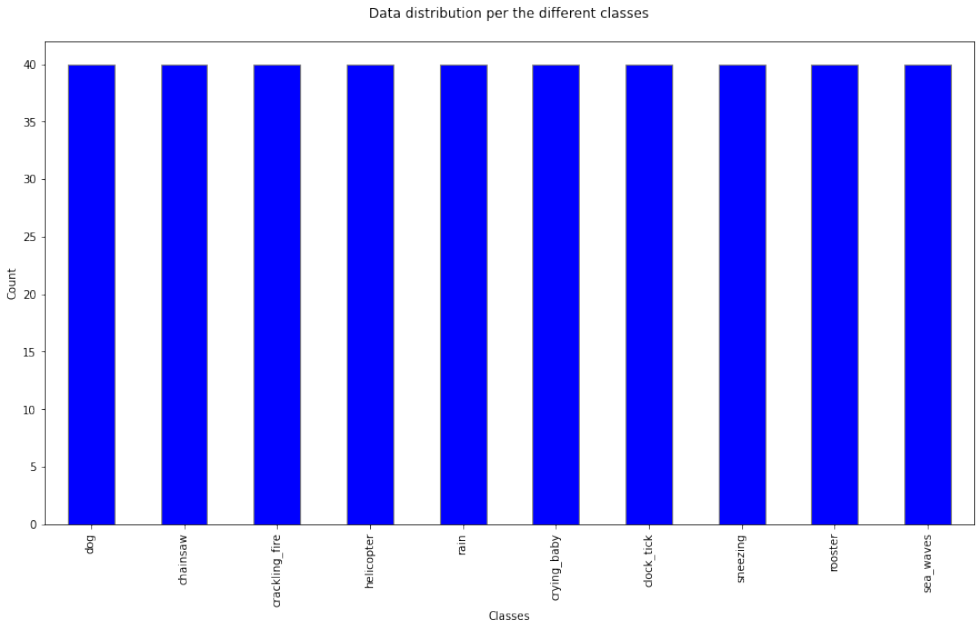


FIGURE 3.3: Distribution of data per class of ESC-10 dataset.

3.2 Feature Extraction Techniques

Different feature extraction techniques were implemented and evaluated to provide the classifier with more distinguishable characteristics and complementary representations

that correctly classify the given audio clips. Next, a description of the implemented spectral audio features is presented, supported by the librosa library (McFee et al. [20]).

- **Zero crossing rate:** is a time-domain feature that considers the sign changing rate of the signal during an audio frame, corresponding to the number of times the signal crosses the zero level.

- **RMS:** measures the energy of a signal by dividing the signal into several windows by applying the following equation to each window:

$$RMS_w = \sqrt{\frac{1}{T}\sum_{t=1}^{T} x_t^2} \tag{3.1}$$

  where T is the window size, and x_t is the amplitude of the t^{th} sample in the window, w.

- **Poly features:** generates a new feature matrix with all polynomial combinations of features raised to a degree less or equal to the specified degree. Therefore, the new matrix consists of bias terms, the features raised to power for each degree until reaching the desired degree and the interactions between all pairs of features.

- **Tonnetz:** computes the tonal centroid features, corresponding to a planar representation of pitch relations that projects chroma features onto a six-dimensional (6D) basis. The 6D tonal centroid vector, ζ, for time frame n is given by the following equation:

$$\zeta_n(d) = \frac{1}{\|c_n\|_1}\sum_{l=0}^{11} \Phi(d,l)c_n(l) \tag{3.2}$$

  where c is the chroma vector, $\|c_n\|_1$ the L_1 norm of c_n, Φ the transformation matrix representing the basis of the 6D space, l the chroma vector pitch class index, and $d \in [0,5]$ denotes which of the six dimensions of ζ_n is being evaluated.

- **Melspectrogram:** is a spectrogram that corresponds to a visual representation of the spectrum of frequencies of a given signal as it varies with time. The signal amplitude is represented by the colour of each point in the image. The frequencies are converted to the mel scale, a scale of pitches (property of sounds that allows their ordering on a frequency-related scale) judged by listeners to be equal in distance from one another.

- **Mel Frequency Cepstral Coefficients (MFCC):** are coefficients that collectively represent a sound's short-term power spectrum. So, the process to develop MFCC is the following:

  1. Apply the discrete Fourier transform to a window of the signal.
  2. Map the magnitudes of the spectrum obtained above into the mel scale.
  3. Take the logarithm of the magnitudes at each mel frequency.
  4. Apply the discrete cosine transform to the logarithmic mel frequency coefficients.

     Since the first few MFCCs coefficients are capable of representing the majority of the signal information, higher-order discrete cosine transform components can be truncated.
  5. Creation of a spectrum over mel frequencies. The amplitudes of the spectrum are the MFCCs.

  Therefore, MFCC can be calculated using the next equation:

  $$c(n) = \sum_{m=0}^{M-1} \log\left(s(m)\right) \cos\left(\frac{\pi n(m-0.5)}{M}\right) \tag{3.3}$$

  where $n = 0, 1, 2, ..., C-1$ and C is the number of MFCCs, $c(n)$ the cepstral coefficients, M the total number of Mel weighting filters, $s(m) = \sum_{k=0}^{N-1}\left[|X(k)|^2 H_m(k)\right]$, with $X(k)$ being the magnitude spectrum, N the number of points used to compute the discrete Fourier transform, and $H_m(k)$ the Mel weighting filter that corresponds to the weight given to the k^{th} energy spectrum bin that contributes to the m^{th} output band.

- **Chroma features:** is the distribution of the signal's energy across a predefined set of pitch classes, allowing to capture harmonic and melodic characteristics of sound while being robust to changes in timbre.

  Its purpose is to represent the harmonic content of a short-time window of audio, which results in the magnitude spectrum from where it is possible to extract a feature vector using Short-Term Fourier Transformation (STFT), Constant Q-transform (CQT), Chroma Energy Normalized Statistics (CENS), among others.

– **Chroma STFT:** is a chroma feature that extracts the feature vector using the STFT, which is a successive evaluation of Fourier transform over short segments of the signal. The values obtained using STFT tell which frequencies are present on the signal and the corresponding time interval they appear. So, STFT is used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time and can be calculated using the following equation:

$$STFT\{x[n]\}(m,w) = \sum_{n=-\infty}^{+\infty} x[n]w[n-m]e^{-jwn} \tag{3.4}$$

where $STFT\{x[n]\}$ represents the STFT of the signal $x[n]$, w the window function and m the time index.

– **Chroma CQT:** is a chroma feature that transforms the audio data to the frequency domain. The transformation is applied to a series of filters logarithmically spaced in frequency. The equation to calculate CQT is the following:

$$X[k] = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} W[k,n]x[n] \exp\left(\frac{-j2\pi Qn}{N[k]}\right) \tag{3.5}$$

where $x[n]$ is the n^{th} sample of the temporal signal, $W[k,n]$ the window function, $N[k]$ the length of the window in samples at the frequency of the k^{th} spectral component, and Q the ratio of frequency to bandwidth.

– **Chroma CENS:** adds a degree of abstraction by considering short-time statistics over energy distributions within the chroma bands. The CENS features correlate to the short-time harmonic content of the audio signal and learn the variations of properties such as dynamics, timbre, and articulations, among others.

  Therefore, to produce CENS features, the following steps need to be followed:

  1. Obtain chroma vectors using chroma CQT;
  2. Normalize the chroma vectors concerning the L_1 norm;
  3. Quantization of amplitude based on log-like amplitude thresholds that introduce a logarithmic compression;
  4. Smoothing using a sliding window;
  5. Downsampling.

- **Spectral features:** extract frequency and power characteristics of a signal, is an excellent tool for analysing repetitive patterns in a given signal, such as vibrations.

  - **Spectral centroid:** indicates where the centre of mass of the spectrum is located and provides a noise-robust estimate of how the dominant frequency of signal changes over time. The higher the centroid, the higher the sound's frequency. The central frequency, f_c, at time frame t is calculated using the following equation:

$$f_c(t) = \frac{\sum_k S(k,t) f(k)}{\sum_k S(k,t)} \tag{3.6}$$

    where $S(k,t)$ is the spectral magnitude at frequency bin k and time frame t, and $f(k)$ the frequency at bin k.

  - **Spectral bandwidth:** is the difference between the upper and lower frequencies in a continuous band of frequencies. To calculate the bandwidth of a signal at a particular time frame is necessary to sum up the maximum deviation of the signal on both sides of the centroid point of the signal, an operation that corresponds to the following equation that computes the p^{th} order spectral bandwidth.

$$\text{Spectral bandwidth} = \left(\sum_k S(k,t) \left(f(k,t) - f_c(t) \right)^p \right)^{1/p} \tag{3.7}$$

    where $S(k,t)$ is the spectral magnitude at frequency bin k and time frame t, $f(k,t)$ the frequency at bin k and time frame t, and f_c is the spectral centroid or central frequency of the signal at time frame t.

  - **Spectral contrast:** divides each frame of a spectrogram into sub-bands and estimates the energy contrast by comparing the mean energy of spectral peaks to spectral valleys in each sub-band separately, with high contrast values corresponding to clear, narrow-band signals and low contrast values to non-harmonic components or broad-band noise.

  - **Spectral flatness:** quantifies how much a sound resembles a pure tone, as opposed to being noise-like. High spectral flatness indicates that the spectrum is similar to white noise and has equivalent power in all spectral bands. On the other hand, low spectral flatness suggests that the power spectrum is concentrated in a relatively small number of bands.

Therefore, spectral flatness can be defined as the ratio of the geometric mean regarding the arithmetic mean of a power spectrum, being expressed as:

$$\text{Spectral flatness} = \frac{\exp\left(\frac{1}{N}\sum_{n=0}^{N-1}\ln\left(x(n)\right)\right)}{\frac{1}{N}\sum_{n=0}^{N-1}x(n)} \tag{3.8}$$

where $x(n)$ represents the magnitude at the frequency bin n and N the length of the samples to consider for the spectral band.

– **Spectral rolloff:** computes the roll-off frequency, defined as the cutoff frequency where a certain defined percentage of the total energy of the spectrum is contained. It is helpful to determine the maximum or minimum frequency of the signal by setting the ratio to a value close to 1 (one) or 0 (zero), respectively.

Besides evaluating the performance with the features input alone, it was also studied the model's behaviour with different combinations of the above features. The combinations performed are presented in Table 3.1.

TABLE 3.1: Summary table of feature combinations.

mfccstft	MFCC, Chroma STFT (nº bins = 40)
mfccstft80	MFCC, Chroma STFT (nº bins = 80)
mfccmel	MFCC, Melspectrogram (nº bins = 80)
mmq	MFCC, Melspectrogram, Chroma CQT (nº bins = 80)
mmcens	MFCC, Melspectrogram, Chroma CENS (nº bins = 80)
mms40	MFCC, Melspectrogram, Chroma STFT (nº bins = 40)
mms60	MFCC, Melspectrogram, Chroma STFT (nº bins = 60)
mms80	MFCC, Melspectrogram, Chroma STFT (nº bins = 80)
mmstftq	MFCC, Melspectrogram, Chroma STFT, Chroma CQT (nº bins = 80)
mmsqc	MFCC, Melspectrogram, Chroma STFT, Chroma CQT, Chroma CENS (nº bins = 40)
mmsqc80	MFCC, Melspectrogram, Chroma STFT, Chroma CQT, Chroma CENS (nº bins = 80)
scontpoly	Spectral Contrast, Poly features (polynomial order = 6)
tsp	Spectral Contrast, Poly features, Tonnetz (nº bands = 5, polynomial order = 5)
zrsp	Zero crossing rate, RMS, Spectral Flatness, Poly features (polynomial order = 0)
zsrssp	Zero crossing rate, Spectral Centroid, Spectral Rolloff, RMS, Spectral Bandwith, Spectral Flatness, Poly features (polynomial order = 0)

3.3 Model's Architecture and Functions

Initially, an architecture consisting of three dense layers with 256 units with ReLu was used as the activation function, which returns 0 (zero) if the input is negative or the value if is positive; one dense layer of 10 units with Softmax function, that is a generalization of

the logistic regression function to multiclass problems by assigning a probability to each class whose probability summation over all classes must be equal to 1 (one), and can be defined by the following equation:

$$\sigma_i(z) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.9}$$

where z is the input vector and K is the number of classes. Furthermore, there is a dropout layer between the dense layers with a rate value of 0.4. The scheme of the model's architecture is shown in Figure 3.4.

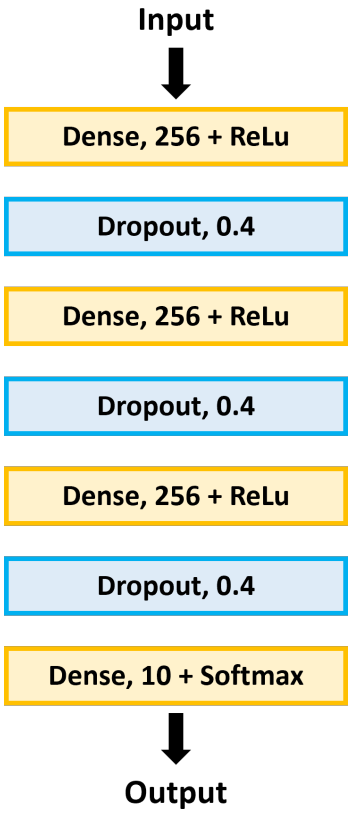


FIGURE 3.4: Baseline model architecture.

The dense layer has the neurons connected to every neuron of the preceding layer, making it deeply connected with its previous layer. Then, the dense layer performs the following operation to deliver the output:

$$output = activation(dot(input, kernel) + bias) \tag{3.10}$$

where activation corresponds to the element-wise activation function, in this case, was used for the first 3 layers, the ReLu function and the last one, the Softmax function; the

kernel corresponds to a weighted matrix created by the layer, with the bias being a bias vector.

The dropout layer prevents overfitting by randomly setting a percentage of the activations to zero with a frequency of rate at each step during training time and scales up the other input values by 1/(1-rate) to the total sum remains unchanged.

To study the influence of the rate on the dropout layer, some experiences were conducted by employing different rate values such as $[0, 0.2, 0.4, 0.6 \text{ and } 0.8]$. Also, model modifications were created to improve the models' performance by adding two extra layers: one dense layer and one dropout layer.

Furthermore, it is important to note that depending on the dataset used, it may be necessary to change the number of units in the last dense layer, as this must be equal to the number of classes in the dataset. Thus, for the ESC-50 dataset, this layer must have 50 units.

After creating the model, it is necessary to configure it for training, requiring the models to be compiled and the loss function, optimization function and metrics defined.

3.3.1 Loss Function

For the loss function, due to the objective of sound classification, it only matters if the output prediction is right or wrong, and the use of datasets has several classes that are as an one-hot encoded, so categorical cross-entropy was the chosen loss function.

Categorical cross-entropy measures the entropy difference between two probability distributions which can be calculated using the following equation:

$$Loss = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{3.11}$$

where N is the output size which is the number of classes, $\hat{y}_i$ the i^{the} scalar value in the model output, which is the Softmax probability for i^{the} class, and y_i is the corresponding truth value.

3.3.2 Optimization Function

Several optimization functions, such as Stochastic Gradient Descent (SGD), Adagrad, Adadelta, Adam, Adamax and Nadam, all minimize the loss function by changing the models' weights.

Next is a description of the characteristics of the different optimization functions employed.

SGD is a stochastic approximation of gradient descent optimization since it is calculated from a randomly selected subset of data (Ruder [27]). The values are calculated following the next algorithm, which should be repeated until the model converges to the desired minimum.

1. Training examples are randomly shuffle
2. For i = 1, ..., M:

$$\theta_i^{(j)} = \theta_{i-1}^{(j)} - \eta \nabla L(\theta_{i-1}^{(j)}) \tag{3.12}$$

for every j = 0, ..., n where n is the total number of weights to be optimized, M the number of training examples, $\theta^{(j)}$ the weight that is being optimized, η the learning rate, and $\nabla L(\theta_{i-1}^{(j)}) = \frac{\partial L}{\partial \theta_{i-1}^{(j)}}$ the partial derivative of the loss function with respect to the weight $\theta_{i-1}^{(j)}$.

Adagrad means adaptive gradient descent and is an optimizer similar to SGD but uses different learning rates for each iteration. This learning rate change is based on how frequently weight updates during training. The more the weights change, the minor changes in the learning rate (Duchi et al. [7]), allowing for the sparse parameter to get more significant updates, improving convergence.

The rule to change the learning rate is given by:

$$\eta_i = \frac{\eta}{\sqrt{\alpha_i} + \epsilon} \tag{3.13}$$

where $\alpha_i = \sum_{t=1}^{i} \left(\frac{\partial L}{\partial \theta_{i-t}^{(j)}} \right)^2$ is the summation of gradient square, being L the loss function, $\theta_{i-t}^{(j)}$ the t-the iteration before the i-the iteration of the $\theta^{(j)}$ weight, ϵ is just a small constant to avoid divisions by zero, η is the initial learning rate manually defined and η_i the updated value of the learning rate i-the iteration, which will be the one used to compute the values of the weights in equation 3.12.

Adadelta is an extension of Adagrad. Instead of summing up all the past squared gradients, it uses a moving window of a fixed size of gradient updates that allows Adadelta to continue learning even after many updates (Zeiler [42]). So, in this case, the learning rate is defined by:

$$\eta_i = \frac{\sqrt{D_{i-1} + \epsilon}}{\sqrt{v_i + \epsilon}} \tag{3.14}$$

where $D_i = \beta \dot{D}_{i-1} + (1-\beta)\left(\theta_i^{(j)} - \theta_{i-1}^{(j)}\right)^2$ and v_i is i-the iteration of the exponential weighted average of squared gradients defined by $v_i = \beta v_{i-1} + (1-\beta)\left(\frac{\partial L}{\partial \theta_{i-1}^{(j)}}\right)^2, \beta \in [0,1[$ is a constant that controls the decay rate, usually set around 0.9 to give more importance to the previous weighted average, allowing to change the v_i slowly with the new values of the squared gradient, L is the loss function, $\theta_{i-1}^{(j)}$ is the iteration previous to the i-the iteration of $\theta^{(j)}$ weight.

Adam is an adaptive moment estimation, a stochastic gradient descent method based on adaptive estimation of the first and second-order moments. This method combines the gradient descent with the momentum algorithm, allowing the gradient descent algorithm to converge much faster by considering the exponentially weighted average of gradients with the root mean square propagation algorithm, similar to Adagrad. However, instead of taking the cumulative sum of squared gradients, it takes the exponential moving average that corresponds in equation 3.13 to replace α_i by v_i (Kingma and Ba [14]).

Therefore, the Adam optimizer algorithm can be defined by:

$$\theta_i^{(j)} = \theta_{i-1}^{(j)} - \hat{m}_i \left(\frac{\eta}{\sqrt{\hat{v}_i} + \epsilon}\right) \tag{3.15}$$

where $\theta^{(j)}$ is the weight that is being optimized, η is the learning rate, ϵ is a small constant to avoid divisions by zero, $\hat{m}_i = m_i/(1-\beta_{1_i})$ is the bias-corrected estimate of momentum, $m_i = \beta_1 m_{i-1} + (1-\beta_1)\left(\frac{\partial L}{\partial \theta_{i-1}^{(j)}}\right)$, $\beta_1 \in [0,1[$ is a constant that controls the rate of decay of m_i, $\hat{v}_i = v_i/(1-\beta_{2_i})$ is the bias-corrected estimate of exponential weighted average of squared gradients, and $\beta_2 \in [0,1[$ is a constant that controls the rate of decay of v_i.

The moving averages require bias correction to prevent the moment estimates from being biased towards zero, a situation that occurs during the initial iterations because the moving averages are initialized as zero and when the decay rates are low (β_1 and β_2 close to 1 (one)).

Adamax is an adaptation of the Adam optimizer that generalizes the approach to the infinite norm of maximum $|(max)|$ (Kingma and Ba [14]) and is defined by:

$$\theta_i^{(j)} = \theta_{i-1}^{(j)} - \left(\frac{\hat{m}_i}{u_i}\frac{\eta}{1-\beta_{1_i}}\right) \tag{3.16}$$

where $\theta^{(j)}$ is the weight that is being optimized, η is the learning rate, $\hat{m}_i$ is the bias-corrected momentum estimate, $u_i = max\left(\beta_2 u_{i-1}, \left|\frac{\partial L}{\partial \theta_i^{(j)}}\right|\right)$ is the exponential weighted infinity norm, and $\beta_2 \in [0, 1[$ is a constant that controls the rate of decay of u_i and L the loss function.

Nadam is the Nesterov-accelerated adaptive moment estimation similar to the Adam optimizer but with a Nesterov moment that applies momentum to the parameters before computing the gradient (Dozat [6]). So, the next equation allows to calculate Nadam optimization:

$$\theta_i^{(j)} = \theta_{i-1}^{(j)} - \bar{m}_i \frac{\eta}{\sqrt{\hat{v}_i} + \epsilon} \tag{3.17}$$

where $\theta^{(j)}$ is the weight that is being optimized, η the learning rate, $\bar{m}_i$ the Nesterov's moment which is defined by $\frac{1-\beta_{1_i}}{1-\prod_{t=1}^{i}\beta_{1_t}}\left(\frac{\partial L}{\partial \theta_{i-1}^{(j)}}\right) + \beta_{1_{i+1}}\hat{m}_i$ and $\hat{m}_i = \frac{m_i}{1-\prod_{t=1}^{i+1}\beta_{1_t}}$, and $\hat{v}_i$ is the bias-corrected estimate of an exponentially weighted average of squared gradients.

3.3.3 Metrics

Metrics are essential to access the model's performance, compare it with other models, and select the best-performing model.

This work used six metrics: Accuracy, AUC, precision, recall, micro and macro $F1$-score.

Accuracy measures how often the algorithm classifies a data point correctly, so it is the number of correctly classified data points out of all data points. Mathematically, it can be defined as:

$$Accuracy = \frac{\text{n}^\text{o} \text{ of correct predictions}}{\text{Total n}^\text{o} \text{ of predictions}} \tag{3.18}$$

AUC measures the ability of a classifier to distinguish between classes. It integrates the ROC curve graph that evaluates the model performance in all classification thresholds.

This metric is preferable because it is scale-invariant. Instead of measuring the absolute values, it measures how well predictions are ranked invariant to classification thresholds, measuring the quality of the model's prediction independent of the chosen classification threshold. However, this is not robust when it is necessary to have well-calibrated probability outputs or when there is a wide disparity in the cost of False Negative (FN) vs False Positive (FP), requiring the need to minimize one type of classification error.

Precision gives the number of correctly classified data points out of all points identified as being of a certain class. In binary, problems can be defined mathematically as:

$$Precision = \frac{TP}{TP + FP} \tag{3.19}$$

True Positive (TP) are the examples that are correctly identified as positive.

Recall gives the number of correctly classified data points out of all points belonging to the class on the dataset. In binary, problems can be defined as:

$$Recall = \frac{TP}{TP + FN} \tag{3.20}$$

$F1$**-score** corresponds to the harmonic mean of precision and recall, which allows assessing model performance based on the values of two metrics:

$$F1\text{-score} = \frac{2(Precision * Recall)}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{3.21}$$

In multiclass problems, it is possible to define different averaging types to calculate the n $F1$-score metric. One is the micro average $F1$-score, which computes the global average score by summing all values across all classes for TP, FN and FP and then plugs it in the $F1$-score equation 3.21. Also, there is a $F1$-score calculated using an average macro scheme corresponding to the computation of the arithmetic mean of all instances per class $F1$-scores.

3.4 Baseline Experiments - Using UrbanSound8K Dataset

Firstly, six different optimizers were evaluated with the architecture presented in Figure 3.4, with each model variation corresponding to a different optimization function. (model 1: Adam; model 2: Adamax; model 3: SGD; model 4: Nadam; model 5: Adagrad and model 6: Adadelta). Also, to quantify which of the features gives better results, a study was carried out using the different spectral features described in Section 3.2. The dataset used to evaluate the baseline model's performance was the UrbanSound8K.

For easy assessment of the results, all tables are arranged in descending order according to the values of the macro $F1$-score, with the best result for each column in the following tables being highlighted. Also, for tables referring to a single input feature, on the name of each feature, a number is added in front to represent the number of chroma bins,

bins in mel scale, or the number of MFCC used respectively, depending on the type of feature and the acronym that is commonly represented. As example, mfcc for MFCC, mel for Melspectrogram, stft for Chroma STFT, cens for Chroma CENS, cqt for Chroma CQT and tonz for Tonnetz. Regarding all figures with graphical representations, the curves represented for the different models correspond to referred feature or combination of features that provided the top result according to the macro $F1$-score for each model presented on the respective table. The legend of each graph shows which model it refers to and the individual input or group of features used.

3.4.1 Models with a Single Feature Input - Baseline Model Architecture

Table 3.2 shows the top 5 best performing features, according to the macro $F1$-score metric, of the six basic models with the architecture presented on Figure 3.4.

TABLE 3.2: Results of the 6 models for different features.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel80	**0.591**	0.873	**0.591**	**0.593**	0.643	0.483	**mfcc60**	**0.522**	**0.886**	**0.522**	**0.551**	**0.774**	0.278
mfcc60	0.572	0.885	0.572	0.578	**0.682**	**0.527**	**mfcc80**	0.516	0.878	0.516	0.528	0.765	**0.314**
mfcc40	0.575	**0.894**	0.575	0.575	0.646	0.527	**mel80**	0.465	0.838	0.465	0.459	0.761	0.228
mfcc80	0.559	0.881	0.559	0.563	0.648	0.520	**mfcc40**	0.441	0.797	0.441	0.440	0.519	0.226
mel60	0.486	0.850	0.486	0.497	0.550	0.381	**mel60**	0.423	0.832	0.423	0.436	0.683	0.170
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	0.611	**0.904**	0.611	**0.622**	**0.722**	0.550	**mfcc80***	**0.637**	**0.908**	**0.637**	**0.628**	**0.747**	**0.572**
mfcc60	**0.616**	0.898	**0.616**	0.618	0.682	**0.564**	**mfcc40**	0.576	0.873	0.576	0.583	0.651	0.535
mfcc40	0.593	0.893	0.593	0.607	0.693	0.519	**mfcc60**	0.573	0.864	0.573	0.571	0.654	0.529
mel80	0.491	0.859	0.491	0.501	0.703	0.294	**mel80**	0.535	0.857	0.535	0.539	0.583	0.419
mel40	0.466	0.862	0.466	0.477	0.688	0.260	**mel60**	0.487	0.836	0.487	0.490	0.569	0.415
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	0.209	0.595	0.209	**0.179**	0.317	**0.118**	**mfcc60**	**0.368**	0.753	**0.368**	**0.361**	**1.000**	0.033
mel60	**0.213**	**0.637**	**0.213**	0.164	**0.556**	0.066	**mel80**	0.323	**0.793**	0.323	0.329	0.538	0.060
tonz20	0.208	0.538	0.208	0.163	0.000	0.000	**mel40**	0.329	0.779	0.329	0.329	0.766	0.059
cens20	0.167	0.597	0.167	0.140	0.000	0.000	**mel60**	0.283	0.786	0.283	0.287	0.803	**0.068**
cens40	0.145	0.554	0.145	0.130	0.000	0.000	**mel20**	0.295	0.762	0.295	0.257	0.723	0.041

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Looking at the results, variations of MFCC and Melspectogram features are, in almost all cases, the top 5 features for all models and MFCC with a number of MFCC equal to 60 and 80 are the features which gives best results, in general. When comparing the results returned by the models according to the different metrics was possible to conclude that the first four models' results are above 0.5 in almost all metrics for most features. The model with Nadam optimizer and MFCC feature input with 80 MFCC, followed by the model with Adamax optimizer and with input one of the top three features for this model

were the cases that allowed to produce better results in almost all metrics when compared to other models configuration.

On Figure 3.5 is represented the curves of AUC and loss function for the different models according to the top feature shown in Table 3.2. By analysis of the loss curves, it is possible to conclude that, except for model 5, all the other models have converged around the 70^{th} epoch. Regarding model 5, looking at the loss function, it is possible to see that the model has not yet converged at the 100^{th} epoch, so it should have been trained for more epochs to see if it can reach convergence. Nevertheless, looking at the tendency of the AUC curve, it does not seem likely that this particular model, after convergence, will give better results than any of the other models.

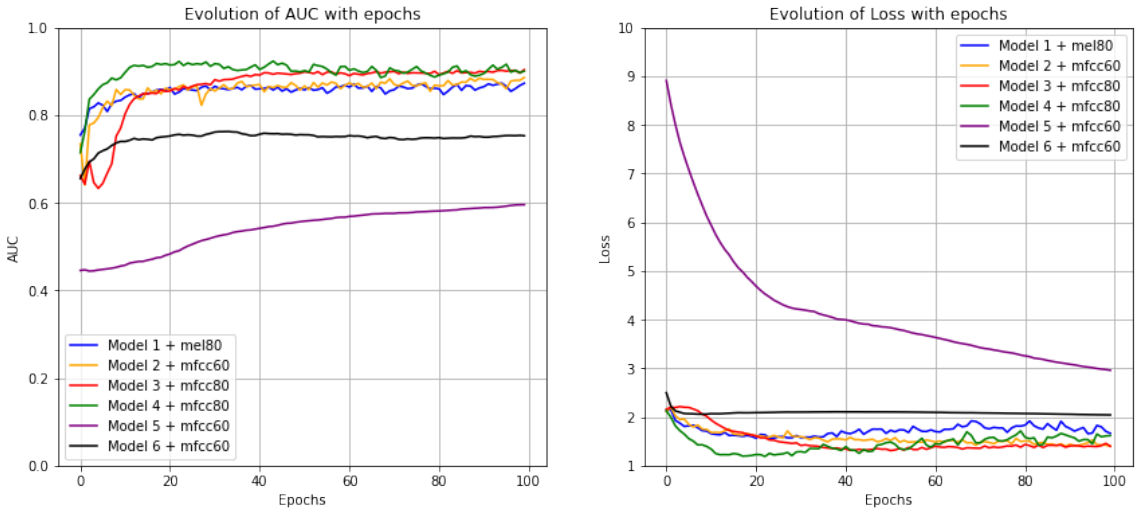


FIGURE 3.5: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the six base models.

Therefore, considering that Adagrad and Adadelta gave inferior results in comparison to the other models, only the top four performing remainder models were considered for the subsequent studies. As result, the following tables show the results for the models' variations based on these four models, with Table 3.3 representing models with an extra layer added, and Table 3.4 and 3.5 the results of the models with a different dropout rate of 0.2 and 0.6, respectively.

3.4.2 Models with a Single Feature Input - Extra Layer

Comparing the results for the models with an extra layer with the base models is possible to conclude that the only model that has benefited from this change was the model with

TABLE 3.3: Results of the 4 best models with an extra layer for different features.

Model 7: (optimizer: Adam)							Model 8: (optimizer: SGD)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	0.573	**0.879**	0.573	**0.584**	0.627	0.524	**mfcc80***	**0.539**	**0.873**	**0.539**	**0.561**	**0.856**	**0.263**
mfcc40	**0.584**	**0.879**	**0.584**	0.579	**0.652**	**0.550**	**mfcc60**	0.514	0.857	0.514	0.539	0.777	0.146
mfcc60	0.522	0.865	0.522	0.539	0.638	0.471	**mfcc40**	0.490	0.838	0.490	0.505	0.725	0.227
mel80	0.526	0.852	0.526	0.534	0.556	0.441	**mel80**	0.430	0.844	0.430	0.440	0.708	0.203
mel20	0.487	0.845	0.487	0.486	0.532	0.373	**mel60**	0.415	0.839	0.415	0.420	0.714	0.194
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.596**	**0.902**	**0.596**	**0.610**	0.656	**0.529**	**mfcc60**	0.577	0.875	0.577	**0.587**	**0.694**	0.532
mfcc60	0.558	0.883	0.558	0.565	0.624	0.511	**mfcc40**	**0.585**	**0.880**	**0.585**	0.574	0.667	**0.534**
mfcc40	0.557	0.892	0.557	0.561	**0.681**	0.458	**mfcc80**	0.556	0.865	0.556	0.566	0.640	0.526
mel40	0.486	0.857	0.486	0.487	0.680	0.297	**mel40**	0.493	0.845	0.493	0.495	0.540	0.384
mel60	0.462	0.857	0.462	0.467	**0.681**	0.294	**mel80**	0.485	0.849	0.485	0.488	0.548	0.419

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

SGD as an optimizer, with the max benefit being 0.082 for precision. For accuracy and micro $F1$-score with a gain of 0.017 and macro $F1$-score of 0.010. In terms of features, MFCC and Melspectrogram are the preferable features, but in this case, in general, the feature that gave better results was MFCC with 80 MFCC.

Figure 3.6 shows the evolution of the AUC and loss function for the selected four base models and the corresponding models with an extra layer. It is possible to observe that the models have converged in all cases. In general, there are no significant differences between the base models or the models with an extra layer for the AUC metric at the 100^{th} epoch.

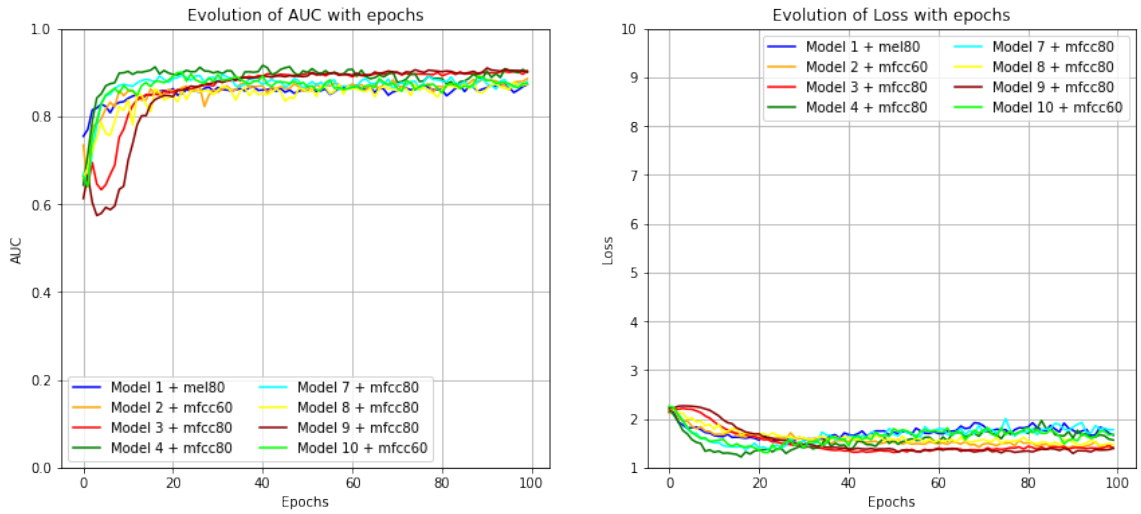


FIGURE 3.6: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four base models and their corresponding ones with an extra layer.

Therefore, since there were no evident benefits for most of the models in changing the depth of the given architecture for the following studies, the base architecture was maintained. Next, it was studied the influence of the dropout rate.

3.4.3 Models with a Single Feature Input - Dropout Rate

The conducted experiments are summarize in Table 3.4 and Table 3.5, showing the results for models with a dropout rate of 0.2 and 0.6, respectively.

TABLE 3.4: Results of the 4 best models for different features and dropout of 0.2.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.620**	**0.862**	**0.620**	**0.616**	**0.639**	**0.606**	**mfcc80**	**0.582**	**0.897**	**0.582**	**0.583**	0.707	0.472
mfcc60	0.582	0.857	0.582	0.587	0.613	0.572	**mfcc40**	0.570	0.876	0.570	0.570	0.708	0.478
mfcc40	0.589	0.855	0.589	0.582	0.618	0.581	**mfcc60**	0.575	0.894	0.575	0.568	**0.729**	**0.492**
mel80	0.503	0.816	0.503	0.505	0.539	0.481	**mel80**	0.498	0.849	0.498	0.505	0.658	0.324
stft60	0.514	0.856	0.514	0.504	0.562	0.425	**mel60**	0.460	0.831	0.460	0.472	0.686	0.258
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.613**	**0.886**	**0.613**	**0.616**	0.638	**0.587**	**mfcc80**	**0.605**	0.849	**0.605**	**0.607**	**0.622**	**0.596**
mfcc40	0.597	0.868	0.597	0.608	0.636	0.579	**mfcc60**	0.587	**0.862**	0.587	0.603	0.616	0.560
mfcc60	0.602	0.875	0.602	0.604	**0.643**	0.583	**mfcc40**	0.539	0.823	0.539	0.524	0.558	0.527
stft40	0.539	0.885	0.539	0.523	0.629	0.429	**stft80**	0.496	0.829	0.496	0.487	0.518	0.386
mel80	0.490	0.849	0.490	0.496	0.609	0.357	**stft40**	0.493	0.846	0.493	0.485	0.563	0.428

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE 3.5: Results of the 4 best models for different features and dropout of 0.6.

Model 15: (optimizer: Adam)							**Model 16: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	0.591	0.894	0.591	**0.611**	0.788	0.448	**mel40**	**0.387**	**0.806**	**0.387**	**0.371**	0.765	**0.074**
mfcc80	**0.608**	**0.902**	**0.608**	0.610	**0.823**	**0.493**	**mel20**	0.386	0.798	0.386	0.350	0.676	0.055
mfcc40	0.601	0.894	0.601	0.609	0.803	0.424	**cqt80**	0.341	0.764	0.341	0.344	0.787	0.044
mel80	0.487	0.862	0.487	0.489	0.643	0.297	**mel60**	0.342	0.788	0.342	0.332	0.775	**0.074**
mel20	0.478	0.847	0.478	0.478	0.629	0.247	**mfcc80**	0.352	0.778	0.352	0.325	**0.934**	0.068
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.569**	**0.887**	**0.569**	**0.596**	0.768	**0.384**	**mfcc60**	**0.599**	**0.898**	**0.599**	**0.612**	**0.848**	0.448
mfcc60	0.530	0.883	0.530	0.551	0.787	0.349	**mfcc80**	0.578	0.882	0.578	0.592	0.810	**0.449**
mfcc40	0.490	0.879	0.490	0.517	**0.830**	0.320	**mfcc40**	0.562	0.887	0.562	0.557	0.827	0.399
stft40	0.460	0.876	0.460	0.456	0.637	0.228	**mel80**	0.495	0.856	0.495	0.503	0.710	0.317
stft60	0.434	0.864	0.434	0.431	0.630	0.220	**mel20**	0.485	0.845	0.485	0.483	0.645	0.269

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

By varying the dropout rate, it can be concluded that having a dropout rate of 0.2 gives better results in most metrics than the dropout rate of 0.6. Compared with the dropout rate of 0.4 selected for the base models, the most beneficial dropout rate depends on the chosen optimization function. For the models with Adam or SGD optimizer, the 0.2 dropout rate was more beneficial. Still, for the models with Adamax or Nadam, the

dropout rate of 0.4 produced better results in most metrics. Comparing the results for the dropout rate of 0.6 and the base models is possible to conclude that, in general, the base models have produced better results except for the model with the Adam optimizer.

In terms of features, MFCC and Melspectrogram continue to be the preferable features for the models with a dropout rate of 0.6 and for the models where the dropout rate was set to 0.2, MFCC with 80 MFCC is the feature that provided better results in most metrics, being the top feature for all the models.

In Figure 3.7, observing the AUC curves, it is evident that the models with SGD as optimizers, when using a lower dropout rate, exhibit better performance. Regarding the loss function, it is possible to observe that all models have converged and that models 11 and 14 have started to overfit.

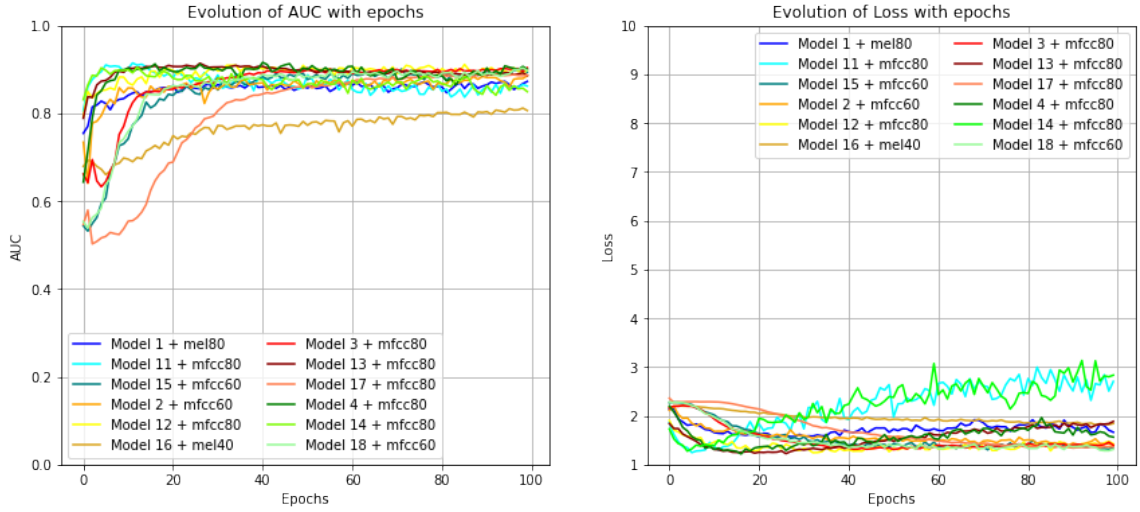


FIGURE 3.7: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four base models and their corresponding ones with a dropout rate of 0.2 and 0.6.

So, these results culminate in the following conclusions; models with Adam, Adamax or Nadam as optimizers are the ones that allowed for achieving better results, in most cases, for all metrics. The dropout rate of 0.2 was the most beneficial for the Adam optimizer models. However, the model has begun to overfit. When comparing the results obtained for the models with the Adamax optimizer for the top 2 input features, the base model gives better results in most metrics than the models with the other selected dropout rates. Finally, for Nadam optimizer models, the dropout rate of 0.4 was the one that allowed the best results out of all models in most metrics, and the change of the dropout rate up or down was inefficient.

To further understand the influence of changing the dropout rate, a study was made for the models with Adam and Adamax as optimizers because models with Adam optimizers have improved their performance with both rate changes. For Adamax optimizer models, the top feature for the dropout rate of 0.2 had similar results to the top feature for the base model; regarding the use of Nadam optimizer in the models, the change in dropout rate was only detrimental. So, the following study uses Adam and Adamax as optimizers and a dropout rate of 0.8 and 0 to evaluate the impact, with results shown in Table 3.6.

TABLE 3.6: Results of the 2 best models for different features and dropout rate of 0.8 and without dropout.

Model 19: (optimizer: Adam; dropout: 0.8)							**Model 20: (optimizer: Adam; dropout: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel20	**0.349**	**0.789**	**0.349**	**0.291**	0.764	0.050	**mfcc60**	**0.566**	**0.820**	**0.566**	**0.565**	**0.577**	**0.564**
mel40	0.308	0.754	0.308	0.275	0.788	**0.062**	**mfcc40**	0.514	0.799	0.514	0.508	0.525	0.508
stft40	0.265	0.750	0.265	0.264	0.683	0.033	**mfcc80**	0.523	0.785	0.523	0.505	0.536	0.519
mfcc80	0.260	0.762	0.260	0.260	0.852	0.055	**mel80**	0.479	0.768	0.479	0.489	0.487	0.464
cens20	0.283	0.717	0.283	0.256	**0.872**	0.041	**stft60**	0.483	0.807	0.483	0.483	0.497	0.453
Model 21: (optimizer: Adamax; dropout: 0.8)							**Model 22: (optimizer: Adamax; dropout: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft20	0.266	**0.807**	0.266	**0.260**	0.521	0.030	**mfcc40**	**0.569**	0.825	**0.569**	**0.570**	**0.580**	**0.564**
cqt80	0.266	0.672	0.266	0.255	0.806	**0.035**	**mfcc80**	0.493	0.790	0.493	0.509	0.503	0.490
stft40	0.253	0.778	0.253	0.255	0.537	0.026	**stft60**	0.487	**0.845**	0.487	0.489	0.538	0.429
cens20	**0.275**	0.699	**0.275**	0.255	**0.917**	0.026	**mel60**	0.491	0.801	0.491	0.488	0.549	0.425
stft80	0.259	0.764	0.259	0.251	0.514	0.023	**mel80**	0.480	0.808	0.480	0.485	0.552	0.437

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Based on the results presented in Table 3.6, it is possible to see that in both cases, the models without dropout gave better results than the models with a dropout rate equal to 0.8, which confirms what was previously observed. Nevertheless, comparing these results with the previous ones, the models that provided better results are still the previously identified baselines. However, the models that provided the worst results have changed to those with a dropout of 0.8.

In Figure 3.8, a graphic plot of the AUC and loss is presented, showing the results of the models with Adam and Adamax as an optimizer and a dropout rate of 0.4, 0.8 and 0. Analysing the graph, both models without a dropout rate suffer from overfitting, and the models with a dropout rate of 0.8 give the worst results.

Therefore, these results show evidence that the value used for the dropout rate greatly influences the final results, and if the dropout rate is too low, the model will suffer from overfitting. Still, if it is too high, the model will not have enough information to distinguish properly between classes due to a large number of activation being set to zero, a fact

that does not allow the network to learn, culminating in low values for the different metrics. These inferences are corroborated by Srivastava et al. [30] which has also explored the effect of varying the dropout rate while maintaining the same architecture using different datasets from several domains, reaching the following conclusions: a high value for the dropout rate leads to underfitting since very few units will turn on during training and the decrease in the dropout rate makes the error go down until the interval where the dropout rate is situated between 0.2 and 0.6 inclusive, in which the error becomes flat, then, after this interval further decreasing the dropout rate will increase the error.

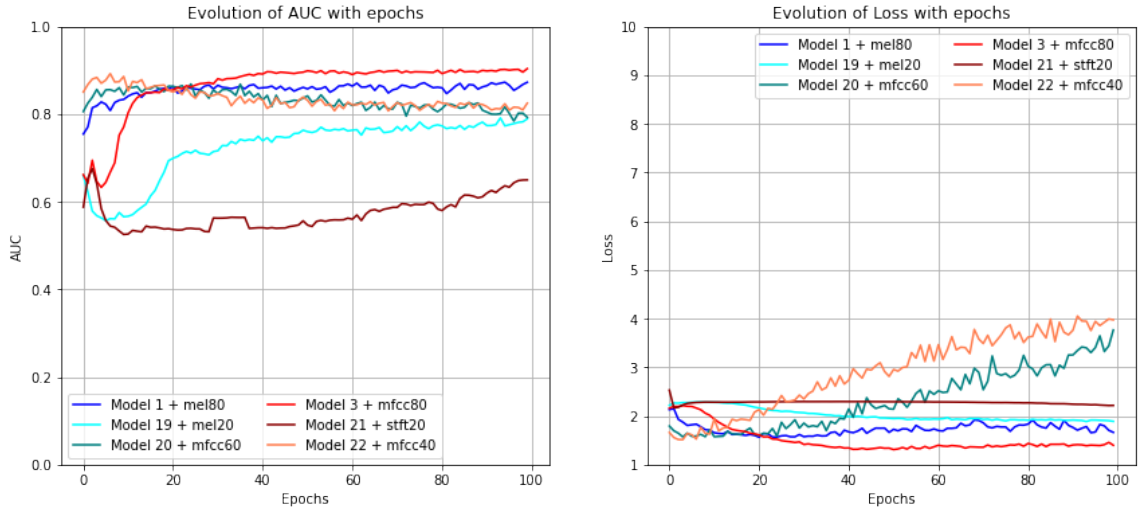


FIGURE 3.8: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the two best base models and their corresponding ones with a dropout rate of 0.8 and 0.

In conclusion, looking at all of the results, the model configuration that gave the best result in most metrics was model 4, which used Nadam optimizer, the base architecture and as input MFCC with 80 MFCC. Also, model 11 has the base architecture with a dropout rate of 0.2, which gave the best results out of all the models with Adam optimizer when evaluated using MFCC with 80 MFCC; however, the model started to suffer from overfitting, indicating that the ideal dropout rate for the model with Adam optimizer should be superior to 0.2 and inferior to 0.4 because the dropout rate of 0.2 has given better results than the dropout rate of 0.4. For the models with Adamax optimizers, having a dropout rate superior to 0.4 shows some degradation in the performance. On the other hand, the dropout rate of 0.2 and 0.4 for the top feature give similar results, but when comparing the top 2 features, the results are better for the dropout rate of 0.4. Finally, for the model with SGD optimizer, the decrease in the dropout rate shows to be

beneficial. Considering that the model did not show overfitting, further decreasing the dropout rate would have given better results before overfitting.

3.4.4 Models with a Combination of Features as Input - Baseline Model Architecture

To determine the best set of features, a study was conducted to determine the best combination of features for this baseline model. All features were combined in many ways as possible to perform this study. Due to the different output shapes of features, some features could not be combined; therefore it was only tried 15 different combinations. Table 3.7 summarizes the performance of the base models.

TABLE 3.7: Results of the 6 models for different feature combinations.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.669**	**0.925**	**0.669**	**0.682**	**0.736**	**0.638**	**mms80**	**0.630**	0.898	**0.630**	**0.662**	0.845	0.312
mmstftq	0.659	0.910	0.659	0.681	0.719	0.634	**mms60**	0.602	**0.909**	0.602	0.624	0.830	**0.367**
mmsqc	0.627	0.912	0.627	0.658	0.696	0.605	**mfccmel**	0.603	0.894	0.603	0.624	**0.855**	0.366
mms40	0.634	0.909	0.634	0.652	0.717	0.600	**mfccstft80**	0.591	0.897	0.591	0.611	0.802	0.305
mmcens	0.627	0.894	0.627	0.652	0.678	0.602	**mmstftq**	0.601	0.898	0.601	0.610	0.820	0.348
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.655**	**0.926**	**0.655**	**0.680**	**0.756**	**0.625**	**mmsqc80**	**0.676**	**0.926**	**0.676**	**0.693**	**0.719**	**0.652**
mmq	0.640	0.912	0.640	0.668	0.697	0.601	**mmstftq**	0.663	0.914	0.663	0.687	0.712	0.639
mmsqc80	0.634	0.900	0.634	0.656	0.677	0.585	**mms80**	0.650	0.901	0.650	0.676	0.687	0.634
mfccstft80	0.628	0.915	0.628	0.654	0.695	0.585	**mmsqc**	0.658	0.919	0.658	0.676	0.709	0.620
mms40	0.637	0.921	0.637	0.651	0.752	0.585	**mmq**	0.645	0.916	0.645	0.669	0.723	0.608
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfccstft80	0.247	0.665	0.247	**0.217**	0.245	0.029	**mmstftq**	0.409	**0.828**	0.409	**0.419**	**0.933**	**0.067**
mfccmel	0.249	0.636	0.249	0.186	**0.295**	0.121	**mmsqc80**	**0.423**	0.827	**0.423**	0.407	0.786	0.053
mms40	**0.251**	0.610	**0.251**	0.180	0.235	0.127	**mmsqc**	0.384	0.780	0.384	0.383	0.830	0.047
mmsqc80	0.208	0.638	0.208	0.175	0.269	0.116	**mms60**	0.338	0.776	0.338	0.348	0.907	0.047
mmq	0.201	**0.688**	0.201	0.169	0.261	**0.145**	**mfccstft**	0.370	0.789	0.370	0.345	0.914	0.038

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Looking at the performance of the six base models is possible to verify that giving as input a combination of features instead of a single feature provides much better results for all models, which would be expected since many of them are very discriminating features. Nevertheless, the models that performed better with individual features are the same with a better performance with a combination of features. In terms of a group of features, the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT and Chroma CENS with 80 bins is the group of features that ranks in the highest positions for the top 5 of most models.

Figure 3.9 shows a graphic representation of the AUC curves and loss function evolution over the defined epochs for the six models, having each one as input the best performing combination of features. These graphics conclude that model 5 is the only one that still has not entirely converged. Still, it seems to be on the verge of convergence, so considering the tendency of both curves is unlikely that this model will outperform any other, so no further training was performed. Observing the performance of the other models is clear that model 5 has an inferior performance compared to the others, evidencing the weak capability of MFCC and Chroma STFT only.

Therefore, the two worst-performing models were not considered for the following studies.

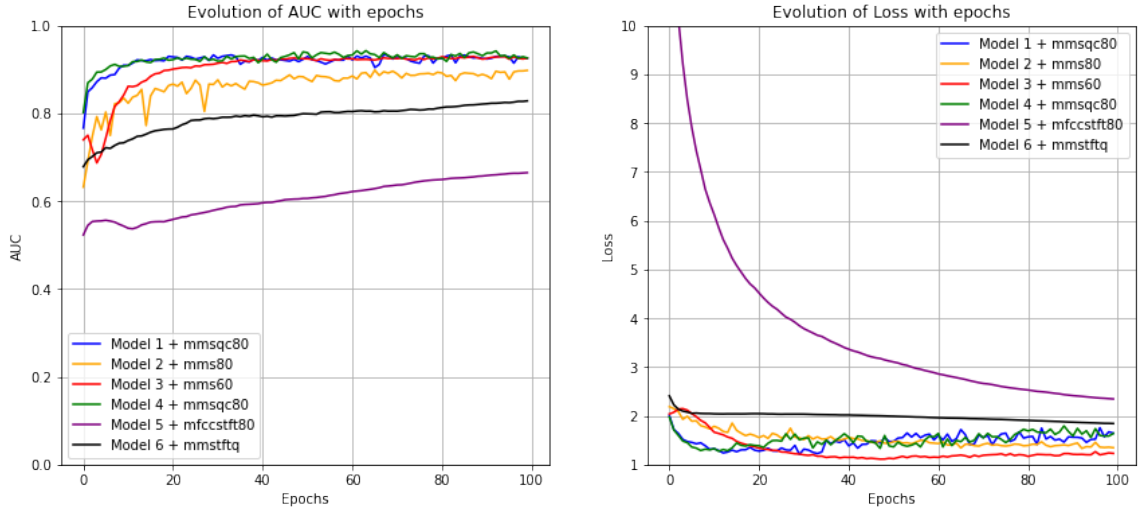


FIGURE 3.9: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the six base models with a group of features as input.

3.4.5 Models with a Combination of Features as Input - Extra Layer

Next, the performance of the models was studied when an extra layer was added, with results presented in Table 3.8.

Comparing the results obtained with the base models and the models with an extra layer can be observed that the top result of the model with Adam optimizer had slightly improved when the extra layer was added in three of the six metrics considered, having a positive influence on accuracy and micro $F1$-score of 0.002, macro $F1$-score of 0.004, recall of 0.010. For precision, the value was the same. The model with Adamax optimizer has also got three metrics with a slight improvement: accuracy and micro $F1$-score of 0.003,

TABLE 3.8: Results of the 4 models with an extra dense and dropout layer for different feature combinations.

Model 7: (optimizer: Adam)							**Model 8: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.671**	0.916	**0.671**	**0.686**	0.736	**0.648**	**mmsqc**	0.582	0.889	0.582	**0.616**	0.804	0.300
mmq	0.658	0.905	0.658	0.685	0.713	0.631	**mmq**	**0.602**	0.893	**0.602**	0.612	0.812	0.274
mms40	0.663	0.912	0.663	0.683	**0.750**	0.644	**mmsqc80**	0.595	0.890	0.595	0.610	0.822	0.299
mfccstft	0.664	**0.918**	0.664	0.683	0.735	0.636	**mfccmel**	0.566	**0.900**	0.566	0.593	**0.832**	**0.302**
mmstftq	0.634	0.914	0.634	0.656	0.700	0.599	**mfccstft80**	0.570	0.876	0.570	0.585	0.829	0.208
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.658**	**0.919**	**0.658**	**0.680**	**0.710**	**0.633**	**mmstftq***	**0.699**	0.911	**0.699**	**0.720**	0.756	**0.664**
mms40	0.628	0.913	0.628	0.655	0.690	0.568	**mmsqc**	0.687	**0.922**	0.687	0.703	**0.777**	0.645
mfccstft	0.626	0.914	0.626	0.650	0.703	0.576	**mmsqc80**	0.676	0.910	0.676	0.689	0.713	0.650
mms60	0.626	0.901	0.626	0.648	0.687	0.591	**mms80**	0.654	0.906	0.654	0.670	0.706	0.599
mmsqc	0.606	0.907	0.606	0.628	0.694	0.578	**mms40**	0.648	0.918	0.648	0.668	0.703	0.625

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

recall of 0.008, and for macro $F1$-score, the result was the same. The model with Nadam optimizer was the model that showed more improvements by having 5 out of 6 metrics improved, with an increase of 0.023 for accuracy and micro $F1$-score, 0.027 for macro $F1$-score, 0.037 for precision and 0.012 for recall. In contrast, the model with SGD optimizer was the only one that didn't improve the results for most metrics with the addition of the extra layer, showing some of the limitations of the SGD for larger sets.

Considering the group of features, the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT, Chroma CENS with a number of bins equal to 80 was the most beneficial combination for the models with Adam or Adamax as an optimizer and is the only combination of features that appears in the top 5 of all models.

In Figure 3.10, as can be concluded, all models have converged, and the model with SGD as the optimizer is the one that shows a slightly worst behaviour.

Since the benefits are relatively small for two models, being significant only for one, the base architecture was maintained in the subsequent studies.

3.4.6 Models with a Combination of Features as Input - Dropout Rate

So, for the next study, only the dropout rate of the model was changed to 0.2 and 0.6 concerning the base model. The results are presented on Table 3.9 for the dropout rate of 0.2 and Table 3.10 for the dropout rate of 0.6.

Analysing the tables of results is possible to conclude that the model with Adam or Nadam as optimizer has shown better performance for a dropout rate of 0.6 and worst for 0.2. The model with the Adamax optimizer obtained better results for 0.2 and worst

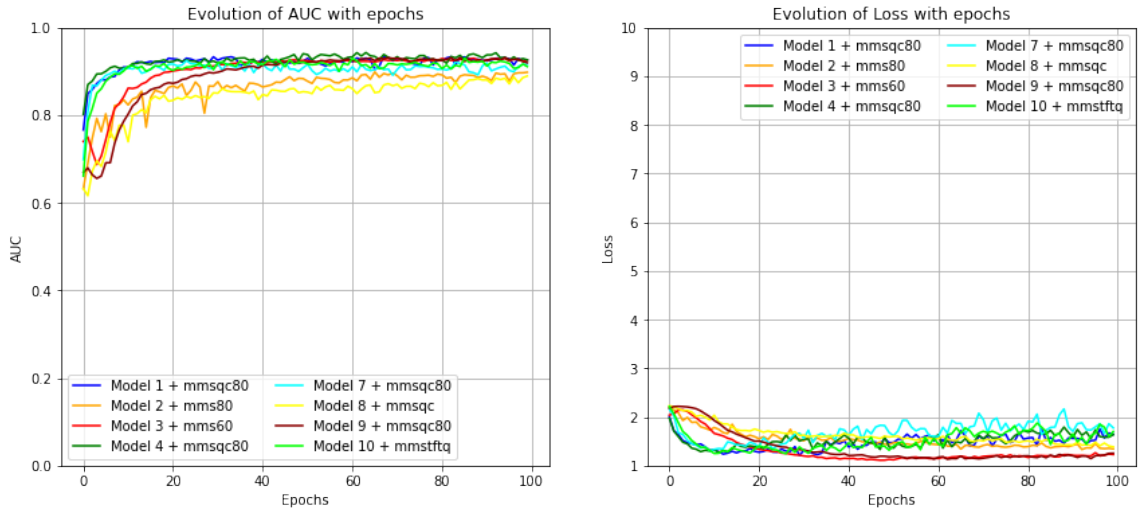


FIGURE 3.10: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four base models and their corresponding ones with an extra layer with a group of features as input.

TABLE 3.9: Results of the 4 models with a dropout rate of 0.2 for different feature combinations.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.668**	**0.895**	**0.668**	**0.693**	**0.691**	**0.658**	**mms80**	**0.627**	**0.904**	**0.627**	**0.641**	**0.731**	**0.550**
mmq	0.646	0.889	0.646	0.670	0.675	0.633	**mfccstft80**	0.594	0.892	0.594	0.593	0.718	0.477
mmstftq	0.642	0.881	0.642	0.667	0.664	0.637	**mmsqc**	0.591	0.890	0.591	0.589	0.706	0.454
mms40	0.637	0.889	0.637	0.655	0.677	0.624	**mfccstft**	0.591	0.869	0.591	0.587	0.697	0.425
mmsqc80	0.622	0.866	0.622	0.641	0.639	0.618	**mms60**	0.588	0.894	0.588	0.586	0.663	0.522
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc	**0.685**	0.907	**0.685**	**0.697**	**0.708**	**0.664**	**mms80**	**0.675**	0.898	**0.675**	**0.695**	**0.697**	**0.668**
mmstftq	0.670	**0.914**	0.670	0.691	0.691	0.657	**mmsqc80**	0.639	0.882	0.639	0.668	0.657	0.624
mmsqc80	0.651	0.894	0.651	0.673	0.670	0.636	**mmsqc**	0.644	0.877	0.644	0.665	0.665	0.636
mmcens	0.632	0.889	0.632	0.655	0.671	0.614	**mms60**	0.650	**0.899**	0.650	0.664	0.665	0.639
mmq	0.645	0.903	0.645	0.647	0.678	0.621	**mmstftq**	0.631	0.882	0.631	0.654	0.652	0.625

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

for 0.6. Finally, the model with SGD optimizer has the worst results for both dropout rates compared with the base models. The improvements for the Adam optimizer model were accuracy and micro $F1$-score of 0.007, macro $F1$-score of 0.014 and precision of 0.045. For Nadam optimizer model, accuracy, micro $F1$-score and macro $F1$-score has improved 0.028, then, 0.003 for AUC and 0.070 for precision, respectively. For the Adamax optimizer using this model, the improvements were 0.030 for accuracy and micro $F1$-score, 0.017 for macro $F1$-score and 0.039 for recall. So, once again, the model that showed more significant improvements in metrics was the model with Nadam optimizer.

Focusing on the most beneficial group of features, for the dropout rate of 0.2 was the

TABLE 3.10: Results of the 4 models with a dropout rate of 0.6 for different feature combinations.

Model 15: (optimizer: Adam)							**Model 16: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc	**0.676**	0.923	**0.676**	**0.696**	0.781	0.551	**mfccmel**	0.271	0.764	0.271	**0.288**	**0.877**	**0.085**
mmsqc80	0.650	**0.927**	0.650	0.684	0.807	0.550	**mfccstft80**	**0.292**	0.779	**0.292**	0.267	0.804	0.049
mms60	0.662	0.916	0.662	0.679	**0.828**	**0.568**	**mms40**	0.257	**0.794**	0.257	0.247	0.865	0.054
mms80	0.651	0.910	0.651	0.674	0.786	0.522	**mfccstft**	0.201	0.756	0.201	0.205	0.769	0.048
mmstftq	0.634	0.905	0.634	0.662	0.757	0.556	**mmstftq**	0.201	0.710	0.201	0.162	0.667	0.017
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.636**	0.911	**0.636**	**0.657**	0.767	**0.489**	**mmsqc***	**0.704**	**0.929**	**0.704**	**0.721**	0.789	**0.594**
mfccstft80	0.621	**0.913**	0.621	0.640	**0.787**	0.478	**mms80**	0.675	0.918	0.675	0.696	**0.827**	0.558
mmq	0.575	0.870	0.575	0.595	**0.787**	0.296	**mms40**	0.654	0.921	0.654	0.680	0.788	0.536
mfccstft	0.557	0.881	0.557	0.585	0.732	0.373	**mmsqc80**	0.640	0.925	0.640	0.677	0.744	0.554
mmcens	0.571	0.879	0.571	0.583	0.746	0.389	**mmstftq**	0.644	0.907	0.644	0.669	0.741	0.541

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT and Chroma CENS with 40 bins because is one of the groups of features that appears in three out of four models and also, is the one that appears in higher positions of the top 5 combinations of features, meaning that are very discriminating features among many parameter variations. For the dropout rate of 0.6, the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT with a number of bins equal to 80 is the one that appears in the top 5 of all models. However, the top combination of features that allowed the models with Adam and Nadam optimizers to achieve the best results was the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT and Chroma CENS with 40 bins which are the two best performing models for this dropout rate.

In Figure 3.11, the graphics plots show that all models have converged, and models 11 and 14, as previously observed when individual features were used as input, started to suffer from overfitting. Analysing the AUC curves, the model with SGD optimizer had the worst performance of all models when the dropout rate was 0.6. For the other models, the performance differences are not evident. Still, it is possible to see that for the models with Adam and Nadam optimizers, the dropout rate of 0.2 provided the worst result. For the Adamax model, the dropout rate of 0.4 was the most beneficial.

3.4.7 Models with a Combination of Features as Input - Extra Layer and Dropout Rate

Using Adam and Adamax optimizers, there were minor improvements when an extra layer was added to the model and when the dropout rate was changed to 0.6 for the

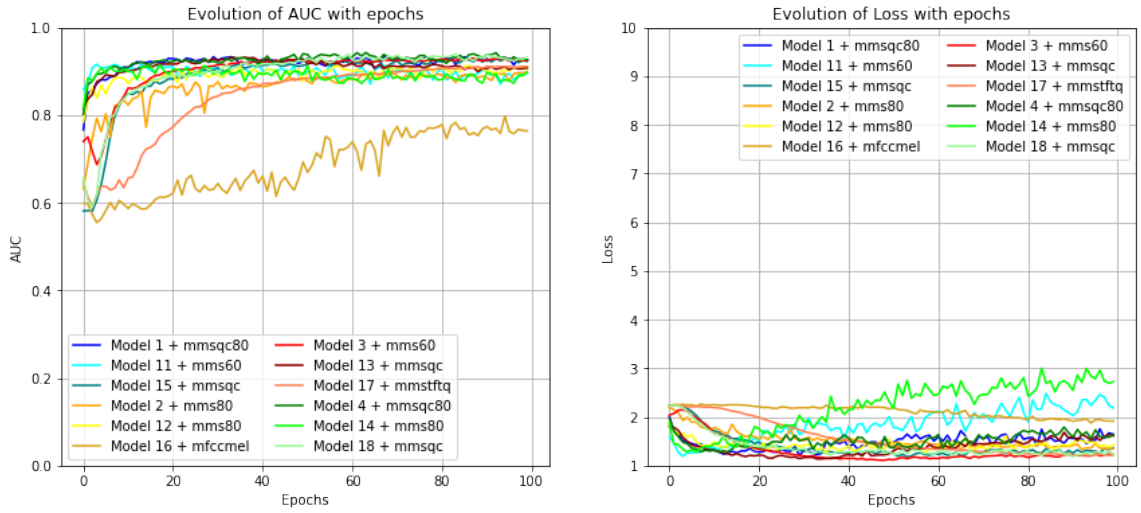


FIGURE 3.11: Graphs of the evolution of AUC (left) and loss function (right) with epochs for the four base models and their corresponding ones with a dropout rate of 0.2 and 0.6 with a group of features as input.

Adam optimizer and 0.2 for the Adamax optimizer. So, the following study considers these improvements to see if combined could produce even better results and evaluate if the dropout rate of 0.8 and an extra layer would work better for the Adamax optimizer model or no dropout rate with an extra layer for the Adam optimizer model. Also, a combination of Nadam optimizer is studied, an extra layer and a dropout rate of 0.6 and the same combination but with a dropout rate of 0.8.

Table 3.11 shows the results for the models with the mentioned modifications.

Analysing the results shown in Table 3.11, it can be concluded that the Adam and Nadam optimizer model with a dropout rate of 0.6 has a much higher performance than the model with 0.8. Then, regarding the models with Adamax as an optimizer, the dropout rate of 0.2 was also more beneficial. Comparing the three best models in this table with the base models is possible to conclude that the results for Adam and Nadam optimizer models were worse than the base model. However, for Adamax optimizer models, in most metrics, the model with an extra layer and a dropout rate of 0.2 has a higher performance when compared to the base model. Also, the three best-performing combinations of features for this model are better than the best-performing combination of features of the base model. The difference between the best performing group of features for the base model and the combination of features with more top values of model 25 for the different metrics are for accuracy and micro $F1$-score of 0.016, for macro $F1$-score of 0.004 and 0.032 for recall.

TABLE 3.11: Results of the 3 models with an extra layer and dropout rate of 0.6 and 0.8 for Adam and Nadam optimizer and 0.2 and 0 for the Adamax optimizer with different feature combinations.

Model 23: (optimizer: Adam and dropout rate: 0.6)							**Model 24: (optimizer: Adam and dropout rate: 0.8)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.626**	0.898	**0.626**	**0.653**	0.775	0.466	**mms40**	**0.214**	0.639	**0.214**	**0.188**	0.875	**0.042**
mmsqc	0.613	0.902	0.613	0.651	0.763	0.430	**mfccstft**	0.189	**0.680**	0.189	0.165	0.854	**0.042**
mmstftq	0.613	0.895	0.613	0.643	0.799	**0.474**	**mms60**	0.166	0.571	0.166	0.143	0.957	0.026
mfccstft80	0.605	**0.907**	0.605	0.615	**0.802**	0.441	**mms80**	0.168	0.611	0.168	0.130	0.952	0.024
mms40	0.588	0.881	0.588	0.614	0.743	0.421	**mmsqc80**	0.154	0.574	0.154	0.123	**1.000**	0.023
Model 25: (optimizer: Adamax and dropout rate: 0.2)							**Model 26: (optimizer: Adamax and dropout rate: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80*	0.662	**0.906**	0.662	**0.690**	0.689	0.650	**mmstftq**	**0.618**	0.850	**0.618**	**0.636**	**0.624**	**0.608**
mms80	0.664	0.894	0.664	0.685	0.692	0.652	**mms80**	0.609	**0.855**	0.609	0.626	0.619	0.602
mms40	**0.671**	0.900	**0.671**	0.684	**0.718**	**0.657**	**mmsqc80**	0.575	0.823	0.575	0.601	0.583	0.572
mmstftq	0.649	0.893	0.649	0.670	0.668	0.638	**mmcens**	0.599	0.849	0.599	0.599	0.612	0.593
mfccmel	0.650	0.890	0.650	0.654	0.687	0.638	**mms40**	0.582	0.825	0.582	0.595	0.594	0.577
Model 27: (optimizer: Nadam and dropout rate: 0.6)							**Model 28: (optimizer: Nadam and dropout rate: 0.8)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.639**	0.897	**0.639**	**0.668**	**0.826**	**0.465**	**mmsqc**	0.182	0.638	0.182	**0.168**	0.879	0.035
mmsqc	0.615	0.898	0.615	0.638	0.807	0.440	**mms80**	**0.204**	**0.651**	**0.204**	0.167	**1.000**	0.026
mmsqc80	0.615	0.893	0.615	0.637	0.780	0.436	**mms40**	0.179	0.632	0.179	0.154	0.857	**0.036**
mms60	0.611	**0.907**	0.611	0.635	0.771	0.458	**mms60**	0.111	0.596	0.111	0.142	**1.000**	0.026
mms40	0.605	0.901	0.605	0.627	0.765	0.464	**mmcens**	0.160	0.615	0.160	0.125	**1.000**	0.026

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Figure 3.12 shows the various AUC and loss curves for Adam, Adamax and Nadam optimizer models with the dropout rates of 0.4, 0 and 0.8.

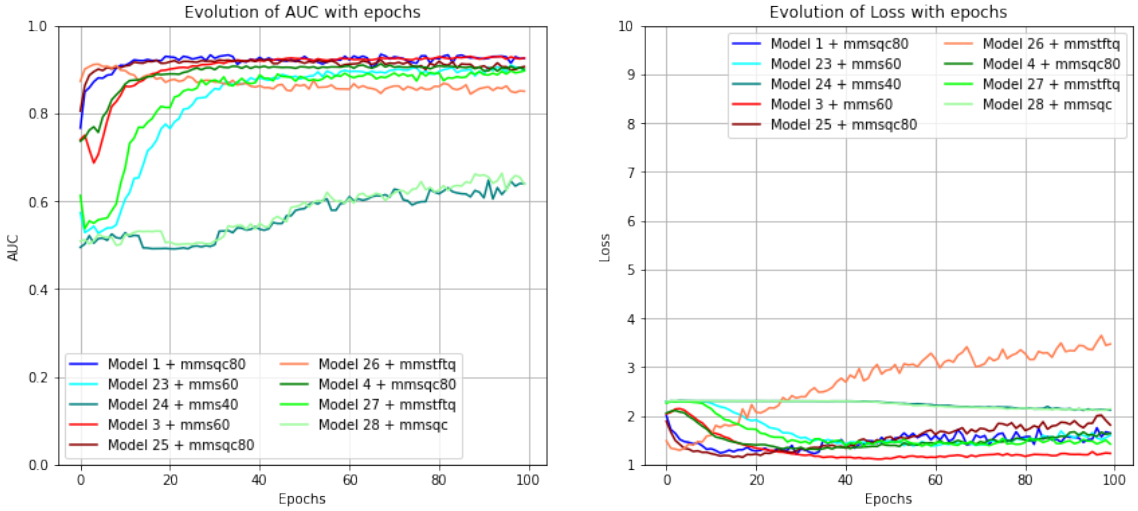


FIGURE 3.12: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the two base models and their corresponding ones with an extra layer and dropout rate of 0.2 and 0 (zero) for Adamax and 0.6 and 0.8 for Adam with a group of features as input.

So, according to Figure 3.12 all models have converged, model 26 has suffered from overfitting and model 24 and 28 had inferior performance. Analysing the AUC curves per

set of models with the same optimizer is possible to conclude that for Adam and Adamax at the end of the 100^{th} epoch, the best performance is given by the base models and the worst by the models with a drop rate of 0.8 or without any dropout rate, respectively. For Nadam optimizer models, the worst performance was clearly of the model with a dropout rate of 0.8, while several models performed equally well based on the AUC curve analysis.

According to the obtained results, it can be concluded that Adam, Adamax and Nadam are the optimizers that proportionate better results, confirming the previous analysis. The results showed that for Adam and Nadam, the best model was when the base architecture was maintained and when the dropout rate was changed to 0.6. On the other hand, for Adamax, when the dropout rate was changed to 0.2 or added an extra layer with a dropout rate changed to 0.2, the models improved 4 out of 6 metrics. However, the one that provided the more significant improvement on most metrics was the model with the base architecture and a dropout rate of 0.2. In terms of the group of features, the one that allowed to produce the best results for these Adam, Nadam and Adamax optimizers was the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT, Chroma CENS with 40 bins. To summarise, out of the three best models, the one that provided the best results was the model with the Nadam optimizer.

3.4.8 Best Models Analysis and Discussion Using UrbanSound8K

In order to be able to compare the results in an unbiased way according to Salamon et al. [28], it is necessary to do 10 folds cross-validation and average out the values to obtain the final results. Table 3.12 shows the results for each folder as the test folder and the average metric values for the best model with a single feature input and the best model with a combination of features as input.

Figure 3.13 presents the confusion matrices for the two performing models previously mentioned to see how well they perform per class to understand which classes generate a higher error as well as the ones that are better identified by the models. Analysing the confusion matrices, it can be seen that even though model 4 with "mfcc80" has an inferior performance compared to model 18 with "mmsqc" for all metrics, model 4 was capable of outperforming model 18 for four classes which were car horns with an improvement of 4 percentage points (pp), 5 pp for a dog bark, 2 pp for drilling and 4 pp for street music. However, model 18, in general, can better predict the class, especially for children playing and gunshot classes for which the percentage error is equal to or less than 23%. Thus,

TABLE 3.12: Results for the 10 folds of the top performing model with a single feature and with a group of features as input.

Model 4: (optimizer: Nadam; dr: 0.4; feature: mfcc80)							**Model 18: (optimizer: Nadam; dr: 0.6; feature: mmsqc)**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1	0.512	0.829	0.512	0.535	0.628	0.301	1	0.584	0.867	0.584	0.608	0.713	0.475
2	0.515	0.858	0.515	0.514	0.619	0.447	2	0.557	0.872	0.557	0.570	0.675	0.417
3	0.524	0.871	0.524	0.523	0.621	0.467	3	0.525	0.837	0.525	0.561	0.630	0.392
4	0.543	0.890	0.543	0.517	0.725	0.337	4	0.606	0.900	0.606	0.603	0.731	0.492
5	0.627	0.919	0.627	0.617	0.778	0.568	5	0.632	0.912	0.632	0.652	0.745	0.534
6	0.542	0.878	0.542	0.551	0.656	0.439	6	0.548	0.869	0.548	0.580	0.662	0.447
7	0.585	0.895	0.585	0.593	0.768	0.372	7	0.641	0.910	0.641	0.652	0.833	0.459
8	0.609	0.896	0.609	0.635	**0.851**	0.333	8	0.660	0.907	0.660	0.694	**0.795**	0.553
9	**0.680**	**0.923**	**0.680**	**0.688**	0.756	**0.651**	9	0.647	0.891	0.647	0.675	0.738	0.559
10	0.637	0.908	0.637	0.628	0.747	0.572	10	**0.704**	**0.929**	**0.704**	**0.721**	0.789	**0.594**
Average	0.577	0.887	0.577	0.580	0.715	0.449	**Average**	0.611	0.889	0.611	0.632	0.731	0.492

dr - dropout rate; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

both models show more difficulties identifying sounds belonging to the air conditioner, drilling, engine idling and jackhammer classes. On the other hand, car horns, children playing and gunshots were the classes with the highest accuracy for both models.

Mu et al. [21] found that the benefits of the attention mechanisms depend on the sound characteristics, being more profitable for transient sounds, temporal attention mechanisms, and for continuous sounds, frequency attention mechanisms. With these findings in mind, the obtained results can indicate a higher facility for the models to identify sounds with pronounced temporal characteristics. Therefore, the models are more capable of identifying transient rather than continuous sounds.

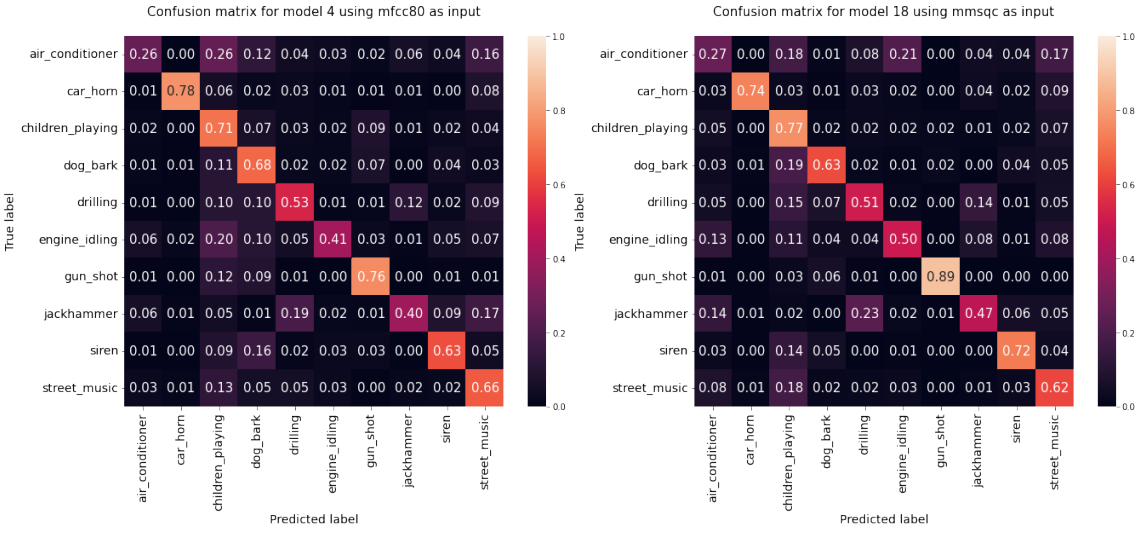


FIGURE 3.13: Confusion matrices for the best model using as input a single feature on the left and the best model using a group of features as input, on the right.

3.5 Baseline Models - Using ESC Dataset

After analysing the results of all the previous models, a similar study was performed for the datasets ESC-10 and ESC-50 to see the models' performance changes for balanced datasets with the same number of classes as UrbanSound8K and with a higher number of classes.

3.5.1 Models with a Single Feature Input - Baseline Model Architecture

Next, Table 3.13 and 3.14 the results for the base models with a single feature input for ESC-10 and ESC-50, respectively, are presented according to the same methodology used in models for UrbanSound8K.

TABLE 3.13: Results of the 6 models for different features - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.613**	0.880	**0.613**	**0.601**	0.642	0.425	**mel60**	**0.463**	0.852	**0.463**	**0.442**	0.526	0.125
mfcc80	0.588	0.879	0.588	0.574	**0.667**	0.425	**mel80**	0.450	0.858	0.450	0.433	**0.714**	**0.188**
mel40	0.563	0.891	0.563	0.567	0.603	0.438	**mel40**	0.450	0.875	0.450	0.427	0.500	0.113
mel20	0.575	**0.897**	0.575	0.558	0.617	**0.463**	**mel20**	0.438	**0.886**	0.438	0.413	0.455	0.063
mel60	0.538	0.843	0.538	0.528	0.548	0.425	**mfcc80**	0.425	0.852	0.425	0.364	0.455	0.125
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel40	**0.488**	0.878	**0.488**	**0.465**	**0.765**	**0.163**	**mfcc40**	**0.650**	0.894	**0.650**	**0.644**	**0.723**	0.425
mfcc60	0.475	**0.890**	0.475	0.410	0.462	0.075	**mel40**	0.588	**0.900**	0.588	0.597	0.603	**0.438**
stft80	0.425	0.859	0.425	0.409	0.667	0.025	**mfcc60**	0.600	0.877	0.600	0.594	0.630	0.363
mel80	0.425	0.831	0.425	0.405	0.579	0.138	**mel20**	0.600	0.899	0.600	0.587	0.545	0.375
stft40	0.400	0.859	0.400	0.385	0.500	0.013	**mfcc80**	0.575	0.874	0.575	0.558	0.630	0.363
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
cens40	0.175	0.532	0.175	**0.116**	0.000	0.000	**mel60**	0.400	0.838	0.400	**0.383**	0.647	0.138
cens80	**0.225**	0.498	**0.225**	0.115	0.000	0.000	**mel40**	0.413	0.828	**0.413**	0.367	0.667	0.125
mfcc40	0.125	0.482	0.125	0.093	0.094	0.075	**mel80**	0.400	0.813	0.400	0.365	0.565	0.163
mel80	0.163	**0.543**	0.163	0.082	**0.417**	**0.125**	**mfcc60**	**0.413**	0.830	**0.413**	0.361	0.571	**0.200**
mel40	0.088	0.479	0.088	0.078	0.000	0.000	**mel20**	0.375	**0.843**	0.375	0.320	**0.778**	0.088

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Similar to what was previously observed for UrbanSound8K, for the ESC datasets, the best model was still the one that used Nadam as an optimizer, and the two worst were the ones that used Adadelta and Adagrad optimizers. So, these two worst optimizers were not considered for the following studies. In terms of features, Melspectrogram and MFCC are the preferable features, however, for ESC-10 the feature which produced the best results was MFCC with 40 MFCC and for ESC-50, MFCC with 80 MFCC.

In Figure 3.14 and 3.15 is shown the AUC and loss curves of the base models for the ESC-10 and ESC-50 datasets, respectively.

TABLE 3.14: Results of the 6 models for different features - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.313**	0.854	**0.313**	**0.293**	0.504	0.170	**mel80**	**0.173**	0.780	**0.173**	**0.139**	0.394	**0.033**
mfcc60	0.305	0.856	0.305	0.290	0.503	**0.188**	**mel60**	0.158	0.781	0.158	0.131	**0.480**	0.030
mfcc40	0.288	0.850	0.288	0.271	0.456	0.130	**mel40**	0.153	0.796	0.153	0.126	0.429	0.030
mel20	0.263	**0.858**	0.263	0.249	0.519	0.105	**mfcc80**	0.143	0.782	0.143	0.123	0.375	0.015
mel40	0.240	0.855	0.240	0.228	**0.540**	0.118	**mel20**	0.148	**0.800**	0.148	0.118	0.450	0.023
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.188**	**0.845**	**0.188**	**0.146**	0.214	0.008	**mfcc80**	**0.313**	0.850	**0.313**	**0.297**	0.511	0.168
stft40	0.148	0.794	0.148	0.119	0.533	**0.020**	**mfcc60**	0.310	0.848	0.310	0.290	0.515	**0.173**
mel40	0.128	0.773	0.128	0.118	0.421	**0.020**	**mfcc40**	0.300	**0.866**	0.300	0.275	0.496	0.145
stft60	0.153	0.797	0.153	0.118	0.538	0.018	**mel20**	0.258	0.854	0.258	0.242	**0.556**	0.113
stft20	0.153	0.795	0.153	0.117	**0.636**	0.018	**mel40**	0.238	0.850	0.238	0.218	0.543	0.110
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
cens80	**0.045**	0.507	**0.045**	**0.020**	0.000	0.000	**mel40**	**0.113**	0.712	**0.113**	**0.078**	0.333	0.013
mel80	0.023	**0.535**	0.023	0.014	0.033	0.005	**mel20**	0.108	**0.713**	0.108	0.074	**0.545**	0.015
mel60	0.020	0.533	0.020	0.013	0.019	0.003	**mel60**	0.103	0.690	0.103	0.064	0.450	**0.023**
mel40	0.018	0.508	0.018	0.008	0.026	0.003	**mel80**	0.095	0.671	0.095	0.063	0.308	0.020
mfcc40	0.025	0.517	0.025	0.007	**0.036**	**0.013**	**cens80**	0.043	0.583	0.043	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

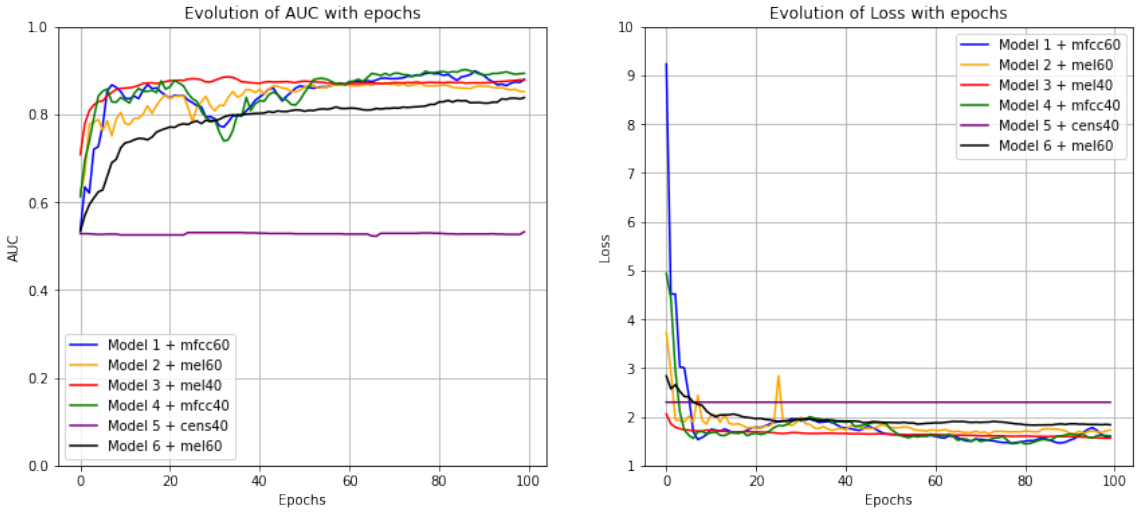


FIGURE 3.14: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the six base models for ESC-10.

Observing the curves is possible to conclude that with the ESC-10 dataset, all models have converged; however, for the ESC-50 dataset, model 3, which used Adamax optimizer, was the only one that has not converged which is an unexpected situation according to the previous results, but can be justified by the more challenging dataset that may require a few more epochs with this configuration.

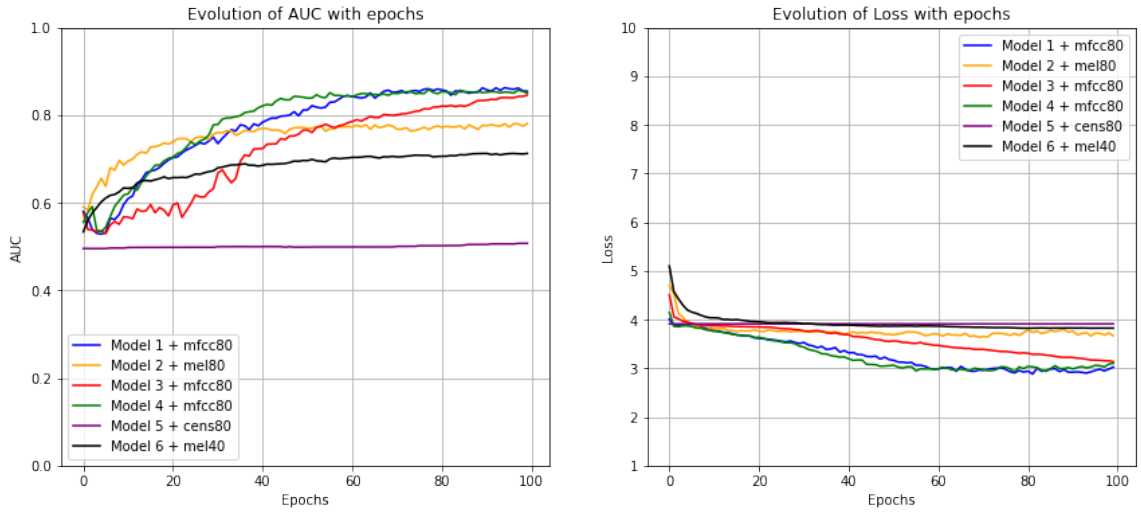


FIGURE 3.15: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the six base models for ESC-50.

Focusing on the AUC curves, model 5 is the worst one, followed by model 6. Nonetheless, models 1 and 4 show identical behaviours corresponding to the best performances. However, model 3 presents a final result slightly worse than the best ones. Considering that this model at the end of the 100^{th} epoch had not fully converged, perhaps with further training, a better convergence and corresponding results could be obtained. So, to confirm that, model 3 was trained until convergence, culminating in the results presented in Table 3.15 and Figure 3.16.

TABLE 3.15: Results of the model with Adamax optimizer for different features - ESC-50.

Model 3: (optimizer: Adamax)						
Features	acc	AUC	micro f1score	macro f1score	precision	recall
mfcc60	0.318	**0.871**	0.318	**0.312**	0.465	0.133
mfcc80	**0.323**	0.869	**0.323**	0.304	0.477	**0.158**
mfcc40	0.290	0.866	0.290	0.275	**0.568**	0.125
mel20	0.233	0.858	0.233	0.209	0.525	0.053
mel80	0.185	0.826	0.185	0.172	0.488	0.053

acc - accuracy; AUC - area under the receiver operating characteristic curve.
All metrics range from $[0, 1]$ (the higher, the better).

Therefore, Figure 3.16 shows that the model has only converged around the 170^{th} epoch, and the difference in the results after convergence is quite considerable, as can be seen graphically by the differences in the loss and AUC value. Table 3.15 further confirms the improvements by comparing either with the same model or the others. In particular, the top 2 features have produced better results than any of the previous models in most metrics.

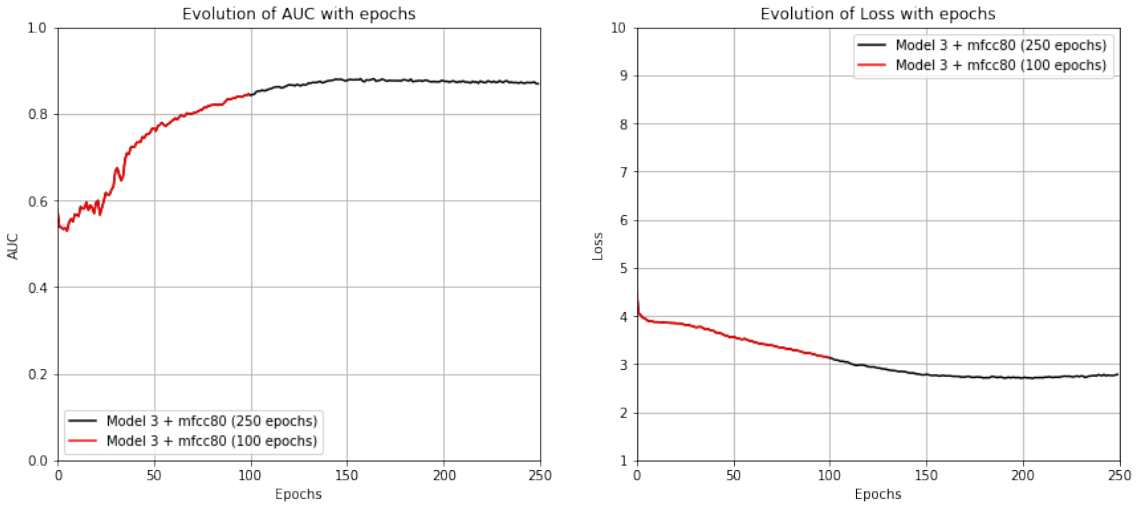


FIGURE 3.16: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the Adamax optimizer model for ESC-50.

For the subsequent studies, only the four best optimizers were used to explore the potential benefits of changing the architecture by adding a new layer and changing the dropout rate could produce. After looking at the results, it was clear that adding an extra layer and the dropout rate of 0.6 did not improve the model's performance, so these results are only shown in Appendix B just for reference.

3.5.2 Models with a Single Feature Input - Dropout Rate of 0.2

Tables 3.16 and 3.17 displays the results for the models when the dropout rate was changed to 0.2.

TABLE 3.16: Results of the 4 best models for single features and dropout of 0.2 - ESC-10.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.650**	0.878	**0.650**	**0.647**	**0.676**	**0.625**	**mel40**	**0.525**	0.881	**0.525**	**0.517**	0.600	0.225
mfcc40	0.625	**0.893**	0.625	0.618	0.644	0.588	**mfcc80**	0.513	**0.891**	0.513	0.497	**0.647**	0.413
mfcc80	0.600	0.873	0.600	0.586	0.613	0.575	**mel80**	0.500	0.860	0.500	0.482	0.600	0.263
mel80	0.525	0.862	0.525	0.533	0.547	0.513	**mfcc60**	0.500	0.888	0.500	0.476	0.583	**0.438**
mel60	0.538	0.851	0.538	0.531	0.557	0.488	**mel60**	0.488	0.865	0.488	0.469	0.583	0.263
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc40	**0.588**	0.909	**0.588**	**0.575**	**0.659**	0.363	**mfcc60**	**0.625**	0.874	**0.625**	**0.621**	**0.657**	**0.575**
mfcc60	0.575	**0.910**	0.575	0.570	0.640	**0.400**	**mfcc40**	0.600	**0.894**	0.600	0.603	0.608	0.563
mfcc80	0.550	0.870	0.550	0.535	0.595	0.313	**mfcc80**	0.613	0.878	0.613	0.594	0.639	**0.575**
mel40	0.525	0.886	0.525	0.533	0.617	0.363	**mel80**	0.550	0.873	0.550	0.552	0.574	0.488
mel20	0.513	0.865	0.513	0.515	0.526	0.250	**mel40**	0.550	0.878	0.550	0.551	0.581	0.538

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE 3.17: Results of the 4 best models for single features and dropout of 0.2 - ESC-50.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.353**	0.811	**0.353**	**0.344**	**0.419**	**0.303**	**mfcc80**	**0.275**	**0.860**	**0.275**	**0.258**	**0.505**	**0.125**
mfcc40	0.335	0.806	0.335	0.328	0.394	0.283	**mfcc60**	0.255	0.859	0.255	0.236	0.448	0.108
mfcc80	0.303	0.790	0.303	0.293	0.359	0.253	**mfcc40**	0.245	0.856	0.245	0.219	0.495	0.113
mel20	0.255	**0.824**	0.255	0.237	0.379	0.138	**mel20**	0.178	0.808	0.178	0.163	0.400	0.040
mel40	0.240	0.817	0.240	0.229	0.406	0.178	**mel40**	0.173	0.799	0.173	0.159	0.387	0.060
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.328**	**0.874**	**0.328**	**0.314**	**0.521**	**0.190**	**mfcc80**	**0.350**	0.802	**0.350**	**0.330**	0.413	**0.315**
mfcc60	0.295	0.845	0.295	0.283	0.484	0.153	**mfcc60**	0.335	0.788	0.335	0.316	0.401	0.288
mfcc40	0.283	0.854	0.283	0.263	0.479	0.145	**mfcc40**	0.305	0.814	0.305	0.295	0.366	0.253
mel20	0.198	0.844	0.198	0.175	0.455	0.050	**mel40**	0.270	**0.825**	0.270	0.259	**0.424**	0.160
mel40	0.185	0.821	0.185	0.169	0.389	0.053	**mel20**	0.250	0.823	0.250	0.234	0.369	0.138

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Analysing the results, all models have improved their performance except for the model with Nadam optimizer for the ESC-10 dataset.

For the ESC-10 dataset, model 11, which has Adam as the optimizer, got results analogous to what was obtained by the Nadam optimizer in the initial model. Then, regarding the results for the ESC-50 dataset, the models with Adam and Nadam optimizers showed identical results for the base models. However, when the dropout rate was decreased to 0.2, the model with Adam optimizer showed an improved capacity to produce the best results by a small margin compared to the results produced by the model with Nadam optimizer. Regarding the features, it is evident that MFCC and Melspectrogram are the preferable features. Nonetheless, MFCC with 60 MFCC stands out because it gives the best results for both datasets when the model uses Adam as an optimizer, corresponding to the presented models with the best results with a dropout rate of 0.2.

In Figure 3.17 and 3.18 is represented the curves for the base models and the correspondent ones with a dropout rate of 0.2.

These figures show that models 11 and 14 have overfitted, being more pronounced for the ESC-50 dataset and unlike in the base models that the model with Adamax optimizer for the ESC-50 dataset has not converged, when the dropout rate is changed to 0.2, the model converges. This model offered the best result when considering the AUC curve metric.

Since models 11 and 14 have been overfitted, further reduction of the dropout rate will not improve the results. Regarding the models with Adamax optimizer, when the dropout rate was changed to 0.2, it showed considerable improvements, allowing convergence in both datasets without overfitting. Due to previous experiments with UrbanSound8K,

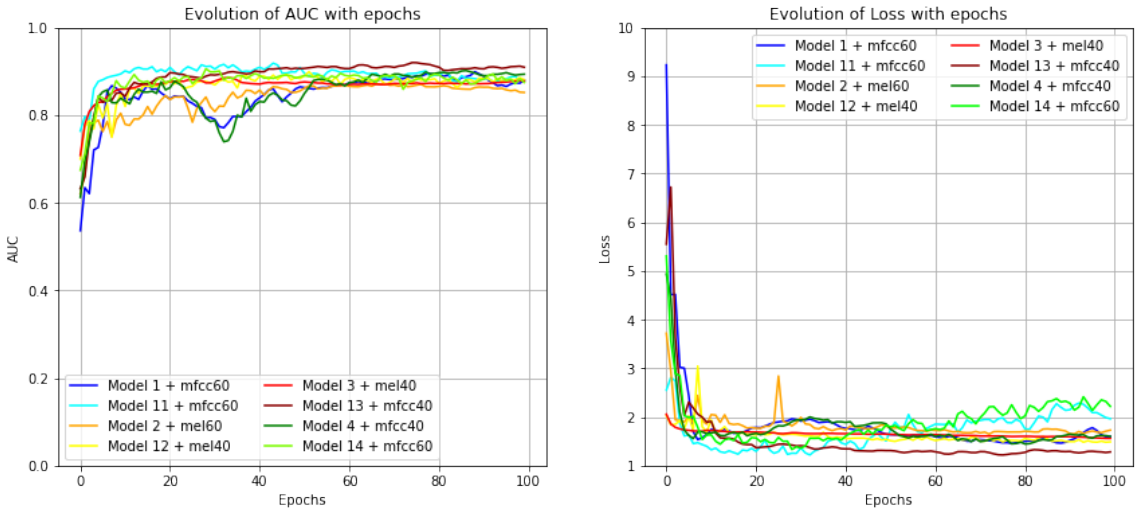


FIGURE 3.17: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four best models with a dropout rate of 0.2 for ESC-10.

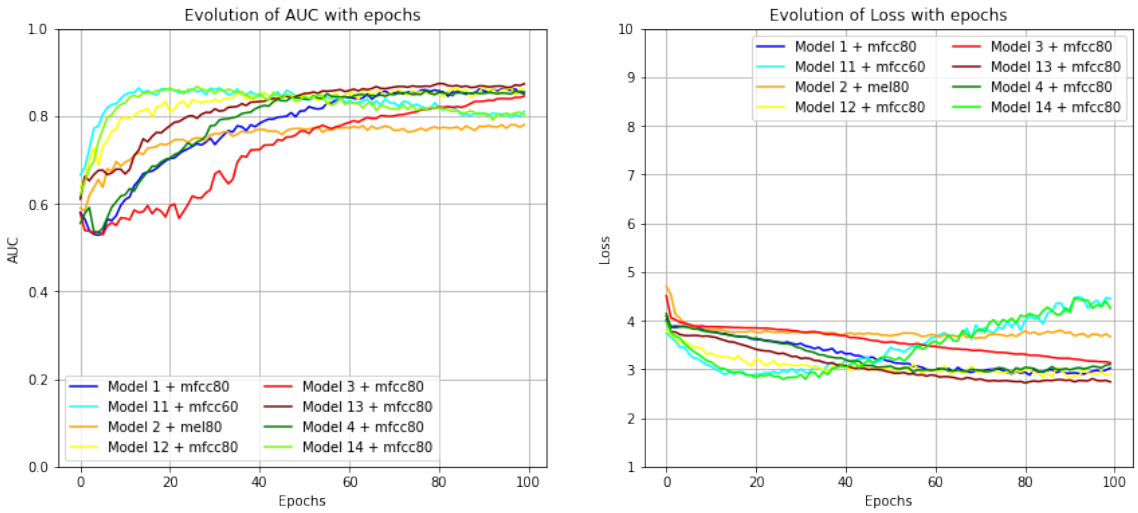


FIGURE 3.18: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four best models with a dropout rate of 0.2 for ESC-50.

the dropout rate change seems to be the next logical step to take to improve the model's performance. However, it would be improbable that a lower dropout rate would improve the results. However, this study was made to rule out this hypothesis properly, as shown in Table 3.18 and Figure 3.19.

Analysing the results, it can be confirmed that the results did not improve when the dropout rate was set to 0 (zero), which is consistent since dropout act as a regularization to avoid model overfitting. Focusing on the loss curves, it is possible to see that model 19 has suffered from overfitting for ESC-50. However, for ESC-10, the same model did not

TABLE 3.18: Results for Adamax optimizer's models for single features and dropout of 0.

Model 19: (optimizer: Adamax; dr: 0; dataset: ESC-10)							Model 19: (optimizer: Adamax; dr: 0; dataset: ESC-50)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.575**	**0.918**	**0.575**	**0.580**	0.592	**0.563**	**mfcc80**	**0.323**	0.771	**0.323**	**0.317**	0.373	**0.295**
mfcc60	0.550	0.909	0.550	0.566	0.609	0.525	**mfcc40**	0.308	0.779	0.308	0.298	0.343	0.260
mfcc40	**0.575**	0.893	**0.575**	0.565	0.584	**0.563**	**mfcc60**	0.293	0.783	0.293	0.283	0.336	0.253
mel80	0.525	0.835	0.525	0.524	0.557	0.488	**mel20**	0.238	**0.827**	0.238	0.224	0.325	0.095
mel20	0.525	0.872	0.525	0.523	0.554	0.450	**mel60**	0.208	0.781	0.208	0.204	0.292	0.113

dr - dropout rate; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

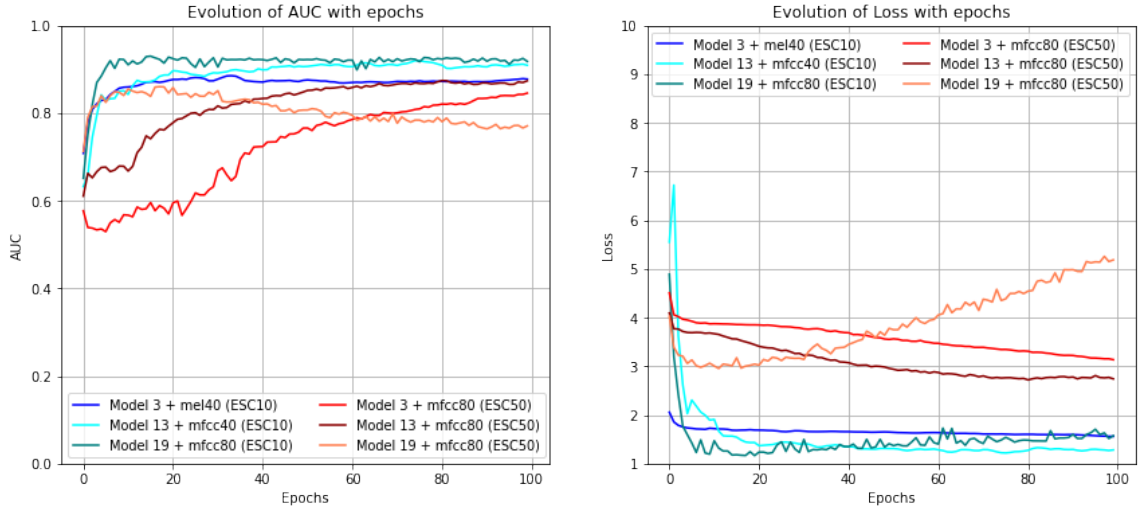


FIGURE 3.19: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the models with Adamax as an optimizer and without dropout rate for ESC-10 and ESC-50.

show the same behaviour, but even though the model has not overfitted, the results were worst than the model trained with a dropout rate of 0.2.

Therefore, out of all these models, the one that allowed the best results was the model with Adam optimizer and a dropout rate of 0.2 for both datasets. For ESC-10, model 11 and model 4 produce similar results. Nonetheless, there is a big difference in the recall value which makes model 11 slightly better than model 4.

3.5.3 Models with a Combination of Features as Input - Baseline Model Architecture

In the study made for UrbanSound8K, the change of models' input to a combination of features significantly boosted the models' performance, so the same approach was employed in the ESC datasets.

Table 3.19 and Table 3.20 show the results of the ESC datasets for the base architecture models with a combination of features as input.

It is necessary to note that in addition to the combination of features already mentioned in Table 3.1, four new features groups were used, which were "mmq40", "mmstftq40", "mmcens40" and "mfccmel40" which correspond to the same combination of features presented on Table 3.1 with the same group of letters. However, the number of bins was set to 40.

TABLE 3.19: Results of the 6 models for different feature combinations - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.713**	0.938	**0.713**	**0.701**	0.740	**0.675**	**mms40**	**0.538**	**0.908**	**0.538**	**0.499**	**0.656**	**0.263**
mmq	0.688	0.941	0.688	0.673	0.729	0.538	**mmstftq40**	0.475	0.858	0.475	0.433	0.640	0.200
mms40	0.675	**0.942**	0.675	0.663	0.707	0.513	**mmcens40**	0.438	0.864	0.438	0.396	0.586	0.213
mmstftq	0.675	0.938	0.675	0.661	0.758	0.625	**mmstftq**	0.413	0.793	0.413	0.383	0.519	0.175
mmq40	0.675	0.940	0.675	0.660	**0.780**	0.488	**mfccstft**	0.425	0.858	0.425	0.375	0.433	0.163
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmcens	**0.613**	0.894	**0.613**	**0.587**	0.708	0.213	**mmsqc**	**0.738**	0.910	**0.738**	**0.727**	**0.773**	**0.725**
mmsqc80	0.538	0.907	0.538	0.525	0.718	**0.350**	**mmstftq40**	0.725	**0.916**	0.725	0.719	0.737	0.700
mms60	0.538	**0.912**	0.538	0.521	0.684	0.163	**mmq40**	0.688	0.902	0.688	0.677	0.692	0.675
mmstftq	0.513	0.903	0.513	0.497	**0.826**	0.238	**mmq**	0.663	0.884	0.663	0.657	0.675	0.650
mms80	0.500	0.893	0.500	0.493	0.667	0.175	**mms80**	0.663	0.894	0.663	0.647	0.688	0.663
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.213**	0.533	**0.213**	**0.103**	**0.205**	**0.188**	**mmsqc**	**0.563**	**0.908**	**0.563**	**0.554**	**0.750**	0.150
mms60	0.113	0.504	0.113	0.102	0.083	0.063	**mmstftq40**	0.450	0.883	0.450	0.426	0.543	0.238
mfccstft	0.150	0.541	0.150	0.090	0.145	0.113	**mmsqc80**	0.450	0.884	0.450	0.426	0.568	**0.263**
mmsqc	0.150	**0.565**	0.150	0.085	0.154	0.125	**mmstftq**	0.388	0.848	0.388	0.359	0.600	0.150
mmcens40	0.163	0.541	0.163	0.056	0.163	0.163	**mms80**	0.388	0.856	0.388	0.344	0.600	0.188

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Once again, for ESC-50, the two models that presented the worst results was model 5 and 6 with Adadelta and Adagrad optimizers, respectively. The best results were given by model 1 with Adam optimizer for ESC-50 and model 4 with Nadam for ESC-10 dataset. Unlike the other experiments, for ESC-10, the two worst models were Adadelta and SGD, conforming to the non-robustness of both optimizers on large datasets. Thus, following the previous methodology, the two worst models for each dataset were not considered in the subsequent analyses.

In terms of the group of features, for ESC-10, the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT with 80 bins correspond to the group of features that appears in most models, and MFCC, Melspectrogram and Chroma CQT with 80 bins

TABLE 3.20: Results of the 6 models for different feature combinations - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq	**0.393**	0.873	**0.393**	**0.379**	0.575	**0.250**	**mmq**	**0.178**	0.789	**0.178**	**0.153**	0.591	**0.033**
mmstftq	0.383	0.875	0.383	0.361	0.584	0.243	**mms60**	0.173	0.770	0.173	0.151	0.529	0.023
mmsqc80	0.365	0.869	0.365	0.352	0.511	0.243	**mms40**	0.155	**0.808**	0.155	0.149	0.667	0.030
mmcens40	0.365	**0.895**	0.365	0.345	**0.635**	0.200	**mmcens40**	0.175	0.795	0.175	0.145	**0.722**	**0.033**
mms40	0.358	0.879	0.358	0.343	0.550	0.193	**mmcens**	0.158	0.776	0.158	0.140	0.650	**0.033**
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.230**	**0.844**	**0.230**	**0.194**	0.565	**0.033**	**mmstftq**	**0.375**	0.878	**0.375**	**0.354**	0.571	**0.223**
mmq	0.200	0.827	0.200	0.168	**0.647**	0.028	**mmstftq40**	0.360	0.882	0.360	0.346	0.497	0.180
mmstftq	0.183	0.787	0.183	0.168	0.462	0.015	**mmsqc**	0.355	**0.884**	0.355	0.345	0.524	0.215
mmsqc	0.180	0.836	0.180	0.156	0.571	0.020	**mms80**	0.345	0.872	0.345	0.339	**0.605**	0.188
mfccstft80	0.163	0.821	0.163	0.155	0.440	0.028	**mms60**	0.355	0.874	0.355	0.337	0.536	0.203
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq40	**0.035**	0.503	**0.035**	**0.011**	**0.051**	**0.033**	**mms60**	0.055	0.589	0.055	**0.057**	0.286	0.005
mmstftq	0.023	**0.523**	0.023	0.009	0.032	0.015	**mmq**	**0.060**	0.614	0.060	0.054	**0.429**	**0.008**
mmsqc	0.030	0.518	0.030	0.009	0.030	0.020	**mms80**	0.060	0.609	**0.060**	0.050	0.250	0.005
mmq	0.015	0.506	0.015	0.009	0.028	0.013	**mfccstft**	0.048	**0.620**	0.048	0.043	0.000	0.000
mmsqc80	0.013	0.503	0.013	0.008	0.014	0.008	**mmcens**	0.055	0.572	0.055	0.042	0.167	0.003

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

appear in the top 5 of almost every model for ESC-50. Furthermore, the mentioned combination of features is also the one that permit obtaining the top result out of all models for the ESC-50 dataset.

Figures 3.20 and 3.21 show the evolution of AUC and loss curves with epochs. It is possible to verify that for ESC-50, model 3 did not reach convergence within the 100 training epochs, so it should be trained until convergence to see the best results it can give, which may overcome the results obtained by model 1.

Because the models that presented the best performance were fully trained, the results are shown next in Figure 3.22 and Table 3.21 for model 3 with ESC-50 fully trained to fairly compare this model results with the others.

Therefore, looking at the Figure mentioned above, it is possible to conclude that model 3 converged around the 170^{th} epoch. The graph presents two curves corresponding to two different combinations of feature inputs; one corresponds to the group of features that gave the best performance for the base architecture model, and the other is the group that gives the best results for the fully trained model.

Focusing on Table 3.21, model 3 shows a considerable performance improvement but could not outperform model 1, so model 1 is still the best for the ESC-50 dataset.

After getting the results for the base architecture models thoroughly trained, other experiments were performed for the four best models for each dataset by adding an extra

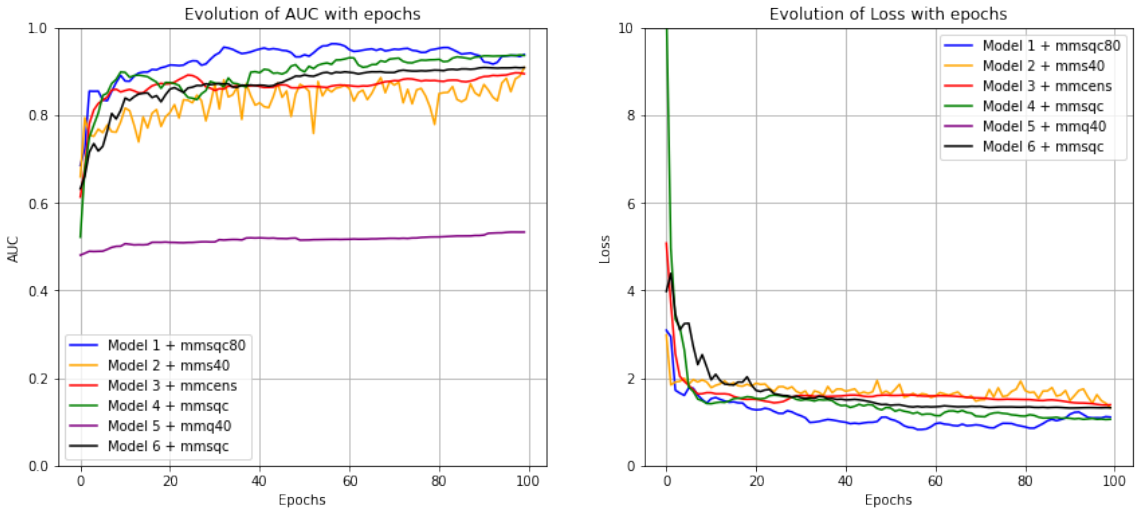


FIGURE 3.20: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four base models with a combination of features as input for ESC-10.

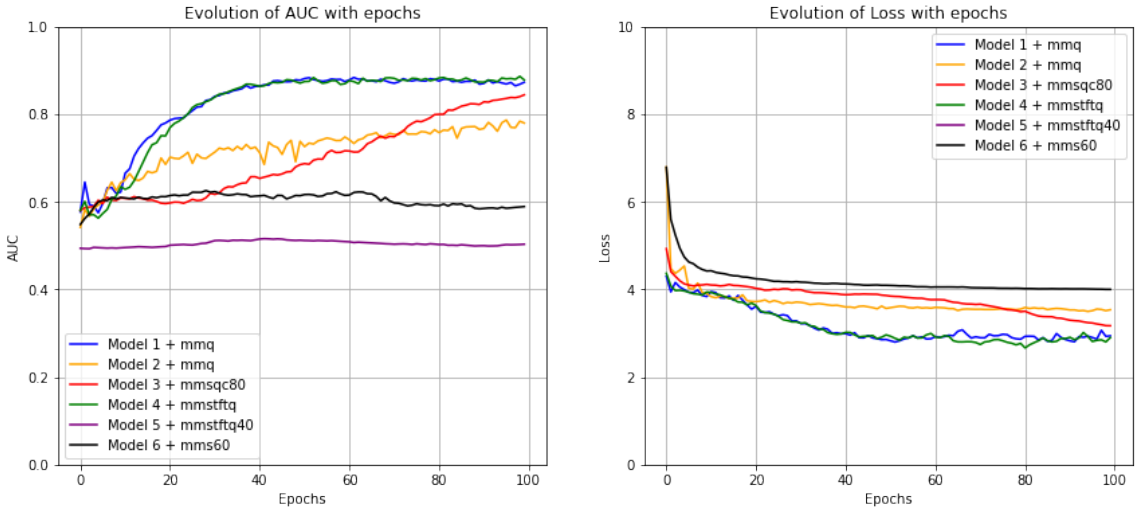


FIGURE 3.21: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for the four base models with a combination of features as input for ESC-50.

TABLE 3.21: Results for model 3 fully trained for different feature combinations - ESC-50.

Model 3: (optimizer: Adamax; dataset: ESC-50)						
Features	acc	AUC	micro f1score	macro f1score	precision	recall
mmstftq	**0.360**	0.885	**0.360**	**0.348**	0.602	0.178
mms80	0.348	0.890	0.348	0.341	0.625	0.188
mfccstft	0.348	0.886	0.348	0.337	0.512	0.158
mmsqc	0.353	**0.902**	0.353	0.335	0.579	0.193
mfccstft80	0.350	0.867	0.350	0.322	0.516	0.205

dr - dropout rate; acc - accuracy; AUC - area under the receiver operating characteristic curve.
All metrics range from $[0,1]$ (the higher, the better).

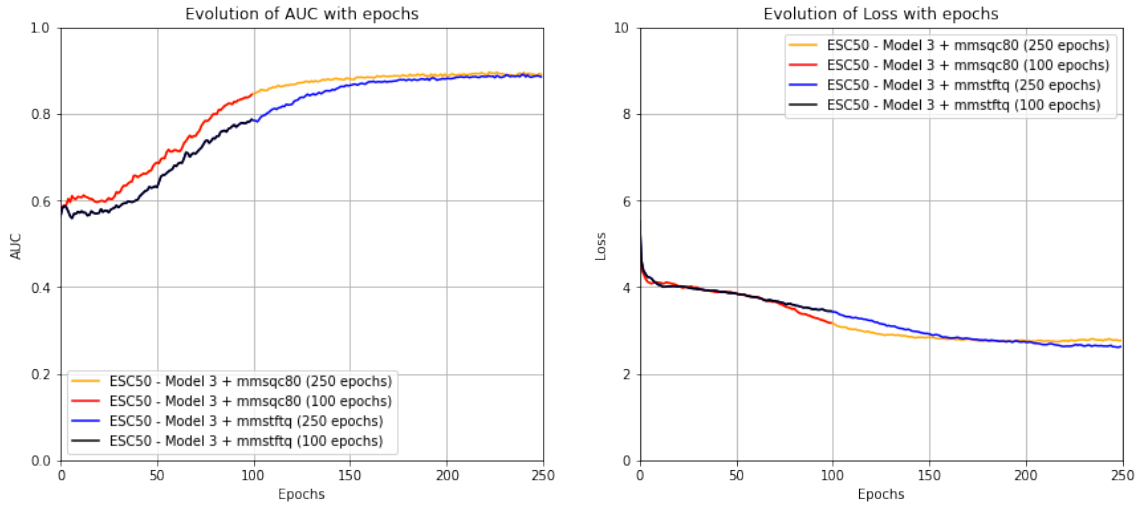


FIGURE 3.22: Graphs of the evolution of AUC (left) and loss function (right) with the epochs for model 4 with ESC-10 dataset and model 3 with ESC-50.

layer and changing the dropout rate to 0.2 and 0.6 to assess the impact of those changes. As previously noted for experiments with a single feature input, the only change that provided better results than the base architecture models has the dropout rate change to 0.2.

So next, the results for the dropout rate of 0.2 for both datasets are presented.

3.5.4 Models with a Combination of Features as Input - Dropout Rate of 0.2

Table 3.22 and Table 3.23 show the results for the models with a dropout rate of 0.2.

TABLE 3.22: Results of the 4 best models for dropout of 0.2 and different feature combinations - ESC-10.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.725**	0.936	**0.725**	**0.724**	**0.750**	**0.713**	**mmstftq**	**0.638**	0.907	**0.638**	**0.639**	0.707	0.513
mmcens40	0.713	**0.952**	0.713	0.708	0.727	0.700	**mmq40**	0.600	**0.928**	0.600	0.569	0.724	**0.525**
mmstftq40	0.713	0.929	0.713	0.699	0.733	0.688	**mmcens**	0.550	0.912	0.550	0.528	0.696	0.400
mmq40	0.700	**0.952**	0.700	0.694	0.724	0.688	**mmcens40**	0.550	0.921	0.550	0.525	0.620	0.388
mmq	0.688	0.932	0.688	0.687	0.705	0.688	**mms40**	0.563	0.919	0.563	0.520	**0.773**	0.425
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq40	**0.700**	0.939	**0.700**	**0.694**	0.758	0.625	**mms60**	**0.775**	**0.956**	**0.775**	**0.774**	**0.787**	**0.738**
mmstftq	**0.700**	**0.958**	**0.700**	0.691	**0.782**	0.538	**mmq40**	0.725	0.942	0.725	0.719	0.753	0.725
mmsqc80	**0.700**	0.942	**0.700**	0.689	0.735	0.625	**mmcens40**	0.700	0.926	0.700	0.691	0.757	0.700
mmq	0.675	0.934	0.675	0.665	0.699	**0.638**	**mmstftq40**	0.688	0.925	0.688	0.682	0.714	0.688
mmcens40	0.663	0.950	0.663	0.654	0.750	0.563	**mmsqc**	0.675	0.921	0.675	0.677	0.761	0.638

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE 3.23: Results of the 4 best models for dropout of 0.2 and different feature combinations - ESC-50.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.398**	0.829	**0.398**	**0.386**	0.472	0.358	**mms40**	**0.313**	0.863	**0.313**	**0.289**	**0.519**	0.140
mmstftq	0.390	0.810	0.390	**0.386**	0.452	0.340	**mmcens40**	0.303	**0.868**	0.303	0.279	0.495	0.130
mmstftq40	0.393	**0.839**	0.393	0.385	**0.476**	0.350	**mms80**	0.275	0.852	0.275	0.263	0.500	0.138
mmsqc80	0.393	0.811	0.393	0.382	0.449	**0.360**	**mmsqc80**	0.275	0.844	0.275	0.262	0.510	0.130
mmsqc	0.390	0.817	0.390	0.376	0.445	0.333	**mmstftq**	0.255	0.852	0.255	0.245	0.405	0.113
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.383**	0.875	**0.383**	**0.373**	**0.583**	**0.228**	**mfccstft80**	**0.403**	0.816	**0.403**	**0.388**	0.459	0.353
mmstftq	0.365	**0.889**	0.365	0.349	0.556	0.223	**mmsqc**	0.385	**0.840**	0.385	0.371	**0.472**	**0.358**
mms80	0.348	0.878	0.348	0.334	0.555	0.215	**mmstftq**	0.378	0.825	0.378	0.371	0.452	0.350
mfccstft80	0.345	0.874	0.345	0.329	0.544	0.218	**mmstftq40**	0.370	0.827	0.370	0.364	0.457	0.323
mmsqc	0.350	0.874	0.350	0.328	0.545	0.198	**mms40**	0.368	0.818	0.368	0.354	0.435	0.310

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Analysing the values for the various metrics can verify that this change improved all models' performance. The model with the Nadam optimizer provided the best results for both datasets.

Regarding the group of features, the combination of MFCC, Melspectrogram, Chroma STFT and Chroma CQT with 80 bins appear to be the preferable group since it appears in almost all models' top 5 and also, produces the top results of two models for ESC-10.

Looking to Figures 3.23 and 3.24, it is possible to verify that all models with a dropout rate of 0.2 have converged. For the ESC-50, models 11 and 14 were overfitted and focused on the results of the AUC curves, these are the only two models showing more bad behaviour than the corresponding base models.

3.5.5 Cross-validating Results

Since the UrbanSound dataset, significant differences in the results were observed depending on the folder considered as the test folder. The same study was done for the ESC datasets. However, it was not expected significant discrepancies in the results depending on the test folder because the datasets are balanced, so the model should have approximately the same difficulty attributing the suitable class to the unseen data regardless of the considered training and test folders.

Table 3.24 shows the results for the 5-fold cross-validation in which each folder is taken as the test folder for the ESC-10 and ESC-50 datasets.

Deep analysis of the results, it is possible to see that using a particular folder as the test folder can give better results with considerable differences in some metrics values

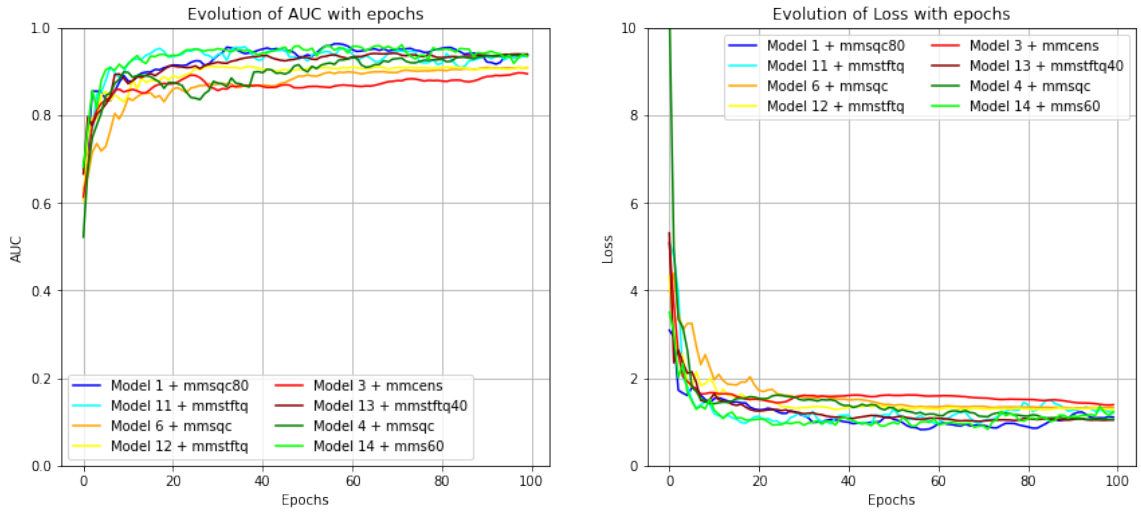


FIGURE 3.23: Graphs of the evolution of AUC (left) and loss function (right) with epochs for the four best base models and their corresponding ones with a dropout rate of 0.2 with a group of features as input for ESC-10.

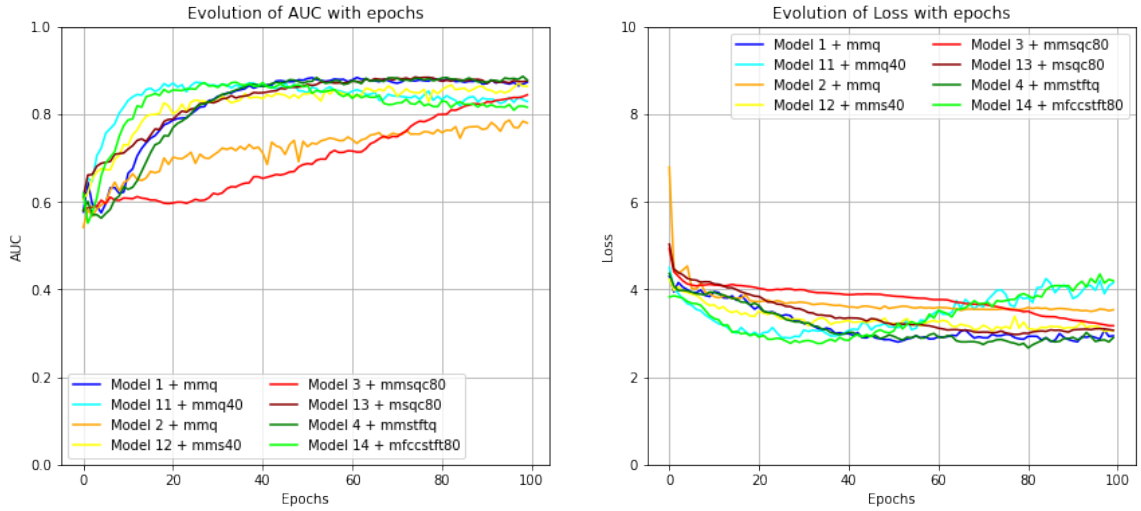


FIGURE 3.24: Graphs of the evolution of AUC (left) and loss function (right) with epochs for the four best base models and their corresponding ones with a dropout rate of 0.2 with a group of features as input for ESC-50.

between the best and the worst performance. Therefore, even though these are balanced datasets, performing cross-validation and averaging out the results seems to be the more fair way to compare the results due to the different difficulties the model has depending on the considered test folder.

Regarding the models' efficiency in distinguish between classes, Figure 3.25 and Figure 3.26 show the confusion matrices of the 5-fold cross-validation model's average used for the ESC-10 dataset and ESC-50 dataset, respectively.

TABLE 3.24: Results for the 5 folds of the top performing model for ESC-10 and ESC-50 dataset.

ESC-10 dataset							**ESC-50 dataset**						
Model 14: (opt: Nadam; dr: 0.2; feature: mms60)							**Model 14: (opt: Nadam; dr: 0.2; feature: mfccstft80)**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1	0.775	0.956	0.775	0.774	0.787	0.738	1	**0.403**	0.816	**0.403**	**0.388**	**0.459**	**0.353**
2	0.675	0.936	0.675	0.663	0.707	0.663	2	0.360	0.805	0.360	0.339	0.411	0.310
3	**0.800**	0.962	**0.800**	**0.797**	**0.842**	**0.800**	3	0.375	0.832	0.375	0.352	0.438	0.338
4	0.775	**0.968**	0.775	0.774	0.827	0.775	4	0.380	**0.848**	0.380	0.361	0.443	0.348
5	0.713	0.921	0.713	0.709	0.724	0.688	5	0.388	0.821	0.388	0.369	0.447	0.345
Average	0.748	0.948	0.748	0.743	0.777	0.733	**Average**	0.381	0.824	0.381	0.362	0.439	0.339

opt - optimizer; dr - dropout rate; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

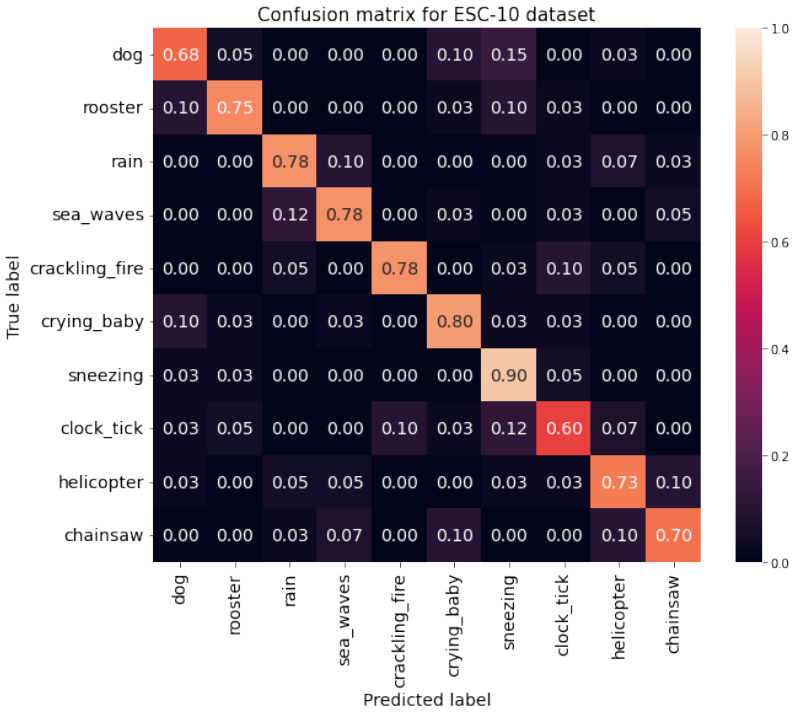


FIGURE 3.25: Confusion matrix for the best model using the ESC-10 dataset.

Analysing the confusion matrices can be concluded that for the ESC-10 dataset, dog and clock tick classes are the most challenging classes with an accuracy inferior to 70%, and the most straightforward class is sneezing, which has 90% of accuracy. Regarding the ESC-50 dataset, the classes with the highest accuracy are sea waves, clapping, and toilet flush, with a value of 70% or superior. However, there are 36 classes with an accuracy inferior to 50%, with the lowest value being 2% for hand saw, water drops, and clock tick classes.

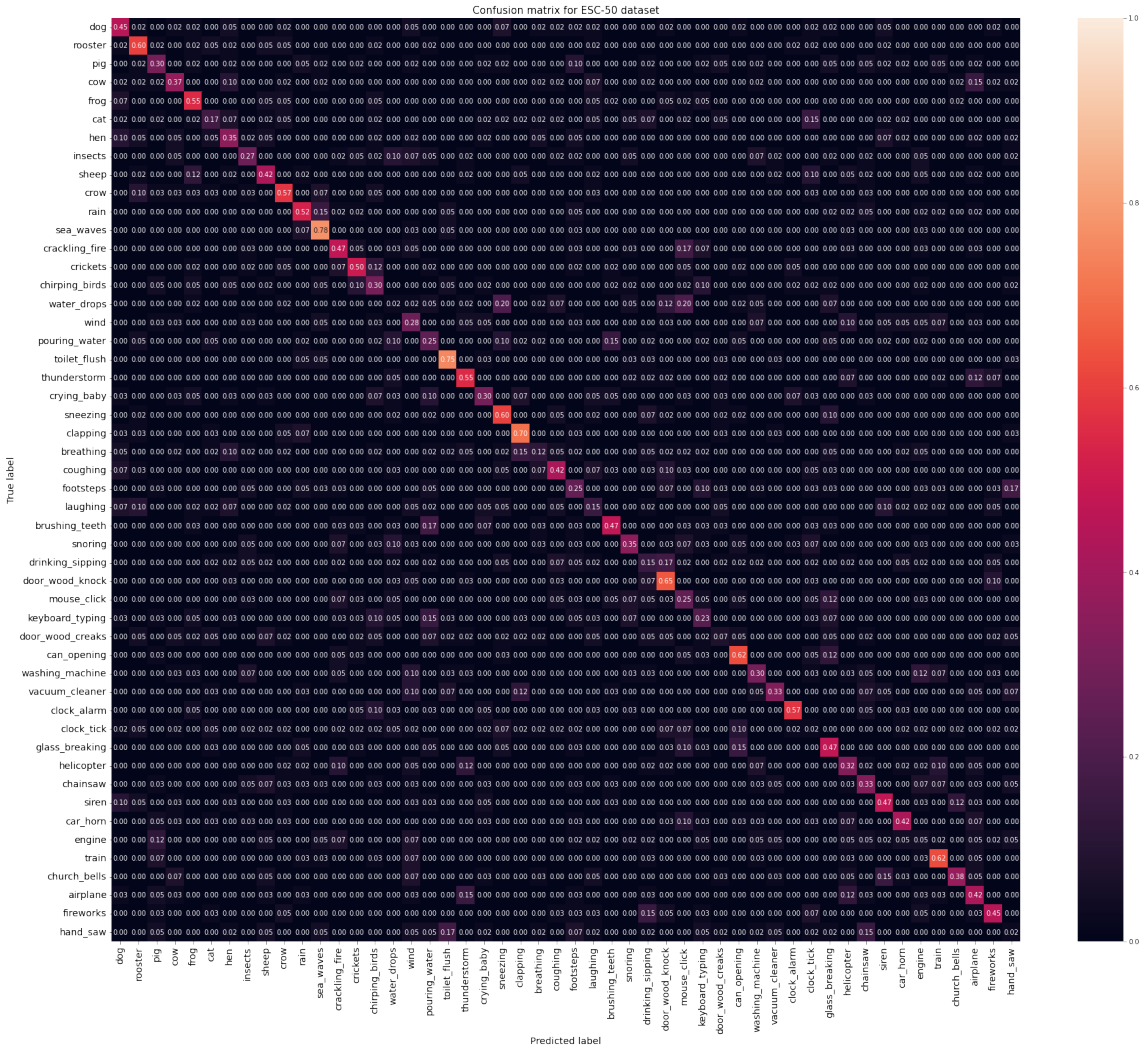


FIGURE 3.26: Confusion matrix for the best model using the ESC-50 dataset.

3.6 Overall Baseline Conclusions

This chapter explored the different optimization functions, feature extraction techniques, the loss function used, and the different used metrics and defined the architecture of the baseline model. To conclude the chapter, several experiences were performed to obtain the best baseline model.

Therefore, models were implemented with different inputs of single and group features, dropout rates and a higher number of layers. The various experiences allow to reach the following conclusions:

- Analysing the base models, Adagrad and Adadelta are the optimizers that produce the worst results regardless if the input is a single feature or a group feature, except in one case with the ESC-10 dataset where SGD performed worst than Adagrad.

- Regarding the UrbanSound8K dataset, adding an extra layer for a single feature input, the only model that benefited from the change was the one that used SGD* as an optimizer. For a combination of features, the models with Adam and Adamax as optimization functions showed a slight improvement in the results. The model with Nadam* as optimizer attained 5 out 6 best metrics. However, unlike in a single feature input, the model with SGD optimizer did not show any performance improvement. For the ESC datasets, adding an extra layer did not improve the models' performances, regardless of model input or optimization function.

- Changing the dropout rate for the single feature input was beneficial for Adam, either by changing the rate to 0.2 or 0.6. For SGD, only the change for 0.2 was better than the base model. For Adamax, the base model and the model with a dropout rate of 0.2 had similar results for the top result. However, when analysing the top 2 features, the base model was better, and for Nadam, neither change was beneficial. Adam and Adamax used dropout rates of 0.8 and 0 to explore the advantages of changing the dropout rate further. These changes revealed that the dropout rate increase started to be too high for the model to have enough information to distinguish correctly between classes. A low rate leads the model to overfit. Regarding the group of features, for Adam and Nadam*, changing to 0.6 improved the results. For Adamax, the change to 0.2 provided better results than the base model, and no change was profitable for SGD. For the ESC datasets, only the change for a dropout rate of 0.2 produced improvements in models' results either for a single or a combination of features as input.

- For the models with a group of features as input trained for UrbanSound8K, it was observed that some results had improved when an extra layer was added and when the dropout rate was changed. However, the combination of these alterations was only profitable for Adamax* when an extra layer was added, and the dropout rate was 0.2.

Out of all these experiences, the optimization function that allowed the best result was Nadam, whether for a single* or a group of features* for UrbanSound8K and the ESC datasets with a combination of features as input, however, for a single feature input, Adam show to be more beneficial. Regarding the input features, MFCC was the single feature that produce better results, particularly, MFCC with 80 MFCCs for UrbanSound8K

and with 60 MFCCs to ESC datasets. For a group of features, the combination of MFCC, Melspectrogram, Chroma STFT, Chroma CQT and Chroma CENS with 40 bins was the combination that gave the best results for UrbanSound8K and the second best result out of all experiences for ESC-10.

Chapter 4

End-to-End Models

This chapter employed the acquired knowledge in implementing the baselines and developing State-of-the-Art (SOTA) end-to-end models to classify Urban Sound events effectively. Extensive experiments and model evaluation with hyper-parameter tuning and different architectures are extensively evaluated and discussed to identify the limitations and potentialities of the different models and improvements regarding the established baselines.

4.1 Residual Neural Network (ResNet)

This architecture was introduced by He et al. [12] to solve the vanishing gradients problem and mitigate the degradation of Deep Neural Network (DNN)s with several layers. The vanishing gradient problem complicates convergence since the gradient is back-propagated to earlier layers. The repeated multiplications may make the gradient go infinitely small, making it challenging for model convergence. This model degradation is noticeable as network depth increases, leading to a degradation of the accuracy, and adding more layers to a suitably deep model leads to higher training error.

To solve the degradation problem is introduced a deep residual learning framework which lets the layers fit a residual mapping, $F(x)$ where $F(x) := H(x) - x \Leftrightarrow H(x) := F(x) + x$ where $H(x)$ is the desired underlying mapping and x is the input of the layer. This approach can be accomplished by a feedforward neural network with shortcut connections.

Shortcut connections prevent the vanishing gradients as they are used to skip one or more layers to perform identity mapping, and their outputs are added to the outputs of the stacked layers.

So, ResNet are constituted by a stack of residual blocks, which are an ensemble of convolutional layers followed by a batch normalization layer and a ReLU activation function with shortcut connections that skips a stack of layers and adds the input directly before the last ReLU activation function of the stack, Figure 4.1 represents a residual block.

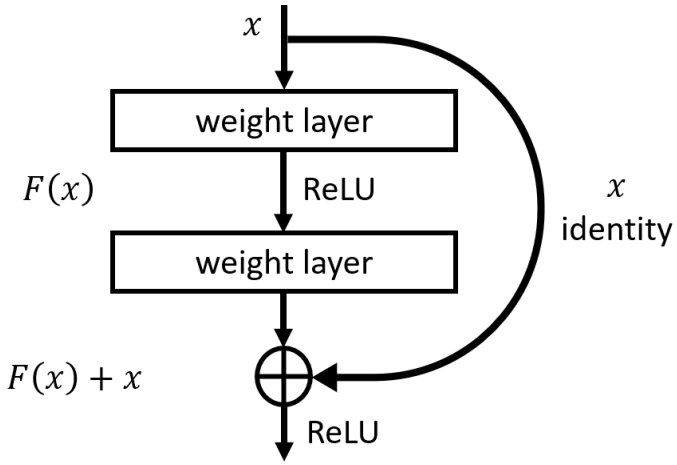


FIGURE 4.1: Residual block (adapted from He et al. [12]).

The model architecture used in this work was ResNet 50, constituted by the following 50 layers:

- convolutional layer with 64 different kernels with a stride of 2 and a kernel size of 7x7.
- max pooling layer with a size of 3x3 and a stride of 2;
- stack of a convolutional layer with 64 different kernels of size 1x1, a convolution layer with 64 different kernels of size 3x3 and a convolution layer with 256 different kernels of size 1x1. There are 3 stacks with this combination, and there is a skip connection between each stack;
- stack of a convolutional layer with 128 different kernels of size 1x1, a convolution layer with 128 different kernels of size 3x3 and a convolution layer with 512 different kernels of size 1x1. There are 4 stacks with this combination, and there is a skip connection between each stack;

- stack of a convolutional layer with 256 different kernels of size 1x1, a convolution layer with 256 different kernels of size 3x3 and a convolution layer with 1024 different kernels of size 1x1. There are 6 stacks with this combination, and there is a skip connection between each stack;
- stack of a convolutional layer with 512 different kernels of size 1x1, a convolution layer with 512 different kernels of size 3x3 and a convolution layer with 2048 different kernels of size 1x1. There are 3 stacks with this combination, and there is a skip connection between each stack;
- then, it has done an average pool, and at the end, there is a fully connected layer with Softmax as the activation function.

A skip connection is performed between the stacks of different layers with a stride of 2.

4.2 Dense Convolutional Network (DenseNet)

Previous Deep Convolutional Neural Networks (DCNN) architectures like ResNet create short paths from early layers to the last layers to solve the vanishing gradient problem. DenseNet, introduced by Huang et al. [13], instead uses a dense connectivity pattern which directly connects all layers, with matching feature-map sizes, with each other. So, each layer obtains additional input from all preceding layers and passes on its feature maps to all subsequent layers. Unlike, in ResNet where features are combined by summation before they serve as input to a layer, DenseNet concatenates the features; thereby, the l^{th} layer has l inputs consisting of the feature maps of all preceding convolutional blocks, so, an L-layer network has $\frac{L(L+1)}{2}$ connections. This allows the final classifier to make decisions based on all feature maps in the network.

Therefore, the advantages of DenseNets are the flow of information and gradients throughout the network, the direct access that each layer has to the gradients from the loss function and the original input signal facilitates the training of DCNN.

The network architecture used in this work was DenseNet201, see Figure 4.2, which is constituted by the following 201 layers:

- Convolutional layer with a kernel size of 7x7 and a stride of 2;
- Max pooling layer with a kernel size of 3x3 and a stride of 2;

- Dense block: a stack of a convolutional layer with a kernel size of 1x1 and a convolutional layer with a kernel size of 3x3. There are 6 stacks formed by the same layers;
- Transition Layer: convolutional layer with a kernel size of 1x1, and it is done an average pool with a kernel size of 2x2 and stride of 2;
- Dense block is composed of the same layers as the first dense block, but there are 12 stacks in a row;
- Transition Layer composed of the same layers as the first transition layer;
- Dense block is composed of the same layers as the first dense block, but there are 48 stacks in a row;
- Transition Layer composed of the same layers as the first transition layer;
- Dense block is composed of the same layers as the first dense block, but there are 32 stacks in a row;
- To finalize, it has done a global average pool with a kernel size of 7x7, and then it is passed to a fully connected layer with Softmax as the activation function.

Before each convolution, it performed the following operations: batch normalization and ReLU.

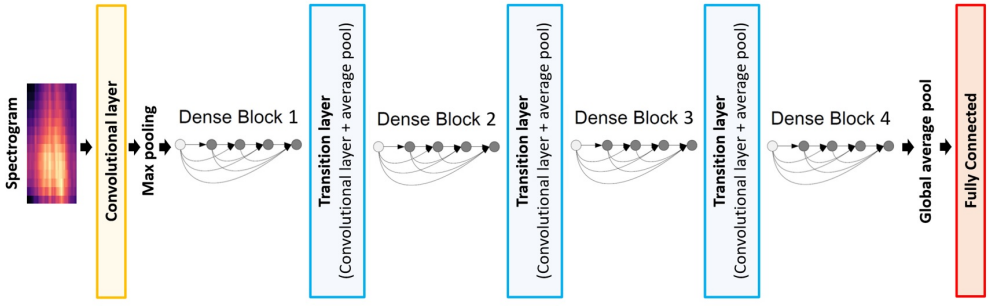


FIGURE 4.2: DenseNet architecture. DenseNet201 has 6, 12, 48 and 32 convolutional layers in each of the dense blocks, respectively.

4.3 Inception

The creation of this architecture was motivated by the fact that the most straightforward way to improve the performance of DNNs is by increasing their depth and width (number of units per level of depth). However, this will usually mean more parameters which make the network more prone to overfit and increase the use of computational resources. Therefore, a network that uses extra sparsity and exploits the current hardware by utilizing dense matrices should be used to solve the mentioned problems.

Inception is introduced by Szegedy et al. [33] to fulfil those requirements and is based on two ideas: find how an optimal local sparse structure in a convolutional network can be approximated and covered by readily available dense components; and judiciously apply dimension reduction and projection whenever the computational requirements would otherwise increase too much. Therefore, an Inception network consists of several Inception layers, which are a combination of a 1x1 convolutional layer, a 3x3 convolutional layer and a 5x5 convolutional layer with their output filter banks concatenated into a single output vector that will serve as input of the next layer, stacked upon each other with occasional max-pooling layers with a stride of 2. For the model to be memory efficient during training, it seems beneficial to start using Inception layers only at higher layers, keeping the others as convolutional layers.

The main benefit of this architecture is that it allows increasing the width of the network without uncontrollably increasing computational complexity. However, suppose the architecture is scaled up. In that case, most computational gains can be immediately lost. Due to the lack of clear reasons why some design decisions were taken, Szegedy et al. [34] introduced Inception-v2 and later, the same group of researchers also developed Inception-v3, Szegedy et al. [35], which was the architecture used in this work.

Therefore, the researchers have proposed the following upgrades: avoid representational bottlenecks, particularly early in the network, because they can cause the loss of too much information, so the input dimension should not be drastically changed; increase the activations per tile as they allow for more disentangled features. As a result, the model will train faster; spatial aggregation is done over lower dimensional embeddings, which won't cause much or any loss in representational power, and it will facilitate the dimension reduction, which will make the learning faster; lastly, balance the width and depth of the network that should be increased in parallel to get higher quality networks. This led to the following changes the 5x5 convolution is replaced by two 3x3 convolutions,

the nxn convolutions are substituted by a 1xn followed by an nx1 convolution, and to solve the bottleneck problem, the filter bank outputs are made wider instead of deeper to prevent excessive dimension reductions which were implemented in Inception-v2. For Inception-v3, besides the changes mentioned for Inception-v2, it was also factorized 7x7 convolutions into three 3x3 convolutions; use auxiliary classifiers along with batch normalization to improve the convergence and avoid the vanishing gradient problem; and ultimately, label smoothing, which is a regularization technique that introduces noise for the labels to account for mistakes that datasets might have in them, to regularize the classifier layer by estimating the effect of label-dropout during training. Thus, the architecture of Inception-v3, Figure 4.3, is 42 layers deep, consisting of the following layers:

- Convolutional layer with a patch size of 3x3 and stride of 2;
- Convolutional layer with a patch size of 3x3;
- Convolutional layer with a patch size of 3x3 and padding;
- Pool layer with a patch size of 3x3 and stride of 2;
- Convolutional layer with a patch size of 3x3;
- Convolutional layer with a patch size of 3x3 and stride of 2;
- Convolutional layer with a patch size of 3x3;
- 3 standard inception modules with 288 filters, each with a grid size of 35x35;
- 5 factorized inception modules;
- 2 inception modules with a concatenated output filter bank size of 2048 for each tile;
- Pool layer with a patch size of 8x8;
- Linear layer to convert the input into logits;
- Softmax layer.

Between the change to different inception, the module always performs grid size reduction, which decreases the feature maps' grid size by expanding the network filters' activation dimension before applying maximum or average pooling to avoid a representational bottleneck.

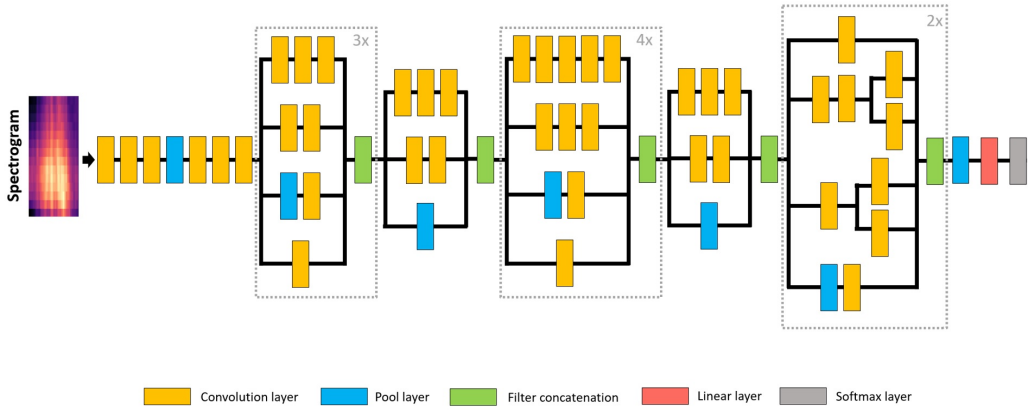


FIGURE 4.3: Inception-v3 architecture.

4.4 Results

In this section, the results for the ResNet, DenseNet and Inception models with different optimization functions are presented, as was done in 3.4 section. However, instead of using Nadam as optimization function, was used AdamW which is a stochastic optimization method that decouples Adam's weight decay from the gradient update, allowing only the gradients of the loss function to be adapted. So, the decouple weight decay regularizes all weights with the same rate (Loshchilov and Hutter [17]).

Then, a detailed analysis of data augmentation's influence on the models' performance is presented, more specifically, the influence of the combination of time stretch with a pitch shift.

4.4.1 USC Dataset - Pre-trained vs. No Pre-trained

This section shows the average results of the 10 folds obtained for the UrbanSound8K dataset for the different model architectures with the following optimization functions: Adam, Adadelta, Adagrad, Adamax, Stochastic Gradient Descent (SGD) and AdamW. Table 4.1, Table 4.2 and Table 4.3 present the results for the ResNet, DenseNet and Inception model with the use of pre-trained model weights and models trained from scratch, respectively and the number of the epoch that gave the best results.

Analysing the results for the pre-trained models, it can be concluded that for ResNet, Adamax provides the best results. On the other hand, Adam is the most beneficial optimization function for DenseNet and Inception models. It is also evident that DenseNet

TABLE 4.1: Results for the average of 10 folds results for ResNet model for the various optimizers.

Model ResNet (Pre-trained)							**Model ResNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (63)	0.723	0.960	0.723	0.723	0.736	0.723	SGD (65)	0.309	0.761	0.309	0.284	0.321	0.309
Adam (57)	0.816	0.975	0.816	0.827	0.823	0.816	Adam (67)	**0.708**	**0.945**	**0.708**	**0.723**	0.712	**0.708**
Adamax (62)	**0.828**	**0.976**	**0.828**	**0.837**	**0.836**	**0.828**	Adamax (63)	0.695	0.937	0.695	0.711	0.705	0.695
AdamW (65)	0.821	**0.976**	0.821	0.830	0.829	0.821	AdamW (67)	0.704	0.945	0.704	0.715	**0.714**	0.704
Adadelta (64)	0.643	0.937	0.643	0.616	0.677	0.643	Adadelta (63)	0.258	0.724	0.258	0.229	0.273	0.258
Adagrad (58)	0.806	0.973	0.806	0.814	0.812	0.806	Adagrad (64)	0.583	0.904	0.583	0.602	0.600	0.583

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.2: Results for the average of 10 folds results for DenseNet model for the various optimizers.

Model DenseNet (Pre-trained)							**Model DenseNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (65)	0.731	0.963	0.731	0.735	0.742	0.731	SGD (66)	0.417	0.832	0.417	0.404	0.453	0.417
Adam (58)	**0.833**	0.977	**0.833**	**0.844**	**0.841**	**0.833**	Adam (68)	0.738	0.950	0.738	0.749	0.742	0.738
Adamax (58)	0.818	0.973	0.818	0.828	0.821	0.818	Adamax (62)	0.722	0.952	0.722	0.732	0.729	0.722
AdamW (66)	0.831	**0.978**	0.831	0.837	0.837	0.831	AdamW (67)	**0.742**	**0.954**	**0.742**	**0.756**	**0.752**	**0.742**
Adadelta (64)	0.625	0.930	0.625	0.612	0.652	0.625	Adadelta (64)	0.361	0.806	0.361	0.310	0.395	0.361
Adagrad (59)	0.811	0.975	0.811	0.818	0.815	0.811	Adagrad (65)	0.709	0.950	0.709	0.719	0.719	0.709

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.3: Results for the average of 10 folds results for the Inception model for the various optimizers.

Model Inception (Pre-trained)							**Model Inception (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (64)	0.593	0.923	0.593	0.541	0.620	0.593	SGD (63)	0.294	0.758	0.294	0.248	0.293	0.294
Adam (62)	**0.827**	**0.973**	**0.827**	**0.840**	**0.833**	**0.827**	Adam (68)	0.713	0.944	0.713	0.725	0.717	0.713
Adamax (56)	0.797	0.968	0.797	0.806	0.803	0.797	Adamax (64)	0.693	0.943	0.693	0.706	0.698	0.693
AdamW (66)	0.812	**0.973**	0.812	0.824	0.816	0.812	AdamW (66)	**0.726**	**0.955**	**0.726**	**0.738**	**0.730**	**0.726**
Adadelta (62)	0.412	0.824	0.412	0.341	0.411	0.412	Adadelta (60)	0.204	0.663	0.204	0.152	0.193	0.204
Adagrad (59)	0.775	0.971	0.775	0.787	0.783	0.775	Adagrad (62)	0.513	0.884	0.513	0.523	0.538	0.513

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

with Adam optimizer is the model that achieves the best results in all metrics. Regarding the best randomly initialized models, it is possible to conclude that in all metrics, the pre-trained models provided better results with a difference of around 10 percentage points (pp) in most metrics. In terms, of optimizers for ResNet, Adam was the preferable optimization function, nonetheless, for the DenseNet and Inception models, AdamW optimization function gave the best results. However, DenseNet gave the best results for the randomly initialized or the model that used pre-trained model weights.

Figure 4.4 shows the graphical representation of the accuracy and loss curves for the different models with the most advantageous optimizer for each model.

The graphics of Figure 4.4 show that all models have converged, ResNet with pre-trained weights around the 10^{th} epoch and all the other models around the 30^{th} epoch.

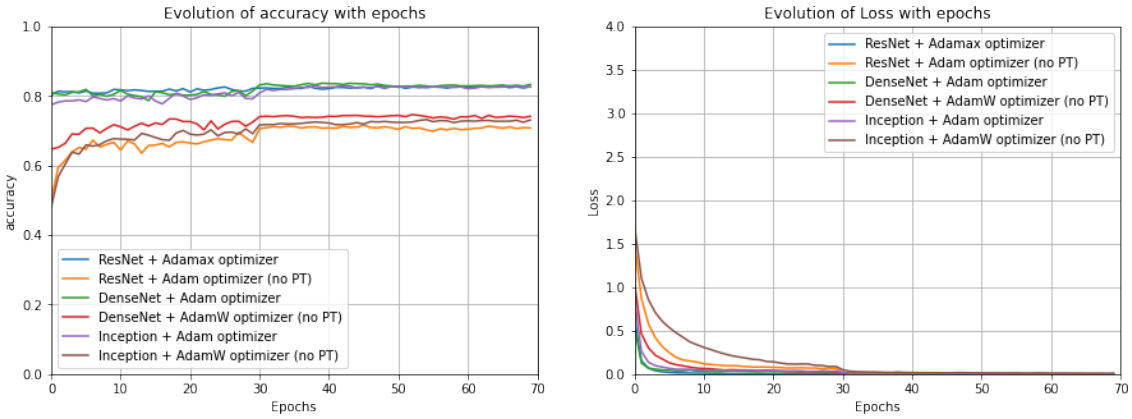


FIGURE 4.4: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model (no PT - no pre-trained).

Focusing on the accuracy's evolution with the epochs is possible to verify that the evolution curve of ResNet is almost straight. For all the other models, the same behaviour is obtained after approximately the 30^{th} epoch, which means that the accuracy value stops improving after the models' convergence significantly; however, all pre-trained models have achieved high accuracy. Nonetheless, DenseNet model shows to be slightly better than the other models. Also, the no pre-trained models because they are trained from scratch the starting accuracy is lower than pre-trained models, which highlights the need to train the models for more epochs to reach convergence.

4.4.2 USC Dataset - Data Augmentation

This section explores the influence that data augmentation has on the models' performance, so, taking into account the results obtained in the previous section, this study was made for the pre-trained models using the optimizer that provided the best results for each model. Table 4.4, Table 4.5 and Table 4.6 show the results for the different models with and without data augmentation for all the folds, respectively.

Comparing the results obtained for data with and without augmentation can be concluded that, on average, the results without data augmentation were better than those obtained with data augmentation. These results are unexpected since data augmentation is one of the most used methods to improve models' performance.

TABLE 4.4: Results for the 10 folds for ResNet model with and without data augmentation.

ResNet: (data aug: no; optimizer: Adamax (62 epochs))							**ResNet: (data aug: yes; optimizer: Adamax (64 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (54)	0.787	0.958	0.787	0.810	0.814	0.787	1 (65)	0.785	0.955	0.785	0.806	0.806	0.785
2 (67)	0.832	0.973	0.832	0.836	0.835	0.832	2 (64)	0.857	0.977	0.857	0.867	0.858	0.857
3 (60)	0.752	0.972	0.752	0.768	0.750	0.752	3 (69)	0.765	0.969	0.765	0.794	0.776	0.765
4 (63)	0.861	0.984	0.861	0.860	0.863	0.861	4 (67)	0.819	0.977	0.819	0.815	0.822	0.819
5 (62)	0.868	**0.991**	0.868	0.875	0.873	0.868	5 (58)	0.864	**0.990**	0.864	0.873	0.876	0.864
6 (58)	0.820	0.971	0.820	0.836	0.832	0.820	6 (58)	0.801	0.973	0.801	0.815	0.808	0.801
7 (67)	0.865	0.972	0.865	0.866	0.869	0.865	7 (62)	0.842	0.970	0.842	0.841	0.846	0.842
8 (68)	0.766	0.969	0.766	0.773	0.773	0.766	8 (64)	0.734	0.962	0.734	0.723	0.679	0.734
9 (64)	0.862	0.983	0.862	0.873	0.867	0.862	9 (68)	**0.881**	0.983	**0.881**	**0.889**	**0.890**	**0.881**
10 (53)	**0.871**	0.990	**0.871**	**0.878**	**0.882**	**0.871**	10 (61)	0.870	0.986	0.870	0.880	0.886	0.870
Average	0.828	0.976	0.828	0.837	0.836	0.828	**Average**	0.822	0.974	0.822	0.830	0.825	0.822

data aug - data augmentation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.5: Results for the 10 folds for DenseNet model with and without data augmentation.

DenseNet: (data aug: no; optimizer: Adam (58 epochs))							**DenseNet: (data aug: yes; optimizer: Adam (67 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (63)	0.809	0.963	0.809	0.833	0.829	0.809	1 (68)	0.771	0.952	0.771	0.795	0.806	0.771
2 (61)	0.827	0.978	0.827	0.827	0.825	0.827	2 (67)	0.850	0.977	0.850	0.868	0.859	0.850
3 (67)	0.720	0.963	0.720	0.742	0.721	0.720	3 (68)	0.735	0.961	0.735	0.761	0.734	0.735
4 (48)	0.834	0.980	0.834	0.833	0.846	0.834	4 (67)	0.791	0.963	0.791	0.785	0.799	0.791
5 (55)	**0.902**	**0.995**	**0.902**	**0.904**	**0.907**	**0.902**	5 (67)	0.841	**0.986**	0.841	0.846	0.852	0.841
6 (69)	0.813	0.971	0.813	0.832	0.822	0.813	6 (67)	0.786	0.965	0.786	0.799	0.791	0.786
7 (44)	0.876	0.978	0.876	0.880	0.884	0.876	7 (68)	0.841	0.970	0.841	0.843	0.850	0.841
8 (69)	0.803	0.964	0.803	0.815	0.815	0.803	8 (62)	0.764	0.958	0.764	0.783	0.776	0.764
9 (51)	0.871	0.986	0.871	0.883	0.876	0.871	9 (69)	0.844	0.977	0.844	0.856	0.846	0.844
10 (55)	0.878	0.989	0.878	0.889	0.888	0.878	10 (69)	**0.886**	0.983	**0.886**	**0.894**	**0.896**	**0.886**
Average	0.833	0.977	0.833	0.844	0.841	0.833	**Average**	0.811	0.969	0.811	0.823	0.821	0.811

data aug - data augmentation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.6: Results for the 10 folds for Inception model with and without data augmentation.

Inception: (data aug: no; optimizer: Adam (62 epochs))							**Inception: (data aug: yes; optimizer: Adam (67 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (65)	0.833	0.961	0.833	0.858	0.855	0.833	1 (67)	0.804	0.955	0.804	0.822	0.818	0.804
2 (52)	0.828	0.975	0.828	0.839	0.828	0.828	2 (68)	0.841	0.975	0.841	0.849	0.844	0.841
3 (65)	0.711	0.958	0.711	0.735	0.705	0.711	3 (67)	0.734	0.966	0.734	0.756	0.724	0.734
4 (67)	0.819	0.976	0.819	0.816	0.828	0.819	4 (67)	0.806	0.964	0.806	0.806	0.811	0.806
5 (68)	**0.897**	**0.987**	**0.897**	**0.906**	**0.905**	**0.897**	5 (64)	0.803	0.972	0.803	0.816	0.817	0.803
6 (62)	0.815	0.973	0.815	0.838	0.828	0.815	6 (68)	0.783	0.974	0.783	0.801	0.780	0.783
7 (68)	0.850	0.972	0.850	0.853	0.859	0.850	7 (67)	0.833	0.964	0.833	0.837	0.832	0.833
8 (61)	0.777	0.967	0.777	0.793	0.774	0.777	8 (64)	0.726	0.952	0.726	0.719	0.679	0.726
9 (67)	0.864	0.979	0.864	0.876	0.874	0.864	9 (69)	0.836	0.974	0.836	0.849	0.838	0.836
10 (46)	0.873	0.984	0.873	0.882	0.879	0.873	10 (68)	**0.878**	**0.987**	**0.878**	**0.888**	**0.889**	**0.878**
Average	0.827	0.973	0.827	0.840	0.833	0.827	**Average**	0.804	0.968	0.804	0.814	0.803	0.804

data aug - data augmentation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Regarding the easiest fold, there is no consensus between all the models. Nonetheless, DenseNet and Inception agreed by giving their best performance for the same folders in both situations.

In Figure 4.5, the behaviour of the models is pretty similar, being even challenging to distinguish between them; however, it is possible to see a slight worst behaviour for the models that were trained with data augmentation techniques.

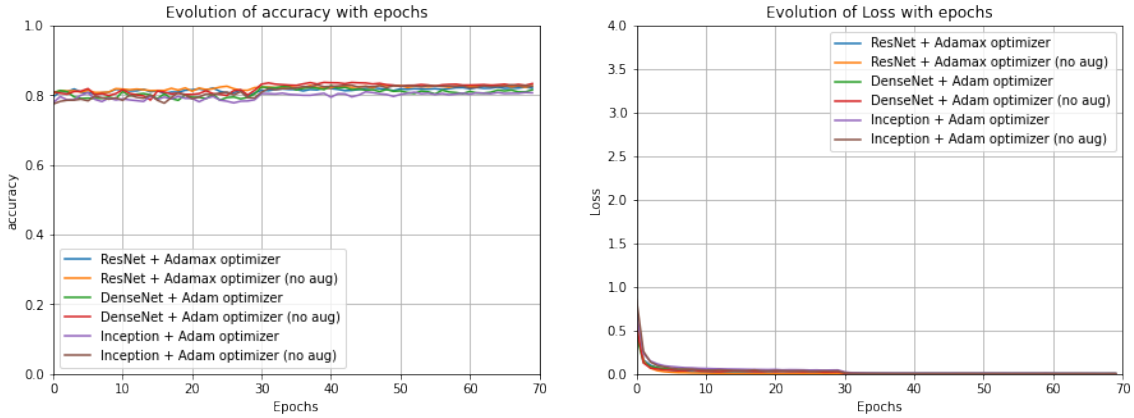


FIGURE 4.5: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model (no aug - no augmentation).

Lastly, analysing the confusion matrix presented in Figure 4.6 for the best model on this dataset which was the DenseNet model with no data augmentation and Adam as the optimization function is possible to understand that the most difficult class is air conditioner that is most misclassified as drilling or engine idling. On the other hand, the model can easily identify sounds from the car horn, children playing, dog bark, gunshot and street music classes giving an accuracy superior to 90%.

4.4.3 ESC Datasets - Pre-trained vs. No Pre-trained

This section presents a similar study to what was done for the UrbanSound8K but now for the ESC datasets. Therefore, Table 4.7, Table 4.8 and Table 4.9 show the results for the ResNet, DenseNet and Inception model, respectively, with the use of pre-trained model weights and models trained from scratch and the number of the epoch that gave the best results for the ESC-10 and ESC-50 dataset.

Unlike what was observed for the UrbanSound8K dataset, in the ESC-50 dataset, for ResNet, the optimizer that provided the best results was Adam, for DenseNet was AdamaW and for Inception, Adam continued to be the most beneficial optimization function. Regarding the best performing model for ESC-50, it is not as evident but ResNet got 4 out of 6 metrics better than the DenseNet model with a maximum difference being

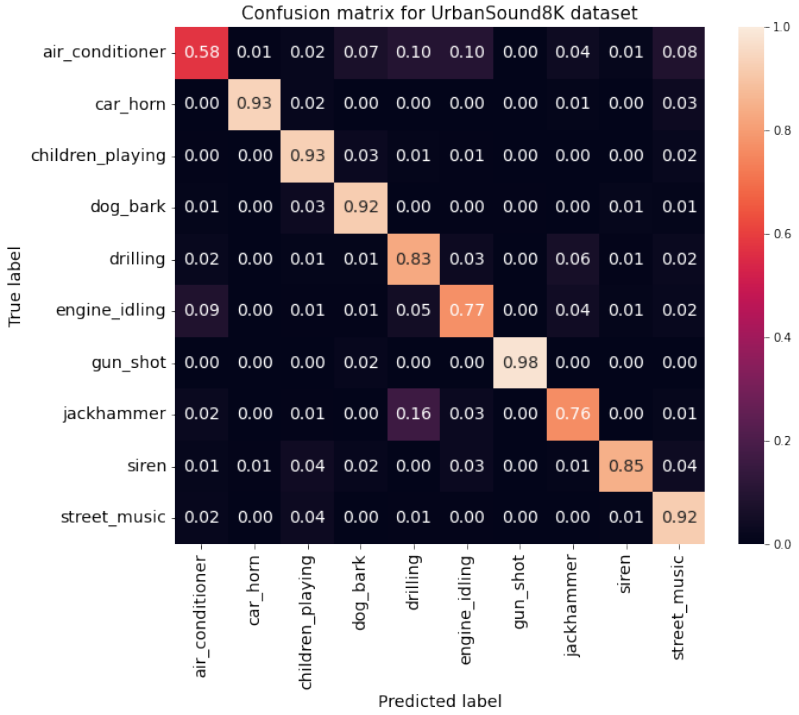


FIGURE 4.6: Confusion matrix for the UrbanSound8K dataset.

TABLE 4.7: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for ResNet model for the various optimizers.

ESC-50 dataset													
Model ResNet (Pre-trained)							**Model ResNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (66)	0.159	0.788	0.159	0.140	0.176	0.159	SGD (64)	0.039	0.630	0.039	0.025	0.024	0.039
Adam (66)	**0.882**	**0.996**	**0.882**	0.877	**0.896**	**0.882**	Adam (62)	**0.683**	0.978	**0.683**	**0.673**	**0.702**	**0.683**
Adamax (64)	0.860	0.995	0.860	0.855	0.874	0.860	Adamax (65)	0.599	0.969	0.599	0.587	0.617	0.599
AdamW (60)	0.881	**0.996**	0.881	**0.878**	0.893	0.881	AdamW (64)	0.669	**0.979**	0.669	0.658	0.681	0.669
Adadelta (65)	0.048	0.644	0.048	0.028	0.031	0.048	Adadelta (63)	0.034	0.614	0.034	0.015	0.014	0.034
Adagrad (63)	0.790	0.992	0.790	0.785	0.812	0.790	Adagrad (64)	0.243	0.865	0.243	0.205	0.211	0.243
ESC-10 dataset													
Model ResNet (Pre-trained)							**Model ResNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (65)	0.410	0.826	0.410	0.392	0.431	0.410	SGD (61)	0.205	0.715	0.205	0.186	0.207	0.205
Adam (58)	0.930	0.997	0.930	0.928	0.939	0.930	Adam (62)	**0.838**	**0.974**	**0.838**	**0.836**	**0.858**	**0.838**
Adamax (61)	0.935	0.996	0.935	0.934	0.946	0.935	Adamax (58)	0.785	0.973	0.785	0.779	0.822	0.785
AdamW (60)	**0.940**	**0.998**	**0.940**	**0.938**	**0.948**	**0.940**	AdamW (64)	0.833	0.971	0.833	0.832	0.855	0.833
Adadelta (67)	0.150	0.677	0.150	0.095	0.091	0.150	Adadelta (57)	0.113	0.610	0.113	0.035	0.043	0.113
Adagrad (59)	0.918	0.993	0.918	0.916	0.927	0.918	Adagrad (56)	0.653	0.938	0.653	0.626	0.645	0.653

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

of 0.2 pp, then, for the macro F1-score and area under the receiver operating characteristic (ROC) curve (AUC) metric, ResNet got 0.1 pp worst result than DenseNet, so overall, ResNet was superior. Inception got the worst results for all metrics compared to the other models. For the ESC-10 dataset, regarding the optimization function that provided the

TABLE 4.8: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for DenseNet model for the various optimizers.

ESC-50 dataset													
Model DenseNet (Pre-trained)							**Model DenseNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (63)	0.130	0.766	0.130	0.115	0.131	0.130	SGD (63)	0.078	0.715	0.078	0.043	0.052	0.078
Adam (66)	0.874	0.996	0.874	0.872	0.890	0.874	Adam (66)	**0.761**	**0.987**	**0.761**	**0.754**	**0.780**	**0.761**
Adamax (67)	0.847	0.994	0.847	0.844	0.861	0.847	Adamax (64)	0.713	0.984	0.713	0.705	0.734	0.713
AdamW (57)	**0.880**	**0.997**	**0.880**	**0.878**	**0.895**	**0.880**	AdamW (67)	0.749	**0.987**	0.749	0.741	0.764	0.749
Adadelta (68)	0.035	0.600	0.035	0.022	0.028	0.035	Adadelta (66)	0.043	0.651	0.043	0.019	0.022	0.043
Adagrad (67)	0.758	0.988	0.758	0.747	0.773	0.758	Adagrad (62)	0.400	0.923	0.400	0.367	0.401	0.400
ESC-10 dataset													
Model DenseNet (Pre-trained)							**Model DenseNet (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (65)	0.395	0.806	0.395	0.374	0.413	0.395	SGD (59)	0.300	0.816	0.300	0.258	0.307	0.300
Adam (58)	**0.938**	**0.997**	**0.938**	**0.936**	**0.945**	**0.938**	Adam (63)	0.888	0.986	0.888	0.883	0.905	0.888
Adamax (59)	0.918	0.995	0.918	0.917	0.930	0.918	Adamax (63)	0.868	0.985	0.868	0.863	0.887	0.868
AdamW (57)	0.930	**0.997**	0.930	0.930	0.938	0.930	AdamW (64)	**0.898**	**0.989**	**0.898**	**0.893**	**0.915**	**0.898**
Adadelta (61)	0.170	0.606	0.170	0.141	0.218	0.170	Adadelta (62)	0.175	0.678	0.175	0.108	0.131	0.175
Adagrad (62)	0.903	0.994	0.903	0.902	0.912	0.903	Adagrad (64)	0.818	0.979	0.818	0.810	0.845	0.818

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.9: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the Inception model for the various optimizers.

ESC-50 dataset													
Model Inception (Pre-trained)							**Model Inception (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (62)	0.047	0.651	0.047	0.031	0.042	0.047	SGD (57)	0.037	0.602	0.037	0.017	0.019	0.037
Adam (65)	**0.821**	**0.990**	**0.821**	0.816	**0.841**	**0.821**	Adam (66)	**0.630**	**0.973**	**0.630**	**0.620**	**0.651**	**0.630**
Adamax (65)	0.778	0.986	0.778	0.772	0.798	0.778	Adamax (63)	0.479	0.951	0.479	0.458	0.476	0.479
AdamW (66)	0.820	0.989	0.820	**0.817**	0.833	0.820	AdamW (68)	0.619	**0.973**	0.619	0.608	0.639	0.619
Adadelta (58)	0.032	0.559	0.032	0.018	0.024	0.032	Adadelta (56)	0.023	0.528	0.023	0.008	0.007	0.023
Adagrad (64)	0.501	0.952	0.501	0.472	0.525	0.501	Adagrad (60)	0.187	0.820	0.187	0.144	0.162	0.187
ESC-10 dataset													
Model Inception (Pre-trained)							**Model Inception (No pre-trained)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
SGD (58)	0.193	0.676	0.193	0.159	0.225	0.193	SGD (58)	0.133	0.631	0.133	0.090	0.110	0.133
Adam (55)	**0.938**	0.993	**0.938**	**0.937**	**0.944**	**0.938**	Adam (61)	**0.830**	**0.981**	**0.830**	**0.824**	**0.849**	**0.830**
Adamax (62)	0.905	**0.994**	0.905	0.905	0.920	0.905	Adamax (65)	0.760	0.971	0.760	0.754	0.779	0.760
AdamW (58)	0.905	0.992	0.905	0.904	0.918	0.905	AdamW (56)	0.825	0.978	0.825	0.821	0.843	0.825
Adadelta (56)	0.123	0.549	0.123	0.107	0.155	0.123	Adadelta (42)	0.135	0.566	0.135	0.086	0.090	0.135
Adagrad (56)	0.830	0.984	0.830	0.831	0.854	0.830	Adagrad (65)	0.573	0.915	0.573	0.555	0.618	0.573

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

best results for each model, the results lead to the conclusion that for DenseNet and Inception, Adam is the preferable optimization function like what was observed for the UrbanSound8K dataset, however, for the ResNet model, AdamW showed to be the best optimization function and also, gave the best results out of all models for this dataset. Inception was again the model that considering all metrics' values, gave the worst results.

Figures 4.7 and 4.8 show the graphical representation of the accuracy and loss curves for the different models with the most advantageous optimizer for each model trained for the ESC-50 and ESC-10 dataset, respectively.

In this case, all models converged after the 30^{th} epoch. Regarding the accuracy curves,

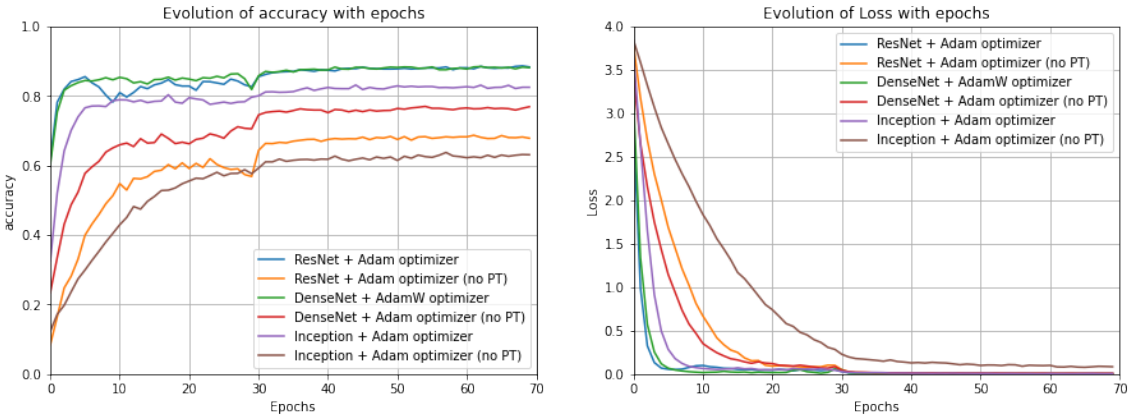


FIGURE 4.7: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model for the ESC-50 dataset. (no PT - no pre-trained).

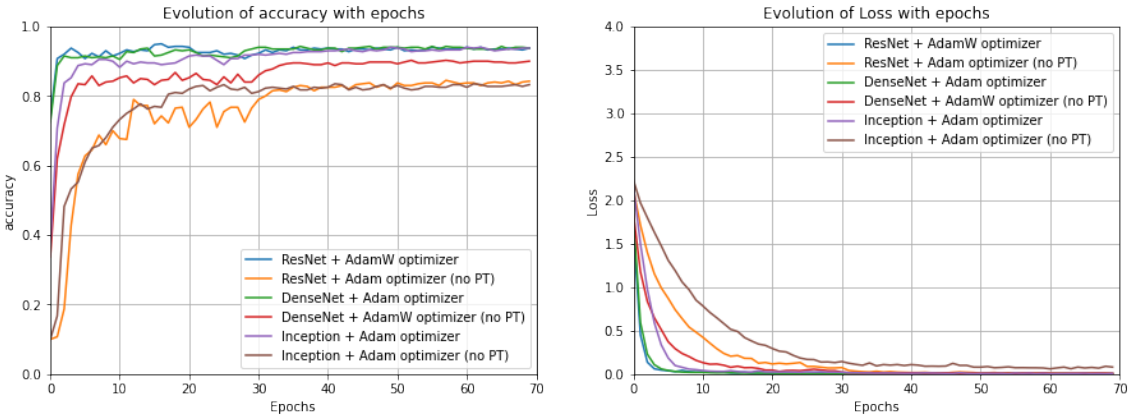


FIGURE 4.8: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model for the ESC-10 dataset. (no PT - no pre-trained).

all models showed similar behaviour. Still, no pre-trained models showed lower accuracy because they were trained from scratch; the initial accuracy is inferior. They cannot get as high results as pre-trained models due to the lack of learned filters to extract meaningful features. Also, the superiority of the DenseNet models is evident for the no pre-trained models and the inferiority of the Inception model for the no pre-trained models with the ESC-50 dataset.

4.4.4 ESC Datasets - Data Augmentation

This section also explored the influence of data augmentation using the ESC datasets. The results are presented in the following tables, Table 4.10, Table 4.11, Table 4.12 for the

different models with and without data augmentation for all the folds.

TABLE 4.10: Results for the 5 folds on ESC-50 and ESC-10 datasets for ResNet model with and without data augmentation.

ESC-50 dataset													
ResNet: (data aug: no; optimizer: Adam (66 epochs))							**ResNet: (data aug: yes; optimizer: Adam (62 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (69)	0.893	0.996	0.893	0.891	0.905	0.893	1 (68)	0.895	0.997	0.895	0.893	0.910	0.895
2 (59)	0.888	0.996	0.888	0.878	0.900	0.888	2 (63)	0.885	0.997	0.885	0.880	0.896	0.885
3 (69)	0.875	0.996	0.875	0.872	0.894	0.875	3 (56)	0.893	0.995	0.893	0.892	0.902	0.893
4 (66)	**0.898**	**0.997**	**0.898**	**0.895**	**0.907**	**0.898**	4 (69)	**0.933**	**0.999**	**0.933**	**0.932**	**0.939**	**0.933**
5 (66)	0.855	0.994	0.855	0.850	0.873	0.855	5 (56)	0.900	0.992	0.900	0.901	0.913	0.900
Average	0.882	0.996	0.882	0.877	0.896	0.882	Average	0.901	0.996	0.901	0.899	0.912	0.901
ESC-10 dataset													
ResNet: (data aug: no; optimizer: AdamW (60 epochs))							**ResNet: (data aug: yes; optimizer: AdamW (65 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (59)	0.963	0.999	0.963	0.962	0.966	0.963	1 (63)	**0.963**	**0.999**	**0.963**	**0.962**	**0.966**	**0.963**
2 (65)	0.925	0.999	0.925	0.925	0.935	0.925	2 (64)	0.938	**0.999**	0.938	0.937	0.941	0.938
3 (62)	0.888	0.993	0.888	0.880	0.905	0.888	3 (68)	0.888	0.996	0.888	0.871	0.883	0.888
4 (56)	**0.988**	**1.000**	**0.988**	**0.987**	**0.989**	**0.988**	4 (62)	0.950	0.996	0.950	0.950	0.954	0.950
5 (60)	0.938	0.999	0.938	0.937	0.946	0.938	5 (67)	0.950	0.997	0.950	0.950	0.952	0.950
Average	0.940	0.998	0.940	0.938	0.948	0.940	Average	0.938	0.997	0.938	0.934	0.939	0.938

data aug - data augmentation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 4.11: Results for the 5 folds on ESC-50 and ESC-10 datasets for DenseNet model with and without data augmentation.

ESC-50 dataset													
DenseNet: (data aug: no; opt: AdamW (57 epochs))							**DenseNet: (data aug: yes; opt: AdamW (66 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (69)	0.895	0.998	0.895	0.894	0.905	0.895	1 (68)	**0.918**	**0.999**	**0.918**	**0.915**	**0.927**	**0.918**
2 (27)	0.843	0.995	0.843	0.839	0.865	0.843	2 (65)	0.895	0.997	0.895	0.892	0.906	0.895
3 (69)	0.875	0.997	0.875	0.874	0.888	0.875	3 (66)	0.893	0.997	0.893	0.890	0.897	0.893
4 (66)	**0.915**	**0.999**	**0.915**	**0.913**	**0.930**	**0.915**	4 (68)	0.915	0.998	0.915	0.912	0.924	0.915
5 (56)	0.870	0.996	0.870	0.869	0.885	0.870	5 (63)	0.885	0.998	0.885	0.881	0.900	0.885
Average	0.880	0.997	0.880	0.878	0.895	0.880	Average	0.901	0.998	0.901	0.898	0.911	0.901
ESC-10 dataset													
DenseNet: (data aug: no; opt: Adam (58 epochs))							**DenseNet: (data aug: yes; opt: Adam (60 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (69)	0.950	0.998	0.950	0.950	0.960	0.950	1 (67)	**0.975**	**0.999**	**0.975**	**0.975**	**0.980**	**0.975**
2 (45)	0.938	0.996	0.938	0.938	0.946	0.938	2 (59)	0.913	0.994	0.913	0.911	0.929	0.913
3 (60)	0.888	0.997	0.888	0.882	0.902	0.888	3 (65)	0.963	**0.999**	0.963	0.962	0.967	0.963
4 (64)	**0.963**	**1.000**	**0.963**	**0.962**	**0.965**	**0.963**	4 (54)	**0.975**	**0.999**	**0.975**	**0.975**	0.978	**0.975**
5 (54)	0.950	0.996	0.950	0.949	0.954	0.950	5 (56)	0.925	0.998	0.925	0.922	0.934	0.925
Average	0.938	0.997	0.938	0.936	0.945	0.938	Average	0.950	0.998	0.950	0.949	0.958	0.950

data aug - data augmentation; opt - optimizer; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Analysing the results tables, unlike observed for the UrbanSound8K dataset experiments, for the ESC datasets, the employment of data augmentation was beneficial in all cases except for the ResNet model trained with the ESC-10 dataset. Thus, for ESC-50, ResNet and DenseNet models got around a 2 pp increase and around 4 pp for the Inception model, in most metrics. For the ESC-10 dataset's results, ResNet got very similar results for both versions. However, the no augmentation data training allowed slightly

TABLE 4.12: Results for the 5 folds on ESC-50 and ESC-10 datasets for Inception model with and without data augmentation.

ESC-50 dataset													
Inception: (data aug: no; optimizer: Adam (65 epochs))							**Inception: (data aug: yes; optimizer: Adam (65 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (62)	0.835	0.989	0.835	0.833	0.849	0.835	1 (68)	0.878	0.992	0.878	0.877	0.893	0.878
2 (65)	0.798	0.987	0.798	0.789	0.829	0.798	2 (58)	0.863	0.993	0.863	0.858	0.875	0.863
3 (67)	0.818	0.989	0.818	0.812	0.835	0.818	3 (69)	0.855	0.991	0.855	0.852	0.867	0.855
4 (69)	**0.875**	**0.994**	**0.875**	**0.873**	**0.890**	**0.875**	4 (63)	**0.903**	**0.996**	**0.903**	**0.902**	**0.913**	**0.903**
5 (63)	0.778	0.988	0.778	0.773	0.804	0.778	5 (65)	0.820	0.989	0.820	0.809	0.840	0.820
Average	0.821	0.990	0.821	0.816	0.841	0.821	Average	0.864	0.992	0.864	0.860	0.877	0.864
ESC-10 dataset													
Inception: (data aug: no; optimizer: Adam (55 epochs))							**Inception: (data aug: yes; optimizer: Adam (62 epochs))**						
Folds	acc	AUC	micro f1score	macro f1score	prec	recall	Folds	acc	AUC	micro f1score	macro f1score	prec	recall
1 (64)	0.925	**0.997**	0.925	0.925	0.933	0.925	1 (54)	0.950	**0.999**	0.950	0.950	0.955	0.950
2 (46)	0.925	0.991	0.925	0.924	0.933	0.925	2 (63)	0.925	0.998	0.925	0.924	0.940	0.925
3 (54)	0.938	0.987	0.938	0.937	0.943	0.938	3 (65)	0.938	0.998	0.938	0.933	0.944	0.938
4 (49)	**0.950**	**0.997**	**0.950**	**0.950**	0.954	**0.950**	4 (68)	**0.975**	**0.999**	**0.975**	**0.975**	**0.978**	**0.975**
5 (62)	**0.950**	0.991	**0.950**	0.949	**0.956**	**0.950**	5 (59)	0.913	0.995	0.913	0.914	0.923	0.913
Average	0.938	0.993	0.938	0.937	0.944	0.938	Average	0.940	0.998	0.940	0.939	0.948	0.940

data aug - data augmentation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

better results, with the maximum difference being 0.9 pp, for DenseNet and Inception, data augmentation was beneficial, giving an improvement of around 1 pp and 0.2 pp, in most metrics, respectively. This can be due to strong data transformation that approximated one class to another among the feature distribution or dispersed the data representation among different class boundaries.

Concerning the easiest folder, the models trained with no augmented data have all given better results for the 4^{th} fold. However, the same was not observed for all the models trained with augmented data. Even though 3 models still had the best performance using fold 4, others identified fold 1 as the easiest.

Figures 4.9 and Figure 4.10 show that all models have converged around the 30^{th} epoch and for ESC-50 dataset, all models trained with data augmentation gave better results and the Inception models show the worst behaviours. For ESC-10, even though it is difficult to distinguish between the curves, DenseNet with data augmentation shows a slightly better result than the others.

Finally, Figure 4.11 and Figure 4.12 show the confusion matrices using the best model for each dataset. So, for the ESC-50 dataset, it was used the ResNet model and for the ESC-10 dataset, the DenseNet model, both using Adam as optimizer and data augmentation.

Analysing the confusion matrices is possible to conclude that for the ESC-50 dataset, there are five classes with an accuracy inferior to 80% which are water drops, laughing, drinking sipping, helicopter and fireworks, being the worst value for drinking sipping,

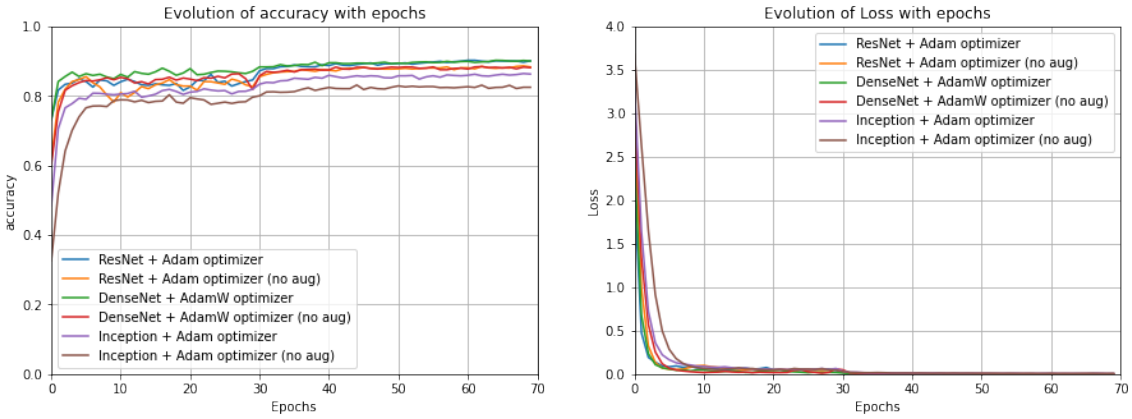


FIGURE 4.9: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model for the ESC-50 dataset (no aug - no augmentation).

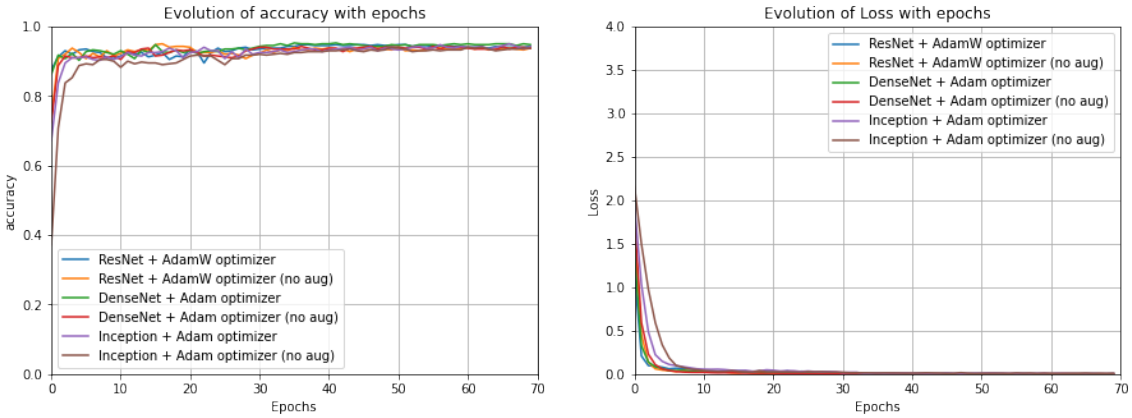


FIGURE 4.10: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the models with the optimizer that allowed the best results for each model for the ESC-10 dataset (no aug - no augmentation).

which is mainly misclassified as keyboard typing and pig. Nonetheless, there are 33 classes with an accuracy equal to or superior to 90%, and three of them got an accuracy score of 100%, which were the class of toilet flush, clock alarm and church bells. Regarding the confusion matrix for the ESC-10 dataset, the dog class is the class with the lowest accuracy score of 82%, which was mostly misclassified as sneezing. On the other hand, all the other classes had an accuracy superior to 90%, with the best results being for rooster, clock tick and chainsaw classes with an accuracy of 100%.

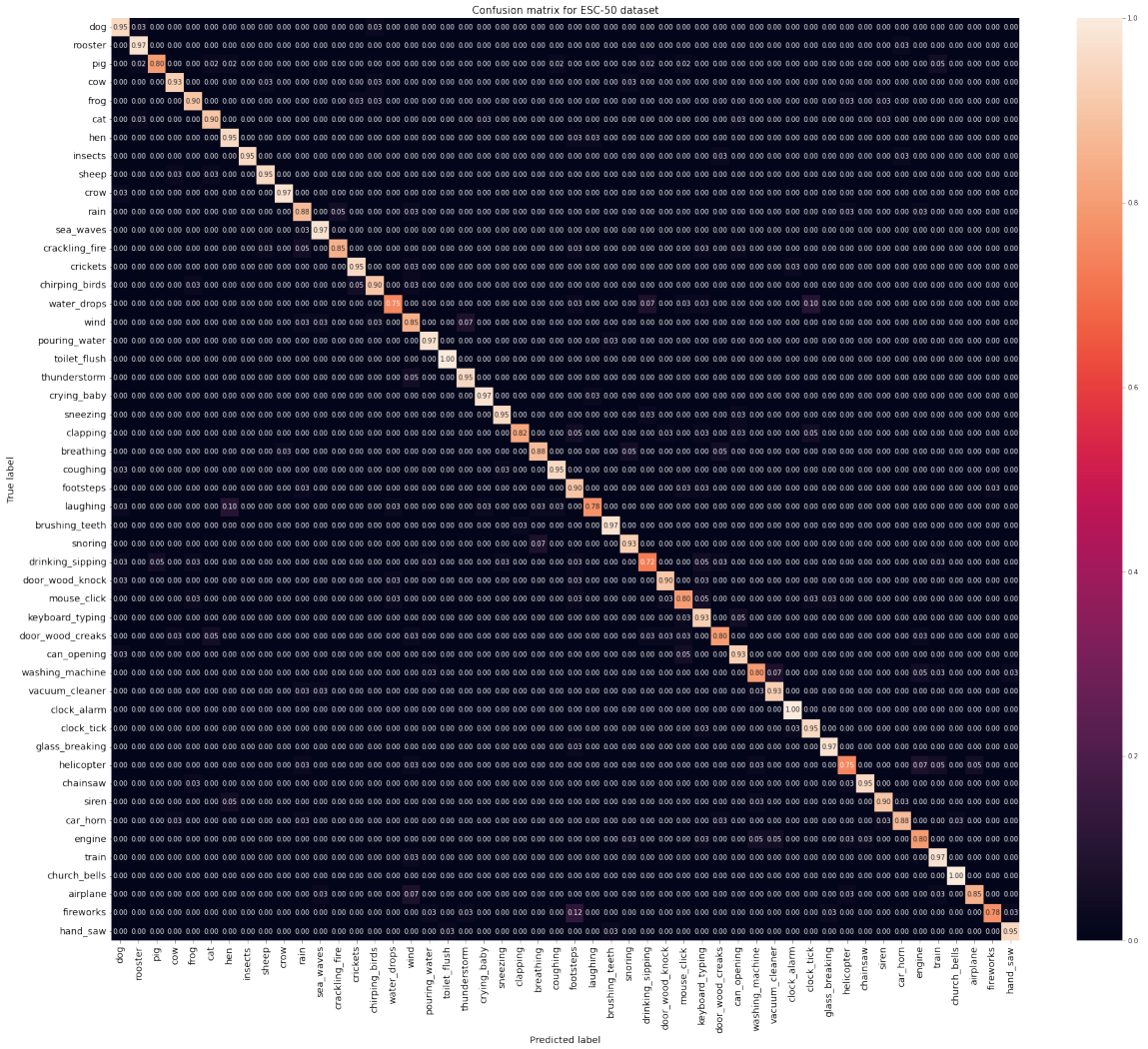


FIGURE 4.11: Confusion matrix for the ESC-50 dataset.

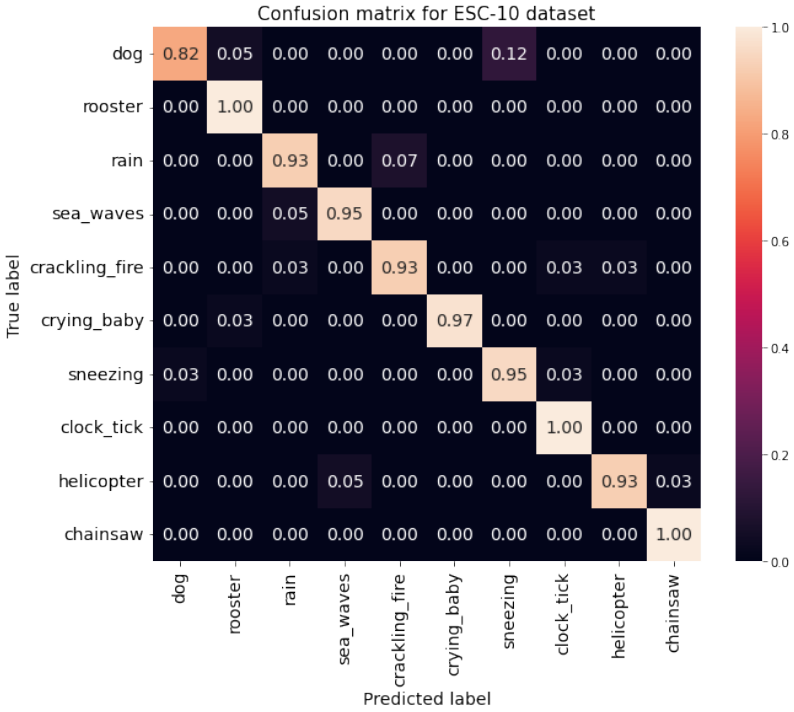


FIGURE 4.12: Confusion matrix for the ESC-10 dataset.

4.5 Conclusion

In conclusion, Inception is the model that exhibits the worst results in most situations, and DenseNet is more capable of giving better results without pre-training by demonstrating that behaviour for all datasets.

Regarding the experiences, the no pre-trained models show the worst behaviour regardless of the dataset, giving around 10 pp for UrbanSound8K, 17 pp for ESC-50 and 8 pp for ESC-10 worst results than the corresponding pre-trained models. Then, considering the results obtained for the models trained with data augmentation, UrbanSound8K got unexpected results by giving worse performances than the no data augmented models with differences between 0.2 pp and 3 pp. On the other hand, the ESC datasets gave the expected results by having their performance improved except for the ResNet model for the ESC-10 dataset, which got around 0.3 pp worst; however, for the others, the benefits range between 1 and 4 pp. Thus, it can be concluded that pre-training has a huge influence on all models' performance regardless of the dataset. The data augmentation techniques depend on the dataset and the model's architecture. However, they can enhance the performance in some situations, even if the impact isn't so pronounced.

Observing the cross-validation results, there is no consensus about the most accessible folder depending on the model, if it is pre-trained and if data augmentation techniques were employed, showing the different capacities of the models to represent the sounds depending on the input.

Concerning the optimizers, Adam was the optimization function capable of giving the best performance for all datasets and, in most situations, for the different models. For ESC-50, the DenseNet model with AdamW optimizer gave very similar results to the ResNet model with Adam optimizer making it challenging to affirm which one is the best.

Compared with the baseline models, there was a huge performance boost of 19.9 pp for UrbanSound8K, 46.4 pp for ESC-50 and 17.6 pp for ESC-10, on average, showing the superiority of the end-to-end pre-trained models.

Chapter 5

Transformers

This chapter explores the model architecture Transformer. Several experiences were done using different optimization functions, data augmentation techniques, batch sizes, and temporal and frequency strides and explored the use of no pre-training, pre-training with ImageNet and pre-training with ImageNet and AudioSet. These were evaluated using different metrics for the UrbanSound8K and ESC datasets.

5.1 Transformer

It is a transduction model that relies entirely on an attention mechanism to compute representations of its input and output, proposed by Vaswani et al. [39].

The model architecture is constituted of an **encoder** which maps an input sequence of symbol representations to a sequence of continuous representations and a **decoder** that generates an output sequence of symbols one at a time. At each step, the model is auto-regressive, so it uses the previously generated symbols as additional input when generating the next one.

In Figure 5.1, on the left side is represented the encoder architecture and on the right is the decoder architecture described below.

The encoder is composed of a stack of identical layers, each one of them has two sub-layers: a **multi-head self-attention mechanism** and a **position-wise fully connected feed-forward network**, and around each of the two sub-layers, there is a residual connection followed by a normalization layer.

The decoder is composed of a stack of identical layers with three sub-layers which are the two mentioned previously in the encoder architecture, and a third one which is

a **multi-head attention over the output of the encoder stack**. Furthermore, there is a modification in the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions, ensuring, in combination with the fact that the output embeddings are offset by one position, that the predictions only depend on the known outputs of previous positions. Finally, similarly to the encoder, it also employs residual connections around each sub-layers, followed by layer normalization.

A brief description of each layer in the model is then presented:

- **Embeddings**: are used to convert the input tokens and output tokens to vectors of a certain dimension;
- **Positional Encoding**: it is the relative or absolute position of the tokens in the sequence that allows the model to use the order of the sequence;
- **Multi-head attention**: performs an attention function parallel to the projected versions of queries, keys and values, allowing the model to jointly attend to information from different representation subspaces at different positions;

  The Transformer uses this layer in three different ways:

  - Self-attention layer in encoder: all keys, values and queries come from the output of the previous layer in the encoder, which allows each position in the encoder to attend to all positions in the previous layer of the encoder;
  - Self-attention layer in decoder: allows each position in the decoder to attend to all positions in the decoder up to and including that position. Then, to preserve the auto-regressive property, this is implemented inside of scaled dot-product attention by masking out all values in the input of the Softmax corresponding to illegal connections;
  - Layer in the decoder over the output of the encoder stack layer: queries come from the previous decoder layer and the memory keys and values from the output of the encoder, which allows the decoder in every position to attend over all positions in the input sequence;

- **Position-wise Feed-Forward Networks**: consists of two linear transformations with a ReLU activation in between, applied to each position separately and identically;
- **Linear and Softmax**: the learned linear transformation and the Softmax function are used to convert the decoder output to predicted next-token probabilities.

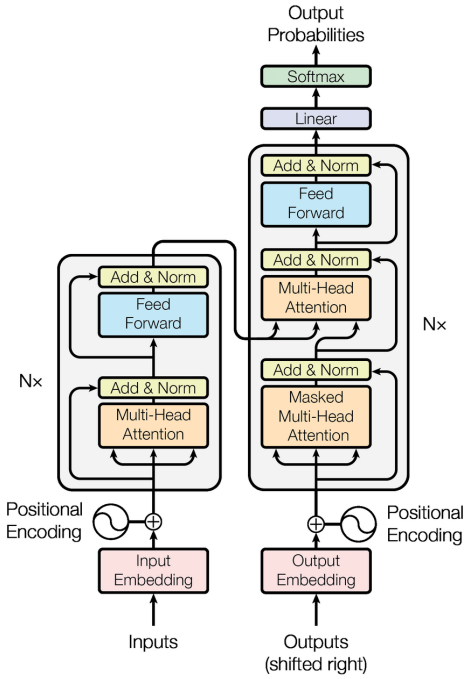


FIGURE 5.1: Transformer architecture (Vaswani et al. [39]).

The employed architecture is based on the Audio Spectrogram Transformer (AST) model introduced by Gong et al. [11]. Its architecture consists of a patch embedding layer which converts the input spectrogram into a sequence of patches and flattens it into a one-dimensional (1D) patch. Then, a trainable positional embedding is added to each patch embedding to capture the input order information and the temporal order of the patch sequence. Also, a classification token is appended at the beginning of the sequence. The resulting sequence serves as input for the standard Transformer's encoder part explained in this section's beginning. Finally, the Transformer encoder's output of the classification token serves as the audio spectrogram representation, which a linear layer will map with sigmoid activation to labels for classification. Figure 5.2 illustrates the models' architecture.

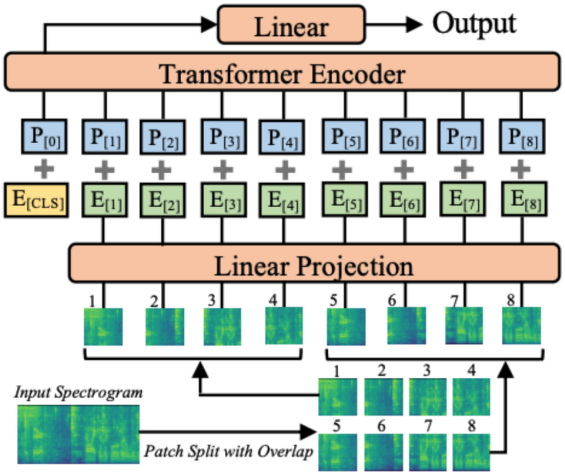


FIGURE 5.2: AST architecture (Gong et al. [11]).

5.2 Experiments and Results

This section presents the results of the different experiences using the Transformer architecture. Starting with the analyses of the effect of pre-training, in particular, the difference between just distilling knowledge from an image domain model or two models: one from the image domain and the other from the audio domain, next, the influence of changing the batch size is explored and finally, the differences that the various data augmentation techniques can provoke.

5.2.1 ESC Datasets - No Pre-trained vs. Pre-trained

As concluded in Section 4, the employment of pre-training gives a faster model convergence resulting in a quicker training process and a huge boost to the models' performance. Therefore, this section presents the results for the ESC datasets with the different optimization functions and with no pre-training, pre-training with ImageNet and pre-training with ImageNet and AudioSet. Tables 5.1 and 5.2 summarize the obtained results.

TABLE 5.1: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the no pre-trained model for the various optimizers.

ESC-50 dataset - No pre-trained							**ESC-10 dataset - No pre-trained**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (23)	0.058	0.710	0.058	0.026	0.083	0.058	Adadelta (24)	0.293	0.808	0.293	0.202	0.380	0.293
Adagrad (19)	0.252	0.879	0.252	0.220	0.277	0.252	Adagrad (24)	0.483	0.916	0.483	0.458	**0.620**	0.483
Adam (24)	0.387	0.924	0.387	0.362	0.418	0.387	Adam (24)	0.458	0.898	0.458	0.428	0.528	0.458
Adamax (23)	0.342	0.911	0.342	0.315	0.374	0.342	Adamax (22)	0.468	0.901	0.468	0.433	0.570	0.468
AdamW (23)	**0.439**	**0.936**	**0.439**	**0.424**	**0.468**	**0.439**	AdamW (23)	**0.528**	**0.918**	**0.528**	**0.507**	0.617	**0.528**
SGD (23)	0.073	0.734	0.073	0.034	0.091	0.073	SGD (23)	0.363	0.847	0.363	0.300	0.448	0.363

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

TABLE 5.2: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the pre-trained models for the various optimizers.

ESC-50 dataset													
ImageNet pre-trained							**ImageNet and AudioSet pre-trained**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (22)	0.033	0.569	0.033	0.022	0.053	0.033	Adadelta (15)	0.025	0.505	0.025	0.012	0.043	0.025
Adagrad (19)	0.843	0.993	0.843	0.838	0.880	0.843	Adagrad (23)	0.814	0.991	0.814	0.806	0.859	0.814
Adam (16)	0.884	0.995	0.884	0.881	0.922	0.884	Adam (18)	0.950	**0.999**	0.950	0.948	0.977	0.950
Adamax (19)	0.885	**0.996**	0.885	0.882	0.926	0.885	Adamax (18)	0.947	0.998	0.947	0.946	0.965	0.947
AdamW (18)	**0.886**	**0.996**	**0.886**	**0.884**	**0.927**	**0.886**	AdamW (18)	**0.958**	**0.999**	**0.958**	**0.956**	**0.978**	**0.958**
SGD (23)	0.055	0.633	0.055	0.036	0.072	0.055	SGD (18)	0.020	0.517	0.020	0.013	0.044	0.020
ESC-10 dataset													
ImageNet pre-trained							**ImageNet and AudioSet pre-trained**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (21)	0.138	0.574	0.138	0.092	0.198	0.138	Adadelta (16)	0.105	0.528	0.105	0.089	0.181	0.105
Adagrad (23)	0.923	0.996	0.923	0.920	0.975	0.923	Adagrad (24)	0.988	0.999	0.988	0.987	0.995	0.988
Adam (21)	0.930	0.997	0.930	0.927	**0.983**	0.930	Adam (20)	**0.990**	**1.000**	**0.990**	**0.990**	**0.999**	**0.990**
Adamax (22)	0.933	0.996	0.933	0.932	0.974	0.933	Adamax (23)	0.988	**1.000**	0.988	0.987	0.998	0.988
AdamW (21)	**0.938**	**0.998**	**0.938**	**0.935**	0.982	**0.938**	AdamW (23)	0.985	**1.000**	0.985	0.985	0.998	0.985
SGD (21)	0.210	0.673	0.210	0.171	0.276	0.210	SGD (19)	0.130	0.543	0.130	0.098	0.214	0.130

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Analysing the results, it can be verified that the pre-training has a big impact on the results showing an improvement of around 43 percentage points (pp), in most metrics, between the no pre-trained and the pre-trained using ImageNet, culminating in a more significant difference than what was observed for the models discussed in Section 4. These results confirm the need for Transformer to have large datasets for competitive results.

Focusing on Table 5.2 can be noticed, for both datasets, a performance improvement when the ImageNet and AudioSet pre-training are used with a difference of 7.2 pp for ESC-50 and approximately 5.3 pp for ESC-10 in 4 out of 6 metrics, which demonstrates the importance of having pre-training models of the same domain has the datasets.

Regarding the optimization function, the one that allowed the best performance was for ESC-50, AdamW and ESC-10, Adam.

Figure 5.3 and Figure 5.4 show the different accuracy and loss curves for the best result of each pre-trained configuration for both datasets. Observing the curves for both datasets is possible to reach the same conclusions that the no pre-trained model cannot give competitive results. Pre-training with both ImageNet and AudioSet shows to be the most beneficial configuration.

5.2.2 ESC Datasets - Batch Size

In this section, the results obtained with the change of the batch size to 24 and 64 for the ESC-50 and ESC-10 datasets are presented in Table 5.3.

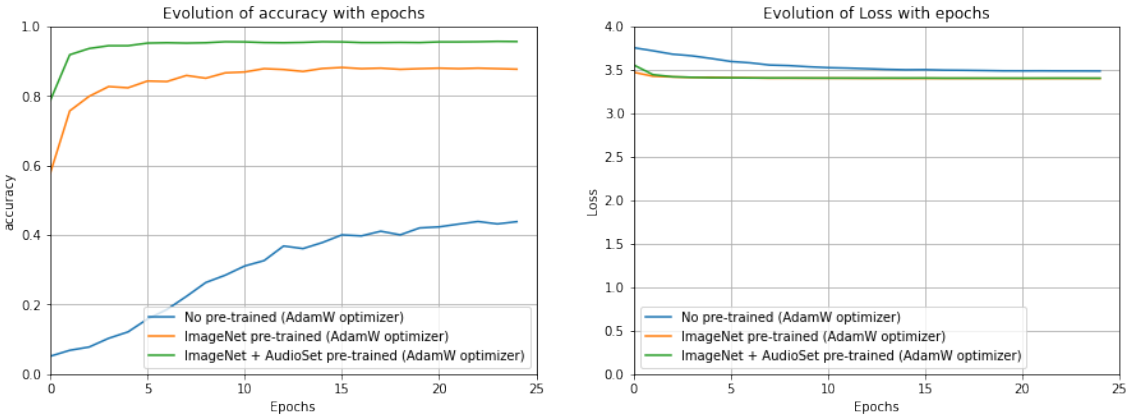


FIGURE 5.3: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different pre-train configurations with the optimizer that allowed the best results for each model for the ESC-50 dataset.

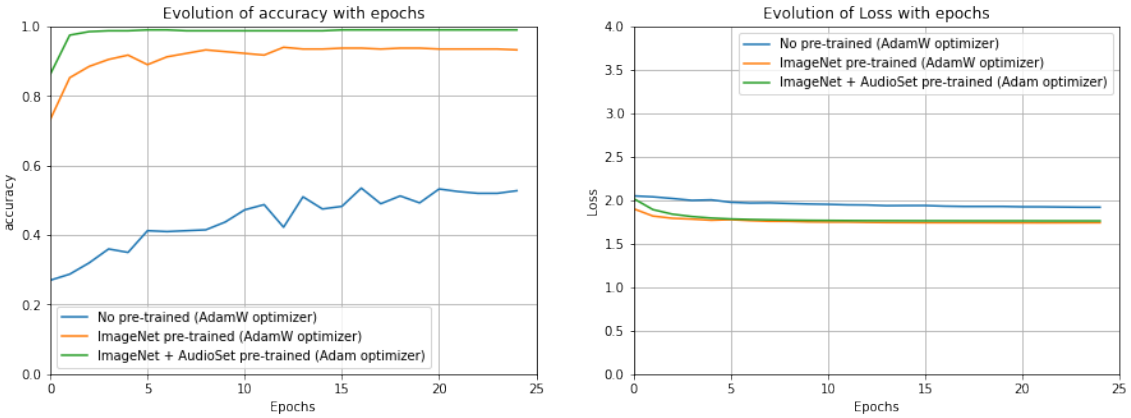


FIGURE 5.4: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different pre-train configurations with the optimizer that allowed the best results for each model for the ESC-10 dataset.

Comparing the results shown in Table 5.3 can be inferred that a batch size of 24 gives a small benefit for the ESC-50 dataset with a maximum difference of 0.2 pp when compared to the batch size of 64. However, the batch size of 48 gives better results by having a difference of 0.1 pp in 3 out of the 6 metrics compared to the batch size of 24. For the ESC-10 dataset, no differences are observed in most metrics except precision, which gives a value 0.1 pp lower than the others for the batch size of 24. Then, for the other batch sizes, there is no difference in the results between them, so it is equally good to use either batch size, but considering the memory capacity depending on the size, a smaller batch size results in a smaller GPU memory occupancy in training. Therefore, for the subsequent studies, pre-training using both models, the batch size of 48 was kept unchanged.

TABLE 5.3: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the pre-trained Transformer with a batch size of 24 and 64 for the various optimizers, respectively.

ESC-50 dataset													
Batch size: 24							**Batch size: 64**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (18)	0.024	0.541	0.024	0.016	0.051	0.024	Adadelta (25)	0.024	0.543	0.024	0.016	0.051	0.024
Adagrad (23)	0.828	0.991	0.828	0.819	0.871	0.828	Adagrad (25)	0.828	0.991	0.828	0.819	0.871	0.828
Adam (18)	**0.957**	**0.999**	**0.957**	**0.956**	**0.979**	**0.957**	Adam (16)	**0.955**	**0.999**	**0.955**	**0.954**	**0.980**	**0.955**
Adamax (18)	0.947	0.998	0.947	0.945	0.971	0.947	Adamax (22)	0.945	0.998	0.945	0.943	0.972	0.945
AdamW (15)	0.955	**0.999**	0.955	0.954	**0.979**	0.955	AdamW (25)	0.954	**0.999**	0.954	0.952	0.979	0.954
SGD (22)	0.037	0.556	0.037	0.029	0.068	0.037	SGD (23)	0.037	0.557	0.037	0.029	0.068	0.037
ESC-10 dataset													
Batch size: 24							**Batch size: 64**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (19)	0.115	0.526	0.115	0.088	0.176	0.115	Adadelta (10)	0.083	0.502	0.083	0.069	0.168	0.083
Adagrad (22)	0.985	0.998	0.985	0.985	0.992	0.985	Adagrad (24)	0.975	0.998	0.975	0.975	0.990	0.975
Adam (18)	**0.990**	**1.000**	**0.990**	**0.990**	0.998	**0.990**	Adam (21)	0.985	**1.000**	0.985	0.985	**1.000**	0.985
Adamax (23)	**0.990**	**1.000**	**0.990**	**0.990**	0.998	**0.990**	Adamax (23)	**0.990**	**1.000**	**0.990**	**0.990**	0.998	**0.990**
AdamW (21)	0.985	**1.000**	0.985	0.985	**0.999**	0.985	AdamW (23)	**0.990**	**1.000**	**0.990**	**0.990**	0.999	**0.990**
SGD (19)	0.125	0.541	0.125	0.094	0.181	0.125	SGD (15)	0.118	0.484	0.118	0.082	0.153	0.118

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Figures 5.5 and 5.6 show the accuracy and loss curves for both datasets, and it is possible to verify that in both cases, changing the batch size was not significative do to the difficulty there is in distinguishing between the curves. However, it shows that the smaller the batch size, the faster it reaches convergence.

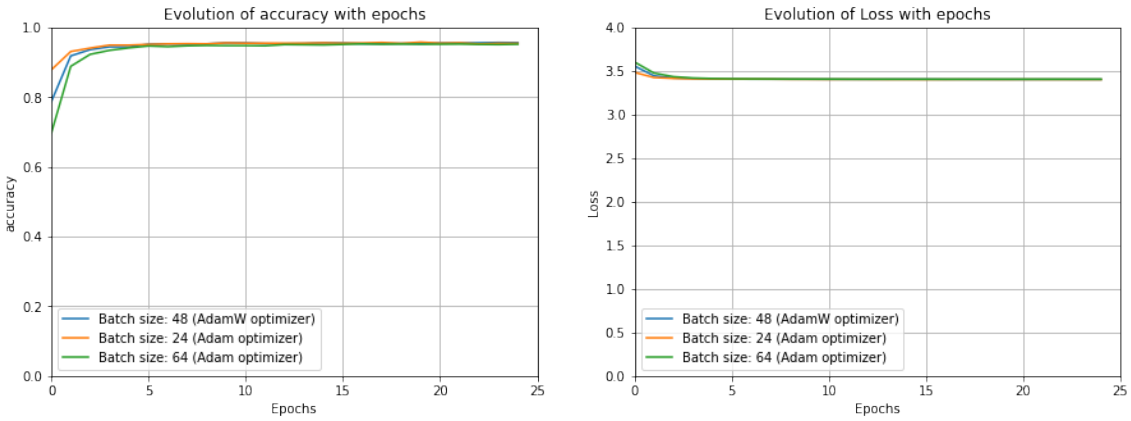


FIGURE 5.5: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different batch sizes for the optimizer that allowed the best results for each model for the ESC-50 dataset.

5.2.3 ESC Datasets - Data Augmentation Techniques

This section explored the influence of different data augmentation techniques such as SpecAugment, noise addition and Mixup. The corresponding results are shown on Table 5.4, Table 5.5 and Table 5.6.

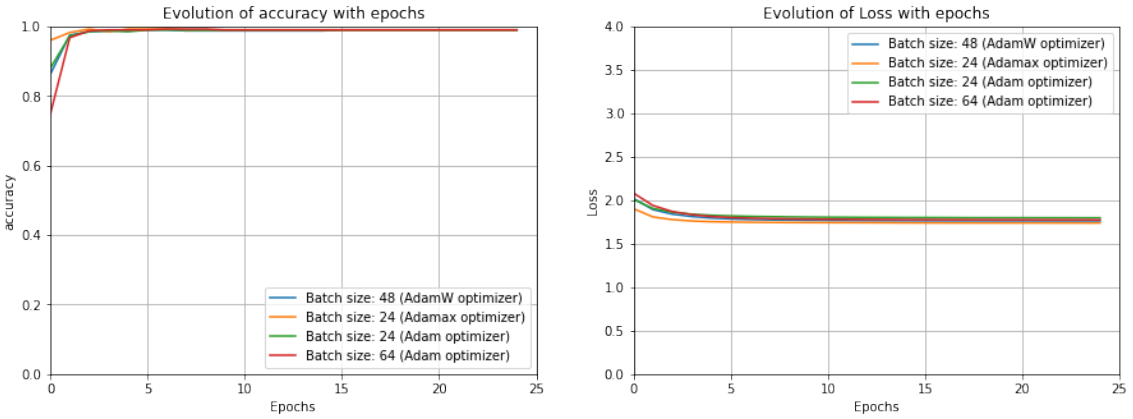


FIGURE 5.6: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different batch sizes for the optimizer that allowed the best results for each model for the ESC-10 dataset.

5.2.3.1 No Data Augmentation

This section presents the results if no data augmentation technique was applied to the model because, in the above situations, it was always used the SpecAugment technique to mask up to 48 frequency bins and 96-time frames.

TABLE 5.4: Results for the average of 5 folds results on the ESC-50 and ESC-10 datasets for the pre-trained models without using any data augmentation technique for the various optimizers.

ESC-50 dataset (data augmentation: no)							**ESC-10 dataset (data augmentation: no)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (15)	0.022	0.530	0.022	0.016	0.045	0.022	Adadelta (17)	0.073	0.470	0.073	0.049	0.142	0.073
Adagrad (23)	0.830	0.991	0.830	0.817	0.872	0.830	Adagrad (23)	0.975	0.998	0.975	0.975	0.989	0.975
Adam (15)	0.948	**0.999**	0.948	0.946	0.975	0.948	Adam (22)	0.985	**1.000**	0.985	0.985	**0.998**	0.985
Adamax (22)	0.947	**0.999**	0.947	0.945	0.964	0.947	Adamax (22)	0.985	0.999	0.985	0.985	0.996	0.985
AdamW (19)	**0.954**	**0.999**	**0.954**	**0.953**	**0.976**	**0.954**	AdamW (22)	**0.988**	**1.000**	**0.988**	**0.987**	0.997	**0.988**
SGD (17)	0.032	0.550	0.032	0.025	0.055	0.032	SGD (18)	0.128	0.563	0.128	0.106	0.213	0.128

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

As can be observed in Table 5.4, the SpecAugment provides a improvement of around 0.3 pp for ESC-50 and of 0.2 pp for ESC-10, in most metrics.

SpecAugment proved to be a beneficial data augmentation technique for both datasets. Thus, this technique was used in subsequent studies, and to improve the results further, random noise and Mixup augmentations were added.

5.2.3.2 Noise

This section considers the addition of random noise to the input waveform. Table 5.5 presents the results obtained for the ESC-50 and ESC-10 datasets.

TABLE 5.5: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the pre-trained models with noise addition for the various optimizers.

ESC-50 dataset (data augmentation: noise)							**ESC-10 dataset (data augmentation: noise)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (12)	0.023	0.538	0.023	0.017	0.051	0.023	Adadelta (13)	0.078	0.445	0.078	0.058	0.135	0.078
Adagrad (24)	0.824	0.987	0.824	0.813	0.856	0.824	Adagrad (23)	0.980	0.998	0.980	0.979	0.990	0.980
Adam (18)	**0.958**	**0.999**	**0.958**	**0.956**	0.978	**0.958**	Adam (23)	**0.988**	**1.000**	**0.988**	0.987	**0.999**	**0.988**
Adamax (19)	0.946	**0.999**	0.946	0.944	0.971	0.946	Adamax (22)	**0.988**	0.999	**0.988**	**0.988**	0.995	**0.988**
AdamW (17)	0.954	**0.999**	0.954	0.952	**0.979**	0.954	AdamW (21)	**0.988**	**1.000**	**0.988**	0.987	0.998	**0.988**
SGD (19)	0.037	0.512	0.037	0.027	0.053	0.037	SGD (16)	0.098	0.541	0.098	0.067	0.182	0.098

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Observing Table 5.5, the addition of noise showed no performance improvements, matching the best model obtained previously for ESC-50 and giving a slightly poorer execution for the ESC-10 dataset. Concerning the optimization functions, Adam gave the best results for both datasets.

5.2.3.3 Mixup

The following section explores the use of the Mixup rate by mixing the raw waveforms randomly from the dataset and the final spectrograms. The used Mixup rate was 0.5, Table 5.6 summarizes the results for the ESC datasets.

TABLE 5.6: Results for the average of 5 folds results on ESC-50 and ESC-10 datasets for the pre-trained models with a Mixup of 0.5 for the various optimizers.

ESC-50 dataset (data augmentation: Mixup: 0.5)							**ESC-10 dataset (data augmentation: Mixup: 0.5)**						
Opt. function	acc	AUC	mf1	Mf1	prec	recall	Opt. function	acc	AUC	mf1	Mf1	prec	recall
Adadelta (9)	0.026	0.520	0.026	0.019	0.045	0.026	Adadelta (12)	0.093	0.463	0.093	0.068	0.151	0.093
Adagrad (22)	0.034	0.943	0.034	0.024	0.596	0.034	Adagrad (24)	0.470	**0.988**	0.470	0.496	0.953	0.470
Adam (21)	0.719	**0.997**	0.719	0.786	**0.952**	0.719	Adam (21)	**0.890**	**0.998**	**0.890**	**0.900**	**0.992**	**0.890**
Adamax (24)	0.628	0.995	0.628	0.702	0.926	0.628	Adamax (24)	0.810	**0.998**	0.810	0.833	0.989	0.810
AdamW (20)	**0.725**	**0.997**	**0.725**	**0.788**	**0.952**	**0.725**	AdamW (23)	0.860	**0.998**	0.860	0.875	0.988	0.860
SGD (15)	0.024	0.525	0.024	0.015	0.044	0.024	SGD (15)	0.113	0.495	0.113	0.084	0.153	0.113

opt. function - optimization function; AUC - area under the receiver operating characteristic curve; mf1 - micro f1score; Mf1 - macro f1score; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

Table 5.6 shows that in both cases, the results get a lot worst with huge differences in the accuracy, micro F1-score and recall by approximately 23 pp and 10 pp and on macro F1-score around 17 pp and 9 pp for ESC-50 and ESC-10, respectively. In this study, Adam for the ESC-10 was also the optimization function that provided the best results. On the other hand, for ESC-50, AdamW has the preferable optimization function.

Lastly, Figure 5.7 and Figure 5.8 show the accuracy and loss curves concerning the different used combinations of augmentation techniques. Observing the curves for both datasets, it is difficult to distinguish between them, except for the model trained with the Mixup, which performed much worse than all the others.

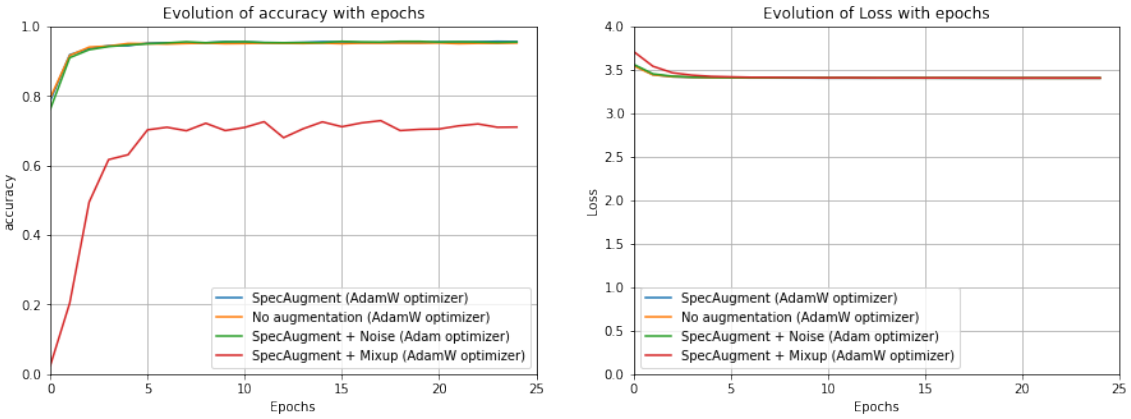


FIGURE 5.7: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different augmentation techniques for the optimizer that allowed the best results for each model for the ESC-50 dataset.

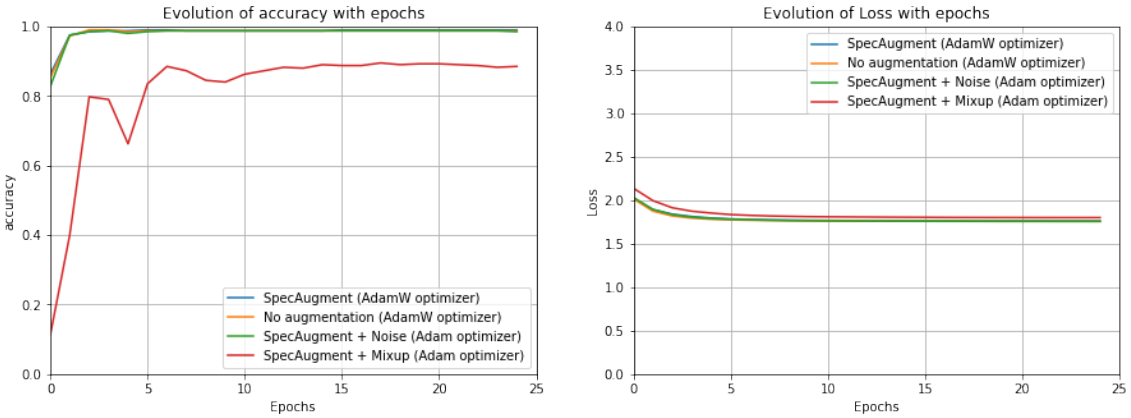


FIGURE 5.8: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer with the different augmentation techniques for the optimizer that allowed the best results for each model for the ESC-10 dataset.

The best results for both datasets were obtained when the optimization function AdamW for ESC-50 and Adam for ESC-10 were used and a Transformer model with the following configuration: pre-trained using ImageNet and AudioSet, a batch size of 48, and SpecAugment as data augmentation technique which allowed to reach an accuracy result of 95.8% for ESC-50 and 99% for ESC-10 evaluated by doing 5-cross-validation and considering the best epoch for each fold. These accuracy results indicate that some classes were misclassified to try to understand which classes the model could not distinguish. In Figure 5.9 and Figure 5.10 the confusion matrices for both datasets are presented.

Analysing the confusion matrices, it is possible to understand that for the ESC-50 dataset, there were 17 classes with an accuracy of 100%, 29 with an accuracy equal or

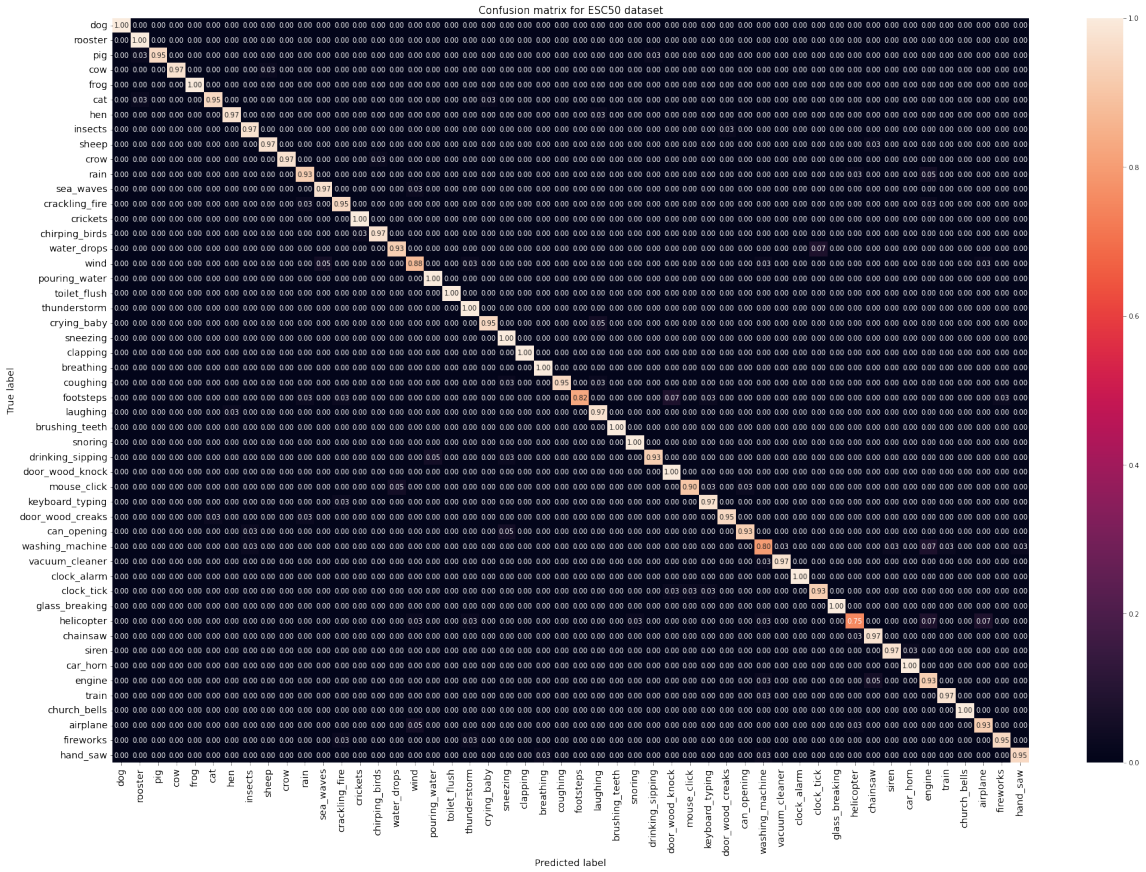


FIGURE 5.9: Confusion matrix for ESC-50 dataset.

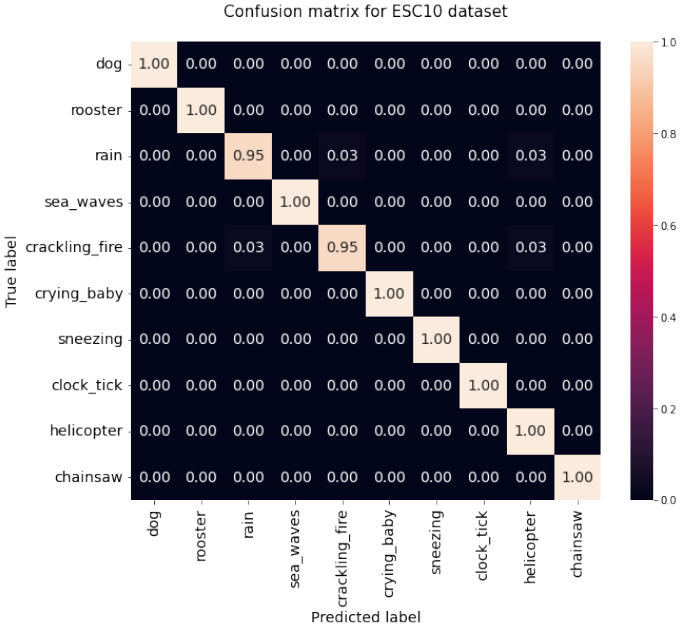


FIGURE 5.10: Confusion matrix for ESC-10 dataset.

superior to 90% and only 4 classes with an accuracy inferior to 90% with the lowest result being for the helicopter class with an accuracy of 75%. The other three most challenging classes to identify were washing machine, footsteps and wind.

Then, for the ESC-10 dataset, the only classes that did not achieve 100% were rain which was misclassified as crackling fire and helicopter, and crackling fire, which was misclassified as rain and helicopter. However, both classes achieved an accuracy of 95%.

To conclude, the model performance for the UrbanSound8K dataset was tested, giving an accuracy, micro F1-score and recall result of 89.8% with Adamax or AdamW optimization functions, as seen in Table 5.7. In Figure 5.11 is represented the accuracy and loss curves for the model using Adamax and AdamW as the optimization function due to the similar results they provide, making it difficult to infer which one is the best. Analysing the accuracy curves is possible to verify that the model trained with Adamax presented a more linear behaviour throughout the epochs than AdamW.

TABLE 5.7: Results for the average of 10 folds results on UrbanSound8K dataset for the pre-trained models and with the use of SpecAugment.

UrbanSound8K						
Opt. function	acc	AUC	micro f1score	macro f1score	prec	recall
Adadelta (24)	0.236	0.669	0.236	0.197	0.226	0.236
Adagrad (15)	0.875	0.984	0.875	0.884	0.928	0.875
Adam (7)	0.897	**0.988**	0.897	0.905	**0.939**	0.897
Adamax (9)	**0.898**	0.986	**0.898**	0.904	0.938	**0.898**
AdamW (8)	**0.898**	0.985	**0.898**	**0.906**	0.937	**0.898**
SGD (25)	0.460	0.821	0.460	0.407	0.436	0.460

opt. function - optimization function; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision. All metrics range from $[0, 1]$ (the higher, the better).

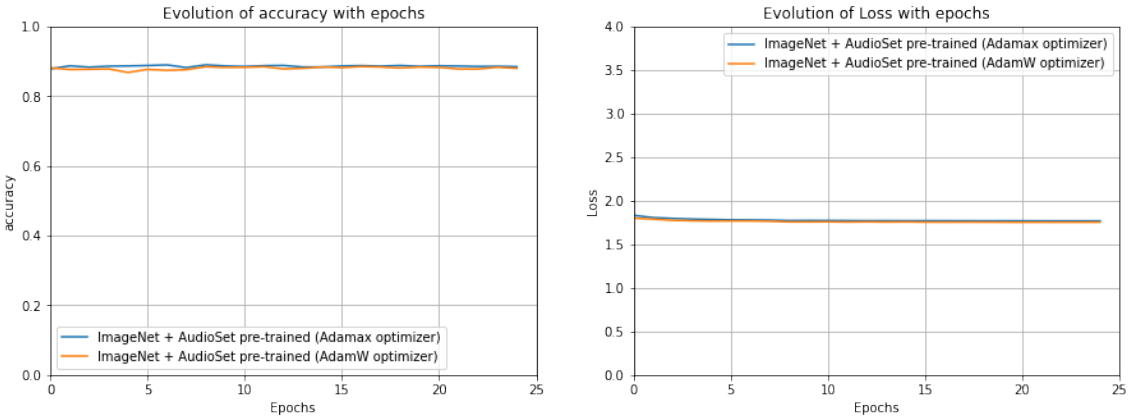


FIGURE 5.11: Graphs of the evolution of accuracy (left) and loss function (right) with epochs for the Transformer pre-trained with ImageNet and AudioSet for the optimizers that allowed the best results for the UrbanSound8K dataset.

Due to the more linear behaviour, Adamax was considered the preferable optimization function, so Figure 5.12 shows the confusion matrix for the Transformer using Adamax as an optimization function trained with the UrbanSound8K dataset, which makes it possible to identify that air conditioner, drilling, engine idling and jackhammer are the most challenging classes to distinguish having an accuracy result inferior to 90% with the worst result being for an air conditioner with 76%. However, the model can perfectly identify gunshot sounds, which was the only class not mistaken for another class, and no class was confused as being of the gunshot class.

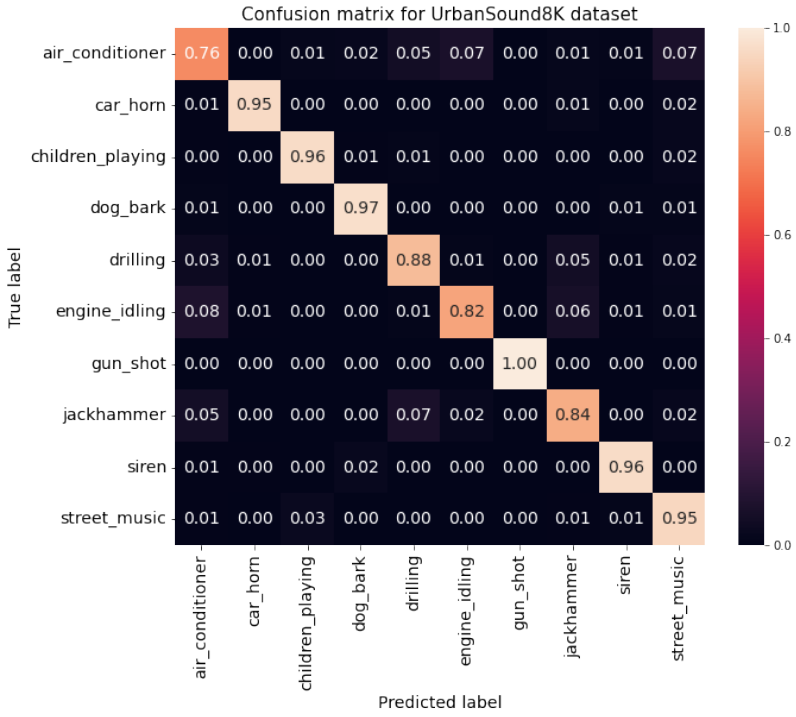


FIGURE 5.12: Confusion matrix for UrbanSound8K dataset.

5.3 Conclusion

This chapter evaluated the Transformer architecture and the influence that pre-training only from the image domain, pre-training from the image and audio domain, changing the batch size, and using different data augmentation techniques can cause depending on the optimizer and dataset.

From these experiments, it was possible to conclude that using pre-trained models can improve the models' results. Nonetheless, when using as optimization function SGD or Adadelta the results were worst, proving the non-robustness to large datasets. However,

the other optimizers' pre-training with both ImageNet and AudioSet produced better results.

Regarding the batch size, by changing the batch size, the model showed no improvements. For the ESC-10 dataset, the increase in batch size produced the same best result, so the smaller value was kept to occupy less memory.

Concerning the data augmentation techniques, using SpecAugment improved the results compared to the no augment results. However, adding more data augmentation techniques produced no benefits, especially disadvantageous with Mixup.

Compared with baseline models, the improvement was, on average, 51.4 pp for ESC-50, 21.0 pp for ESC-10 and 25.9 pp for UrbanSound8K. Also, compared with the models presented in the previous section can be observed that for all datasets, Transformer was the model that provided the best results with differences of 4.95 pp for ESC-50, 3.40 pp for ESC-10 and 6.02 pp for UrbanSound8K.

Chapter 6

Overall Discussion and Conclusions

This work presented different model architectures: the baseline models were based on a simple set of dense layers and explored the use of dropout rate, which significantly influences the model's ability to learn. If the rate is too high, the model hasn't enough information to be able to learn, and if it is too low, the model is more prone to overfit; changes in the architecture, in most cases, the addition of the extra layers was injurious; and different input features being possible to conclude that the combination of features is more advantageous. Then, the end-to-end models were evaluated, particularly the Dense Convolutional Network (DenseNet), Residual Neural Network (ResNet) and Inception, which revealed to be the least capable model out of these three models that can learn the features from an input spectrogram.

The use of pre-training based on ImageNet greatly increased the models' performance regardless of the chosen architecture. Using data augmentation techniques depending on the dataset or chosen architecture might not always be advantageous, sometimes leading to poorer results due to the dramatic modification in sound characteristics that approximates one class to another. Finally, a more recent architecture called Transformer was used, relying entirely on attention mechanisms to compute its input and output representations. This architecture showed the importance of having large datasets by showing the different model performances depending on the pre-training configuration, ranging from no pre-training to pre-training using ImageNet and pre-training using ImageNet and AudioSet.

Besides, the batch size was changed, and some combinations of different data augmentation techniques were used. The experiments done with the Transformer model confirm

the benefits of having pre-training and allowed to understand that a more significant improvement can be obtained when it is used in-task domain pre-training models, in this case, from the audio domain. The batch size is not a significant parameter for the other experiments. Once again, using data augmentation techniques depending on the combination might not be beneficial, concluding that SpecAugment alone was the best option.

Concerning the various optimization functions, these parameters dramatically depend on the model architecture, dataset, input, pre-training, and data augmentation. However, three optimization functions consistently gave the worst performances: Stochastic Gradient Descent (SGD), Adagrad and Adadelta. For the baseline model, it was shown a preference for the Nadam optimizer. For the end-to-end models, the highlight is to Adam optimizer. For the Transformer, the preferable optimization function was AdamW for the ESC-50, Adam for the ESC-10 and Adamax for the UrbanSound8K.

However, there is a common difficulty between all the models concerning the accuracy per class for the UrbanSound8K dataset. All models exhibit lower accuracy values to the same classes. The most challenging class was the air conditioner, followed by engine idling, jackhammer and drilling, which must be pretty similar classes because the referred classes are mostly misclassified as being of each other's classes. In addition, the gunshot class was also the most straightforward class for all models. Regarding the ESC datasets, there is no consensus; however, helicopter seems to be the most challenging class by appearing with a low score in most situations, and the most straightforward class is sneezing for ESC-10 and toilet flush for ESC-50.

Table 6.1 presents a discussion between the most relevant models aggregated by each dataset under study.

After reviewing the results, Transformer is shown to be the most capable of providing better results by showing significant improvements compared with the best baseline and end-to-end model for each dataset with an average difference of 32.8 percentage points (pp) and 4.79 pp, respectively. Thus, the best accuracy results obtained for each dataset were for ESC-10, 99%, for ESC-50, 95.8% and for UrbanSound8K, 89.8%. Although it was not capable of achieving State-of-the-Art (SOTA) results, it gave very competitive results by being the second best result on the ESC-10 dataset with only a difference of 0.22 pp to the top result, the third best score for the ESC-50 dataset and the fourth best for UrbanSound8K considering the official splits, as can be confirmed in Table 6.2.

TABLE 6.1: Summary and discussion of several of the proposed models.

Model	DA	PI	PA	Metrics	Discussion
Dataset: ESC-10					
Baseline model + mms60 + Nadam + dr: 0.2	-	-	-	acc: 74.8%, AUC: 94.8%, mf1: 74.8%, Mf1: 74.3%, prec: 77.7%, rec: 73.3%.	Combination of features gives more discriminating information to the baseline model.
DenseNet + AdamW	-	-	-	acc: 89.8%, AUC: 98.9%, mf1: 89.8%, Mf1: 89.3%, prec: 91.5%, rec: 89.8%.	Improves the baseline performance by 13.23 pp, on average.
ResNet + AdamW	-	✓	-	acc: 94.0%, AUC: 99.8%, mf1: 94.0%, Mf1: 93.8%, prec: 94.8%, rec: 94.0%.	The use of pre-training from ImageNet improves, on average, the end-to-end model performance by 3.55 pp.
DenseNet + Adam	✓	✓	-	acc: 95.0%, AUC: 99.8%, mf1: 95.0%, Mf1: 94.9%, prec: 95.8%, rec: 95.0%.	The addition of data augmentation techniques provides a slight improvement of 0.85 pp, on average.
Transformer + AdamW	✓	-	-	acc: 52.8%, AUC: 91.8%, mf1: 52.8%, Mf1: 50.7%, prec: 61.7%, rec: 52.8%.	The use of a Transformer model without pre-training cannot give competitive results.
Transformer + AdamW	✓	✓	-	acc: 93.8%, AUC: 99.8%, mf1: 93.8%, Mf1: 93.5%, prec: 98.2%, rec: 93.8%.	The use of pre-training from ImageNet gives the Transformer model an average boost of 35.05 pp. Showing the need for large datasets to train.
Transformer + AdamW	-	✓	✓	acc: 98.8%, AUC: 100%, mf1: 98.8%, Mf1: 98.7%, prec: 99.7%, rec: 98.8%.	Using pre-training from ImageNet and AudioSet gives a better performance than just an ImageNet pre-trained Transformer with an average increase of 3.65 pp.
Transformer + Adam	✓	✓	✓	acc: 99.0%, AUC: 100%, mf1: 99.0%, Mf1: 99.0%, prec: 99.9%, rec: 99.0%.	The addition of data augmentation to the pre-train from both domains gives, on average, a slight improvement of 0.18 pp. The average boost for the baseline model is 21.03 pp and for the best end-to-end model of 3.40 pp.
Dataset: ESC-50					
Baseline model + mfcc-stft80 + Nadam + dr: 0.2	-	-	-	acc: 38.1%, AUC: 82.4%, mf1: 38.1%, Mf1: 36.2%, prec: 43.9%, rec: 33.9%.	Combination of features gives more discriminating information to the baseline model.
DenseNet + Adam	-	-	-	acc: 76.1%, AUC: 98.7%, mf1: 76.1%, Mf1: 75.4%, prec: 78.0%, rec: 76.1%.	Improves the baseline performance by 34.63 pp, on average.
ResNet + Adam	-	✓	-	acc: 88.2%, AUC: 99.6%, mf1: 88.2%, Mf1: 87.7%, prec: 89.6%, rec: 88.2%.	The use of pre-training from ImageNet improves, on average, the end-to-end model performance by 10.18 pp.
ResNet + Adam	✓	✓	-	acc: 90.1%, AUC: 99.6%, mf1: 90.1%, Mf1: 89.9%, prec: 91.2%, rec: 90.1%.	The addition of data augmentation techniques gives a small increase of 1.58 pp, on average.
Transformer + AdamW	✓	-	-	acc: 43.9%, AUC: 93.6%, mf1: 43.9%, Mf1: 42.4%, prec: 46.8%, rec: 43.9%.	The use of a Transformer model without pre-training cannot provide good results; however, better than the baseline model.
Transformer + AdamW	✓	✓	-	acc: 88.6%, AUC: 99.6%, mf1: 88.6%, Mf1: 88.4%, prec: 92.7%, rec: 88.6%.	The use of pre-training from ImageNet gives a huge performance boost of 38.67 pp, on average, compared to the Transformer model without pre-training.
Transformer + AdamW	-	✓	✓	acc: 95.4%, AUC: 99.9%, mf1: 95.4%, Mf1: 95.3%, prec: 97.6%, rec: 95.4%.	Using pre-training from ImageNet and AudioSet gives an average improvement of 5.42 pp compared to the ImageNet pre-trained Transformer.
Transformer + AdamW	✓	✓	✓	acc: 95.8%, AUC: 99.9%, mf1: 95.8%, Mf1: 95.6%, prec: 97.8%, rec: 95.8%.	The addition of data augmentation gives a small improvement of 0.28 pp, on average. The average boost is to the baseline model of 51.35 pp and 4.95 pp for the best end-to-end model.
Dataset: UrbanSound8K					
Baseline model + mmsqc + Nadam + dr: 0.6	-	-	-	acc: 61.1%, AUC: 88.9%, mf1: 61.1%, Mf1: 63.2%, prec: 73.1%, rec: 49.2%.	Combination of features gives more discriminating information to the baseline model.
DenseNet + AdamW	-	-	-	acc: 74.2%, AUC: 95.4%, mf1: 74.2%, Mf1: 75.6%, prec: 75.2%, rec: 74.2%.	Improves the baseline performance by 12.03 pp, on average.
DenseNet + Adam	-	✓	-	acc: 83.3%, AUC: 97.7%, mf1: 83.3%, Mf1: 84.4%, prec: 84.1%, rec: 83.3%.	The use of pre-training from ImageNet improves the end-to-end model performance by 7.88 pp, on average.
ResNet + Adamax	✓	✓	-	acc: 82.2%, AUC: 97.4%, mf1: 82.2%, Mf1: 83.0%, prec: 82.5%, rec: 82.2%.	The use of data augmentation techniques was detrimental.
Transformer + Adamax	✓	✓	✓	acc: 89.8%, AUC: 98.6%, mf1: 89.8%, Mf1: 90.4%, prec: 93.8%, rec: 89.8%.	The Transformer model pre-trained with datasets from both domains and using data augmentation gives an average boost of 25.93 pp regarding the baseline model and of 6.02 pp compared to the best end-to-end model.

DA: data augmentation; PI: Pre-trained ImageNet; PA: Pre-trained AudioSet; dr: dropout rate; acc: accuracy; AUC: area under the receiver operating characteristic curve; mf1: micro f1score; Mf1: macro f1score; prec: precision; rec: recall; pp: percentage points.

TABLE 6.2: Accuracy results of all models considered for the literature review and proposed models.

Authors/Year	**UrbanSound8k**	**ESC-50**	**ESC-10**
J. K. Das, A. Ghosh, A. K. Pal, S. Dutta and A. Chakrabarty (2020) [2]	98.81% (unofficial split)	-	-
J.K. Das, A. Chakrabarty and M.J. Piran (2021) [3]	99.60% (unofficial split)	-	-
T. Giannakopoulos, E. Spyrou and S. J. Perantonis (2019) [9]	73.1%	52.2%	-
I. Martin-Morato, M. Cobos and F. J. Ferri (2020) [19]	73.96%	-	-
P. Zinemanas, M. Rocamora, M. Miron, F. Font and X. Serra (2021) [45]	76.2%	-	-
J. Salamon and J. P. Bello (2017) [29]	79%	-	-
W. Mu, B. Yin, X. Huang, J. Xu and Z. Du (2021) [21]	93.1%	84.4%	-
J.S. Luz, M.C. Oliveira, F.H.D. Araújo and D.M.V. Magalhães (2021) [18]	96.8%	-	86.2%
T. M. S. Tax, J. L. D. Antich, H. Purwins and L. Maaløe (2017) [36]	-	$\approx$ 50%	-
David Elliott, Carlos E. Otero, Steven Wyatt and Evan Martino (2021) [8]	-	67.71%	-
H. Akbari, L. Yuan, R. Qian, W. Chuang, S. Chang, Y. Cui and B. Gong (2021) [1]	-	84.9%	-
Z. Zhang, S. Xu, S. Zhang, T. Qiao and S. Cao (2020) [44]	-	86.1%	93.7%
T. Qiao, S. Zhang, S. Cao and S. Xu (2021) [25]	-	86.4%	-
Z. Zhang, S. Xu, S. Zhang, T. Qiao and S. Cao (2019) [43]	-	86.5%	94.2%
N.-C. Ristea, R. T. Ionescu and F. S. Khan (2022) [26]	-	91.13%	-
Y. Gong, Yu-An Chung and J. Glass (2021) [11]	-	95.6%	-
K. Koutini, J. Schlüter, H. Eghbal-zadeh and G. Widmer (2021) [16]	-	96.8%	-
A. M. Tripathi and A. Mishra (2021) [38]	-	-	92.16%
İ. Türker and S. Aksu (2022) [46]	-	-	96.46%
Z. Mushtaq and S.-F. Su (2020) [22]	**97.98%**	**98.52%**	**99.22%**
Proposed models			
Baseline model	61.1%	38.1%	74.8%
Inception	82.7%	86.4%	94.0%
ResNet	82.8%	90.1%	94.0%
DenseNet	83.3%	90.1%	95.0%
Transformer	**89.8%**	**95.8%**	**99.0%**

Future research will explore the inclusion of a dropout method like Patchout introduced by Koutini et al. [16] to obligate the model to perform classification using incomplete sequences, which improves the Transformer's performance. Furthermore, for the end-to-end models, instead of introducing a simple Melspectrogram, it might be beneficial the introduction of a logarithmic spectrogram such as Log(Log-Melspectrogram) or Log(Log(Log-Melspectrogram)) as proved by Mushtaq and Su [22].

Appendix A

Description of Autoencoder Model

The following section presents a brief description of the basic architecture of the autoencoder model.

A.1 Autoencoder

An autoencoder is a feed-forward neural network that is trained to attempt to copy approximately its input to its output in an unsupervised manner. The architecture of an autoencoder, as it is possible to see in Figure A.1*, consists of 3 components:

- Encoder: compresses the input data into an encoded representation in a latent space which is typically several orders of magnitude smaller than the input data;
- Code: contains the compressed knowledge representations;
- Decoder: reconstructs the data back from its encoded form using the latent space attributes. The output and the input must be of the same dimension.

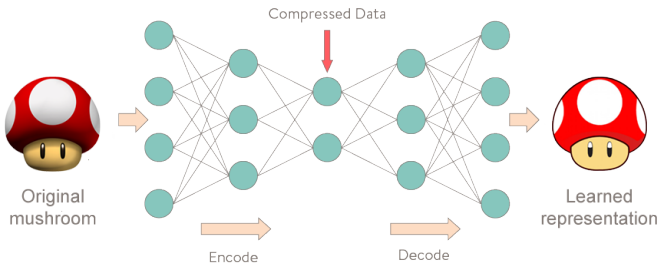


FIGURE A.1: Autoencoder architecture.

*https://www.metamaven.com/beyond-backpropagation-can-go-deeper-deep-learning/autoencoder_800px_web/ (accessed September 2022).

Appendix B

Complete Tables of the Baseline Models

In the following appendix, it's presented the complete tables of the features and combinations of features used to achieve the best baseline models for the UrbanSound8K, the ESC-10 and the ESC-50 datasets.

The used abbreviations and the meaning of the number that appears in front of each feature:

- zcr - zero-crossing rate;
- rms - root mean square;
- rol - spectral rolloff; roll-off percentage;
- poly - poly features; order of the polynomial to fit;
- sflat - spectral flatness;
- scontrast - spectral contrast; number of frequency bands;
- scent - spectral centroid; FFT window size;
- sband - spectral bandwidth; FFT window size;
- mel - melspectrogram, mfcc - Mel Frequency Cepstral Coefficients (MFCC); number of Mel bands;

- stft - chroma Short-Term Fourier Transformation (STFT), cqt - chroma Constant Q-transform (CQT), cens - chroma Chroma Energy Normalized Statistics (CENS), tonz - tonnetz; number of chroma bins.

If a feature has no number in front means that the default parameters were used. The abbreviations for the combinations of features are summarized in Table 3.1.

B.1 UrbanSound8K - Single Feature Input

B.1.1 Baseline Model Architecture

TABLE B.1: Results of the 6 models for different features.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel80	**0.591**	0.873	**0.591**	**0.593**	0.643	0.483	**mfcc60**	**0.522**	**0.886**	**0.522**	**0.551**	0.774	0.278
mfcc60	0.572	0.885	0.572	0.578	0.682	**0.527**	**mfcc80**	0.516	0.878	0.516	0.528	0.765	**0.314**
mfcc40	0.575	**0.894**	0.575	0.575	0.646	**0.527**	**mel80**	0.465	0.838	0.465	0.459	0.761	0.228
mfcc80	0.559	0.881	0.559	0.563	0.648	0.520	**mfcc40**	0.441	0.797	0.441	0.440	0.519	0.226
mel60	0.486	0.850	0.486	0.497	0.550	0.381	**mel60**	0.423	0.832	0.423	0.436	0.683	0.170
stft60	0.507	0.862	0.507	0.492	0.582	0.305	**mel40**	0.412	0.830	0.412	0.411	0.767	0.177
mel40	0.481	0.821	0.481	0.478	0.517	0.367	**mel20**	0.400	0.826	0.400	0.386	0.696	0.140
stft80	0.483	0.859	0.483	0.472	0.591	0.330	**stft60**	0.374	0.835	0.374	0.367	0.589	0.118
mel20	0.468	0.839	0.468	0.472	0.510	0.338	**stft20**	0.348	0.833	0.348	0.349	0.381	0.067
stft20	0.452	0.865	0.452	0.451	0.598	0.348	**stft40**	0.345	0.842	0.345	0.336	0.636	0.100
stft40	0.459	0.865	0.459	0.449	0.606	0.345	**cqt40**	0.344	0.782	0.344	0.332	0.607	0.078
cqt20	0.435	0.817	0.435	0.430	0.640	0.219	**cqt80**	0.331	0.773	0.331	0.331	0.603	0.109
cqt80	0.397	0.811	0.397	0.396	0.607	0.210	**cens20**	0.341	0.764	0.341	0.329	0.745	0.042
cqt40	0.391	0.782	0.391	0.387	0.547	0.179	**stft80**	0.325	0.822	0.325	0.319	0.636	0.134
cens20	0.362	0.788	0.362	0.343	0.512	0.246	**cqt20**	0.300	0.774	0.300	0.290	0.339	0.044
cens40	0.342	0.792	0.342	0.334	0.451	0.249	**scontrast**	0.305	0.754	0.305	0.269	0.467	0.025
scontrast	0.324	0.764	0.324	0.291	0.465	0.157	**cens40**	0.264	0.763	0.264	0.243	0.609	0.084
poly10	0.281	0.777	0.281	0.243	0.432	0.023	**scontrast4**	0.204	0.695	0.204	0.185	0.368	0.008
scontrast4	0.253	0.732	0.253	0.242	0.266	0.025	**tonz20**	0.201	0.647	0.201	0.168	0.333	0.002
tonz20	0.232	0.694	0.232	0.231	0.622	0.082	**tonz40**	0.171	0.655	0.171	0.167	0.357	0.006
poly5	0.274	0.752	0.274	0.227	0.000	0.000	**tonz80**	0.147	0.654	0.147	0.150	0.455	0.006
cens80	0.211	0.687	0.211	0.219	0.405	0.140	**poly10**	0.258	0.735	0.258	0.145	**1.000**	0.005
tonz80	0.205	0.685	0.205	0.200	0.409	0.043	**sflat**	0.173	0.685	0.173	0.130	0.359	0.017
tonz40	0.198	0.671	0.198	0.198	0.419	0.062	**zcr**	0.190	0.716	0.190	0.121	0.636	0.042
zcr	0.223	0.667	0.223	0.164	0.660	0.042	**cens80**	0.128	0.673	0.128	0.118	0.320	0.010
sflat	0.192	0.652	0.192	0.150	0.650	0.016	**poly5**	0.214	0.722	0.214	0.108	0.000	0.000
poly	0.225	0.650	0.225	0.149	**1.000**	0.022	**poly3**	0.222	0.676	0.222	0.106	0.000	0.000
poly3	0.188	0.645	0.188	0.143	0.900	0.022	**poly**	0.209	0.640	0.209	0.094	0.000	0.000
poly2	0.165	0.625	0.165	0.125	**1.000**	0.024	**poly2**	0.205	0.648	0.205	0.091	0.000	0.000
rms	0.227	0.653	0.227	0.112	**1.000**	0.017	**rms**	0.190	0.625	0.190	0.081	0.000	0.000
rol0.25	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.25**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.45	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.45**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.65	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.65**	0.119	0.500	0.119	0.021	0.000	0.000
scent2048	0.119	0.500	0.119	0.021	0.000	0.000	**scent2048**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.85	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.85**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.99	0.119	0.511	0.119	0.021	0.000	0.000	**rol0.99**	0.119	0.500	0.119	0.021	0.000	0.000
sband	0.119	0.500	0.119	0.021	0.000	0.000	**scent8000**	0.119	0.500	0.119	0.021	0.000	0.000
scent8000	0.111	0.500	0.111	0.020	0.000	0.000	**sband8000**	0.115	0.500	0.115	0.021	0.000	0.000
sband8000	0.039	0.500	0.039	0.008	0.000	0.000	**sband**	0.099	0.500	0.099	0.018	0.000	0.000

Model 3: (optimizer: Adamax)							Model 4: (optimizer: Nadam)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	0.611	**0.904**	0.611	**0.622**	0.722	0.550	**mfcc80**	**0.637**	**0.908**	**0.637**	**0.628**	0.747	**0.572**
mfcc60	**0.616**	0.898	**0.616**	0.618	0.682	**0.564**	**mfcc40**	0.576	0.873	0.576	0.583	0.651	0.535
mfcc40	0.593	0.893	0.593	0.607	0.693	0.519	**mfcc60**	0.573	0.864	0.573	0.571	0.654	0.529
mel80	0.491	0.859	0.491	0.501	0.703	0.294	**mel80**	0.535	0.857	0.535	0.539	0.583	0.419
mel40	0.466	0.862	0.466	0.477	0.688	0.260	**mel60**	0.487	0.836	0.487	0.490	0.569	0.415
stft80	0.478	0.870	0.478	0.471	0.631	0.341	**stft40**	0.478	0.865	0.478	0.474	0.569	0.319
stft40	0.477	0.886	0.477	0.456	0.613	0.344	**stft60**	0.492	0.864	0.492	0.471	0.609	0.341
stft20	0.455	0.877	0.455	0.447	0.591	0.297	**mel40**	0.448	0.836	0.448	0.458	0.503	0.368
stft60	0.456	0.874	0.456	0.439	0.590	0.323	**stft20**	0.458	0.870	0.458	0.455	0.591	0.323
mel60	0.411	0.846	0.411	0.434	0.602	0.235	**mel20**	0.446	0.832	0.446	0.448	0.499	0.313
mel20	0.431	0.840	0.431	0.423	0.648	0.204	**stft80**	0.430	0.851	0.430	0.421	0.560	0.300
cqt20	0.423	0.815	0.423	0.416	0.560	0.161	**cqt20**	0.411	0.817	0.411	0.398	0.556	0.214
cqt80	0.401	0.809	0.401	0.399	0.592	0.216	**cqt40**	0.389	0.799	0.389	0.380	0.579	0.219
cqt40	0.395	0.809	0.395	0.391	0.580	0.183	**cqt80**	0.376	0.790	0.376	0.371	0.528	0.214
cens20	0.393	0.791	0.393	0.375	0.528	0.158	**cens20**	0.384	0.790	0.384	0.361	0.513	0.233
cens40	0.369	0.799	0.369	0.358	0.549	0.182	**scontrast**	0.367	0.771	0.367	0.348	0.472	0.119
scontrast	0.362	0.785	0.362	0.337	0.486	0.127	**cens40**	0.344	0.785	0.344	0.339	0.460	0.252
poly10	0.337	0.789	0.337	0.263	0.769	0.036	**scontrast4**	0.280	0.747	0.280	0.258	0.389	0.033
scontrast4	0.271	0.732	0.271	0.261	0.298	0.030	**tonz20**	0.237	0.694	0.237	0.236	0.640	0.087
zcr	0.270	0.682	0.270	0.239	0.610	0.030	**poly10**	0.277	0.777	0.277	0.223	0.625	0.042
tonz20	0.229	0.687	0.229	0.220	0.676	0.030	**poly5**	0.233	0.733	0.233	0.216	0.833	0.018
tonz40	0.232	0.709	0.232	0.216	0.520	0.031	**tonz80**	0.223	0.685	0.223	0.213	0.371	0.051
tonz80	0.214	0.704	0.214	0.210	0.558	0.035	**tonz40**	0.211	0.679	0.211	0.207	0.492	0.072
cens80	0.178	0.712	0.178	0.169	0.353	0.078	**cens80**	0.186	0.690	0.186	0.192	0.348	0.116
poly5	0.191	0.710	0.191	0.162	0.778	0.017	**zcr**	0.229	0.677	0.229	0.177	0.654	0.041
poly	0.205	0.648	0.205	0.140	**1.000**	0.017	**sflat**	0.229	0.656	0.229	0.169	0.838	0.037
sflat	0.179	0.660	0.179	0.138	0.650	0.016	**poly3**	0.210	0.661	0.210	0.152	0.333	0.001
poly3	0.198	0.671	0.198	0.133	0.222	0.005	**poly**	0.186	0.643	0.186	0.132	0.909	0.024
poly2	0.168	0.649	0.168	0.119	0.000	0.000	**poly2**	0.191	0.640	0.191	0.129	**1.000**	0.024
rms	0.219	0.654	0.219	0.105	0.000	0.000	**rms**	0.216	0.654	0.216	0.107	**1.000**	0.017
rol0.25	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.99**	0.119	0.485	0.119	0.021	0.000	0.000
rol0.85	0.119	0.500	0.119	0.021	0.000	0.000	**sband**	0.119	0.500	0.119	0.021	0.000	0.000
sband	0.111	0.500	0.111	0.020	0.000	0.000	**scent8000**	0.119	0.500	0.119	0.021	0.000	0.000
scent2048	0.111	0.500	0.111	0.020	0.000	0.000	**rol2**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.65	0.099	0.500	0.099	0.018	0.000	0.000	**sband8000**	0.119	0.500	0.119	0.021	0.000	0.000
scent8000	0.039	0.500	0.039	0.008	0.000	0.000	**rol0.45**	0.099	0.500	0.099	0.018	0.000	0.000
rol0.99	0.039	0.500	0.039	0.008	0.000	0.000	**scent2048**	0.039	0.500	0.039	0.008	0.000	0.000
rol0.45	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.25**	0.038	0.500	0.038	0.007	0.000	0.000
sband8000	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.65**	0.038	0.500	0.038	0.007	0.000	0.000

Model 5: (optimizer: Adadelta)							Model 6: (optimizer: Adagrad)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	0.209	0.595	0.209	**0.179**	0.317	0.118	**mfcc60**	**0.368**	0.753	**0.368**	**0.361**	**1.000**	0.033
mel60	**0.213**	0.637	**0.213**	0.164	0.556	0.066	**mel80**	0.323	**0.793**	0.323	0.329	0.538	0.060
tonz20	0.208	0.538	0.208	0.163	0.000	0.000	**mel40**	0.329	0.779	0.329	0.329	0.766	0.059
cens20	0.167	0.597	0.167	0.140	0.000	0.000	**mel60**	0.283	0.786	0.283	0.287	0.803	**0.068**
cens40	0.145	0.554	0.145	0.130	0.000	0.000	**mel20**	0.295	0.762	0.295	0.257	0.723	0.041
mel80	0.123	0.564	0.123	0.128	0.136	0.018	**mfcc40**	0.220	0.676	0.220	0.227	0.818	0.032
tonz80	0.131	0.555	0.131	0.117	0.000	0.000	**stft80**	0.223	0.731	0.223	0.204	0.545	0.014
mel20	0.134	**0.661**	0.134	0.108	0.553	0.025	**scontrast4**	0.214	0.666	0.214	0.189	0.000	0.000
mel40	0.160	0.589	0.160	0.104	0.534	0.056	**stft60**	0.198	0.724	0.198	0.185	0.917	0.013
mfcc40	0.153	0.645	0.153	0.100	0.227	0.098	**stft40**	0.213	0.754	0.213	0.183	0.875	0.017
mfcc80	0.146	0.572	0.146	0.096	0.197	0.075	**cens40**	0.201	0.698	0.201	0.177	0.000	0.000
cens80	0.104	0.558	0.104	0.082	0.000	0.000	**mfcc80**	0.200	0.638	0.200	0.176	**1.000**	0.030
poly10	0.184	0.632	0.184	0.082	0.000	0.000	**stft20**	0.177	0.732	0.177	0.168	**1.000**	0.001
rms	0.174	0.541	0.174	0.081	0.000	0.000	**scontrast**	0.226	0.689	0.226	0.159	0.500	0.001
sflat	0.158	0.605	0.158	0.081	0.000	0.000	**cqt80**	0.146	0.658	0.146	0.133	**1.000**	0.001
stft40	0.104	0.576	0.104	0.073	0.000	0.000	**tonz40**	0.195	0.600	0.195	0.126	0.000	0.000
zcr	0.148	0.545	0.148	0.063	0.000	0.000	**cqt40**	0.143	0.656	0.143	0.124	0.000	0.000
poly5	0.145	0.594	0.145	0.057	**1.000**	0.017	**poly10**	0.214	0.728	0.214	0.112	**1.000**	0.010
tonz40	0.109	0.554	0.109	0.055	0.000	0.000	**poly3**	0.209	0.642	0.209	0.104	0.000	0.000
scontrast4	0.116	0.536	0.116	0.050	0.000	0.000	**cqt20**	0.121	0.641	0.121	0.100	0.000	0.000
stft20	0.074	0.572	0.074	0.045	0.000	0.000	**poly5**	0.134	0.686	0.134	0.095	0.000	0.000
stft60	0.074	0.588	0.074	0.042	0.000	0.000	**cens20**	0.108	0.670	0.108	0.092	0.000	0.000
poly2	0.059	0.500	0.059	0.037	0.000	0.000	**tonz80**	0.129	0.611	0.129	0.090	0.000	0.000

Model 5: (optimizer: Adadelta) - continuation							Model 6: (optimizer: Adagrad) - continuation						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft80	0.075	0.573	0.075	0.037	0.000	0.000	**zcr**	0.195	0.662	0.195	0.085	0.000	0.000
cqt40	0.053	0.523	0.053	0.034	0.000	0.000	**poly2**	0.183	0.587	0.183	0.079	0.000	0.000
scontrast	0.060	0.509	0.060	0.031	0.000	0.000	**tonz20**	0.084	0.597	0.084	0.079	0.000	0.000
poly	0.054	0.502	0.054	0.031	0.000	0.000	**rms**	0.192	0.580	0.192	0.075	0.000	0.000
poly3	0.042	0.548	0.042	0.028	0.000	0.000	**sflat**	0.153	0.627	0.153	0.071	0.000	0.000
cqt20	0.045	0.530	0.045	0.022	0.000	0.000	**poly**	0.146	0.598	0.146	0.065	0.000	0.000
rol0.25	0.119	0.514	0.119	0.021	0.125	**0.119**	**cens80**	0.075	0.590	0.075	0.049	0.000	0.000
rol0.65	0.119	0.507	0.119	0.021	0.119	**0.119**	**rol0.85**	0.119	0.515	0.119	0.021	0.000	0.000
rol0.85	0.119	0.508	0.119	0.021	0.119	**0.119**	**rol0.45**	0.119	0.502	0.119	0.021	0.000	0.000
sband8000	0.111	0.508	0.111	0.020	0.111	0.111	**scent8000**	0.119	0.532	0.119	0.021	0.000	0.000
scent2048	0.099	0.468	0.099	0.018	0.099	0.099	**rol0.99**	0.119	0.488	0.119	0.021	0.000	0.000
cqt80	0.042	0.547	0.042	0.013	0.000	0.000	**rol0.25**	0.111	0.528	0.111	0.020	0.000	0.000
rol0.45	0.039	0.397	0.039	0.008	0.036	0.036	**rol0.65**	0.111	0.525	0.111	0.020	0.000	0.000
scent8000	0.039	0.466	0.039	0.008	0.039	0.039	**sband8000**	0.099	0.493	0.099	0.018	0.000	0.000
sband	0.039	0.456	0.039	0.008	0.040	0.039	**scent2048**	0.099	0.482	0.099	0.018	0.000	0.000
rol0.99	0.038	0.474	0.038	0.007	0.038	0.038	**sband**	0.039	0.489	0.039	0.008	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.1.2 Extra Layer

TABLE B.2: Results of the 4 best models with an extra layer for different features.

Model 7: (optimizer: Adam)							Model 8: (optimizer: SGD)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	0.573	**0.879**	0.573	**0.584**	0.627	0.524	**mfcc80**	**0.539**	**0.873**	**0.539**	**0.561**	0.856	**0.263**
mfcc40	**0.584**	**0.879**	**0.584**	0.579	0.652	**0.550**	**mfcc60**	0.514	0.857	0.514	0.539	0.777	0.146
mfcc60	0.522	0.865	0.522	0.539	0.638	0.471	**mfcc40**	0.490	0.838	0.490	0.505	0.725	0.227
mel80	0.526	0.852	0.526	0.534	0.556	0.441	**mel80**	0.430	0.844	0.430	0.440	0.708	0.203
mel20	0.487	0.845	0.487	0.486	0.532	0.373	**mel60**	0.415	0.839	0.415	0.420	0.714	0.194
mel40	0.479	0.829	0.479	0.478	0.499	0.360	**mel20**	0.409	0.823	0.409	0.393	0.675	0.168
stft40	0.468	0.874	0.468	0.465	0.611	0.306	**mel40**	0.375	0.816	0.375	0.371	0.692	0.161
stft60	0.476	0.859	0.476	0.464	0.613	0.307	**cqt40**	0.336	0.778	0.336	0.339	0.604	0.080
mel60	0.449	0.837	0.449	0.454	0.548	0.405	**cqt80**	0.346	0.786	0.346	0.333	0.673	0.081
stft80	0.434	0.843	0.434	0.436	0.601	0.278	**stft60**	0.331	0.821	0.331	0.331	0.615	0.099
stft20	0.431	0.864	0.431	0.428	0.555	0.303	**cens20**	0.332	0.761	0.332	0.323	0.612	0.036
cqt20	0.407	0.804	0.407	0.396	0.599	0.192	**stft80**	0.331	0.821	0.331	0.321	0.496	0.134
cqt40	0.389	0.797	0.389	0.387	0.615	0.162	**cqt20**	0.309	0.775	0.309	0.307	0.471	0.029
cqt80	0.385	0.792	0.385	0.387	0.605	0.155	**stft40**	0.305	0.819	0.305	0.298	0.347	0.073
scontrast	0.356	0.777	0.356	0.334	0.511	0.115	**stft20**	0.308	0.822	0.308	0.284	0.440	0.061
cens20	0.337	0.756	0.337	0.317	0.458	0.171	**scontrast**	0.295	0.756	0.295	0.235	0.423	0.013
cens40	0.323	0.790	0.323	0.316	0.439	0.244	**cens40**	0.256	0.761	0.256	0.234	0.527	0.082
tonz80	0.245	0.697	0.245	0.233	0.456	0.062	**tonz80**	0.182	0.666	0.182	0.172	**1.000**	0.002
poly10	0.266	0.778	0.266	0.228	0.644	0.035	**tonz40**	0.201	0.661	0.201	0.160	**1.000**	0.002
tonz20	0.228	0.695	0.228	0.224	0.636	0.092	**poly10**	0.260	0.736	0.260	0.145	**1.000**	0.005
tonz40	0.216	0.689	0.216	0.214	0.435	0.060	**scontrast4**	0.134	0.655	0.134	0.142	0.444	0.005
scontrast4	0.215	0.718	0.215	0.204	0.204	0.013	**poly5**	0.221	0.713	0.221	0.133	0.000	0.000
cens80	0.180	0.681	0.180	0.183	0.369	0.129	**zcr**	0.197	0.712	0.197	0.131	0.621	0.043
poly5	0.221	0.743	0.221	0.182	0.000	0.000	**sflat**	0.166	0.685	0.166	0.126	0.684	0.016
sflat	0.214	0.647	0.214	0.175	0.923	0.014	**rms**	0.217	0.637	0.217	0.111	0.000	0.000
zcr	0.225	0.678	0.225	0.170	0.615	0.029	**tonz20**	0.125	0.602	0.125	0.103	0.222	0.002
poly	0.227	0.647	0.227	0.143	**1.000**	0.022	**cens80**	0.122	0.673	0.122	0.101	0.259	0.008
poly2	0.170	0.631	0.170	0.125	**1.000**	0.018	**poly**	0.210	0.642	0.210	0.094	0.000	0.000
poly3	0.192	0.629	0.192	0.124	0.889	0.010	**poly3**	0.213	0.671	0.213	0.088	0.000	0.000
rol0.25	0.184	0.615	0.184	0.123	0.000	0.000	**poly2**	0.204	0.650	0.204	0.080	0.000	0.000
rms	0.214	0.650	0.214	0.104	0.000	0.000	**sband**	0.119	0.500	0.119	0.021	0.000	0.000
sband	0.119	0.500	0.119	0.021	0.000	0.000	**rol2**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.45	0.119	0.500	0.119	0.021	0.000	0.000	**sband8000**	0.119	0.500	0.119	0.021	0.000	0.000
rol2	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.65**	0.115	0.500	0.115	0.021	0.000	0.000
sband8000	0.111	0.500	0.111	0.020	0.000	0.000	**scent8000**	0.115	0.500	0.115	0.021	0.000	0.000
rol0.65	0.039	0.500	0.039	0.008	0.000	0.000	**rol0.25**	0.039	0.500	0.039	0.008	0.000	0.000
scent8000	0.039	0.451	0.039	0.008	0.000	0.000	**rol0.45**	0.039	0.500	0.039	0.008	0.000	0.000
scent2048	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.99**	0.039	0.500	0.039	0.008	0.000	0.000
rol0.99	0.038	0.466	0.038	0.007	0.000	0.000	**scent2048**	0.038	0.500	0.038	0.007	0.000	0.000

Model 9: (optimizer: Adamax)							Model 10: (optimizer: Nadam)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.596**	**0.902**	**0.596**	**0.610**	0.656	**0.529**	**mfcc60**	0.577	0.875	0.577	**0.587**	0.694	0.532
mfcc60	0.558	0.883	0.558	0.565	0.624	0.511	**mfcc40**	**0.585**	**0.880**	**0.585**	0.574	0.667	**0.534**
mfcc40	0.557	0.892	0.557	0.561	0.681	0.458	**mfcc80**	0.556	0.865	0.556	0.566	0.640	0.526
mel40	0.486	0.857	0.486	0.487	0.680	0.297	**mel40**	0.493	0.845	0.493	0.495	0.540	0.384
mel60	0.462	0.857	0.462	0.467	0.681	0.294	**mel80**	0.485	0.849	0.485	0.488	0.548	0.419
stft60	0.472	0.868	0.472	0.464	0.617	0.337	**mel20**	0.472	0.831	0.472	0.473	0.504	0.348
mel80	0.458	0.859	0.458	0.457	0.657	0.300	**stft20**	0.460	0.869	0.460	0.457	0.612	0.355
stft20	0.466	0.874	0.466	0.456	0.596	0.326	**stft60**	0.456	0.855	0.456	0.453	0.551	0.271
stft40	0.460	0.877	0.460	0.450	0.592	0.330	**stft40**	0.455	0.859	0.455	0.445	0.592	0.330
stft80	0.446	0.857	0.446	0.441	0.577	0.308	**mel60**	0.441	0.835	0.441	0.441	0.543	0.397
cqt20	0.449	0.822	0.449	0.438	0.626	0.184	**stft80**	0.436	0.855	0.436	0.427	0.581	0.282
mel20	0.430	0.837	0.430	0.425	0.608	0.262	**cqt40**	0.406	0.806	0.406	0.407	0.582	0.203
cqt40	0.419	0.814	0.419	0.424	0.640	0.213	**cqt20**	0.409	0.807	0.409	0.400	0.617	0.221
cqt80	0.404	0.814	0.404	0.401	0.639	0.190	**cqt80**	0.398	0.800	0.398	0.397	0.638	0.194
cens20	0.372	0.781	0.372	0.355	0.539	0.155	**cens40**	0.351	0.783	0.351	0.351	0.460	0.233
cens40	0.344	0.805	0.344	0.340	0.527	0.176	**cens20**	0.355	0.779	0.355	0.341	0.507	0.203
scontrast	0.317	0.763	0.317	0.301	0.470	0.122	**scontrast**	0.317	0.775	0.317	0.308	0.475	0.116
poly10	0.315	0.789	0.315	0.260	0.714	0.024	**poly10**	0.270	0.778	0.270	0.232	0.605	0.031
scontrast4	0.263	0.732	0.263	0.239	0.281	0.019	**cens80**	0.208	0.682	0.208	0.217	0.411	0.146
tonz20	0.229	0.687	0.229	0.221	0.523	0.027	**tonz20**	0.228	0.690	0.228	0.217	0.673	0.081
tonz80	0.239	0.707	0.239	0.221	0.476	0.036	**tonz80**	0.217	0.687	0.217	0.212	0.431	0.053
tonz40	0.223	0.695	0.223	0.211	0.544	0.037	**tonz40**	0.217	0.683	0.217	0.199	0.426	0.059
cens80	0.192	0.711	0.192	0.194	0.417	0.099	**scontrast4**	0.190	0.710	0.190	0.190	0.182	0.012
zcr	0.211	0.675	0.211	0.173	0.660	0.042	**zcr**	0.217	0.676	0.217	0.171	0.654	0.041
sflat	0.203	0.657	0.203	0.160	0.722	0.016	**sflat**	0.202	0.650	0.202	0.157	0.909	0.012
poly	0.245	0.656	0.245	0.152	**1.000**	0.016	**poly5**	0.198	0.730	0.198	0.157	0.000	0.000
poly2	0.189	0.647	0.189	0.145	0.000	0.000	**poly**	0.235	0.652	0.235	0.151	0.000	0.000
poly5	0.172	0.699	0.172	0.135	0.000	0.000	**poly3**	0.168	0.637	0.168	0.121	0.563	0.011
poly3	0.177	0.659	0.177	0.111	**1.000**	0.002	**rol0.25**	0.172	0.625	0.172	0.115	0.000	0.000
rms	0.210	0.651	0.210	0.100	0.000	0.000	**poly2**	0.139	0.630	0.139	0.112	**1.000**	0.022
rol0.25	0.151	0.622	0.151	0.093	0.000	0.000	**rms**	0.217	0.651	0.217	0.106	**1.000**	0.017
sband	0.119	0.500	0.119	0.021	0.000	0.000	**sband**	0.119	0.511	0.119	0.021	0.000	0.000
rol0.65	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.45**	0.119	0.511	0.119	0.021	0.000	0.000
rol0.45	0.111	0.517	0.111	0.020	0.000	0.000	**sband8000**	0.119	0.500	0.119	0.021	0.000	0.000
sband8000	0.099	0.500	0.099	0.018	0.000	0.000	**scent2048**	0.119	0.500	0.119	0.021	0.000	0.000
rol2	0.039	0.500	0.039	0.008	0.000	0.000	**rol0.65**	0.111	0.506	0.111	0.020	0.000	0.000
rol0.99	0.039	0.500	0.039	0.008	0.000	0.000	**rol2**	0.111	0.500	0.111	0.020	0.000	0.000
scent8000	0.038	0.500	0.038	0.007	0.000	0.000	**scent8000**	0.099	0.500	0.099	0.018	0.000	0.000
scent2048	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.99**	0.039	0.477	0.039	0.008	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.1.3 Dropout Rate of 0.2

TABLE B.3: Results of the 4 best models for different features and dropout of 0.2.

Model 11: (optimizer: Adam)							Model 12: (optimizer: SGD)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.620**	0.862	**0.620**	**0.616**	0.639	**0.606**	**mfcc80**	**0.582**	**0.897**	**0.582**	**0.583**	0.707	0.472
mfcc60	0.582	0.857	0.582	0.587	0.613	0.572	**mfcc40**	0.570	0.876	0.570	0.570	0.708	0.478
mfcc40	0.589	0.855	0.589	0.582	0.618	0.581	**mfcc60**	0.575	0.894	0.575	0.568	0.729	**0.492**
mel80	0.503	0.816	0.503	0.505	0.539	0.481	**mel80**	0.498	0.849	0.498	0.505	0.658	0.324
stft60	0.514	0.856	0.514	0.504	0.562	0.425	**mel60**	0.460	0.831	0.460	0.472	0.686	0.258
stft80	0.507	0.854	0.507	0.504	0.560	0.415	**stft40**	0.456	0.862	0.456	0.453	0.623	0.176
stft20	0.505	**0.863**	0.505	0.492	0.565	0.436	**stft60**	0.453	0.863	0.453	0.449	0.678	0.161
stft40	0.481	0.854	0.481	0.481	0.531	0.404	**mel40**	0.438	0.828	0.438	0.441	0.667	0.249
mel60	0.467	0.783	0.467	0.466	0.480	0.409	**mel20**	0.423	0.824	0.423	0.418	0.626	0.166
mel40	0.424	0.766	0.424	0.428	0.441	0.364	**cqt80**	0.388	0.792	0.388	0.390	0.656	0.121
mel20	0.417	0.790	0.417	0.420	0.459	0.373	**stft80**	0.379	0.834	0.379	0.367	0.479	0.186
cqt40	0.400	0.789	0.400	0.401	0.482	0.296	**cqt20**	0.363	0.784	0.363	0.355	0.495	0.066
cqt80	0.387	0.783	0.387	0.387	0.468	0.305	**stft20**	0.351	0.846	0.351	0.354	0.514	0.114
cens20	0.364	0.751	0.364	0.362	0.405	0.307	**cens20**	0.337	0.758	0.337	0.332	0.750	0.068
cqt20	0.368	0.774	0.368	0.358	0.485	0.287	**cqt40**	0.333	0.786	0.333	0.319	0.610	0.103
scontrast	0.374	0.755	0.374	0.349	0.449	0.237	**scontrast**	0.306	0.746	0.306	0.277	0.457	0.108
scontrast4	0.329	0.764	0.329	0.321	0.459	0.154	**cens40**	0.290	0.769	0.290	0.270	0.600	0.104
cens40	0.331	0.747	0.331	0.321	0.371	0.296	**scontrast4**	0.249	0.695	0.249	0.231	0.287	0.035
poly10	0.275	0.760	0.275	0.238	0.493	0.043	**tonz40**	0.192	0.678	0.192	0.182	0.278	0.006
cens80	0.214	0.670	0.214	0.223	0.397	0.173	**tonz80**	0.189	0.670	0.189	0.167	0.292	0.008
poly5	0.229	0.734	0.229	0.199	0.533	0.029	**poly10**	0.251	0.735	0.251	0.145	**1.000**	0.002
tonz20	0.205	0.674	0.205	0.197	0.443	0.092	**cens80**	0.147	0.684	0.147	0.136	0.263	0.012
tonz40	0.178	0.676	0.178	0.178	0.322	0.081	**sflat**	0.166	0.699	0.166	0.125	0.326	0.017
zcr	0.215	0.667	0.215	0.177	0.660	0.042	**zcr**	0.191	0.713	0.191	0.124	0.597	0.044
tonz80	0.179	0.677	0.179	0.174	0.266	0.059	**poly5**	0.217	0.717	0.217	0.118	0.000	0.000
sflat	0.195	0.657	0.195	0.152	0.896	0.072	**tonz20**	0.111	0.625	0.111	0.110	0.400	0.002
rol0.65	0.191	0.655	0.191	0.148	**1.000**	0.010	**rms**	0.204	0.635	0.204	0.107	0.000	0.000
poly	0.186	0.641	0.186	0.130	**1.000**	0.024	**poly3**	0.221	0.686	0.221	0.103	0.000	0.000
poly3	0.178	0.669	0.178	0.127	0.880	0.026	**poly**	0.204	0.642	0.204	0.096	0.000	0.000
poly2	0.162	0.625	0.162	0.115	**1.000**	0.023	**poly2**	0.204	0.655	0.204	0.090	0.000	0.000
rms	0.214	0.656	0.214	0.108	**1.000**	0.017	**rol0.25**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.25	0.119	0.500	0.119	0.021	0.000	0.000	**scent8000**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.45	0.119	0.500	0.119	0.021	0.000	0.000	**sband8000**	0.115	0.500	0.115	0.021	0.000	0.000
scent2048	0.119	0.500	0.119	0.021	0.000	0.000	**sband**	0.111	0.500	0.111	0.020	0.000	0.000
sband	0.119	0.500	0.119	0.021	0.000	0.000	**rol2**	0.111	0.500	0.111	0.020	0.000	0.000
rol2	0.111	0.506	0.111	0.020	0.000	0.000	**rol0.65**	0.099	0.500	0.099	0.018	0.000	0.000
rol0.99	0.111	0.500	0.111	0.020	0.000	0.000	**rol0.99**	0.039	0.500	0.039	0.008	0.000	0.000
scent8000	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.45**	0.038	0.500	0.038	0.007	0.000	0.000
sband8000	0.038	0.500	0.038	0.007	0.000	0.000	**scent2048**	0.038	0.500	0.038	0.007	0.000	0.000
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.613**	**0.886**	**0.613**	**0.616**	0.638	**0.587**	**mfcc80**	**0.605**	0.849	**0.605**	**0.607**	0.622	**0.596**
mfcc40	0.597	0.868	0.597	0.608	0.636	0.579	**mfcc60**	0.587	**0.862**	0.587	0.603	0.616	0.560
mfcc60	0.602	0.875	0.602	0.604	0.643	0.583	**mfcc40**	0.539	0.823	0.539	0.524	0.558	0.527
stft40	0.539	0.885	0.539	0.523	0.629	0.429	**stft80**	0.496	0.829	0.496	0.487	0.518	0.386
mel80	0.490	0.849	0.490	0.496	0.609	0.357	**stft40**	0.493	0.846	0.493	0.485	0.563	0.428
stft60	0.505	0.878	0.505	0.492	0.594	0.399	**stft20**	0.491	0.854	0.491	0.480	0.563	0.437
mel60	0.471	0.839	0.471	0.484	0.604	0.329	**mel80**	0.479	0.787	0.479	0.480	0.503	0.458
stft80	0.471	0.857	0.471	0.458	0.570	0.362	**stft60**	0.472	0.845	0.472	0.466	0.512	0.398
stft20	0.456	0.872	0.456	0.451	0.581	0.349	**mel60**	0.461	0.804	0.461	0.462	0.467	0.401
mel40	0.428	0.834	0.428	0.437	0.562	0.297	**mel40**	0.434	0.789	0.434	0.447	0.446	0.384
cqt40	0.447	0.819	0.447	0.437	0.574	0.307	**cqt40**	0.405	0.790	0.405	0.410	0.476	0.320
mel20	0.419	0.829	0.419	0.422	0.537	0.249	**cqt20**	0.409	0.791	0.409	0.402	0.500	0.335
cqt20	0.404	0.800	0.404	0.401	0.549	0.221	**mel20**	0.376	0.771	0.376	0.375	0.403	0.332
cqt80	0.398	0.808	0.398	0.397	0.523	0.288	**cqt80**	0.362	0.776	0.362	0.362	0.426	0.284
cens20	0.380	0.784	0.380	0.367	0.503	0.234	**cens20**	0.358	0.752	0.358	0.341	0.432	0.317
cens40	0.369	0.788	0.369	0.365	0.491	0.240	**cens40**	0.341	0.745	0.341	0.337	0.371	0.293
scontrast	0.346	0.781	0.346	0.330	0.472	0.191	**scontrast**	0.360	0.764	0.360	0.333	0.432	0.232
scontrast4	0.303	0.750	0.303	0.291	0.381	0.088	**scontrast4**	0.341	0.754	0.341	0.318	0.448	0.153
tonz40	0.244	0.705	0.244	0.240	0.494	0.047	**poly10**	0.274	0.763	0.274	0.235	0.388	0.039
tonz80	0.219	0.696	0.219	0.220	0.446	0.039	**tonz20**	0.207	0.684	0.207	0.212	0.462	0.103

Model 13: (optimizer: Adamax) - continuation							Model 14: (optimizer: Nadam) - continuation						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
poly10	0.256	0.780	0.256	0.217	0.443	0.047	**poly5**	0.235	0.728	0.235	0.209	0.586	0.020
tonz20	0.222	0.685	0.222	0.212	0.556	0.060	**cens80**	0.201	0.680	0.201	0.207	0.361	0.170
cens80	0.198	0.699	0.198	0.199	0.410	0.125	**tonz80**	0.198	0.676	0.198	0.191	0.416	0.104
zcr	0.221	0.677	0.221	0.192	0.595	0.030	**tonz40**	0.197	0.666	0.197	0.191	0.381	0.090
poly5	0.195	0.722	0.195	0.175	0.600	0.018	**zcr**	0.209	0.667	0.209	0.176	0.660	0.042
sflat	0.196	0.656	0.196	0.156	0.591	0.016	**rol0.25**	0.208	0.696	0.208	0.162	0.526	0.012
poly3	0.200	0.659	0.200	0.155	0.955	0.025	**poly3**	0.186	0.666	0.186	0.140	0.800	0.029
poly2	0.183	0.637	0.183	0.147	**1.000**	0.024	**poly**	0.178	0.643	0.178	0.140	0.742	0.027
rol0.25	0.191	0.676	0.191	0.143	0.435	0.012	**sflat**	0.177	0.649	0.177	0.139	0.483	0.017
poly	0.177	0.640	0.177	0.132	0.808	0.025	**poly2**	0.173	0.631	0.173	0.128	**1.000**	0.024
rms	0.214	0.646	0.214	0.110	0.000	0.000	**rms**	0.217	0.654	0.217	0.114	0.778	0.017
sband8000	0.119	0.500	0.119	0.021	0.000	0.000	**sband8000**	0.115	0.500	0.115	0.021	0.000	0.000
rol0.99	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.45**	0.115	0.500	0.115	0.021	0.000	0.000
rol0.45	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.65**	0.115	0.500	0.115	0.021	0.000	0.000
scent2048	0.119	0.500	0.119	0.021	0.000	0.000	**scent2048**	0.099	0.510	0.099	0.018	0.000	0.000
scent8000	0.099	0.500	0.099	0.018	0.000	0.000	**sband**	0.099	0.500	0.099	0.018	0.000	0.000
sband	0.099	0.500	0.099	0.018	0.000	0.000	**rol2**	0.099	0.500	0.099	0.018	0.000	0.000
rol0.65	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.99**	0.039	0.500	0.039	0.008	0.000	0.000
rol2	0.039	0.500	0.039	0.008	0.000	0.000	**scent8000**	0.038	0.500	0.038	0.007	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.1.4 Dropout Rate of 0.6

TABLE B.4: Results of the 4 best models for different features and dropout of 0.6.

Model 15: (optimizer: Adam)							Model 16: (optimizer: SGD)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	0.591	0.894	0.591	**0.611**	0.788	0.448	**mel40**	**0.387**	0.806	**0.387**	**0.371**	0.765	**0.074**
mfcc80	**0.608**	**0.902**	**0.608**	0.610	0.823	**0.493**	**mel20**	0.386	0.798	0.386	0.350	0.676	0.055
mfcc40	0.601	0.894	0.601	0.609	0.803	0.424	**cqt80**	0.341	0.764	0.341	0.344	0.787	0.044
mel80	0.487	0.862	0.487	0.489	0.643	0.297	**mel60**	0.342	0.788	0.342	0.332	0.775	**0.074**
mel20	0.478	0.847	0.478	0.478	0.629	0.247	**mfcc80**	0.352	0.778	0.352	0.325	0.934	0.068
stft20	0.458	0.869	0.458	0.459	0.655	0.258	**mel80**	0.301	0.781	0.301	0.325	0.795	**0.074**
stft60	0.446	0.860	0.446	0.458	0.671	0.222	**mfcc60**	0.305	0.786	0.305	0.297	0.898	0.053
mel60	0.444	0.851	0.444	0.458	0.687	0.275	**stft60**	0.272	0.800	0.272	0.271	0.536	0.036
mel40	0.437	0.846	0.437	0.441	0.636	0.265	**cqt20**	0.263	0.750	0.263	0.260	0.345	0.023
stft40	0.436	0.868	0.436	0.437	0.645	0.232	**cqt40**	0.270	0.759	0.270	0.258	0.619	0.047
stft80	0.409	0.851	0.409	0.416	0.706	0.172	**stft40**	0.264	0.809	0.264	0.252	0.538	0.033
cqt40	0.363	0.783	0.363	0.354	0.596	0.111	**stft20**	0.249	**0.810**	0.249	0.243	0.596	0.037
cqt20	0.349	0.792	0.349	0.348	0.538	0.085	**stft80**	0.249	0.776	0.249	0.241	0.417	0.048
cqt80	0.331	0.780	0.331	0.333	0.655	0.093	**cens20**	0.252	0.722	0.252	0.229	0.750	0.011
cens40	0.335	0.801	0.335	0.331	0.491	0.158	**cens40**	0.244	0.731	0.244	0.214	0.424	0.017
cens20	0.343	0.769	0.343	0.327	0.632	0.103	**mfcc40**	0.201	0.740	0.201	0.203	0.875	0.050
poly10	0.337	0.785	0.337	0.265	0.800	0.033	**tonz80**	0.208	0.654	0.208	0.184	0.625	0.006
tonz80	0.235	0.698	0.235	0.235	0.545	0.029	**tonz40**	0.214	0.660	0.214	0.179	0.400	0.002
tonz20	0.235	0.687	0.235	0.231	0.630	0.020	**tonz20**	0.177	0.644	0.177	0.163	0.286	0.002
scontrast	0.292	0.746	0.292	0.230	0.452	0.017	**poly10**	0.269	0.736	0.269	0.162	**1.000**	0.002
tonz40	0.250	0.694	0.250	0.229	0.574	0.032	**poly5**	0.222	0.706	0.222	0.142	0.000	0.000
zcr	0.246	0.676	0.246	0.199	0.600	0.029	**sflat**	0.151	0.687	0.151	0.112	0.467	0.017
cens80	0.188	0.716	0.188	0.190	0.397	0.092	**cens80**	0.136	0.646	0.136	0.110	0.800	0.005
scontrast4	0.166	0.677	0.166	0.163	0.333	0.005	**zcr**	0.188	0.709	0.188	0.109	0.660	0.042
sflat	0.203	0.641	0.203	0.157	0.909	0.012	**scontrast**	0.168	0.673	0.168	0.103	0.000	0.000
poly5	0.213	0.731	0.213	0.157	0.000	0.000	**rms**	0.200	0.624	0.200	0.095	0.000	0.000
poly	0.251	0.654	0.251	0.152	**1.000**	0.017	**poly2**	0.216	0.639	0.216	0.089	0.000	0.000
poly2	0.214	0.646	0.214	0.139	0.000	0.000	**poly3**	0.204	0.659	0.204	0.088	0.000	0.000
poly3	0.158	0.651	0.158	0.091	0.889	0.010	**poly**	0.205	0.636	0.205	0.081	0.000	0.000
rms	0.194	0.646	0.194	0.075	0.000	0.000	**scontrast4**	0.063	0.608	0.063	0.050	0.000	0.000
rol0.25	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.25**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.65	0.119	0.500	0.119	0.021	0.000	0.000	**sband**	0.119	0.500	0.119	0.021	0.000	0.000
rol2	0.119	0.500	0.119	0.021	0.000	0.000	**scent2048**	0.119	0.500	0.119	0.021	0.000	0.000
sband8000	0.119	0.500	0.119	0.021	0.000	0.000	**rol2**	0.115	0.500	0.115	0.021	0.000	0.000

Model 15: (optimizer: Adam) - continuation							Model 16: (optimizer: SGD) - continuation						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
sband	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.65**	0.111	0.500	0.111	0.020	0.000	0.000
scent2048	0.119	0.500	0.119	0.021	0.000	0.000	**scent8000**	0.039	0.500	0.039	0.008	0.000	0.000
rol0.99	0.119	0.500	0.119	0.021	0.000	0.000	**sband8000**	0.038	0.500	0.038	0.007	0.000	0.000
rol0.45	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.99**	0.038	0.500	0.038	0.007	0.000	0.000
scent8000	0.039	0.500	0.039	0.008	0.000	0.000	**rol0.45**	0.038	0.500	0.038	0.007	0.000	0.000
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.569**	**0.887**	**0.569**	**0.596**	0.768	**0.384**	**mfcc60**	**0.599**	**0.898**	**0.599**	**0.612**	0.848	0.448
mfcc60	0.530	0.883	0.530	0.551	0.787	0.349	**mfcc80**	0.578	0.882	0.578	0.592	0.810	**0.449**
mfcc40	0.490	0.879	0.490	0.517	0.830	0.320	**mfcc40**	0.562	0.887	0.562	0.557	0.827	0.399
stft40	0.460	0.876	0.460	0.456	0.637	0.228	**mel80**	0.495	0.856	0.495	0.503	0.710	0.317
stft60	0.434	0.864	0.434	0.431	0.630	0.220	**mel20**	0.485	0.845	0.485	0.483	0.645	0.269
stft80	0.430	0.862	0.430	0.429	0.628	0.222	**mel40**	0.467	0.853	0.467	0.470	0.632	0.283
mel60	0.440	0.827	0.440	0.424	0.777	0.146	**mel60**	0.460	0.827	0.460	0.469	0.614	0.276
stft20	0.428	0.865	0.428	0.412	0.549	0.188	**stft20**	0.454	0.870	0.454	0.463	0.662	0.271
mel40	0.415	0.828	0.415	0.396	0.753	0.153	**stft40**	0.458	0.871	0.458	0.448	0.616	0.266
mel80	0.404	0.834	0.404	0.384	0.837	0.153	**stft80**	0.432	0.858	0.432	0.441	0.636	0.223
cqt80	0.363	0.803	0.363	0.366	0.701	0.137	**stft60**	0.427	0.843	0.427	0.428	0.609	0.231
cqt20	0.358	0.801	0.358	0.358	0.553	0.106	**cqt80**	0.364	0.790	0.364	0.377	0.608	0.104
cqt40	0.351	0.803	0.351	0.345	0.671	0.119	**cqt40**	0.356	0.786	0.356	0.364	0.628	0.103
cens20	0.354	0.788	0.354	0.340	0.600	0.108	**cens40**	0.344	0.805	0.344	0.345	0.519	0.146
mel20	0.357	0.815	0.357	0.318	0.731	0.127	**cens20**	0.342	0.766	0.342	0.324	0.543	0.106
cens40	0.294	0.785	0.294	0.284	0.519	0.128	**cqt20**	0.319	0.787	0.319	0.308	0.565	0.084
scontrast	0.286	0.768	0.286	0.258	0.714	0.018	**poly10**	0.332	0.792	0.332	0.272	0.741	0.024
scontrast4	0.249	0.704	0.249	0.216	0.357	0.006	**scontrast**	0.309	0.764	0.309	0.262	0.632	0.014
poly10	0.271	0.754	0.271	0.194	**1.000**	0.001	**tonz40**	0.258	0.704	0.258	0.232	0.527	0.035
tonz80	0.207	0.705	0.207	0.170	0.400	0.017	**tonz80**	0.234	0.702	0.234	0.227	0.615	0.029
poly5	0.234	0.715	0.234	0.165	0.000	0.000	**scontrast4**	0.225	0.708	0.225	0.211	0.429	0.004
cens80	0.172	0.718	0.172	0.162	0.432	0.049	**zcr**	0.260	0.672	0.260	0.211	0.610	0.030
tonz20	0.182	0.681	0.182	0.158	0.346	0.011	**tonz20**	0.213	0.690	0.213	0.201	0.593	0.019
tonz40	0.191	0.698	0.191	0.158	0.486	0.020	**cens80**	0.178	0.711	0.178	0.179	0.388	0.087
zcr	0.219	0.690	0.219	0.150	0.636	0.042	**sflat**	0.183	0.636	0.183	0.154	**1.000**	0.011
poly	0.243	0.660	0.243	0.131	0.000	0.000	**poly**	0.251	0.657	0.251	0.152	**1.000**	0.016
poly3	0.226	0.683	0.226	0.126	0.000	0.000	**poly2**	0.191	0.648	0.191	0.144	**1.000**	0.007
poly2	0.219	0.657	0.219	0.114	0.000	0.000	**poly5**	0.205	0.730	0.205	0.120	0.000	0.000
sflat	0.140	0.662	0.140	0.108	0.467	0.017	**rms**	0.222	0.649	0.222	0.106	0.000	0.000
rms	0.215	0.660	0.215	0.104	0.000	0.000	**poly3**	0.171	0.665	0.171	0.101	0.154	0.002
sband	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.25**	0.119	0.511	0.119	0.021	0.000	0.000
rol0.99	0.119	0.500	0.119	0.021	0.000	0.000	**rol2**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.25	0.111	0.500	0.111	0.020	0.000	0.000	**sband8000**	0.119	0.500	0.119	0.021	0.000	0.000
rol2	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.65**	0.119	0.511	0.119	0.021	0.000	0.000
sband8000	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.45**	0.119	0.500	0.119	0.021	0.000	0.000
scent2048	0.039	0.500	0.039	0.008	0.000	0.000	**sband**	0.111	0.500	0.111	0.020	0.000	0.000
rol0.65	0.039	0.500	0.039	0.008	0.000	0.000	**scent2048**	0.039	0.500	0.039	0.008	0.000	0.000
scent8000	0.039	0.500	0.039	0.008	0.000	0.000	**scent8000**	0.039	0.500	0.039	0.008	0.000	0.000
rol0.45	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.99**	0.038	0.500	0.038	0.007	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.1.5 Dropout Rate of 0.8 and 0 for Adam and Adamax

TABLE B.5: Results of the 2 best models for different features and dropout rate of 0.8 and without dropout.

Model 19: (optimizer: Adam; dropout: 0.8)							**Model 20: (optimizer: Adam; dropout: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel20	**0.349**	0.789	**0.349**	**0.291**	0.764	0.050	**mfcc60**	**0.566**	**0.820**	**0.566**	**0.565**	0.577	**0.564**
mel40	0.308	0.754	0.308	0.275	0.788	0.062	**mfcc40**	0.514	0.799	0.514	0.508	0.525	0.508
stft40	0.265	0.750	0.265	0.264	0.683	0.033	**mfcc80**	0.523	0.785	0.523	0.505	0.536	0.519
mfcc80	0.260	0.762	0.260	0.260	0.852	0.055	**mel80**	0.479	0.768	0.479	0.489	0.487	0.464
cens20	0.283	0.717	0.283	0.256	0.872	0.041	**stft60**	0.483	0.807	0.483	0.483	0.497	0.453
cens40	0.270	0.774	0.270	0.245	0.624	**0.081**	**stft20**	0.460	0.792	0.460	0.459	0.483	0.431
mfcc40	0.257	0.739	0.257	0.242	0.544	0.044	**mel60**	0.455	0.772	0.455	0.454	0.471	0.447
stft60	0.251	0.739	0.251	0.238	0.529	0.022	**stft40**	0.444	0.798	0.444	0.442	0.464	0.419
mfcc60	0.262	0.783	0.262	0.237	0.577	0.054	**stft80**	0.435	0.776	0.435	0.424	0.453	0.418
mel80	0.276	0.745	0.276	0.235	0.745	0.049	**cqt80**	0.388	0.743	0.388	0.386	0.400	0.372
stft20	0.245	**0.791**	0.245	0.234	0.741	0.024	**mel40**	0.392	0.732	0.392	0.379	0.403	0.380
mel60	0.272	0.723	0.272	0.222	0.786	0.039	**cqt20**	0.362	0.742	0.362	0.361	0.375	0.333
cqt40	0.174	0.648	0.174	0.197	0.700	0.008	**cqt40**	0.355	0.730	0.355	0.357	0.368	0.348
cqt20	0.208	0.699	0.208	0.185	0.889	0.010	**mel20**	0.358	0.720	0.358	0.353	0.368	0.344
stft80	0.197	0.713	0.197	0.175	0.548	0.020	**scontrast**	0.354	0.730	0.354	0.332	0.395	0.314
cqt80	0.143	0.620	0.143	0.160	0.444	0.005	**cens20**	0.337	0.694	0.337	0.328	0.341	0.327
cens80	0.142	0.680	0.142	0.134	0.229	0.013	**cens40**	0.309	0.685	0.309	0.310	0.317	0.300
poly10	0.235	0.732	0.235	0.133	0.125	0.001	**scontrast4**	0.290	0.726	0.290	0.279	0.344	0.198
zcr	0.191	0.652	0.191	0.122	**1.000**	0.004	**poly10**	0.234	0.721	0.234	0.206	0.293	0.043
tonz80	0.143	0.647	0.143	0.118	0.333	0.004	**cens80**	0.201	0.651	0.201	0.205	0.329	0.170
tonz40	0.142	0.651	0.142	0.114	0.250	0.002	**zcr**	0.220	0.666	0.220	0.199	0.595	0.030
tonz20	0.127	0.630	0.127	0.112	0.500	0.005	**tonz20**	0.201	0.656	0.201	0.192	0.341	0.112
poly5	0.177	0.692	0.177	0.079	0.000	0.000	**tonz40**	0.185	0.650	0.185	0.181	0.212	0.082
sflat	0.099	0.616	0.099	0.078	0.000	0.000	**poly5**	0.198	0.707	0.198	0.176	0.690	0.024
poly2	0.195	0.633	0.195	0.071	0.000	0.000	**tonz80**	0.176	0.631	0.176	0.169	0.237	0.091
rms	0.198	0.642	0.198	0.066	0.000	0.000	**rol0.25**	0.225	0.688	0.225	0.165	0.388	0.023
poly	0.203	0.599	0.203	0.065	0.000	0.000	**sflat**	0.186	0.648	0.186	0.147	0.816	0.037
scontrast	0.116	0.546	0.116	0.029	0.000	0.000	**poly3**	0.162	0.664	0.162	0.132	0.686	0.029
scontrast4	0.053	0.524	0.053	0.026	0.000	0.000	**poly2**	0.179	0.630	0.179	0.132	0.913	0.025
poly3	0.119	0.575	0.119	0.022	0.000	0.000	**poly**	0.147	0.615	0.147	0.130	**1.000**	0.024
rol0.65	0.119	0.500	0.119	0.021	0.000	0.000	**rms**	0.172	0.631	0.172	0.124	0.538	0.008
scent8000	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.65**	0.153	0.659	0.153	0.107	**1.000**	0.008
rol0.99	0.119	0.500	0.119	0.021	0.000	0.000	**scent8000**	0.160	0.650	0.160	0.105	0.900	0.022
rol0.45	0.115	0.500	0.115	0.021	0.000	0.000	**rol0.45**	0.160	0.650	0.160	0.102	0.560	0.033
rol2	0.115	0.500	0.115	0.021	0.000	0.000	**sband**	0.194	0.660	0.194	0.090	0.000	0.000
sband	0.111	0.500	0.111	0.020	0.000	0.000	**scent2048**	0.141	0.646	0.141	0.090	0.619	0.031
scent2048	0.099	0.500	0.099	0.018	0.000	0.000	**sband8000**	0.122	0.653	0.122	0.061	0.000	0.000
sband8000	0.099	0.500	0.099	0.018	0.000	0.000	**rol0.99**	0.119	0.500	0.119	0.021	0.000	0.000
rol0.25	0.038	0.500	0.038	0.007	0.000	0.000	**rol2**	0.119	0.526	0.119	0.021	0.000	0.000
Model 21: (optimizer: Adamax; dropout: 0.8)							**Model 22: (optimizer: Adamax; dropout: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft20	0.266	**0.807**	0.266	**0.260**	0.521	0.030	**mfcc40**	**0.569**	0.825	**0.569**	**0.570**	0.580	**0.564**
cqt80	0.266	0.672	0.266	0.255	0.806	0.035	**mfcc80**	0.493	0.790	0.493	0.509	0.503	0.490
stft40	0.253	0.778	0.253	0.255	0.537	0.026	**stft60**	0.487	0.845	0.487	0.489	0.538	0.429
cens20	**0.275**	0.699	**0.275**	0.255	0.917	0.026	**mel60**	0.491	0.801	0.491	0.488	0.549	0.425
stft80	0.259	0.764	0.259	0.251	0.514	0.023	**mel80**	0.480	0.808	0.480	0.485	0.552	0.437
cqt40	0.258	0.688	0.258	0.246	0.926	0.030	**stft40**	0.476	0.844	0.476	0.476	0.525	0.406
stft60	0.239	0.784	0.239	0.233	0.514	0.023	**stft20**	0.489	**0.854**	0.489	0.475	0.559	0.399
cqt20	0.245	0.666	0.245	0.209	0.889	0.010	**mfcc60**	0.462	0.780	0.462	0.455	0.470	0.458
cens40	0.235	0.692	0.235	0.187	**1.000**	0.020	**stft80**	0.428	0.816	0.428	0.426	0.469	0.385
mfcc60	0.221	0.650	0.221	0.165	0.000	0.000	**mel40**	0.412	0.785	0.412	0.403	0.486	0.344
tonz80	0.161	0.671	0.161	0.151	0.500	0.004	**cqt20**	0.391	0.772	0.391	0.382	0.486	0.288
zcr	0.221	0.719	0.221	0.148	0.654	**0.041**	**cqt40**	0.386	0.784	0.386	0.380	0.469	0.317
tonz40	0.161	0.675	0.161	0.136	0.375	0.004	**mel20**	0.378	0.788	0.378	0.373	0.453	0.290
mel20	0.229	0.612	0.229	0.134	0.714	0.012	**cens40**	0.352	0.750	0.352	0.365	0.388	0.307
tonz20	0.128	0.649	0.128	0.117	0.421	0.010	**cqt80**	0.350	0.771	0.350	0.361	0.408	0.313
poly10	0.222	0.709	0.222	0.115	**1.000**	0.002	**cens20**	0.355	0.750	0.355	0.340	0.405	0.277
cens80	0.123	0.660	0.123	0.111	0.500	0.004	**scontrast**	0.354	0.771	0.354	0.332	0.446	0.265
sflat	0.139	0.680	0.139	0.108	0.722	0.016	**scontrast4**	0.326	0.762	0.326	0.314	0.421	0.184
mfcc80	0.176	0.633	0.176	0.087	0.000	0.000	**cens80**	0.215	0.676	0.215	0.231	0.376	0.162

Model 21: (optimizer: Adamax; dropout: 0.8) - cont.							Model 22: (optimizer: Nadam; dropout: 0) - cont.						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc40	0.155	0.599	0.155	0.084	0.000	0.000	**poly10**	0.263	0.764	0.263	0.222	0.700	0.050
poly2	0.179	0.603	0.179	0.082	0.000	0.000	**tonz80**	0.227	0.693	0.227	0.221	0.405	0.063
rms	0.198	0.650	0.198	0.071	0.000	0.000	**tonz40**	0.217	0.690	0.217	0.217	0.452	0.067
mel40	0.134	0.542	0.134	0.068	0.000	0.000	**tonz20**	0.214	0.660	0.214	0.206	0.477	0.073
mel60	0.128	0.551	0.128	0.060	0.000	0.000	**poly5**	0.244	0.734	0.244	0.195	0.789	0.018
poly5	0.140	0.649	0.140	0.046	0.000	0.000	**zcr**	0.228	0.673	0.228	0.190	0.654	0.041
poly	0.135	0.586	0.135	0.046	0.000	0.000	**sflat**	0.192	0.650	0.192	0.154	0.843	0.051
mel80	0.122	0.572	0.122	0.041	0.000	0.000	**poly3**	0.183	0.650	0.183	0.153	0.957	0.026
scontrast4	0.124	0.552	0.124	0.035	0.000	0.000	**rol0.25**	0.201	0.697	0.201	0.137	0.382	0.016
poly3	0.127	0.582	0.127	0.032	0.000	0.000	**poly2**	0.172	0.626	0.172	0.137	**1.000**	0.024
scontrast	0.116	0.556	0.116	0.029	0.000	0.000	**rms**	0.219	0.640	0.219	0.133	0.000	0.000
rol0.25	0.119	0.500	0.119	0.021	0.000	0.000	**poly**	0.168	0.631	0.168	0.130	0.710	0.026
scent8000	0.119	0.500	0.119	0.021	0.000	0.000	**scent2048**	0.174	0.628	0.174	0.126	0.929	0.016
sband	0.119	0.500	0.119	0.021	0.000	0.000	**rol0.45**	0.190	0.669	0.190	0.112	0.512	0.025
sband8000	0.119	0.500	0.119	0.021	0.000	0.000	**rol2**	0.178	0.649	0.178	0.096	0.000	0.000
rol0.99	0.119	0.500	0.119	0.021	0.000	0.000	**scent8000**	0.143	0.632	0.143	0.095	0.673	0.044
rol2	0.119	0.500	0.119	0.021	0.000	0.000	**sband8000**	0.135	0.617	0.135	0.083	0.000	0.000
rol0.45	0.115	0.500	0.115	0.021	0.000	0.000	**rol0.65**	0.141	0.645	0.141	0.071	0.506	0.051
rol0.65	0.039	0.500	0.039	0.008	0.000	0.000	**sband**	0.141	0.631	0.141	0.052	0.000	0.000
scent2048	0.038	0.500	0.038	0.007	0.000	0.000	**rol0.99**	0.111	0.500	0.111	0.020	0.000	0.000

cont. - continuation; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.2 UrbanSound8K - Combination of Features as Input

B.2.1 Baseline Model Architecture

TABLE B.6: Results of the 6 models for different feature combinations.

Model 1: (optimizer: Adam)							Model 2: (optimizer: SGD)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.669**	**0.925**	**0.669**	**0.682**	**0.736**	**0.638**	**mms80**	**0.630**	0.898	**0.630**	**0.662**	0.845	0.312
mmstftq	0.659	0.910	0.659	0.681	0.719	0.634	**mms60**	0.602	**0.909**	0.602	0.624	0.830	**0.367**
mmsqc	0.627	0.912	0.627	0.658	0.696	0.605	**mfccmel**	0.603	0.894	0.603	0.624	0.855	0.366
mms40	0.634	0.909	0.634	0.652	0.717	0.600	**mfccstft80**	0.591	0.897	0.591	0.611	0.802	0.305
mmcens	0.627	0.894	0.627	0.652	0.678	0.602	**mmstftq**	0.601	0.898	0.601	0.610	0.820	0.348
mmq	0.626	0.916	0.626	0.650	0.688	0.607	**mmcens**	0.597	0.893	0.597	0.610	0.860	0.315
mfccstft80	0.652	0.894	0.652	0.646	0.701	0.618	**mmsqc80**	0.608	0.894	0.608	0.598	**0.862**	0.351
mms80	0.614	0.902	0.614	0.632	0.659	0.585	**mms40**	0.566	0.886	0.566	0.591	0.815	0.305
mms60	0.603	0.893	0.603	0.613	0.671	0.571	**mmq**	0.563	0.890	0.563	0.575	0.832	0.349
mfccstft	0.601	0.899	0.601	0.606	0.679	0.575	**mmsqc**	0.538	0.873	0.538	0.574	0.726	0.275
mfccmel	0.579	0.888	0.579	0.598	0.640	0.559	**mfccstft**	0.485	0.858	0.485	0.501	0.706	0.195
zrsp	0.258	0.675	0.258	0.238	0.647	0.039	**zsrssp**	0.197	0.715	0.197	0.128	0.607	0.044
zsrssp	0.216	0.666	0.216	0.173	0.654	0.041	**zrsp**	0.196	0.718	0.196	0.128	0.607	0.044
scontpoly	0.119	0.511	0.119	0.021	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000
tsp	0.039	0.466	0.039	0.008	0.000	0.000	**scontpoly**	0.038	0.500	0.038	0.007	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.655**	0.926	**0.655**	**0.680**	**0.756**	**0.625**	**mmsqc80**	**0.676**	**0.926**	**0.676**	**0.693**	0.719	**0.652**
mmq	0.640	0.912	0.640	0.668	0.697	0.601	**mmstftq**	0.663	0.914	0.663	0.687	0.712	0.639
mmsqc80	0.634	0.900	0.634	0.656	0.677	0.585	**mms80**	0.650	0.901	0.650	0.676	0.687	0.634
mfccstft80	0.628	0.915	0.628	0.654	0.695	0.585	**mmsqc**	0.658	0.919	0.658	0.676	0.709	0.620
mms40	0.637	0.921	0.637	0.651	0.752	0.585	**mmq**	0.645	0.916	0.645	0.669	**0.723**	0.608
mmstftq	0.611	0.912	0.611	0.639	0.694	0.579	**mms40**	0.638	0.913	0.638	0.650	0.695	0.620
mmsqc	0.621	**0.927**	0.621	0.636	0.714	0.588	**mms60**	0.618	0.902	0.618	0.647	0.674	0.590
mms80	0.614	0.909	0.614	0.629	0.683	0.571	**mmcens**	0.602	0.884	0.602	0.618	0.635	0.595
mfccstft	0.612	0.915	0.612	0.629	0.712	0.575	**mfccmel**	0.599	0.878	0.599	0.617	0.650	0.569
mmcens	0.609	0.904	0.609	0.626	0.671	0.577	**mfccstft**	0.597	0.897	0.597	0.605	0.673	0.545
mfccmel	0.596	0.903	0.596	0.623	0.666	0.559	**mfccstft80**	0.590	0.886	0.590	0.604	0.637	0.569
zrsp	0.243	0.678	0.243	0.205	0.600	0.032	**zrsp**	0.225	0.678	0.225	0.200	0.583	0.025
zsrssp	0.222	0.684	0.222	0.172	0.647	0.039	**zsrssp**	0.221	0.659	0.221	0.194	0.605	0.027
tsp	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
scontpoly	0.038	0.500	0.038	0.007	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfccstft80	0.247	0.665	0.247	**0.217**	0.245	0.029	**mmstftq**	0.409	**0.828**	0.409	**0.419**	0.933	**0.067**
mfccmel	0.249	0.636	0.249	0.186	0.295	0.121	**mmsqc80**	**0.423**	0.827	**0.423**	0.407	0.786	0.053
mms40	**0.251**	0.610	**0.251**	0.180	0.235	0.127	**mmsqc**	0.384	0.780	0.384	0.383	0.830	0.047
mmsqc80	0.208	0.638	0.208	0.175	0.269	0.116	**mms60**	0.338	0.776	0.338	0.348	0.907	0.047
mmq	0.201	**0.688**	0.201	0.169	0.261	**0.145**	**mfccstft**	0.370	0.789	0.370	0.345	0.914	0.038
mmstftq	0.221	0.647	0.221	0.163	**0.403**	0.104	**mfccmel**	0.343	0.783	0.343	0.314	0.929	0.047
mms80	0.191	0.609	0.191	0.146	0.235	0.092	**mmq**	0.294	0.765	0.294	0.300	**0.935**	0.051
mmsqc	0.191	0.618	0.191	0.146	0.318	0.092	**mms40**	0.288	0.730	0.288	0.285	0.921	0.042
mms60	0.177	0.601	0.177	0.124	0.202	0.087	**mfccstft80**	0.277	0.723	0.277	0.282	0.892	0.039
mmcens	0.173	0.643	0.173	0.122	0.236	0.100	**mmcens**	0.258	0.730	0.258	0.257	0.907	0.047
mfccstft	0.152	0.597	0.152	0.072	0.195	0.124	**mms80**	0.243	0.706	0.243	0.235	0.923	0.043
zsrssp	0.159	0.562	0.159	0.064	0.000	0.000	**zsrssp**	0.179	0.688	0.179	0.080	0.000	0.000
zrsp	0.131	0.549	0.131	0.047	0.000	0.000	**zrsp**	0.168	0.668	0.168	0.064	0.000	0.000
scontpoly	0.119	0.511	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.522	0.119	0.021	0.000	0.000
tsp	0.099	0.465	0.099	0.018	0.000	0.000	**tsp**	0.039	0.500	0.039	0.008	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.2.2 Extra Layer

TABLE B.7: Results of the 4 models with an extra dense and dropout layer for the different feature combinations.

Model 7: (optimizer: Adam)							**Model 8: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.671**	0.916	**0.671**	**0.686**	0.736	**0.648**	**mmsqc**	0.582	0.889	0.582	**0.616**	0.804	0.300
mmq	0.658	0.905	0.658	0.685	0.713	0.631	**mmq**	**0.602**	0.893	**0.602**	0.612	0.812	0.274
mms40	0.663	0.912	0.663	0.683	**0.750**	0.644	**mmsqc80**	0.595	0.890	0.595	0.610	0.822	0.299
mfccstft	0.664	**0.918**	0.664	0.683	0.735	0.636	**mfccmel**	0.566	**0.900**	0.566	0.593	0.832	**0.302**
mmstftq	0.634	0.914	0.634	0.656	0.700	0.599	**mfccstft80**	0.570	0.876	0.570	0.585	0.829	0.208
mms60	0.625	0.906	0.625	0.653	0.697	0.594	**mmcens**	0.572	0.884	0.572	0.583	0.811	0.257
mfccstft80	0.625	0.895	0.625	0.651	0.699	0.593	**mmstftq**	0.527	0.875	0.527	0.567	0.807	0.286
mfccmel	0.626	0.896	0.626	0.646	0.690	0.571	**mms80**	0.548	0.878	0.548	0.565	0.805	0.276
mmsqc	0.609	0.896	0.609	0.643	0.682	0.571	**mms40**	0.532	0.872	0.532	0.545	**0.833**	0.286
mmcens	0.624	0.899	0.624	0.642	0.683	0.590	**mms60**	0.502	0.871	0.502	0.523	0.824	0.240
mms80	0.596	0.887	0.596	0.618	0.664	0.573	**mfccstft**	0.502	0.864	0.502	0.520	0.744	0.177
zrsp	0.246	0.668	0.246	0.222	0.615	0.029	**zsrssp**	0.190	0.709	0.190	0.125	0.607	0.044
zsrssp	0.213	0.674	0.213	0.174	0.614	0.032	**zrsp**	0.182	0.706	0.182	0.119	0.585	0.045
scontpoly	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
tsp	0.038	0.500	0.038	0.007	0.000	0.000	**tsp**	0.039	0.500	0.039	0.008	0.000	0.000
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.658**	**0.919**	**0.658**	**0.680**	**0.710**	**0.633**	**mmstftq**	**0.699**	0.911	**0.699**	**0.720**	0.756	**0.664**
mms40	0.628	0.913	0.628	0.655	0.690	0.568	**mmsqc**	0.687	**0.922**	0.687	0.703	**0.777**	0.645
mfccstft	0.626	0.914	0.626	0.650	0.703	0.576	**mmsqc80**	0.676	0.910	0.676	0.689	0.713	0.650
mms60	0.626	0.901	0.626	0.648	0.687	0.591	**mms80**	0.654	0.906	0.654	0.670	0.706	0.599
mmsqc	0.606	0.907	0.606	0.628	0.694	0.578	**mms40**	0.648	0.918	0.648	0.668	0.703	0.625
mmcens	0.596	0.897	0.596	0.615	0.655	0.552	**mfccstft**	0.649	0.899	0.649	0.659	0.707	0.583
mms80	0.594	0.905	0.594	0.608	0.669	0.575	**mmq**	0.632	0.900	0.632	0.650	0.670	0.605
mmstftq	0.582	0.898	0.582	0.602	0.636	0.559	**mfccstft80**	0.622	0.912	0.622	0.649	0.691	0.581
mfccmel	0.578	0.901	0.578	0.593	0.657	0.529	**mms60**	0.602	0.894	0.602	0.629	0.659	0.559
mfccstft80	0.571	0.890	0.571	0.580	0.655	0.545	**mfccmel**	0.602	0.878	0.602	0.616	0.628	0.551
mmq	0.575	0.892	0.575	0.575	0.624	0.550	**mmcens**	0.581	0.886	0.581	0.590	0.624	0.540
zrsp	0.250	0.670	0.250	0.214	0.605	0.027	**zsrssp**	0.249	0.671	0.249	0.224	0.583	0.025
zsrssp	0.217	0.676	0.217	0.168	0.647	0.039	**zrsp**	0.210	0.667	0.210	0.171	0.617	0.035
scontpoly	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
tsp	0.039	0.500	0.039	0.008	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.2.3 Dropout Rate of 0.2

TABLE B.8: Results of the 4 models with a dropout rate of 0.2 for different feature combinations.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.668**	**0.895**	**0.668**	**0.693**	**0.691**	**0.658**	**mms80**	**0.627**	0.904	**0.627**	**0.641**	**0.731**	**0.550**
mmq	0.646	0.889	0.646	0.670	0.675	0.633	**mfccstft80**	0.594	0.892	0.594	0.593	0.718	0.477
mmstftq	0.642	0.881	0.642	0.667	0.664	0.637	**mmsqc**	0.591	0.890	0.591	0.589	0.706	0.454
mms40	0.637	0.889	0.637	0.655	0.677	0.624	**mfccstft**	0.591	0.869	0.591	0.587	0.697	0.425
mmsqc80	0.622	0.866	0.622	0.641	0.639	0.618	**mms60**	0.588	0.894	0.588	0.586	0.663	0.522
mfccstft80	0.627	0.879	0.627	0.636	0.657	0.619	**mmcens**	0.579	0.897	0.579	0.579	0.689	0.476
mmcens	0.619	0.856	0.619	0.630	0.649	0.608	**mmq**	0.581	0.886	0.581	0.577	0.704	0.498
mmsqc	0.618	0.869	0.618	0.628	0.644	0.603	**mfccmel**	0.587	0.900	0.587	0.570	0.696	0.497
mms80	0.602	0.866	0.602	0.619	0.624	0.575	**mms40**	0.558	0.905	0.558	0.563	0.671	0.470
mfccmel	0.584	0.841	0.584	0.593	0.604	0.563	**mmsqc80**	0.535	0.884	0.535	0.562	0.658	0.450
mfccstft	0.589	0.863	0.589	0.592	0.610	0.579	**mmstftq**	0.554	0.885	0.554	0.561	0.638	0.459
zsrssp	0.240	0.674	0.240	0.221	0.583	0.025	**zsrssp**	0.195	0.720	0.195	0.124	0.607	0.044
zrsp	0.216	0.667	0.216	0.199	0.422	0.023	**zrsp**	0.189	0.711	0.189	0.121	0.585	0.045
scontpoly	0.119	0.500	0.119	0.021	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000
tsp	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.038	0.500	0.038	0.007	0.000	0.000
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc	**0.685**	0.907	**0.685**	**0.697**	**0.708**	**0.664**	**mms80**	**0.675**	0.898	**0.675**	**0.695**	**0.697**	**0.668**
mmstftq	0.670	**0.914**	0.670	0.691	0.691	0.657	**mmsqc80**	0.639	0.882	0.639	0.668	0.657	0.624
mmsqc80	0.651	0.894	0.651	0.673	0.670	0.636	**mmsqc**	0.644	0.877	0.644	0.665	0.665	0.636
mmcens	0.632	0.889	0.632	0.655	0.671	0.614	**mms60**	0.650	**0.899**	0.650	0.664	0.665	0.639
mmq	0.645	0.903	0.645	0.647	0.678	0.621	**mmstftq**	0.631	0.882	0.631	0.654	0.652	0.625
mfccmel	0.631	0.901	0.631	0.646	0.679	0.620	**mms40**	0.636	0.870	0.636	0.653	0.663	0.625
mms80	0.618	0.887	0.618	0.644	0.647	0.601	**mfccmel**	0.639	0.870	0.639	0.642	0.664	0.632
mms40	0.625	0.885	0.625	0.639	0.671	0.599	**mfccstft80**	0.624	0.865	0.624	0.640	0.646	0.609
mms60	0.612	0.877	0.612	0.633	0.634	0.588	**mmq**	0.603	0.854	0.603	0.634	0.617	0.591
mfccstft	0.602	0.873	0.602	0.609	0.638	0.577	**mmcens**	0.611	0.866	0.611	0.626	0.625	0.601
mfccstft80	0.597	0.858	0.597	0.604	0.623	0.583	**mfccstft**	0.612	0.872	0.612	0.605	0.627	0.600
zrsp	0.240	0.674	0.240	0.225	0.615	0.029	**zsrssp**	0.235	0.672	0.235	0.208	0.615	0.029
zsrssp	0.211	0.668	0.211	0.172	0.647	0.039	**zrsp**	0.208	0.661	0.208	0.184	0.595	0.026
scontpoly	0.115	0.500	0.115	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
tsp	0.111	0.500	0.111	0.020	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.2.4 Dropout Rate of 0.6

TABLE B.9: Results of the 4 models with a dropout rate of 0.6 for different feature combinations.

Model 15: (optimizer: Adam)							**Model 16: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc	**0.676**	0.923	**0.676**	**0.696**	0.781	0.551	**mfccmel**	0.271	0.764	0.271	**0.288**	**0.877**	**0.085**
mmsqc80	0.650	**0.927**	0.650	0.684	0.807	0.550	**mfccstft80**	**0.292**	0.779	**0.292**	0.267	0.804	0.049
mms60	0.662	0.916	0.662	0.679	**0.828**	**0.568**	**mms40**	0.257	**0.794**	0.257	0.247	0.865	0.054
mms80	0.651	0.910	0.651	0.674	0.786	0.522	**mfccstft**	0.201	0.756	0.201	0.205	0.769	0.048
mmstftq	0.634	0.905	0.634	0.662	0.757	0.556	**mmstftq**	0.201	0.710	0.201	0.162	0.667	0.017
mmq	0.632	0.912	0.632	0.647	0.782	0.536	**mms80**	0.198	0.717	0.198	0.160	0.727	0.029
mmcens	0.622	0.897	0.622	0.644	0.735	0.547	**mmsqc80**	0.190	0.625	0.190	0.143	0.833	0.042
mfccmel	0.628	0.912	0.628	0.644	0.821	0.511	**mmsqc**	0.246	0.720	0.246	0.140	0.684	0.047
mfccstft80	0.634	0.917	0.634	0.642	0.786	0.519	**mmcens**	0.222	0.667	0.222	0.132	0.483	0.035
mms40	0.603	0.912	0.603	0.628	0.733	0.472	**zrsp**	0.194	0.705	0.194	0.126	0.647	0.039
mfccstft	0.569	0.905	0.569	0.595	0.781	0.455	**zsrssp**	0.186	0.702	0.186	0.122	0.660	0.042
zsrssp	0.257	0.678	0.257	0.202	0.533	0.019	**mms60**	0.080	0.538	0.080	0.056	0.250	0.002
zrsp	0.249	0.675	0.249	0.198	0.563	0.022	**mmq**	0.116	0.496	0.116	0.029	0.250	0.002
tsp	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
scontpoly	0.099	0.500	0.099	0.018	0.000	0.000	**tsp**	0.111	0.506	0.111	0.020	0.000	0.000
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.636**	0.911	**0.636**	**0.657**	0.767	**0.489**	**mmsqc**	**0.704**	**0.929**	**0.704**	**0.721**	0.789	**0.594**
mfccstft80	0.621	**0.913**	0.621	0.640	0.787	0.478	**mms80**	0.675	0.918	0.675	0.696	0.827	0.558
mmq	0.575	0.870	0.575	0.595	0.787	0.296	**mms40**	0.654	0.921	0.654	0.680	0.788	0.536
mfccstft	0.557	0.881	0.557	0.585	0.732	0.373	**mmsqc80**	0.640	0.925	0.640	0.677	0.744	0.554
mmcens	0.571	0.879	0.571	0.583	0.746	0.389	**mmstftq**	0.644	0.907	0.644	0.669	0.741	0.541
mmsqc	0.557	0.896	0.557	0.578	0.695	0.427	**mmcens**	0.638	0.909	0.638	0.658	0.825	0.546
mms80	0.563	0.891	0.563	0.576	0.778	0.361	**mms60**	0.625	0.910	0.625	0.652	0.800	0.521
mmsqc80	0.545	0.888	0.545	0.572	0.701	0.362	**mmq**	0.631	0.918	0.631	0.650	0.777	0.523
mms60	0.557	0.900	0.557	0.569	0.755	0.397	**mfccstft80**	0.624	0.915	0.624	0.645	0.806	0.522
mfccmel	0.558	0.889	0.558	0.562	**0.809**	0.335	**mfccmel**	0.603	0.896	0.603	0.628	0.746	0.503
mms40	0.542	0.892	0.542	0.555	0.763	0.338	**mfccstft**	0.600	0.898	0.600	0.616	**0.845**	0.458
zrsp	0.222	0.687	0.222	0.154	0.647	0.039	**zsrssp**	0.211	0.668	0.211	0.160	0.633	0.037
zsrssp	0.221	0.687	0.221	0.152	0.647	0.039	**zrsp**	0.220	0.670	0.220	0.153	0.600	0.032
tsp	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.119	0.500	0.119	0.021	0.000	0.000
scontpoly	0.111	0.500	0.111	0.020	0.000	0.000	**tsp**	0.039	0.500	0.039	0.008	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.2.5 Extra Layer and Dropout Rate

TABLE B.10: Results of the 3 models with an extra layer and dropout rate of 0.6 and 0.8 for Adam and Nadam optimizer and 0.2 and 0 for Adamax optimizer with different feature combinations.

Model 23: (optimizer: Adam and dropout rate: 0.6)							**Model 24: (optimizer: Adam and dropout rate: 0.8)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.626**	0.898	**0.626**	**0.653**	0.775	0.466	**mms40**	**0.214**	0.639	**0.214**	**0.188**	0.875	0.042
mmsqc	0.613	0.902	0.613	0.651	0.763	0.430	**mfccstft**	0.189	**0.680**	0.189	0.165	0.854	0.042
mmstftq	0.613	0.895	0.613	0.643	0.799	**0.474**	**mms60**	0.166	0.571	0.166	0.143	0.957	0.026
mfccstft80	0.605	**0.907**	0.605	0.615	0.802	0.441	**mms80**	0.168	0.611	0.168	0.130	0.952	0.024
mms40	0.588	0.881	0.588	0.614	0.743	0.421	**mmsqc80**	0.154	0.574	0.154	0.123	**1.000**	0.023
mfccstft	0.587	0.891	0.587	0.612	**0.817**	0.373	**mmsqc**	0.189	0.614	0.189	0.113	0.867	**0.047**
mmsqc80	0.591	0.879	0.591	0.610	0.741	0.399	**mfccstft80**	0.125	0.614	0.125	0.108	0.958	0.027
mms80	0.572	0.891	0.572	0.588	0.725	0.416	**mmcens**	0.121	0.577	0.121	0.107	**1.000**	0.026
mfccmel	0.570	0.889	0.570	0.586	0.786	0.458	**mmq**	0.128	0.572	0.128	0.104	**1.000**	0.024
mmq	0.541	0.881	0.541	0.569	0.765	0.330	**mmstftq**	0.141	0.573	0.141	0.103	**1.000**	0.018
mmcens	0.551	0.882	0.551	0.565	0.715	0.368	**mfccmel**	0.124	0.560	0.124	0.097	**1.000**	0.017
zsrssp	0.215	0.649	0.215	0.149	0.714	0.006	**zsrssp**	0.165	0.622	0.165	0.070	0.000	0.000
zrsp	0.210	0.659	0.210	0.144	0.526	0.012	**zrsp**	0.162	0.608	0.162	0.060	0.000	0.000
tsp	0.038	0.466	0.038	0.007	0.000	0.000	**tsp**	0.119	0.500	0.119	0.021	0.000	0.000
scontpoly	0.038	0.500	0.038	0.007	0.000	0.000	**scontpoly**	0.115	0.500	0.115	0.021	0.000	0.000
Model 25: (optimizer: Adamax and dropout rate: 0.2)							**Model 26: (optimizer: Adamax and dropout rate: 0)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	0.662	**0.906**	0.662	**0.690**	0.689	0.650	**mmstftq**	0.618	0.850	0.618	0.636	0.624	0.608
mms80	0.664	0.894	0.664	0.685	0.692	0.652	**mms80**	0.609	0.855	0.609	0.626	0.619	0.602
mms40	**0.671**	0.900	**0.671**	0.684	**0.718**	**0.657**	**mmsqc80**	0.575	0.823	0.575	0.601	0.583	0.572
mmstftq	0.649	0.893	0.649	0.670	0.668	0.638	**mmcens**	0.599	0.849	0.599	0.599	0.612	0.593
mfccmel	0.650	0.890	0.650	0.654	0.687	0.638	**mms40**	0.582	0.825	0.582	0.595	0.594	0.577
mfccstft80	0.642	0.886	0.642	0.653	0.661	0.631	**mmsqc**	0.575	0.845	0.575	0.592	0.586	0.572
mmcens	0.636	0.887	0.636	0.642	0.662	0.607	**mms60**	0.564	0.819	0.564	0.583	0.573	0.556
mmsqc	0.621	0.876	0.621	0.637	0.646	0.599	**mfccmel**	0.570	0.837	0.570	0.581	0.575	0.560
mms60	0.608	0.886	0.608	0.622	0.642	0.572	**mmq**	0.539	0.831	0.539	0.561	0.549	0.526
mfccstft	0.614	0.875	0.614	0.618	0.639	0.591	**mfccstft80**	0.560	0.827	0.560	0.554	0.565	0.552
mmq	0.565	0.867	0.565	0.583	0.610	0.548	**mfccstft**	0.540	0.829	0.540	0.531	0.548	0.533
zrsp	0.249	0.664	0.249	0.230	0.633	0.037	**zsrssp**	0.211	0.666	0.211	0.184	**0.647**	0.039
zsrssp	0.220	0.680	0.220	0.185	0.647	0.039	**zrsp**	0.209	0.661	0.209	0.182	0.617	0.035
scontpoly	0.119	0.500	0.119	0.021	0.000	0.000	**scontpoly**	0.115	0.500	0.115	0.021	0.000	0.000
tsp	0.099	0.500	0.099	0.018	0.000	0.000	**tsp**	0.039	0.500	0.039	0.008	0.000	0.000
Model 27: (optimizer: Nadam and dropout rate: 0.6)							**Model 28: (optimizer: Nadam and dropout rate: 0.8)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.639**	0.897	**0.639**	**0.668**	0.826	**0.465**	**mmsqc**	0.182	0.638	0.182	**0.168**	0.879	0.035
mmsqc	0.615	0.898	0.615	0.638	0.807	0.440	**mms80**	**0.204**	0.651	**0.204**	0.167	**1.000**	0.026
mmsqc80	0.615	0.893	0.615	0.637	0.780	0.436	**mms40**	0.179	0.632	0.179	0.154	0.857	**0.036**
mms60	0.611	**0.907**	0.611	0.635	0.771	0.458	**mms60**	0.111	0.596	0.111	0.142	**1.000**	0.026
mms40	0.605	0.901	0.605	0.627	0.765	0.464	**mmcens**	0.160	0.615	0.160	0.125	**1.000**	0.026
mfccstft	0.603	0.904	0.603	0.625	**0.843**	0.404	**mfccstft80**	0.141	**0.653**	0.141	0.119	0.957	0.026
mms80	0.578	0.887	0.578	0.608	0.691	0.437	**mmstftq**	0.155	0.578	0.155	0.117	**1.000**	0.014
mfccmel	0.593	0.886	0.593	0.606	0.741	0.437	**mfccstft**	0.124	0.605	0.124	0.111	0.857	0.022
mfccstft80	0.570	0.891	0.570	0.600	0.713	0.404	**mfccmel**	0.136	0.570	0.136	0.105	**1.000**	0.017
mmq	0.569	0.887	0.569	0.597	0.798	0.430	**mmq**	0.145	0.550	0.145	0.102	**1.000**	0.022
mmcens	0.559	0.879	0.559	0.576	0.735	0.362	**mmsqc80**	0.065	0.548	0.065	0.087	**1.000**	0.016
zsrssp	0.232	0.663	0.232	0.184	0.524	0.013	**zsrssp**	0.157	0.638	0.157	0.067	0.000	0.000
zrsp	0.214	0.654	0.214	0.150	0.522	0.014	**zrsp**	0.157	0.589	0.157	0.059	0.000	0.000
tsp	0.119	0.511	0.119	0.021	0.000	0.000	**tsp**	0.038	0.500	0.038	0.007	0.000	0.000
scontpoly	0.038	0.500	0.038	0.007	0.000	0.000	**scontpoly**	0.038	0.500	0.038	0.007	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.3 ESC Datasets - Single Feature Input

B.3.1 Baseline Model Architecture

TABLE B.11: Results of the 6 models for different features - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.613**	0.880	**0.613**	**0.601**	0.642	0.425	**mel60**	**0.463**	0.852	**0.463**	**0.442**	0.526	0.125
mfcc80	0.588	0.879	0.588	0.574	**0.667**	0.425	**mel80**	0.450	0.858	0.450	0.433	**0.714**	**0.188**
mel40	0.563	0.891	0.563	0.567	0.603	0.438	**mel40**	0.450	0.875	0.450	0.427	0.500	0.113
mel20	0.575	**0.897**	0.575	0.558	0.617	**0.463**	**mel20**	0.438	**0.886**	0.438	0.413	0.455	0.063
mel60	0.538	0.843	0.538	0.528	0.548	0.425	**mfcc80**	0.425	0.852	0.425	0.364	0.455	0.125
mel80	0.525	0.895	0.525	0.520	0.611	0.413	**mfcc60**	0.388	0.863	0.388	0.314	0.517	**0.188**
mfcc40	0.538	0.881	0.538	0.516	**0.667**	0.350	**mfcc40**	0.350	0.795	0.350	0.294	0.375	0.113
stft80	0.513	0.888	0.513	0.490	0.643	0.225	**stft80**	0.325	0.754	0.325	0.258	0.000	0.000
stft40	0.488	0.885	0.488	0.482	0.571	0.150	**stft60**	0.263	0.737	0.263	0.197	0.000	0.000
stft60	0.500	0.883	0.500	0.480	0.615	0.200	**cens40**	0.275	0.754	0.275	0.176	0.000	0.000
stft20	0.413	0.868	0.413	0.406	0.588	0.125	**cens80**	0.225	0.706	0.225	0.154	0.000	0.000
cens20	0.350	0.704	0.350	0.335	0.405	0.188	**cqt40**	0.175	0.636	0.175	0.115	0.000	0.000
cqt80	0.350	0.784	0.350	0.320	0.385	0.125	**stft40**	0.188	0.721	0.188	0.114	0.000	0.000
cqt40	0.313	0.777	0.313	0.288	0.366	0.188	**stft20**	0.213	0.715	0.213	0.113	0.000	0.000
cqt20	0.288	0.799	0.288	0.269	0.333	0.100	**cqt80**	0.163	0.637	0.163	0.107	0.000	0.000
cens80	0.313	0.773	0.313	0.252	0.370	0.213	**cens20**	0.125	0.568	0.125	0.087	0.000	0.000
cens40	0.250	0.795	0.250	0.242	0.234	0.188	**cqt20**	0.138	0.609	0.138	0.075	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel40	**0.488**	0.878	**0.488**	**0.465**	0.765	**0.163**	**mfcc40**	**0.650**	0.894	**0.650**	**0.644**	**0.723**	0.425
mfcc60	0.475	**0.890**	0.475	0.410	0.462	0.075	**mel40**	0.588	**0.900**	0.588	0.597	0.603	**0.438**
stft80	0.425	0.859	0.425	0.409	0.667	0.025	**mfcc60**	0.600	0.877	0.600	0.594	0.630	0.363
mel80	0.425	0.831	0.425	0.405	0.579	0.138	**mel20**	0.600	0.899	0.600	0.587	0.545	0.375
stft40	0.400	0.859	0.400	0.385	0.500	0.013	**mfcc80**	0.575	0.874	0.575	0.558	0.630	0.363
mel60	0.400	0.836	0.400	0.383	0.533	0.100	**mel80**	0.550	0.889	0.550	0.549	0.635	0.413
mel20	0.400	0.870	0.400	0.373	0.714	0.063	**mel60**	0.500	0.857	0.500	0.493	0.534	0.388
stft60	0.388	0.850	0.388	0.351	0.500	0.013	**stft40**	0.463	0.874	0.463	0.450	0.500	0.188
cqt40	0.350	0.754	0.350	0.317	0.167	0.013	**stft80**	0.488	0.857	0.488	0.436	0.613	0.238
cens80	0.363	0.820	0.363	0.312	0.444	0.150	**stft20**	0.413	0.868	0.413	0.383	0.524	0.138
cqt80	0.325	0.746	0.325	0.303	0.000	0.000	**stft60**	0.388	0.873	0.388	0.355	0.536	0.188
mfcc40	0.313	0.823	0.313	0.282	0.600	0.075	**cqt20**	0.338	0.807	0.338	0.330	0.452	0.175
stft20	0.313	0.839	0.313	0.262	**1.000**	0.013	**cens80**	0.363	0.774	0.363	0.317	0.422	0.238
cens40	0.275	0.847	0.275	0.250	0.478	0.138	**cqt40**	0.325	0.789	0.325	0.303	0.385	0.188
cqt20	0.263	0.767	0.263	0.245	0.000	0.000	**cqt80**	0.325	0.773	0.325	0.289	0.366	0.188
cens20	0.263	0.682	0.263	0.221	0.500	0.050	**cens20**	0.288	0.676	0.288	0.281	0.333	0.175
mfcc80	0.225	0.799	0.225	0.177	0.308	0.050	**cens40**	0.225	0.792	0.225	0.216	0.239	0.200
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
cens40	0.175	0.532	0.175	**0.116**	0.000	0.000	**mel60**	0.400	0.838	0.400	**0.383**	0.647	0.138
cens80	**0.225**	0.498	**0.225**	0.115	0.000	0.000	**mel40**	**0.413**	0.828	**0.413**	0.367	0.667	0.125
mfcc40	0.125	0.482	0.125	0.093	0.094	0.075	**mel80**	0.400	0.813	0.400	0.365	0.565	0.163
mel80	0.163	0.543	0.163	0.082	**0.417**	**0.125**	**mfcc60**	**0.413**	0.830	**0.413**	0.361	0.571	**0.200**
mel40	0.088	0.479	0.088	0.078	0.000	0.000	**mel20**	0.375	0.843	0.375	0.320	**0.778**	0.088
stft40	0.150	**0.576**	0.150	0.075	0.000	0.000	**mfcc80**	0.388	**0.847**	0.388	0.305	0.464	0.163
cens20	0.100	0.453	0.100	0.049	0.000	0.000	**mfcc40**	0.350	0.842	0.350	0.300	0.450	0.113
mfcc60	0.075	0.425	0.075	0.037	0.017	0.013	**cens80**	0.288	0.704	0.288	0.197	0.000	0.000
mel60	0.050	0.463	0.050	0.037	0.083	0.013	**cens40**	0.200	0.620	0.200	0.118	0.000	0.000
cqt20	0.100	0.523	0.100	0.037	0.000	0.000	**cens20**	0.125	0.520	0.125	0.048	0.000	0.000
mel20	0.100	0.545	0.100	0.036	0.182	0.025	**stft40**	0.100	0.616	0.100	0.024	0.000	0.000
stft60	0.088	0.525	0.088	0.036	0.000	0.000	**stft20**	0.100	0.590	0.100	0.023	0.000	0.000
mfcc80	0.050	0.443	0.050	0.020	0.046	0.038	**stft80**	0.100	0.622	0.100	0.021	0.000	0.000
cqt40	0.100	0.491	0.100	0.018	0.000	0.000	**stft60**	0.100	0.596	0.100	0.020	0.000	0.000
stft20	0.088	0.537	0.088	0.017	0.000	0.000	**cqt80**	0.088	0.560	0.088	0.019	0.000	0.000
stft80	0.025	0.501	0.025	0.017	0.000	0.000	**cqt40**	0.100	0.538	0.100	0.019	0.000	0.000
cqt80	0.075	0.530	0.075	0.016	0.000	0.000	**cqt20**	0.100	0.550	0.100	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.12: Results of the 6 models for different features - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.313**	0.854	**0.313**	**0.293**	0.504	0.170	**mel80**	**0.173**	0.780	**0.173**	**0.139**	0.394	**0.033**
mfcc60	0.305	0.856	0.305	0.290	0.503	**0.188**	**mel60**	0.158	0.781	0.158	0.131	**0.480**	0.030
mfcc40	0.288	0.850	0.288	0.271	0.456	0.130	**mel40**	0.153	0.796	0.153	0.126	0.429	0.030
mel20	0.263	**0.858**	0.263	0.249	0.519	0.105	**mfcc80**	0.143	0.782	0.143	0.123	0.375	0.015
mel40	0.240	0.855	0.240	0.228	**0.540**	0.118	**mel20**	0.148	**0.800**	0.148	0.118	0.450	0.023
mel60	0.225	0.833	0.225	0.207	0.402	0.088	**mfcc40**	0.138	0.783	0.138	0.114	0.235	0.010
mel80	0.220	0.835	0.220	0.202	0.388	0.095	**mfcc60**	0.133	0.782	0.133	0.104	0.333	0.015
stft60	0.153	0.779	0.153	0.130	0.395	0.038	**cens40**	0.060	0.594	0.060	0.037	0.000	0.000
stft20	0.138	0.789	0.138	0.119	0.444	0.040	**cens20**	0.048	0.550	0.048	0.036	0.000	0.000
stft40	0.145	0.786	0.145	0.113	0.395	0.043	**stft80**	0.065	0.685	0.065	0.030	0.000	0.000
stft80	0.130	0.775	0.130	0.109	0.289	0.033	**cens80**	0.050	0.599	0.050	0.027	0.000	0.000
cqt80	0.123	0.758	0.123	0.099	0.257	0.023	**cqt80**	0.045	0.645	0.045	0.021	0.000	0.000
cqt20	0.110	0.768	0.110	0.090	0.393	0.028	**cqt20**	0.035	0.571	0.035	0.018	0.000	0.000
cqt40	0.110	0.743	0.110	0.084	0.324	0.028	**stft60**	0.055	0.682	0.055	0.015	0.000	0.000
cens20	0.093	0.715	0.093	0.079	0.235	0.020	**stft20**	0.045	0.642	0.045	0.015	0.000	0.000
cens40	0.078	0.719	0.078	0.070	0.141	0.028	**stft40**	0.045	0.675	0.045	0.013	0.000	0.000
cens80	0.078	0.680	0.078	0.054	0.153	0.045	**cqt40**	0.028	0.612	0.028	0.004	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.188**	**0.845**	**0.188**	**0.146**	0.214	0.008	**mfcc80**	**0.313**	0.850	**0.313**	**0.297**	0.511	0.168
stft40	0.148	0.794	0.148	0.119	0.533	0.020	**mfcc60**	0.310	0.848	0.310	0.290	0.515	**0.173**
mel40	0.128	0.773	0.128	0.118	0.421	0.020	**mfcc40**	0.300	**0.866**	0.300	0.275	0.496	0.145
stft60	0.153	0.797	0.153	0.118	0.538	0.018	**mel20**	0.258	0.854	0.258	0.242	**0.556**	0.113
stft20	0.153	0.795	0.153	0.117	**0.636**	0.018	**mel40**	0.238	0.850	0.238	0.218	0.543	0.110
stft80	0.153	0.793	0.153	0.117	0.563	0.023	**mel80**	0.228	0.845	0.228	0.215	0.488	0.103
mel80	0.133	0.752	0.133	0.116	0.619	**0.033**	**mel60**	0.200	0.844	0.200	0.188	0.468	0.093
mel20	0.125	0.795	0.125	0.106	0.333	0.018	**stft60**	0.168	0.780	0.168	0.136	0.381	0.040
mfcc60	0.115	0.775	0.115	0.098	0.077	0.003	**stft20**	0.143	0.799	0.143	0.120	0.441	0.038
mel60	0.113	0.745	0.113	0.097	0.550	0.028	**stft80**	0.150	0.767	0.150	0.118	0.366	0.038
mfcc40	0.103	0.747	0.103	0.088	0.400	0.005	**stft40**	0.135	0.786	0.135	0.108	0.359	0.035
cens40	0.100	0.765	0.100	0.086	0.533	0.020	**cqt80**	0.125	0.744	0.125	0.103	0.267	0.030
cqt80	0.108	0.751	0.108	0.079	0.438	0.018	**cqt40**	0.123	0.748	0.123	0.102	0.297	0.028
cqt40	0.108	0.756	0.108	0.077	0.429	0.015	**cens40**	0.093	0.720	0.093	0.086	0.194	0.033
cqt20	0.105	0.752	0.105	0.072	0.429	0.008	**cens20**	0.100	0.708	0.100	0.083	0.324	0.030
cens80	0.088	0.764	0.088	0.065	0.423	0.028	**cqt20**	0.095	0.762	0.095	0.076	0.333	0.025
cens20	0.080	0.707	0.080	0.060	0.556	0.013	**cens80**	0.088	0.695	0.088	0.063	0.192	0.048
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
cens80	**0.045**	0.507	**0.045**	**0.020**	0.000	0.000	**mel40**	**0.113**	0.712	**0.113**	**0.078**	0.333	0.013
mel80	0.023	**0.535**	0.023	0.014	0.033	0.005	**mel20**	0.108	**0.713**	0.108	0.074	**0.545**	0.015
mel60	0.020	0.533	0.020	0.013	0.019	0.003	**mel60**	0.103	0.690	0.103	0.064	0.450	**0.023**
mel40	0.018	0.508	0.018	0.008	0.026	0.003	**mel80**	0.095	0.671	0.095	0.063	0.308	0.020
mfcc40	0.025	0.517	0.025	0.007	0.036	0.013	**cens80**	0.043	0.583	0.043	0.018	0.000	0.000
cens40	0.013	0.510	0.013	0.006	0.000	0.000	**cens40**	0.033	0.533	0.033	0.016	0.000	0.000
mfcc80	0.028	0.530	0.028	0.005	**0.046**	**0.023**	**mfcc80**	0.023	0.524	0.023	0.012	0.000	0.000
cens20	0.025	0.514	0.025	0.005	0.000	0.000	**stft80**	0.033	0.545	0.033	0.011	0.000	0.000
mfcc60	0.020	0.507	0.020	0.005	0.026	0.013	**mfcc40**	0.023	0.519	0.023	0.009	0.000	0.000
cqt80	0.025	0.499	0.025	0.005	0.000	0.000	**mfcc60**	0.023	0.515	0.023	0.008	0.000	0.000
mel20	0.013	0.503	0.013	0.004	0.000	0.000	**cqt80**	0.033	0.518	0.033	0.003	0.000	0.000
cqt40	0.025	0.491	0.025	0.003	0.000	0.000	**cens20**	0.023	0.505	0.023	0.002	0.000	0.000
stft40	0.020	0.503	0.020	0.002	0.000	0.000	**cqt40**	0.020	0.509	0.020	0.001	0.000	0.000
stft80	0.025	0.515	0.025	0.002	0.000	0.000	**stft60**	0.020	0.541	0.020	0.001	0.000	0.000
cqt20	0.023	0.498	0.023	0.002	0.000	0.000	**stft40**	0.018	0.529	0.018	0.001	0.000	0.000
stft60	0.020	0.505	0.020	0.001	0.000	0.000	**stft20**	0.020	0.524	0.020	0.001	0.000	0.000
stft20	0.015	0.503	0.015	0.001	0.000	0.000	**cqt20**	0.020	0.513	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.13: Results of the model with Adamax optimizer for different features - ESC-50.

Model 3: (optimizer: Adamax)						
Features	acc	AUC	micro f1score	macro f1score	precision	recall
mfcc60	0.318	**0.871**	0.318	**0.312**	0.465	0.133
mfcc80	**0.323**	0.869	**0.323**	0.304	0.477	**0.158**
mfcc40	0.290	0.866	0.290	0.275	**0.568**	0.125
mel20	0.233	0.858	0.233	0.209	0.525	0.053
mel80	0.185	0.826	0.185	0.172	0.488	0.053
mel40	0.190	0.834	0.190	0.170	0.462	0.045
mel60	0.175	0.832	0.175	0.160	0.450	0.045
stft20	0.158	0.810	0.158	0.138	0.560	0.035
stft60	0.165	0.792	0.165	0.135	0.367	0.028
stft40	0.150	0.802	0.150	0.124	0.414	0.030
stft80	0.148	0.793	0.148	0.119	0.289	0.028
cqt40	0.135	0.758	0.135	0.110	0.345	0.025
cens40	0.105	0.745	0.105	0.095	0.387	0.030
cqt80	0.115	0.757	0.115	0.094	0.281	0.023
cens20	0.108	0.725	0.108	0.092	0.412	0.018
cqt20	0.118	0.773	0.118	0.091	0.550	0.028
cens80	0.088	0.728	0.088	0.069	0.242	0.040

acc - accuracy; AUC - area under the receiver operating characteristic curve.
All metrics range from $[0, 1]$ (the higher, the better).

B.3.2 Extra Layer

TABLE B.14: Results of the 4 models with extra layer for different features - ESC-10.

Model 7: (optimizer: Adam)							**Model 8: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc40	**0.600**	0.870	**0.600**	**0.587**	0.667	0.325	**mel20**	**0.463**	**0.880**	**0.463**	**0.432**	**0.750**	0.075
mel40	0.550	0.865	0.550	0.560	0.604	0.400	**mel40**	0.413	0.865	0.413	0.381	0.667	0.100
mfcc80	0.550	0.876	0.550	0.551	0.641	0.313	**mel80**	0.400	0.853	0.400	0.355	0.714	0.125
mfcc60	0.575	**0.890**	0.575	0.550	**0.700**	0.350	**mel60**	0.375	0.842	0.375	0.334	0.615	0.100
mel20	0.563	**0.890**	0.563	0.550	0.589	**0.413**	**mfcc40**	0.388	0.829	0.388	0.310	0.400	0.125
stft80	0.513	0.884	0.513	0.499	0.640	0.200	**stft80**	0.325	0.773	0.325	0.273	0.000	0.000
stft60	0.500	0.879	0.500	0.491	0.542	0.163	**mfcc60**	0.313	0.818	0.313	0.270	0.400	**0.200**
mel60	0.488	0.853	0.488	0.485	0.510	0.325	**mfcc80**	0.238	0.806	0.238	0.167	0.308	0.050
mel80	0.488	0.889	0.488	0.477	0.612	0.375	**cens80**	0.213	0.651	0.213	0.138	0.000	0.000
stft40	0.488	0.870	0.488	0.469	0.655	0.238	**stft60**	0.200	0.735	0.200	0.138	0.000	0.000
stft20	0.463	0.881	0.463	0.453	0.625	0.188	**stft20**	0.238	0.737	0.238	0.137	0.000	0.000
cens20	0.313	0.686	0.313	0.303	0.317	0.163	**cens40**	0.188	0.682	0.188	0.128	0.000	0.000
cqt20	0.313	0.820	0.313	0.287	0.323	0.125	**cqt40**	0.188	0.634	0.188	0.105	0.000	0.000
cqt40	0.300	0.794	0.300	0.270	0.440	0.138	**cqt80**	0.150	0.665	0.150	0.082	0.000	0.000
cqt80	0.275	0.782	0.275	0.239	0.355	0.138	**cqt20**	0.125	0.638	0.125	0.082	0.000	0.000
cens80	0.288	0.746	0.288	0.224	0.326	0.175	**stft40**	0.163	0.742	0.163	0.072	0.000	0.000
cens40	0.213	0.774	0.213	0.206	0.229	0.200	**cens20**	0.088	0.542	0.088	0.030	0.000	0.000
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft40	**0.488**	0.871	**0.488**	**0.467**	0.667	0.025	**mfcc80**	**0.575**	0.880	**0.575**	**0.565**	0.620	**0.388**
stft80	0.463	0.864	0.463	0.437	0.500	0.025	**mel40**	0.538	0.873	0.538	0.530	0.547	0.363
mel40	0.425	0.856	0.425	0.404	0.727	0.100	**mfcc40**	0.538	0.872	0.538	0.529	0.667	0.275
mel60	0.388	0.831	0.388	0.371	0.750	0.113	**mel20**	0.525	0.883	0.525	0.524	0.564	**0.388**
stft60	0.400	0.847	0.400	0.368	0.500	0.013	**stft80**	0.550	0.887	0.550	0.516	0.645	0.250
mel20	0.413	**0.875**	0.413	0.363	**0.857**	0.075	**mfcc60**	0.525	0.881	0.525	0.495	0.622	0.288
mel80	0.338	0.823	0.338	0.303	0.545	0.075	**stft40**	0.513	**0.890**	0.513	0.485	**0.710**	0.275
cens40	0.338	0.837	0.338	0.298	0.355	0.138	**mel60**	0.463	0.860	0.463	0.461	0.490	0.300
mfcc60	0.300	0.793	0.300	0.294	0.000	0.000	**mel80**	0.475	0.882	0.475	0.458	0.620	**0.388**
cens80	0.338	0.805	0.338	0.290	0.382	**0.163**	**stft60**	0.475	0.873	0.475	0.449	0.563	0.225
cqt80	0.325	0.754	0.325	0.289	0.000	0.000	**stft20**	0.413	0.858	0.413	0.394	0.500	0.250
stft20	0.350	0.848	0.350	0.288	0.000	0.000	**cens40**	0.325	0.781	0.325	0.317	0.324	0.275
cqt40	0.338	0.763	0.338	0.287	0.500	0.038	**cqt40**	0.338	0.769	0.338	0.300	0.289	0.138
mfcc40	0.325	0.810	0.325	0.270	0.000	0.000	**cqt20**	0.313	0.803	0.313	0.293	0.364	0.150
cens20	0.313	0.687	0.313	0.269	0.667	0.050	**cens80**	0.325	0.750	0.325	0.268	0.391	0.225
cqt20	0.263	0.758	0.263	0.216	0.000	0.000	**cqt80**	0.250	0.757	0.250	0.204	0.241	0.088
mfcc80	0.250	0.774	0.250	0.212	0.000	0.000	**cens20**	0.200	0.653	0.200	0.192	0.226	0.150

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.15: Results of the 4 models with extra layer for different features - ESC-50.

Model 7: (optimizer: Adam)							**Model 8: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.300**	0.856	**0.300**	**0.281**	0.505	**0.130**	**mfcc80**	0.133	0.775	0.133	**0.114**	0.667	0.010
mfcc60	0.288	0.849	0.288	0.272	0.474	0.113	**mel20**	**0.140**	**0.783**	**0.140**	0.112	0.389	0.018
mel20	0.270	**0.865**	0.270	0.261	0.533	0.100	**mfcc40**	0.135	0.782	0.135	0.108	0.500	0.008
mfcc40	0.288	0.853	0.288	0.260	0.549	0.125	**mel40**	0.133	0.778	0.133	0.099	**0.722**	**0.033**
mel40	0.238	0.857	0.238	0.213	**0.574**	0.088	**mfcc60**	0.120	0.780	0.120	0.095	0.500	0.008
mel60	0.220	0.850	0.220	0.199	0.538	0.070	**mel80**	0.113	0.775	0.113	0.086	0.412	0.018
mel80	0.213	0.833	0.213	0.182	0.397	0.058	**mel60**	0.115	0.775	0.115	0.070	0.550	0.028
stft40	0.145	0.780	0.145	0.117	0.406	0.033	**cens20**	0.053	0.518	0.053	0.028	0.000	0.000
stft20	0.138	0.801	0.138	0.116	0.382	0.033	**cens40**	0.055	0.588	0.055	0.020	0.000	0.000
stft80	0.143	0.773	0.143	0.109	0.433	0.033	**cqt80**	0.040	0.627	0.040	0.016	0.000	0.000
stft60	0.148	0.780	0.148	0.109	0.368	0.035	**cqt40**	0.038	0.625	0.038	0.012	0.000	0.000
cqt20	0.115	0.768	0.115	0.087	0.375	0.023	**cens80**	0.045	0.604	0.045	0.011	0.000	0.000
cens20	0.103	0.709	0.103	0.084	0.306	0.028	**stft40**	0.040	0.667	0.040	0.010	0.000	0.000
cens40	0.088	0.721	0.088	0.082	0.132	0.025	**stft80**	0.045	0.672	0.045	0.008	0.000	0.000
cqt40	0.103	0.751	0.103	0.080	0.296	0.020	**stft20**	0.045	0.651	0.045	0.007	0.000	0.000
cqt80	0.090	0.737	0.090	0.075	0.267	0.020	**stft60**	0.048	0.667	0.048	0.007	0.000	0.000
cens80	0.090	0.684	0.090	0.073	0.161	0.045	**cqt20**	0.025	0.595	0.025	0.005	0.000	0.000
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft40	**0.158**	0.798	**0.158**	**0.128**	0.571	0.020	**mfcc80**	**0.283**	0.851	**0.283**	**0.271**	0.509	**0.138**
stft60	0.153	0.790	0.153	0.117	**0.750**	0.023	**mfcc40**	0.280	0.860	0.280	0.256	0.486	0.090
stft80	0.143	0.794	0.143	0.115	0.571	0.020	**mel20**	0.258	**0.867**	0.258	0.237	**0.559**	0.095
mfcc80	0.135	**0.824**	0.135	0.112	0.333	0.003	**mel40**	0.223	0.858	0.223	0.205	0.500	0.085
stft20	0.143	0.799	0.143	0.103	0.538	0.018	**mfcc60**	0.235	0.853	0.235	0.204	0.394	0.070
cens40	0.120	0.786	0.120	0.096	0.474	0.023	**mel80**	0.203	0.840	0.203	0.184	0.456	0.078
mfcc60	0.128	0.797	0.128	0.089	0.400	0.005	**mel60**	0.203	0.843	0.203	0.180	0.520	0.065
cens80	0.113	0.759	0.113	0.086	0.448	**0.033**	**stft20**	0.163	0.794	0.163	0.140	0.436	0.043
mel20	0.115	0.785	0.115	0.084	0.438	0.018	**stft40**	0.163	0.791	0.163	0.138	0.366	0.038
cqt80	0.105	0.769	0.105	0.078	0.333	0.010	**stft60**	0.160	0.781	0.160	0.133	0.405	0.038
cqt20	0.110	0.767	0.110	0.074	0.400	0.005	**stft80**	0.165	0.786	0.165	0.126	0.293	0.030
mfcc40	0.113	0.762	0.113	0.072	0.400	0.005	**cqt80**	0.135	0.746	0.135	0.108	0.310	0.023
mel60	0.080	0.704	0.080	0.069	0.250	0.005	**cqt40**	0.133	0.754	0.133	0.106	0.306	0.028
cqt40	0.105	0.761	0.105	0.069	0.444	0.020	**cens20**	0.120	0.701	0.120	0.098	0.385	0.038
mel80	0.080	0.682	0.080	0.067	0.636	0.018	**cens40**	0.098	0.728	0.098	0.088	0.195	0.040
mel40	0.085	0.734	0.085	0.062	0.700	0.018	**cqt20**	0.110	0.773	0.110	0.081	0.360	0.023
cens20	0.088	0.700	0.088	0.061	0.545	0.015	**cens80**	0.098	0.690	0.098	0.079	0.182	0.055

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.3.3 Dropout Rate of 0.2

TABLE B.16: Results of the 4 best models for single features and dropout of 0.2 - ESC-10.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.650**	0.878	**0.650**	**0.647**	**0.676**	**0.625**	**mel40**	**0.525**	0.881	**0.525**	**0.517**	0.600	0.225
mfcc40	0.625	**0.893**	0.625	0.618	0.644	0.588	**mfcc80**	0.513	**0.891**	0.513	0.497	**0.647**	0.413
mfcc80	0.600	0.873	0.600	0.586	0.613	0.575	**mel80**	0.500	0.860	0.500	0.482	0.600	0.263
mel80	0.525	0.862	0.525	0.533	0.547	0.513	**mfcc60**	0.500	0.888	0.500	0.476	0.583	**0.438**
mel60	0.538	0.851	0.538	0.531	0.557	0.488	**mel60**	0.488	0.865	0.488	0.469	0.583	0.263
mel40	0.538	0.861	0.538	0.530	0.557	0.488	**mel20**	0.450	0.872	0.450	0.435	0.364	0.100
mel20	0.538	0.858	0.538	0.526	0.571	0.500	**mfcc40**	0.425	0.850	0.425	0.390	0.481	0.313
stft80	0.525	0.877	0.525	0.515	0.649	0.300	**stft40**	0.250	0.745	0.250	0.181	0.000	0.000
stft60	0.513	0.865	0.513	0.498	0.524	0.275	**cens80**	0.263	0.721	0.263	0.170	0.000	0.000
stft40	0.475	0.874	0.475	0.471	0.605	0.325	**stft20**	0.238	0.739	0.238	0.143	0.000	0.000
stft20	0.400	0.855	0.400	0.399	0.450	0.225	**cens40**	0.213	0.710	0.213	0.142	0.000	0.000
cqt20	0.375	0.780	0.375	0.371	0.395	0.213	**stft60**	0.213	0.731	0.213	0.141	0.000	0.000
cqt40	0.375	0.781	0.375	0.364	0.393	0.300	**stft80**	0.200	0.746	0.200	0.137	0.000	0.000
cens20	0.300	0.693	0.300	0.288	0.353	0.225	**cqt40**	0.188	0.648	0.188	0.137	0.000	0.000
cens80	0.325	0.763	0.325	0.281	0.370	0.213	**cqt80**	0.175	0.673	0.175	0.111	0.000	0.000
cqt80	0.300	0.754	0.300	0.254	0.305	0.225	**cens20**	0.125	0.589	0.125	0.066	0.000	0.000
cens40	0.225	0.747	0.225	0.219	0.221	0.188	**cqt20**	0.138	0.632	0.138	0.066	0.000	0.000
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc40	**0.588**	0.909	**0.588**	**0.575**	0.659	0.363	**mfcc60**	**0.625**	0.874	**0.625**	**0.621**	**0.657**	**0.575**
mfcc60	0.575	**0.910**	0.575	0.570	0.640	**0.400**	**mfcc40**	0.600	**0.894**	0.600	0.603	0.608	0.563
mfcc80	0.550	0.870	0.550	0.535	0.595	0.313	**mfcc80**	0.613	0.878	0.613	0.594	0.639	**0.575**
mel40	0.525	0.886	0.525	0.533	0.617	0.363	**mel80**	0.550	0.873	0.550	0.552	0.574	0.488
mel20	0.513	0.865	0.513	0.515	0.526	0.250	**mel40**	0.550	0.878	0.550	0.551	0.581	0.538
stft80	0.538	0.884	0.538	0.511	0.739	0.213	**mel60**	0.550	0.851	0.550	0.541	0.600	0.525
mel60	0.500	0.870	0.500	0.491	0.614	0.338	**mel20**	0.525	0.846	0.525	0.522	0.562	0.513
stft40	0.500	0.886	0.500	0.488	**0.833**	0.125	**stft40**	0.463	0.879	0.463	0.463	0.565	0.325
mel80	0.475	0.844	0.475	0.462	0.605	0.288	**stft80**	0.475	0.859	0.475	0.455	0.587	0.338
stft60	0.450	0.885	0.450	0.423	0.692	0.225	**stft60**	0.463	0.857	0.463	0.430	0.585	0.300
stft20	0.375	0.866	0.375	0.349	0.583	0.088	**cqt20**	0.400	0.793	0.400	0.391	0.326	0.188
cqt80	0.313	0.762	0.313	0.302	0.313	0.063	**cqt40**	0.363	0.774	0.363	0.339	0.375	0.225
cqt20	0.313	0.773	0.313	0.289	0.400	0.050	**stft20**	0.350	0.851	0.350	0.325	0.436	0.213
cens80	0.325	0.805	0.325	0.283	0.395	0.188	**cens80**	0.363	0.757	0.363	0.306	0.435	0.250
cens20	0.288	0.688	0.288	0.270	0.429	0.075	**cens40**	0.250	0.728	0.250	0.246	0.235	0.200
cqt40	0.275	0.756	0.275	0.269	0.286	0.050	**cens20**	0.263	0.679	0.263	0.246	0.281	0.200
cens40	0.275	0.821	0.275	0.262	0.333	0.175	**cqt80**	0.263	0.737	0.263	0.221	0.293	0.213

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.17: Results of the 4 best models for single features and dropout of 0.2 - ESC-50.

Model 11: (optimizer: Adam)							**Model 12: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc60	**0.353**	0.811	**0.353**	**0.344**	**0.419**	**0.303**	**mfcc80**	**0.275**	**0.860**	**0.275**	**0.258**	**0.505**	**0.125**
mfcc40	0.335	0.806	0.335	0.328	0.394	0.283	**mfcc60**	0.255	0.859	0.255	0.236	0.448	0.108
mfcc80	0.303	0.790	0.303	0.293	0.359	0.253	**mfcc40**	0.245	0.856	0.245	0.219	0.495	0.113
mel20	0.255	**0.824**	0.255	0.237	0.379	0.138	**mel20**	0.178	0.808	0.178	0.163	0.400	0.040
mel40	0.240	0.817	0.240	0.229	0.406	0.178	**mel40**	0.173	0.799	0.173	0.159	0.387	0.060
mel60	0.238	0.799	0.238	0.225	0.328	0.148	**mel60**	0.173	0.785	0.173	0.157	0.364	0.060
mel80	0.235	0.800	0.235	0.224	0.330	0.150	**mel80**	0.170	0.782	0.170	0.151	0.338	0.058
stft40	0.145	0.748	0.145	0.133	0.288	0.053	**cens80**	0.080	0.603	0.080	0.030	0.000	0.000
stft60	0.148	0.740	0.148	0.127	0.271	0.058	**cqt40**	0.053	0.653	0.053	0.025	0.000	0.000
stft80	0.140	0.730	0.140	0.121	0.315	0.058	**cqt80**	0.048	0.679	0.048	0.024	0.000	0.000
cqt80	0.118	0.703	0.118	0.110	0.163	0.043	**stft80**	0.053	0.704	0.053	0.023	0.000	0.000
stft20	0.123	0.750	0.123	0.110	0.339	0.050	**cens20**	0.038	0.536	0.038	0.023	0.000	0.000
cqt20	0.105	0.731	0.105	0.093	0.209	0.035	**stft60**	0.060	0.702	0.060	0.021	0.000	0.000
cqt40	0.108	0.720	0.108	0.093	0.222	0.050	**stft40**	0.053	0.696	0.053	0.018	0.000	0.000
cens40	0.090	0.684	0.090	0.081	0.160	0.065	**cqt20**	0.035	0.606	0.035	0.012	0.000	0.000
cens20	0.088	0.678	0.088	0.080	0.159	0.043	**stft20**	0.055	0.665	0.055	0.011	0.000	0.000
cens80	0.088	0.658	0.088	0.067	0.134	0.055	**cens40**	0.050	0.627	0.050	0.010	0.000	0.000
Model 13: (optimizer: Adamax)							**Model 14: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.328**	**0.874**	**0.328**	**0.314**	0.521	**0.190**	**mfcc80**	**0.350**	0.802	**0.350**	**0.330**	0.413	**0.315**
mfcc60	0.295	0.845	0.295	0.283	0.484	0.153	**mfcc60**	0.335	0.788	0.335	0.316	0.401	0.288
mfcc40	0.283	0.854	0.283	0.263	0.479	0.145	**mfcc40**	0.305	0.814	0.305	0.295	0.366	0.253
mel20	0.198	0.844	0.198	0.175	0.455	0.050	**mel40**	0.270	**0.825**	0.270	0.259	**0.424**	0.160
mel40	0.185	0.821	0.185	0.169	0.389	0.053	**mel20**	0.250	0.823	0.250	0.234	0.369	0.138
mel80	0.180	0.799	0.180	0.162	0.353	0.060	**mel60**	0.235	0.796	0.235	0.225	0.364	0.148
stft20	0.160	0.796	0.160	0.139	**0.588**	0.025	**mel80**	0.230	0.794	0.230	0.216	0.343	0.148
mel60	0.158	0.799	0.158	0.138	0.368	0.053	**stft60**	0.148	0.732	0.148	0.130	0.205	0.043
stft40	0.163	0.793	0.163	0.129	0.476	0.025	**stft40**	0.148	0.740	0.148	0.128	0.253	0.050
stft60	0.145	0.790	0.145	0.122	0.440	0.028	**stft20**	0.133	0.765	0.133	0.120	0.233	0.043
stft80	0.145	0.791	0.145	0.111	0.333	0.025	**stft80**	0.118	0.727	0.118	0.104	0.200	0.040
cqt80	0.125	0.742	0.125	0.100	0.350	0.018	**cens40**	0.103	0.677	0.103	0.093	0.142	0.060
cqt40	0.115	0.756	0.115	0.087	0.444	0.020	**cqt80**	0.110	0.705	0.110	0.092	0.226	0.060
cens20	0.090	0.708	0.090	0.078	0.545	0.015	**cqt40**	0.100	0.708	0.100	0.091	0.193	0.043
cqt20	0.105	0.756	0.105	0.078	0.400	0.015	**cqt20**	0.093	0.727	0.093	0.088	0.215	0.035
cens40	0.073	0.752	0.073	0.063	0.476	0.025	**cens20**	0.090	0.668	0.090	0.078	0.162	0.045
cens80	0.075	0.741	0.075	0.058	0.235	0.030	**cens80**	0.075	0.664	0.075	0.052	0.112	0.048

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.3.4 Dropout Rate of 0.6

TABLE B.18: Results of the 4 best models for single features and dropout of 0.6 - ESC-10.

Model 15: (optimizer: Adam)							**Model 16: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel40	**0.463**	**0.888**	**0.463**	**0.460**	**0.818**	0.113	**mel40**	**0.375**	0.831	**0.375**	**0.338**	**0.857**	**0.075**
mel80	0.438	0.827	0.438	0.420	0.643	0.113	**mel20**	0.350	**0.867**	0.350	0.291	0.714	0.063
stft60	0.450	0.858	0.450	0.405	0.667	0.100	**mel80**	0.325	0.797	0.325	0.282	0.500	0.063
stft40	0.425	0.870	0.425	0.384	0.583	0.088	**mel60**	0.300	0.820	0.300	0.270	0.444	0.050
mel60	0.400	0.846	0.400	0.368	0.667	0.075	**cens40**	0.238	0.669	0.238	0.186	0.000	0.000
mel20	0.388	0.869	0.388	0.350	0.429	0.113	**mfcc60**	0.263	0.763	0.263	0.181	0.200	0.013
stft20	0.375	0.849	0.375	0.330	0.625	0.063	**stft80**	0.250	0.767	0.250	0.157	0.000	0.000
mfcc80	0.350	0.749	0.350	0.300	0.385	0.063	**mfcc80**	0.225	0.756	0.225	0.155	0.250	0.013
cqt80	0.350	0.751	0.350	0.299	0.000	0.000	**cens80**	0.238	0.647	0.238	0.155	0.000	0.000
cens20	0.313	0.698	0.313	0.290	0.304	0.088	**stft60**	0.250	0.744	0.250	0.153	0.000	0.000
stft80	0.313	0.834	0.313	0.288	0.500	0.013	**cqt40**	0.213	0.679	0.213	0.131	0.000	0.000
cens40	0.288	0.815	0.288	0.264	0.294	**0.188**	**stft20**	0.200	0.745	0.200	0.102	0.000	0.000
cens80	0.313	0.782	0.313	0.260	0.357	**0.188**	**cqt20**	0.188	0.679	0.188	0.099	0.000	0.000
cqt20	0.288	0.797	0.288	0.247	0.364	0.050	**stft40**	0.213	0.761	0.213	0.094	0.000	0.000
cqt40	0.263	0.782	0.263	0.209	0.111	0.013	**cqt80**	0.163	0.699	0.163	0.084	0.000	0.000
mfcc40	0.188	0.715	0.188	0.164	0.500	0.013	**mfcc40**	0.163	0.749	0.163	0.084	0.250	0.013
mfcc60	0.238	0.707	0.238	0.162	0.200	0.013	**cens20**	0.138	0.563	0.138	0.080	0.000	0.000
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel20	**0.425**	0.852	**0.425**	**0.408**	0.500	0.025	**stft40**	**0.500**	0.872	**0.500**	**0.476**	**0.692**	0.113
mel40	0.413	**0.873**	0.413	0.393	**1.000**	**0.125**	**stft80**	0.488	0.843	0.488	0.451	0.600	0.075
mel80	0.375	0.808	0.375	0.357	0.643	0.113	**mel80**	0.413	0.858	0.413	0.415	0.571	0.100
mel60	0.338	0.813	0.338	0.334	0.615	0.100	**mel20**	0.438	**0.878**	0.438	0.407	0.450	0.113
cens80	0.325	0.829	0.325	0.271	0.500	0.075	**stft60**	0.425	0.873	0.425	0.402	0.611	0.138
cens40	0.325	0.845	0.325	0.250	0.833	0.063	**mel40**	0.425	0.858	0.425	0.401	0.571	0.100
mfcc60	0.225	0.684	0.225	0.227	0.000	0.000	**mel60**	0.400	0.840	0.400	0.380	0.500	0.075
cqt80	0.250	0.736	0.250	0.208	0.000	0.000	**stft20**	0.363	0.853	0.363	0.334	0.563	0.113
mfcc80	0.225	0.673	0.225	0.207	0.000	0.000	**cens20**	0.338	0.710	0.338	0.307	0.353	0.075
stft20	0.275	0.793	0.275	0.193	0.000	0.000	**cqt40**	0.325	0.785	0.325	0.306	0.500	0.050
stft60	0.250	0.803	0.250	0.184	0.000	0.000	**cqt80**	0.325	0.765	0.325	0.277	0.600	0.038
stft80	0.300	0.804	0.300	0.180	0.000	0.000	**mfcc80**	0.275	0.825	0.275	0.276	0.429	0.075
stft40	0.225	0.788	0.225	0.168	0.000	0.000	**cens40**	0.275	0.825	0.275	0.266	0.283	**0.188**
cens20	0.200	0.620	0.200	0.158	**1.000**	0.038	**cqt20**	0.300	0.798	0.300	0.260	0.313	0.063
cqt20	0.200	0.719	0.200	0.143	0.000	0.000	**cens80**	0.300	0.776	0.300	0.253	0.357	**0.188**
cqt40	0.175	0.722	0.175	0.116	0.000	0.000	**mfcc40**	0.213	0.699	0.213	0.203	0.250	0.013
mfcc40	0.100	0.638	0.100	0.092	0.000	0.000	**mfcc60**	0.163	0.753	0.163	0.158	0.143	0.013

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.19: Results of the 4 best models for single features and dropout of 0.6 - ESC-50.

Model 15: (optimizer: Adam)							**Model 16: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mel20	**0.193**	**0.855**	**0.193**	**0.168**	0.467	0.018	**mel40**	0.093	**0.683**	0.093	**0.081**	0.571	0.010
mel40	0.158	0.824	0.158	0.137	0.476	0.025	**mel20**	**0.098**	0.671	**0.098**	0.076	0.750	0.008
mfcc60	0.133	0.783	0.133	0.120	0.556	0.013	**mel60**	0.090	0.677	0.090	0.063	**0.875**	**0.018**
mel60	0.135	0.821	0.135	0.112	**0.647**	0.028	**cens20**	0.050	0.566	0.050	0.035	0.000	0.000
mel80	0.133	0.810	0.133	0.112	0.600	0.015	**mfcc60**	0.050	0.558	0.050	0.030	0.000	0.000
mfcc80	0.123	0.781	0.123	0.101	0.571	0.010	**stft80**	0.078	0.668	0.078	0.029	0.000	0.000
stft20	0.133	0.805	0.133	0.099	0.579	0.028	**stft60**	0.068	0.660	0.068	0.027	0.000	0.000
stft60	0.138	0.786	0.138	0.097	0.500	0.030	**cens40**	0.060	0.608	0.060	0.026	0.000	0.000
stft40	0.133	0.793	0.133	0.095	0.450	0.023	**mfcc80**	0.043	0.599	0.043	0.021	0.000	0.000
cens20	0.115	0.719	0.115	0.088	0.375	0.015	**cens80**	0.060	0.632	0.060	0.020	0.000	0.000
stft80	0.123	0.771	0.123	0.085	0.500	0.023	**mfcc40**	0.043	0.619	0.043	0.020	0.000	0.000
cens40	0.100	0.770	0.100	0.084	0.333	0.023	**stft40**	0.038	0.655	0.038	0.017	0.000	0.000
cqt40	0.108	0.763	0.108	0.081	0.364	0.010	**stft20**	0.050	0.637	0.050	0.013	0.000	0.000
cqt20	0.103	0.770	0.103	0.068	0.455	0.013	**cqt20**	0.035	0.570	0.035	0.011	0.000	0.000
mfcc40	0.083	0.720	0.083	0.066	0.600	0.008	**cqt40**	0.030	0.590	0.030	0.011	0.000	0.000
cens80	0.085	0.734	0.085	0.063	0.250	**0.038**	**cqt80**	0.038	0.601	0.038	0.011	0.000	0.000
cqt80	0.085	0.737	0.085	0.056	0.273	0.008	**mel80**	0.018	0.501	0.018	0.004	0.000	0.000
Model 17: (optimizer: Adamax)							**Model 18: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
stft40	0.118	0.777	0.118	**0.085**	**1.000**	0.010	**mel20**	**0.160**	**0.853**	**0.160**	**0.132**	0.615	0.020
stft60	**0.128**	0.777	**0.128**	0.081	**1.000**	0.010	**stft20**	0.153	0.799	0.153	0.130	0.550	0.028
stft80	0.123	0.772	0.123	0.079	**1.000**	0.008	**mel40**	0.153	0.834	0.153	0.126	0.545	0.015
stft20	0.103	0.774	0.103	0.068	**1.000**	0.010	**mfcc80**	0.130	0.764	0.130	0.114	0.500	0.005
cens80	0.098	0.764	0.098	0.066	0.667	**0.020**	**mel80**	0.115	0.796	0.115	0.104	0.500	0.018
mel20	0.065	0.589	0.065	0.061	0.000	0.000	**mel60**	0.120	0.806	0.120	0.096	0.353	0.015
cens40	0.098	**0.780**	0.098	0.059	0.667	0.015	**cens40**	0.118	0.772	0.118	0.096	0.440	0.028
cens20	0.088	0.678	0.088	0.055	0.600	0.008	**stft60**	0.120	0.783	0.120	0.094	0.476	0.025
cqt40	0.083	0.728	0.083	0.043	0.667	0.005	**stft80**	0.128	0.782	0.128	0.093	0.429	0.023
cqt20	0.083	0.736	0.083	0.040	0.000	0.000	**stft40**	0.118	0.791	0.118	0.089	0.500	0.025
cqt80	0.083	0.730	0.083	0.038	**1.000**	0.003	**cqt20**	0.115	0.772	0.115	0.083	0.444	0.010
mel40	0.043	0.585	0.043	0.032	**1.000**	0.003	**mfcc60**	0.100	0.738	0.100	0.081	**0.667**	0.005
mel80	0.038	0.595	0.038	0.027	0.200	0.003	**cens20**	0.108	0.718	0.108	0.081	0.333	0.015
mel60	0.035	0.582	0.035	0.021	0.333	0.003	**cqt80**	0.105	0.755	0.105	0.075	0.364	0.010
mfcc60	0.028	0.513	0.028	0.009	0.000	0.000	**cens80**	0.093	0.733	0.093	0.071	0.315	**0.043**
mfcc80	0.025	0.506	0.025	0.007	0.000	0.000	**mfcc40**	0.090	0.709	0.090	0.065	0.500	0.003
mfcc40	0.020	0.507	0.020	0.005	0.000	0.000	**cqt40**	0.093	0.765	0.093	0.063	0.400	0.020

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.3.5 Dropout Rate of 0 and Adamax

TABLE B.20: Results of Adamax optimizer's models for single features and dropout of 0.

Model 19: (optimizer: Adamax; dr: 0; dataset: ESC-10)							**Model 19: (optimizer: Adamax; dr: 0; dataset: ESC-50)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfcc80	**0.575**	**0.918**	**0.575**	**0.580**	0.592	**0.563**	**mfcc80**	**0.323**	0.771	**0.323**	**0.317**	0.373	**0.295**
mfcc60	0.550	0.909	0.550	0.566	0.609	0.525	**mfcc40**	0.308	0.779	0.308	0.298	0.343	0.260
mfcc40	**0.575**	0.893	**0.575**	0.565	0.584	**0.563**	**mfcc60**	0.293	0.783	0.293	0.283	0.336	0.253
mel80	0.525	0.835	0.525	0.524	0.557	0.488	**mel20**	0.238	**0.827**	0.238	0.224	0.325	0.095
mel20	0.525	0.872	0.525	0.523	0.554	0.450	**mel60**	0.208	0.781	0.208	0.204	0.292	0.113
mel40	0.525	0.877	0.525	0.520	0.543	0.475	**mel40**	0.205	0.810	0.205	0.202	0.301	0.103
stft60	0.525	0.894	0.525	0.498	0.690	0.250	**mel80**	0.205	0.795	0.205	0.201	0.304	0.130
mel60	0.500	0.859	0.500	0.495	0.521	0.463	**stft20**	0.163	0.794	0.163	0.149	0.441	0.038
stft40	0.463	0.874	0.463	0.446	0.591	0.163	**stft40**	0.160	0.779	0.160	0.138	**0.545**	0.045
stft80	0.413	0.874	0.413	0.398	0.656	0.263	**stft80**	0.143	0.784	0.143	0.117	0.333	0.038
stft20	0.375	0.854	0.375	0.362	**0.692**	0.113	**stft60**	0.140	0.778	0.140	0.115	0.326	0.035
cqt40	0.338	0.773	0.338	0.335	0.357	0.125	**cqt80**	0.113	0.737	0.113	0.097	0.275	0.028
cqt80	0.325	0.782	0.325	0.321	0.323	0.125	**cens20**	0.103	0.694	0.103	0.089	0.406	0.033
cens20	0.288	0.688	0.288	0.289	0.333	0.088	**cqt40**	0.113	0.730	0.113	0.086	0.324	0.028
cens80	0.338	0.800	0.338	0.278	0.450	0.225	**cens80**	0.090	0.699	0.090	0.078	0.222	0.045
cqt20	0.275	0.775	0.275	0.255	0.333	0.100	**cens40**	0.073	0.731	0.073	0.068	0.239	0.028
cens40	0.250	0.787	0.250	0.236	0.273	0.188	**cqt20**	0.085	0.742	0.085	0.065	0.333	0.023

dr - dropout rate; acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.4 ESC Datasets - Combination of Features as Input

B.4.1 Baseline Model Architecture

TABLE B.21: Results of the 6 models for different feature combinations - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.713**	0.938	**0.713**	**0.701**	0.740	**0.675**	**mms40**	**0.538**	**0.908**	**0.538**	**0.499**	0.656	**0.263**
mmq	0.688	0.941	0.688	0.673	0.729	0.538	**mmstftq40**	0.475	0.858	0.475	0.433	0.640	0.200
mms40	0.675	**0.942**	0.675	0.663	0.707	0.513	**mmcens40**	0.438	0.864	0.438	0.396	0.586	0.213
mmstftq	0.675	0.938	0.675	0.661	0.758	0.625	**mmstftq**	0.413	0.793	0.413	0.383	0.519	0.175
mmq40	0.675	0.940	0.675	0.660	**0.780**	0.488	**mfccstft**	0.425	0.858	0.425	0.375	0.433	0.163
mmcens	0.663	0.906	0.663	0.648	0.725	0.625	**mms60**	0.400	0.809	0.400	0.350	0.591	0.163
mmstftq40	0.650	0.925	0.650	0.644	0.759	0.513	**mmcens**	0.363	0.761	0.363	0.325	0.563	0.113
mms80	0.663	0.932	0.663	0.641	0.700	0.525	**mmq**	0.363	0.817	0.363	0.314	0.647	0.138
mms60	0.650	0.928	0.650	0.637	0.690	0.500	**mmq40**	0.375	0.843	0.375	0.309	0.440	0.138
mmsqc	0.650	0.928	0.650	0.632	0.763	0.563	**mms80**	0.338	0.816	0.338	0.301	0.588	0.125
mfccstft	0.638	0.897	0.638	0.631	0.695	0.513	**mmsqc**	0.350	0.834	0.350	0.297	0.619	0.163
mmcens40	0.650	0.915	0.650	0.622	0.725	0.463	**mfccstft80**	0.250	0.746	0.250	0.197	0.440	0.138
mfccstft80	0.625	0.887	0.625	0.606	0.677	0.550	**mmsqc80**	0.263	0.773	0.263	0.192	**1.000**	0.075
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmcens	**0.613**	0.894	**0.613**	**0.587**	0.708	0.213	**mmsqc**	**0.738**	0.910	**0.738**	**0.727**	**0.773**	**0.725**
mmsqc80	0.538	0.907	0.538	0.525	0.718	**0.350**	**mmstftq40**	0.725	0.916	0.725	0.719	0.737	0.700
mms60	0.538	**0.912**	0.538	0.521	0.684	0.163	**mmq40**	0.688	0.902	0.688	0.677	0.692	0.675
mmstftq	0.513	0.903	0.513	0.497	0.826	0.238	**mmq**	0.663	0.884	0.663	0.657	0.675	0.650
mms80	0.500	0.893	0.500	0.493	0.667	0.175	**mms80**	0.663	0.894	0.663	0.647	0.688	0.663
mmcens40	0.488	0.868	0.488	0.482	0.824	0.175	**mmsqc80**	0.662	0.928	0.662	0.647	0.704	**0.625**
mmq40	0.488	0.891	0.488	0.463	0.778	0.175	**mmcens**	0.650	**0.938**	0.650	0.647	0.689	0.525
mmq	0.463	0.865	0.463	0.447	0.696	0.200	**mms60**	0.638	0.924	0.638	0.618	0.679	0.475
mms40	0.425	0.835	0.425	0.396	0.714	0.125	**mmstftq**	0.625	0.935	0.625	0.615	0.701	0.588
mfccstft	0.400	0.813	0.400	0.387	0.438	0.088	**mfccstft80**	0.625	0.890	0.625	0.607	0.661	0.463
mmsqc	0.388	0.827	0.388	0.362	**0.909**	0.125	**mms40**	0.625	0.931	0.625	0.605	0.684	0.488
mmstftq40	0.388	0.823	0.388	0.345	0.813	0.163	**mfccstft**	0.613	0.913	0.613	0.582	0.744	0.400
mfccstft80	0.288	0.807	0.288	0.278	0.556	0.125	**mmcens40**	0.600	0.930	0.600	0.577	0.750	0.450
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.213**	0.533	**0.213**	**0.103**	**0.205**	**0.188**	**mmsqc**	**0.563**	**0.908**	**0.563**	**0.554**	**0.750**	0.150
mms60	0.113	0.504	0.113	0.102	0.083	0.063	**mmstftq40**	0.450	0.883	0.450	0.426	0.543	0.238
mfccstft	0.150	0.541	0.150	0.090	0.145	0.113	**mmsqc80**	0.450	0.884	0.450	0.426	0.568	**0.263**
mmsqc	0.150	**0.565**	0.150	0.085	0.154	0.125	**mmstftq**	0.388	0.848	0.388	0.359	0.600	0.150
mmcens40	0.163	0.541	0.163	0.056	0.163	0.163	**mms80**	0.388	0.856	0.388	0.344	0.600	0.188
mms80	0.088	0.493	0.088	0.051	0.052	0.038	**mmq40**	0.350	0.832	0.350	0.343	0.706	0.150
mmsqc80	0.088	0.546	0.088	0.047	0.085	0.075	**mmcens**	0.400	0.836	0.400	0.343	0.588	0.250
mmstftq	0.138	0.516	0.138	0.047	0.118	0.113	**mfccstft**	0.375	0.855	0.375	0.339	0.625	0.125
mmcens	0.100	0.524	0.100	0.045	0.110	0.100	**mmq**	0.350	0.819	0.350	0.337	0.448	0.163
mmq	0.113	0.501	0.113	0.042	0.118	0.113	**mmcens40**	0.375	0.853	0.375	0.299	0.529	0.113
mms40	0.075	0.472	0.075	0.034	0.052	0.038	**mfccstft80**	0.400	0.857	0.400	0.291	0.444	0.100
mfccstft80	0.100	0.518	0.100	0.020	0.107	0.100	**mms40**	0.350	0.852	0.350	0.289	0.500	0.163
mfccmel40	0.100	0.500	0.100	0.018	0.100	0.100	**mms60**	0.338	0.831	0.338	0.271	0.667	0.100
mfccmel80	0.100	0.500	0.100	0.018	0.100	0.100	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mmstftq40	0.063	0.468	0.063	0.015	0.066	0.063	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.22: Results of the 6 models for different feature combinations - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq	**0.393**	0.873	**0.393**	**0.379**	0.575	**0.250**	**mmq**	**0.178**	0.789	**0.178**	**0.153**	0.591	0.033
mmstftq	0.383	0.875	0.383	0.361	0.584	0.243	**mms60**	0.173	0.770	0.173	0.151	0.529	0.023
mmsqc80	0.365	0.869	0.365	0.352	0.511	0.243	**mms40**	0.155	**0.808**	0.155	0.149	0.667	0.030
mmcens40	0.365	**0.895**	0.365	0.345	**0.635**	0.200	**mmcens40**	0.175	0.795	0.175	0.145	0.722	0.033
mms40	0.358	0.879	0.358	0.343	0.550	0.193	**mmcens**	0.158	0.776	0.158	0.140	0.650	0.033
mms60	0.353	0.878	0.353	0.342	0.574	0.203	**mfccstft80**	0.145	0.783	0.145	0.129	0.615	0.020
mmsqc	0.348	0.881	0.348	0.334	0.617	0.238	**mmq40**	0.135	0.784	0.135	0.123	0.714	0.025
mmq40	0.340	0.886	0.340	0.331	0.542	0.195	**mmsqc80**	0.148	0.773	0.148	0.122	0.500	0.023
mms80	0.343	0.875	0.343	0.331	0.557	0.208	**mmstftq40**	0.135	0.728	0.135	0.114	**0.750**	**0.038**
mmstftq40	0.348	0.868	0.348	0.327	0.527	0.218	**mfccstft**	0.133	0.768	0.133	0.113	0.600	0.023
mfccstft	0.328	0.876	0.328	0.313	0.500	0.175	**mmstftq**	0.138	0.765	0.138	0.107	0.500	0.020
mmcens	0.315	0.869	0.315	0.293	0.554	0.180	**mms80**	0.128	0.783	0.128	0.103	0.455	0.025
mfccstft80	0.310	0.862	0.310	0.287	0.452	0.190	**mmsqc**	0.115	0.790	0.115	0.100	0.471	0.020
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.230**	**0.844**	**0.230**	**0.194**	0.565	**0.033**	**mmstftq**	**0.375**	0.878	**0.375**	**0.354**	0.571	0.223
mmq	0.200	0.827	0.200	0.168	**0.647**	0.028	**mmstftq40**	0.360	0.882	0.360	0.346	0.497	0.180
mmstftq	0.183	0.787	0.183	0.168	0.462	0.015	**mmsqc**	0.355	**0.884**	0.355	0.345	0.524	0.215
mmsqc	0.180	0.836	0.180	0.156	0.571	0.020	**mms80**	0.345	0.872	0.345	0.339	**0.605**	0.188
mfccstft80	0.163	0.821	0.163	0.155	0.440	0.028	**mms60**	0.355	0.874	0.355	0.337	0.536	0.203
mmstftq40	0.190	0.840	0.190	0.155	0.429	0.015	**mmq40**	0.343	0.881	0.343	0.333	0.549	0.195
mfccstft	0.170	0.792	0.170	0.148	0.333	0.010	**mmq**	0.343	0.866	0.343	0.327	0.566	0.203
mms40	0.173	0.809	0.173	0.145	0.500	0.015	**mms40**	0.343	0.877	0.343	0.327	0.569	0.195
mms80	0.138	0.777	0.138	0.125	0.364	0.010	**mmsqc80**	0.348	0.871	0.348	0.321	0.508	**0.233**
mms60	0.128	0.751	0.128	0.114	0.533	0.020	**mfccstft80**	0.335	0.858	0.335	0.319	0.516	0.205
mmcens	0.118	0.798	0.118	0.100	0.455	0.013	**mmcens**	0.335	0.869	0.335	0.316	0.503	0.185
mmcens40	0.110	0.736	0.110	0.095	0.636	0.018	**mmcens40**	0.318	0.876	0.318	0.309	0.527	0.170
mmq40	0.088	0.748	0.088	0.077	0.308	0.010	**mfccstft**	0.313	0.868	0.313	0.299	0.485	0.165
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000
Model 5: (optimizer: Adadelta)							**Model 6: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq40	**0.035**	0.503	**0.035**	**0.011**	**0.051**	**0.033**	**mms60**	0.055	0.589	0.055	**0.057**	0.286	0.005
mmstftq	0.023	**0.523**	0.023	0.009	0.032	0.015	**mmq**	0.060	0.614	0.060	0.054	0.429	0.008
mmsqc	0.030	0.518	0.030	0.009	0.030	0.020	**mms80**	0.060	0.609	0.060	0.050	0.250	0.005
mmq	0.015	0.506	0.015	0.009	0.028	0.013	**mfccstft**	0.048	**0.620**	0.048	0.043	0.000	0.000
mmsqc80	0.013	0.503	0.013	0.008	0.014	0.008	**mmcens**	0.055	0.572	0.055	0.042	0.167	0.003
mmcens40	0.020	0.510	0.020	0.007	0.015	0.010	**mmq40**	0.045	0.535	0.045	0.037	**0.571**	0.010
mfccstft80	0.028	0.496	0.028	0.007	0.032	0.020	**mfccstft80**	**0.063**	0.552	**0.063**	0.036	0.000	0.000
mms40	0.023	0.503	0.023	0.006	0.032	0.020	**mmsqc80**	0.028	0.588	0.028	0.032	0.125	0.003
mmq40	0.028	0.485	0.028	0.006	0.021	0.013	**mmstftq**	0.048	0.579	0.048	0.031	0.500	**0.013**
mfccstft	0.013	0.502	0.013	0.006	0.005	0.003	**mmstftq40**	0.030	0.551	0.030	0.029	0.143	0.003
mms60	0.015	0.494	0.015	0.004	0.021	0.015	**mmsqc**	0.048	0.590	0.048	0.028	0.000	0.000
mms80	0.018	0.474	0.018	0.003	0.023	0.013	**mms40**	0.045	0.565	0.045	0.027	0.000	0.000
mmcens	0.005	0.495	0.005	0.002	0.014	0.005	**mmcens40**	0.035	0.528	0.035	0.018	0.000	0.000
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.23: Results for model 3 fully trained with ESC-50 dataset.

Model 3: (optimizer: Adamax; dataset: ESC-50)						
Features	acc	AUC	micro f1score	macro f1score	precision	recall
mmsqc80	**0.333**	0.881	**0.333**	**0.322**	0.609	0.098
mmq	0.310	0.872	0.310	0.289	0.614	0.088
mfccstft80	0.283	0.874	0.283	0.267	0.656	**0.100**
mmsqc	0.265	**0.888**	0.265	0.254	0.704	0.095
mmstftq40	0.258	0.871	0.258	0.248	0.705	0.078
mfccstft	0.268	0.868	0.268	0.240	0.620	0.078
mmq40	0.253	0.872	0.253	0.234	0.571	0.060
mmcens40	0.238	0.874	0.238	0.226	**0.745**	0.088
mmstftq	0.233	0.844	0.233	0.210	0.659	0.068
mms80	0.240	0.844	0.240	0.204	0.618	0.053
mms40	0.210	0.842	0.210	0.188	0.640	0.040
mmcens	0.210	0.835	0.210	0.179	0.719	0.058
mms60	0.200	0.833	0.200	0.174	0.520	0.033
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve.
All metrics range from $[0, 1]$ (the higher, the better).

B.4.2 Extra Layer

TABLE B.24: Results of the 4 models with extra layer for different feature combinations - ESC-10.

Model 7: (optimizer: Adam)							Model 8: (optimizer: Adagrad)						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.700**	0.931	**0.700**	**0.684**	0.766	0.450	**mms40**	0.375	0.812	0.375	**0.347**	0.000	0.000
mms80	0.688	**0.952**	0.688	0.681	0.759	0.513	**mms60**	**0.400**	0.824	**0.400**	0.330	**1.000**	0.025
mms60	0.675	0.931	0.675	0.663	**0.860**	0.463	**mmstftq**	0.388	0.837	0.388	0.322	0.333	0.013
mms40	0.675	0.946	0.675	0.661	0.773	0.425	**mms80**	0.375	0.813	0.375	0.315	0.600	0.038
mmstftq40	0.663	0.943	0.663	0.640	0.796	0.488	**mmsqc**	0.375	0.834	0.375	0.300	**1.000**	0.013
mfccstft80	0.638	0.881	0.638	0.632	0.686	0.438	**mmq40**	0.325	**0.858**	0.325	0.289	0.600	0.038
mmstftq	0.650	0.917	0.650	0.629	0.750	0.488	**mfccstft80**	0.375	0.783	0.375	0.284	0.000	0.000
mmcens	0.638	0.923	0.638	0.628	0.667	0.475	**mmsqc80**	0.375	0.837	0.375	0.281	0.667	0.025
mmq	0.638	0.929	0.638	0.624	0.722	0.488	**mmcens**	0.313	0.823	0.313	0.267	0.600	0.038
mmsqc	0.638	0.932	0.638	0.624	0.717	0.475	**mmq**	0.300	0.827	0.300	0.241	0.143	0.013
mmsqc80	0.625	0.929	0.625	0.607	0.692	**0.563**	**mmstftq40**	0.275	0.751	0.275	0.199	0.833	**0.063**
mmcens40	0.600	0.928	0.600	0.594	0.705	0.388	**mfccstft**	0.175	0.787	0.175	0.131	**1.000**	0.013
mfccstft	0.588	0.870	0.588	0.562	0.675	0.338	**mmcens40**	0.175	0.800	0.175	0.111	0.500	0.013
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc	**0.500**	0.879	**0.500**	**0.468**	**1.000**	0.013	**mmstftq**	**0.725**	**0.953**	**0.725**	**0.714**	0.792	0.525
mfccstft80	0.475	0.874	0.475	0.420	0.750	0.038	**mmsqc**	0.700	0.934	0.700	0.688	0.774	0.513
mmstftq	0.425	0.878	0.425	0.414	**1.000**	0.038	**mms80**	0.688	0.938	0.688	0.665	0.731	0.475
mms80	0.425	0.836	0.425	0.406	0.667	0.025	**mms60**	0.663	0.923	0.663	0.658	0.783	0.450
mmq	0.450	0.852	0.450	0.387	0.800	0.050	**mmsqc80**	0.663	0.926	0.663	0.653	0.703	**0.563**
mmcens40	0.388	0.832	0.388	0.370	0.800	0.050	**mmstftq40**	0.663	0.912	0.663	0.652	0.776	0.475
mmsqc80	0.388	**0.889**	0.388	0.369	0.545	**0.075**	**mmq**	0.663	**0.953**	0.663	0.643	**0.813**	0.488
mms60	0.388	0.832	0.388	0.352	**1.000**	0.013	**mmq40**	0.638	0.921	0.638	0.636	0.735	0.450
mmcens	0.300	0.756	0.300	0.293	0.833	0.063	**mmcens**	0.600	0.933	0.600	0.599	0.760	0.475
mmq40	0.313	0.811	0.313	0.265	**1.000**	0.050	**mmcens40**	0.613	0.910	0.613	0.594	0.763	0.363
mmstftq40	0.300	0.785	0.300	0.244	0.000	0.000	**mms40**	0.600	0.917	0.600	0.579	0.756	0.388
mms40	0.250	0.791	0.250	0.241	0.000	0.000	**mfccstft80**	0.588	0.882	0.588	0.568	0.640	0.400
mfccstft	0.150	0.717	0.150	0.129	**1.000**	0.025	**mfccstft**	0.550	0.893	0.550	0.523	0.639	0.288
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.25: Results of the 4 models with extra layer for different feature combinations - ESC-50.

Model 7: (optimizer: Adam)							**Model 8: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.353**	0.880	**0.353**	**0.343**	0.636	0.140	**mmsqc**	**0.143**	0.761	**0.143**	**0.127**	0.700	0.018
mmsqc80	0.350	0.882	0.350	0.329	0.652	0.183	**mms40**	**0.143**	0.774	**0.143**	0.119	**0.857**	0.015
mmstftq40	0.335	0.879	0.335	0.321	0.593	0.168	**mmstftq**	0.118	0.764	0.118	0.108	0.667	0.010
mmstftq	0.328	0.879	0.328	0.315	0.664	**0.188**	**mmq40**	0.128	0.777	0.128	0.101	0.500	0.013
mmq	0.328	0.865	0.328	0.311	0.652	0.150	**mmstftq40**	0.110	0.746	0.110	0.096	0.556	0.013
mmsqc	0.338	0.880	0.338	0.311	0.628	0.178	**mmcens**	0.103	0.761	0.103	0.094	0.500	0.010
mfccstft	0.313	0.873	0.313	0.306	0.602	0.133	**mmcens40**	0.123	0.764	0.123	0.093	0.800	**0.020**
mmcens	0.323	0.870	0.323	0.304	**0.683**	0.140	**mmq**	0.100	0.750	0.100	0.084	0.600	0.008
mmcens40	0.318	0.875	0.318	0.302	0.682	0.145	**mms60**	0.115	0.750	0.115	0.082	0.429	0.008
mms60	0.325	0.862	0.325	0.300	0.579	0.138	**mfccstft80**	0.105	0.764	0.105	0.078	0.600	0.008
mms80	0.310	0.873	0.310	0.296	0.602	0.155	**mfccstft**	0.118	**0.780**	0.118	0.077	0.700	0.018
mfccstft80	0.315	0.860	0.315	0.295	0.573	0.168	**mmsqc80**	0.093	0.750	0.093	0.071	0.625	0.013
mms40	0.318	**0.891**	0.318	0.293	0.626	0.143	**mms80**	0.093	0.754	0.093	0.068	0.533	0.020
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000
Model 9: (optimizer: Adamax)							**Model 10: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mfccstft80	**0.158**	**0.829**	**0.158**	**0.123**	0.600	0.008	**mmsqc80**	**0.355**	0.875	**0.355**	**0.340**	0.676	**0.188**
mmsqc	0.145	0.802	0.145	0.118	0.000	0.000	**mmstftq**	0.353	0.879	0.353	0.332	0.613	0.183
mmsqc80	0.133	0.786	0.133	0.096	0.000	0.000	**mmsqc**	0.343	**0.889**	0.343	0.326	0.646	0.155
mmq	0.113	0.741	0.113	0.081	0.800	**0.010**	**mms80**	0.343	0.881	0.343	0.322	0.575	0.153
mmq40	0.103	0.776	0.103	0.077	0.000	0.000	**mmstftq40**	0.340	0.881	0.340	0.315	**0.758**	0.173
mfccstft	0.118	0.783	0.118	0.075	0.333	0.003	**mmcens40**	0.323	0.879	0.323	0.309	0.653	0.123
mms40	0.093	0.757	0.093	0.072	0.000	0.000	**mmq40**	0.310	0.865	0.310	0.289	0.637	0.145
mms80	0.090	0.749	0.090	0.062	0.000	0.000	**mfccstft**	0.305	0.869	0.305	0.286	0.573	0.138
mmstftq	0.085	0.758	0.085	0.062	0.500	0.003	**mmcens**	0.318	0.868	0.318	0.282	0.585	0.138
mmstftq40	0.085	0.759	0.085	0.062	0.000	0.000	**mms40**	0.310	0.884	0.310	0.280	0.614	0.128
mmcens	0.085	0.691	0.085	0.058	**1.000**	0.003	**mmq**	0.293	0.876	0.293	0.262	0.700	0.105
mmcens40	0.070	0.706	0.070	0.055	0.000	0.000	**mfccstft80**	0.285	0.856	0.285	0.261	0.508	0.150
mms60	0.068	0.760	0.068	0.041	0.000	0.000	**mms60**	0.295	0.879	0.295	0.259	0.588	0.125
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.4.3 Dropout Rate of 0.2

TABLE B.26: Results of the 4 best models for dropout of 0.2 and different feature combinations - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq	**0.725**	0.936	**0.725**	**0.724**	**0.750**	**0.713**	**mmstftq**	**0.638**	0.907	**0.638**	**0.639**	0.707	0.513
mmcens40	0.713	**0.952**	0.713	0.708	0.727	0.700	**mmq40**	0.600	0.928	0.600	0.569	0.724	**0.525**
mmstftq40	0.713	0.929	0.713	0.699	0.733	0.688	**mmcens**	0.550	0.912	0.550	0.528	0.696	0.400
mmq40	0.700	**0.952**	0.700	0.694	0.724	0.688	**mmcens40**	0.550	0.921	0.550	0.525	0.620	0.388
mmq	0.688	0.932	0.688	0.687	0.705	0.688	**mms40**	0.563	0.919	0.563	0.520	**0.773**	0.425
mms60	0.700	0.926	0.700	0.684	0.718	0.700	**mmq**	0.550	0.894	0.550	0.519	0.609	0.350
mms40	0.688	0.941	0.688	0.681	0.689	0.638	**mmsqc**	0.563	0.911	0.563	0.515	0.681	0.400
mmsqc	0.675	0.916	0.675	0.670	0.707	0.663	**mfccstft**	0.525	0.891	0.525	0.490	0.625	0.313
mms80	0.675	0.911	0.675	0.664	0.703	0.650	**mfccstft80**	0.513	0.912	0.513	0.489	0.604	0.400
mmcens	0.663	0.896	0.663	0.658	0.680	0.638	**mms80**	0.500	0.891	0.500	0.479	0.574	0.388
mmsqc80	0.675	0.923	0.675	0.656	0.697	0.663	**mmsqc80**	0.500	0.901	0.500	0.476	0.547	0.363
mfccstft	0.613	0.881	0.613	0.611	0.632	0.538	**mms60**	0.488	0.904	0.488	0.465	0.581	0.313
mfccstft80	0.613	0.877	0.613	0.603	0.618	0.588	**mmstftq40**	0.500	**0.931**	0.500	0.463	0.654	0.425
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq40	**0.700**	0.939	**0.700**	**0.694**	0.758	0.625	**mms60**	**0.775**	0.956	**0.775**	**0.774**	**0.787**	0.738
mmstftq	**0.700**	**0.958**	**0.700**	0.691	**0.782**	0.538	**mmq40**	0.725	0.942	0.725	0.719	0.753	**0.725**
mmsqc80	**0.700**	0.942	**0.700**	0.689	0.735	0.625	**mmcens40**	0.700	0.926	0.700	0.691	0.757	0.700
mmq	0.675	0.934	0.675	0.665	0.699	**0.638**	**mmstftq40**	0.688	0.925	0.688	0.682	0.714	0.688
mmcens40	0.663	0.950	0.663	0.654	0.750	0.563	**mmsqc**	0.675	0.921	0.675	0.677	0.761	0.638
mmsqc	0.663	0.946	0.663	0.644	0.742	0.613	**mmsqc80**	0.675	0.939	0.675	0.675	0.693	0.650
mms60	0.650	0.949	0.650	0.642	**0.782**	0.538	**mmq**	0.675	0.917	0.675	0.669	0.712	0.650
mmq40	0.663	0.947	0.663	0.641	0.772	0.550	**mms80**	0.675	0.913	0.675	0.665	0.701	0.675
mfccstft	0.638	0.908	0.638	0.624	0.667	0.525	**mfccstft80**	0.663	0.867	0.663	0.652	0.699	0.638
mms40	0.625	0.945	0.625	0.616	0.724	0.525	**mmcens**	0.650	0.916	0.650	0.646	0.662	0.613
mms80	0.613	0.937	0.613	0.612	0.689	0.525	**mmstftq**	0.650	0.899	0.650	0.638	0.649	0.600
mmcens	0.613	0.938	0.613	0.592	0.701	0.588	**mms40**	0.650	**0.953**	0.650	0.637	0.658	0.625
mfccstft80	0.575	0.914	0.575	0.554	0.609	0.488	**mfccstft**	0.588	0.896	0.588	0.583	0.634	0.563
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.27: Results of the 4 best models for dropout of 0.2 and different feature combinations - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmq40	**0.398**	0.829	**0.398**	**0.386**	0.472	0.358	**mms40**	**0.313**	0.863	**0.313**	**0.289**	**0.519**	0.140
mmstftq	0.390	0.810	0.390	**0.386**	0.452	0.340	**mmcens40**	0.303	**0.868**	0.303	0.279	0.495	0.130
mmstftq40	0.393	**0.839**	0.393	0.385	**0.476**	0.350	**mms80**	0.275	0.852	0.275	0.263	0.500	0.138
mmsqc80	0.393	0.811	0.393	0.382	0.449	**0.360**	**mmsqc80**	0.275	0.844	0.275	0.262	0.510	0.130
mmsqc	0.390	0.817	0.390	0.376	0.445	0.333	**mmstftq**	0.255	0.852	0.255	0.245	0.405	0.113
mmq	0.383	0.817	0.383	0.373	0.445	0.358	**mmq**	0.253	0.858	0.253	0.244	0.496	**0.148**
mms60	0.368	0.828	0.368	0.350	0.439	0.308	**mfccstft**	0.270	0.856	0.270	0.243	0.468	0.110
mfccstft80	0.358	0.811	0.358	0.349	0.429	0.333	**mfccstft80**	0.248	0.852	0.248	0.241	0.506	0.105
mmcens	0.353	0.820	0.353	0.341	0.416	0.305	**mmstftq40**	0.238	0.849	0.238	0.223	0.500	0.120
mms80	0.360	0.805	0.360	0.339	0.405	0.308	**mmcens**	0.238	0.827	0.238	0.220	0.435	0.125
mmcens40	0.345	0.823	0.345	0.335	0.405	0.283	**mms60**	0.250	0.856	0.250	0.219	0.429	0.113
mfccstft	0.338	0.804	0.338	0.334	0.397	0.300	**mmq40**	0.243	0.844	0.243	0.218	0.486	0.128
mms40	0.343	0.826	0.343	0.327	0.436	0.315	**mmsqc**	0.228	0.854	0.228	0.212	0.490	0.118
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.383**	0.875	**0.383**	**0.373**	**0.583**	**0.228**	**mfccstft80**	**0.403**	0.816	**0.403**	**0.388**	0.459	0.353
mmstftq	0.365	**0.889**	0.365	0.349	0.556	0.223	**mmsqc**	0.385	**0.840**	0.385	0.371	**0.472**	**0.358**
mms80	0.348	0.878	0.348	0.334	0.555	0.215	**mmstftq**	0.378	0.825	0.378	0.371	0.452	0.350
mfccstft80	0.345	0.874	0.345	0.329	0.544	0.218	**mmstftq40**	0.370	0.827	0.370	0.364	0.457	0.323
mmsqc	0.350	0.874	0.350	0.328	0.545	0.198	**mms40**	0.368	0.818	0.368	0.354	0.435	0.310
mmq	0.335	0.866	0.335	0.316	0.536	0.188	**mms80**	0.375	0.821	0.375	0.352	0.424	0.330
mmcens	0.330	0.865	0.330	0.314	0.492	0.160	**mmsqc80**	0.363	0.810	0.363	0.348	0.417	0.335
mms60	0.323	0.872	0.323	0.312	0.514	0.178	**mmcens**	0.360	0.809	0.360	0.346	0.427	0.308
mfccstft	0.325	0.863	0.325	0.307	0.504	0.155	**mmq**	0.348	0.823	0.348	0.345	0.401	0.310
mmcens40	0.323	0.878	0.323	0.306	0.538	0.175	**mmq40**	0.343	0.813	0.343	0.337	0.403	0.300
mmstftq40	0.313	0.883	0.313	0.304	0.526	0.178	**mmcens40**	0.353	0.823	0.353	0.334	0.421	0.295
mmq40	0.318	0.875	0.318	0.300	0.537	0.183	**mfccstft**	0.333	0.800	0.333	0.319	0.401	0.290
mms40	0.308	0.877	0.308	0.292	0.553	0.158	**mms60**	0.333	0.801	0.333	0.319	0.410	0.308
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

B.4.4 Dropout Rate of 0.6

TABLE B.28: Results of the 4 best models for dropout of 0.6 and different feature combinations - ESC-10.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: Adagrad)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmstftq40	**0.638**	0.921	**0.638**	**0.623**	0.852	**0.288**	**mmsqc**	**0.375**	0.813	**0.375**	**0.350**	0.750	0.038
mmsqc	**0.638**	0.923	**0.638**	0.622	**1.000**	0.238	**mmstftq**	0.350	0.804	0.350	0.301	0.700	0.088
mmsqc80	0.613	0.925	0.613	0.599	0.696	0.200	**mmstftq40**	0.363	**0.833**	0.363	0.290	0.714	0.125
mmq	0.613	0.925	0.613	0.599	0.810	0.213	**mms60**	0.350	0.797	0.350	0.280	**0.786**	0.138
mmcens	0.613	**0.931**	0.613	0.586	0.850	0.213	**mms40**	0.300	0.788	0.300	0.250	0.727	0.100
mmstftq	0.588	0.913	0.588	0.559	0.870	0.250	**mms80**	0.313	0.776	0.313	0.244	0.500	0.075
mms80	0.575	0.895	0.575	0.540	0.733	0.138	**mmcens**	0.300	0.766	0.300	0.240	0.400	0.050
mms60	0.488	0.877	0.488	0.490	0.813	0.163	**mmsqc80**	0.313	0.768	0.313	0.234	0.556	**0.188**
mms40	0.475	0.914	0.475	0.452	0.769	0.125	**mmcens40**	0.250	0.746	0.250	0.176	0.412	0.088
mfccstft80	0.400	0.835	0.400	0.329	0.533	0.100	**mmq**	0.263	0.825	0.263	0.174	0.542	0.163
mmq40	0.325	0.802	0.325	0.319	**1.000**	0.113	**mmq40**	0.225	0.781	0.225	0.161	0.692	0.113
mfccstft	0.388	0.831	0.388	0.309	0.529	0.113	**mfccstft**	0.150	0.769	0.150	0.101	0.500	0.038
mmcens40	0.313	0.784	0.313	0.285	0.800	0.050	**mfccstft80**	0.163	0.737	0.163	0.084	0.625	0.063
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.525**	**0.877**	**0.525**	**0.512**	0.250	0.013	**mmsqc80**	0.600	**0.923**	0.600	**0.597**	0.857	0.225
mmstftq	0.413	0.831	0.413	0.399	0.833	0.063	**mmstftq**	**0.613**	0.909	**0.613**	**0.597**	0.731	**0.238**
mmq	0.375	0.846	0.375	0.353	0.857	**0.075**	**mfccstft80**	0.575	0.858	0.575	0.545	0.593	0.200
mms40	0.388	0.847	0.388	0.327	0.000	0.000	**mms60**	0.563	0.907	0.563	0.513	0.857	0.150
mms60	0.325	0.834	0.325	0.292	0.750	0.038	**mmsqc**	0.525	0.914	0.525	0.512	**1.000**	0.150
mmstftq40	0.288	0.693	0.288	0.263	**1.000**	0.013	**mmcens**	0.513	0.877	0.513	0.475	0.625	0.125
mmcens40	0.288	0.805	0.288	0.250	**1.000**	0.013	**mmq**	0.438	0.859	0.438	0.444	0.800	0.150
mms80	0.275	0.728	0.275	0.249	0.400	0.025	**mmq40**	0.425	0.877	0.425	0.424	0.750	0.113
mmcens	0.263	0.747	0.263	0.248	0.857	**0.075**	**mms80**	0.400	0.869	0.400	0.411	0.692	0.113
mmq40	0.263	0.691	0.263	0.235	**1.000**	0.013	**mmstftq40**	0.425	0.837	0.425	0.367	0.867	0.163
mfccstft80	0.188	0.667	0.188	0.183	0.000	0.000	**mms40**	0.288	0.843	0.288	0.278	0.538	0.088
mfccstft	0.175	0.644	0.175	0.175	0.000	0.000	**mmcens40**	0.263	0.762	0.263	0.269	**1.000**	0.088
mmsqc	0.200	0.648	0.200	0.171	**1.000**	0.013	**mfccstft**	0.213	0.735	0.213	0.188	0.375	0.038
mfccmel40	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel40**	0.100	0.500	0.100	0.018	0.000	0.000
mfccmel80	0.100	0.500	0.100	0.018	0.000	0.000	**mfccmel80**	0.100	0.500	0.100	0.018	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

TABLE B.29: Results of the 4 best models for dropout of 0.6 and different feature combinations - ESC-50.

Model 1: (optimizer: Adam)							**Model 2: (optimizer: SGD)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mms60	**0.180**	0.810	**0.180**	**0.167**	0.625	0.013	**mmstftq40**	**0.053**	0.553	**0.053**	**0.034**	0.750	0.008
mmstftq40	0.160	0.771	0.160	0.156	0.500	0.005	**mmq40**	**0.053**	0.544	**0.053**	**0.034**	**1.000**	0.008
mfccstft80	0.165	0.778	0.165	0.146	0.533	0.020	**mmcens40**	0.050	0.547	0.050	0.031	0.800	**0.010**
mmsqc	0.153	0.782	0.153	0.141	0.667	0.020	**mfccstft80**	0.045	0.622	0.045	0.031	0.000	0.000
mmsqc80	0.155	0.815	0.155	0.139	**0.818**	0.023	**mms40**	**0.053**	0.545	**0.053**	0.030	0.667	0.005
mms80	0.158	**0.822**	0.158	0.135	0.800	0.020	**mfccstft**	**0.053**	**0.647**	**0.053**	0.029	0.000	0.000
mmstftq	0.143	0.783	0.143	0.135	0.800	**0.030**	**mms60**	0.048	0.538	0.048	0.028	0.600	0.008
mmcens40	0.145	0.792	0.145	0.133	0.778	0.018	**mmsqc**	0.048	0.554	0.048	0.026	**1.000**	0.005
mmq	0.148	0.783	0.148	0.130	0.778	0.018	**mmcens**	0.035	0.533	0.035	0.019	0.000	0.000
mmcens	0.120	0.779	0.120	0.112	0.625	0.013	**mmsqc80**	0.035	0.529	0.035	0.019	**1.000**	0.003
mmq40	0.120	0.749	0.120	0.103	0.667	0.005	**mms80**	0.033	0.545	0.033	0.018	**1.000**	0.003
mfccstft	0.113	0.743	0.113	0.094	0.200	0.003	**mmq**	0.033	0.525	0.033	0.011	**1.000**	0.003
mms40	0.088	0.737	0.088	0.081	0.667	0.005	**mmstftq**	0.023	0.503	0.023	0.009	0.200	0.003
mfccmel40	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	0.000	0.000	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000
Model 3: (optimizer: Adamax)							**Model 4: (optimizer: Nadam)**						
Features	acc	AUC	micro f1score	macro f1score	prec	recall	Features	acc	AUC	micro f1score	macro f1score	prec	recall
mmsqc80	**0.050**	0.555	**0.050**	**0.036**	**0.000**	**0.000**	**mmsqc80**	**0.203**	**0.837**	**0.203**	**0.159**	0.818	0.023
mmstftq	0.035	0.574	0.035	0.025	**0.000**	**0.000**	**mmstftq40**	0.170	0.794	0.170	0.150	0.600	0.008
mmsqc	0.038	0.548	0.038	0.024	**0.000**	**0.000**	**mmcens40**	0.160	0.807	0.160	0.147	0.429	0.008
mfccstft	0.033	**0.584**	0.033	0.014	**0.000**	**0.000**	**mmq**	0.165	0.805	0.165	0.141	0.667	0.015
mfccstft80	0.035	0.529	0.035	0.013	**0.000**	**0.000**	**mmq40**	0.158	0.803	0.158	0.137	**1.000**	**0.028**
mmcens	0.028	0.523	0.028	0.012	**0.000**	**0.000**	**mms80**	0.145	0.770	0.145	0.129	0.857	0.015
mms40	0.025	0.526	0.025	0.009	**0.000**	**0.000**	**mfccstft80**	0.138	0.786	0.138	0.127	0.714	0.013
mms60	0.023	0.503	0.023	0.005	**0.000**	**0.000**	**mmstftq**	0.138	0.775	0.138	0.120	**1.000**	0.013
mmcens40	0.023	0.530	0.023	0.003	**0.000**	**0.000**	**mmcens**	0.123	0.765	0.123	0.111	0.900	0.023
mmq40	0.020	0.511	0.020	0.001	**0.000**	**0.000**	**mmsqc**	0.130	0.797	0.130	0.108	0.667	0.005
mms80	0.020	0.502	0.020	0.001	**0.000**	**0.000**	**mms40**	0.118	0.789	0.118	0.093	**1.000**	0.003
mmstftq40	0.020	0.504	0.020	0.001	**0.000**	**0.000**	**mfccstft**	0.105	0.731	0.105	0.091	0.600	0.008
mmq	0.020	0.502	0.020	0.001	**0.000**	**0.000**	**mms60**	0.093	0.700	0.093	0.076	**1.000**	0.005
mfccmel40	0.020	0.500	0.020	0.001	**0.000**	**0.000**	**mfccmel40**	0.020	0.500	0.020	0.001	0.000	0.000
mfccmel80	0.020	0.500	0.020	0.001	**0.000**	**0.000**	**mfccmel80**	0.020	0.500	0.020	0.001	0.000	0.000

acc - accuracy; AUC - area under the receiver operating characteristic curve; prec - precision.
All metrics range from $[0, 1]$ (the higher, the better).

Bibliography

[1] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. VATT: transformers for multimodal self-supervised learning from raw video, audio and text. *CoRR*, abs/2104.11178, 2021. URL https://arxiv.org/abs/2104.11178. [Cited on pages 2, 13, 19, 21, 31, 34, and 122.]

[2] Joy Krishan Das, Arka Ghosh, Abhijit Kumar Pal, Sumit Dutta, and Amitabha Chakrabarty. Urban sound classification using convolutional neural network and long short term memory based on multiple features. In *2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS)*, pages 1–9, 2020. doi: 10.1109/ICDS50568.2020.9268723. [Cited on pages 2, 9, 10, 14, 30, 34, 35, and 122.]

[3] Joy Krishan Das, Amitabha Chakrabarty, and Md. Jalil Piran. Environmental sound classification using convolution neural networks with different integrated loss functions. *Expert Systems*, 09 2021. doi: https://doi.org/10.1111/exsy.12804. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.12804. [Cited on pages 1, 2, 9, 10, 14, 34, 35, and 122.]

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423. [Cited on page 15.]

[5] Dolby E Bitstreams. Standards and practices for authoring dolby digital and dolby e bitstreams. 2002. [Cited on page 9.]

[6] Timothy Dozat. Incorporating Nesterov Momentum into Adam. In *Proceedings of the 4th International Conference on Learning Representations*, pages 1–4, 2016. [Cited on page 49.]

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12: 2121–2159, 07 2011. [Cited on page 47.]

[8] David Elliott, Carlos E. Otero, Steven Wyatt, and Evan Martino. Tiny transformers for environmental sound classification at the edge. *CoRR*, abs/2103.12157, 2021. URL https://arxiv.org/abs/2103.12157. [Cited on pages 2, 13, 15, 16, 21, 31, 34, and 122.]

[9] Theodore Giannakopoulos, Evaggelos Spyrou, and Stavros J. Perantonis. Recognition of urban sound events using deep context-aware feature extractors and handcrafted features. In John MacIntyre, Ilias Maglogiannis, Lazaros Iliadis, and Elias Pimenidis, editors, *Artificial Intelligence Applications and Innovations*, pages 184–195, Cham, 2019. Springer International Publishing. ISBN 978-3-030-19909-8. [Cited on pages 2, 24, 26, 34, 35, and 122.]

[10] Pablo Gimeno, Ignacio Viñals, Alfonso Ortega, Antonio Miguel, and Eduardo Lleida. Multiclass audio segmentation based on recurrent neural networks for broadcast domain data. 2020. [Cited on pages 2, 23, 26, 34, and 35.]

[11] Yuan Gong, Yu-An Chung, and James R. Glass. AST: audio spectrogram transformer. *CoRR*, abs/2104.01778, 2021. URL https://arxiv.org/abs/2104.01778. [Cited on pages xix, 2, 13, 17, 21, 34, 107, 108, and 122.]

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385. [Cited on pages xviii, 85, and 86.]

[13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2016. URL https://arxiv.org/abs/1608.06993. [Cited on page 87.]

[14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980. [Cited on page 48.]

[15] Qiuqiang Kong, Yong Xu, and Mark Plumbley. Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, PP:1–1, 08 2020. doi: 10.1109/TASLP.2020.3014737. [Cited on pages 2, 13, 14, 21, 31, and 34.]

[16] Khaled Koutini, Jan Schlüter, Hamid Eghbal-zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. *CoRR*, abs/2110.05069, 2021. URL https://arxiv.org/abs/2110.05069. [Cited on pages 2, 14, 20, 21, 31, 34, and 122.]

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. URL https://arxiv.org/abs/1711.05101. [Cited on page 91.]

[18] Jederson S. Luz, Myllena C. Oliveira, Flávio H.D. Araújo, and Deborah M.V. Magalhães. Ensemble of handcrafted and deep features for urban sound classification. *Applied Acoustics*, 175:107819, 2021. ISSN 0003-682X. doi: https://doi.org/10.1016/j.apacoust.2020.107819. URL https://www.sciencedirect.com/science/article/pii/S0003682X20309245. [Cited on pages 2, 25, 26, 34, 35, and 122.]

[19] Irene Martín-Morató, Maximo Cobos, and Francesc J. Ferri. Adaptive distance-based pooling in convolutional neural networks for audio event classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1925–1935, 2020. doi: 10.1109/TASLP.2020.3001683. [Cited on pages 23, 26, 34, 35, and 122.]

[20] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, pages 18–25, 2015. [Cited on pages 10 and 40.]

[21] Wenjie Mu, Bo Yin, Xianqing Huang, Jiali Xu, and Zehua Du. Environmental sound classification using temporal-frequency attention based convolutional neural network. *Scientific Reports*, 11, 11 2021. doi: 10.1038/s41598-021-01045-4. [Cited on pages 2, 9, 12, 14, 31, 34, 66, and 122.]

[22] Zohaib Mushtaq and Shun-Feng Su. Efficient classification of environmental sounds through multiple features aggregation and data enhancement techniques for spectrogram images. *Symmetry*, 12(11), 2020. ISSN 2073-8994. doi: 10.3390/sym12111822. URL https://www.mdpi.com/2073-8994/12/11/1822. [Cited on pages 1, 2, 32, 33, 34, 35, and 122.]

[23] Sooyoung Park, Youngho Jeong, and TaeJin Lee. Many-to-many audio spectrogram tansformer: Transformer for sound event localization and detection. In *DCASE*, pages 105–109, 2021. URL http://dcase.community/documents/workshop2021/proceedings/DCASE2021Workshop_Park_39.pdf. [Cited on pages 2, 13, 18, 21, and 34.]

[24] Karol J. Piczak. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia*, pages 1015–1018. ACM Press. ISBN 978-1-4503-3459-4. doi: 10.1145/2733373.2806390. URL http://dl.acm.org/citation.cfm?doid=2733373.2806390. [Cited on page 38.]

[25] Tianhao Qiao, Shunqing Zhang, Shan Cao, and Shugong Xu. High accurate environmental sound classification: Sub-spectrogram segmentation versus temporal-frequency attention mechanism. *Sensors*, 21(16), 2021. ISSN 1424-8220. doi: 10.3390/s21165500. URL https://www.mdpi.com/1424-8220/21/16/5500. [Cited on pages 2, 25, 27, 29, 34, and 122.]

[26] Nicolae-Catalin Ristea, Radu Tudor Ionescu, and Fahad Shahbaz Khan. Septr: Separable transformer for audio spectrogram processing, 2022. URL https://arxiv.org/abs/2203.09581. [Cited on pages 2, 28, 29, 34, and 122.]

[27] Sebastian Ruder. An overview of gradient descent optimization algorithms, 2016. URL https://arxiv.org/abs/1609.04747. [Cited on page 47.]

[28] J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *22nd ACM International Conference on Multimedia (ACM-MM'14)*, pages 1041–1044, Orlando, FL, USA, Nov. 2014. [Cited on pages 37 and 65.]

[29] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24:279–283, 2017. [Cited on pages 2, 9, 14, 34, 35, and 122.]

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html. [Cited on page 57.]

[31] Yui Sudo, Katsutoshi Itoyama, Kenji Nishida, and Kazuhiro Nakadai. Multichannel environmental sound segmentation. *Applied Intelligence*, 51:8245–8259, 2021. doi: 10.1007/s10489-021-02314-5. [Cited on pages 28, 31, and 34.]

[32] Abbas Shah Syed, Daniel Sierra-Sosa, Anup Kumar, and Adel Elmaghraby. Iot in smart cities: A survey of technologies, practices and challenges. *Smart Cities*, 4(2): 429–475, 2021. ISSN 2624-6511. doi: 10.3390/smartcities4020024. URL https://www.mdpi.com/2624-6511/4/2/24. [Cited on page 1.]

[33] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. URL https://arxiv.org/abs/1409.4842. [Cited on page 89.]

[34] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015. URL https://arxiv.org/abs/1512.00567. [Cited on page 89.]

[35] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. doi: 10.1109/CVPR.2016.308. [Cited on page 89.]

[36] Tycho Max Sylvester Tax, Jose Luis Diez Antich, Hendrik Purwins, and Lars Maaløe. Utilizing domain knowledge in end-to-end audio processing. *ArXiv*, abs/1712.00254, 2017. [Cited on pages 22, 26, 34, 35, and 122.]

[37] Theodoros Theodorou, Iosif Mporas, and Nikos Fakotakis. An overview of automatic audio segmentation. *International Journal of Information Technology and Computer Science*, 6:1–9, 10 2014. doi: 10.5815/ijitcs.2014.11.01. [Cited on page 22.]

[38] Achyut Mani Tripathi and Aakansha Mishra. Environment sound classification using an attention-based residual neural network. *Neurocomputing*, 460:409–423, 2021. ISSN 0925-2312. doi: 10.1016/j.neucom.2021.06.031. URL https://www.sciencedirect.com/science/article/pii/S0925231221009358. [Cited on pages 2, 27, 29, 34, and 122.]

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf. [Cited on pages xix, 15, 105, and 107.]

[40] Satvik Venkatesh, David Moffat, and Eduardo Reck Miranda. You only hear once: A yolo-like algorithm for audio segmentation and sound event detection, 2021. [Cited on pages 30, 31, and 34.]

[41] Steven Wyatt, David Elliott, Akshay Aravamudan, Carlos E. Otero, Luis D. Otero, Georgios C. Anagnostopoulos, Anthony O. Smith, Adrian M. Peter, Wesley Jones, Steven Leung, and Eric Lam. Environmental sound classification with tiny transformers in noisy edge environments. In *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, pages 309–314, 2021. doi: 10.1109/WF-IoT51360.2021.9596007. [Cited on pages 2, 13, 15, 16, 21, and 34.]

[42] Matthew D. Zeiler. Adadelta: An adaptive learning rate method, 2012. URL https://arxiv.org/abs/1212.5701. [Cited on page 47.]

[43] Zhichao Zhang, Shugong Xu, Shunqing Zhang, Tianhao Qiao, and Shan Cao. Learning attentive representations for environmental sound classification. *IEEE Access*, 7: 130327–130339, 2019. doi: 10.1109/ACCESS.2019.2939495. [Cited on pages 2, 25, 29, 34, and 122.]

[44] Zhichao Zhang, Shugong Xu, Shunqing Zhang, Tianhao Qiao, and Shan Cao. Attention based convolutional recurrent neural network for environmental sound classification. *Neurocomputing*, 453, 09 2020. doi: 10.1016/j.neucom.2020.08.069. [Cited on pages 2, 25, 29, 34, and 122.]

[45] Pablo Zinemanas, Martín Rocamora, Marius Miron, Frederic Font, and Xavier Serra. An interpretable deep learning model for automatic sound classification. *Electronics*, 10(7), 2021. ISSN 2079-9292. doi: 10.3390/electronics10070850. URL https://www.mdpi.com/2079-9292/10/7/850. [Cited on pages 9, 11, 14, 30, 34, 35, and 122.]

[46] İlker Türker and Serkan Aksu. Connectogram – a graph-based time dependent representation for sounds. *Applied Acoustics*, 191:108660, 2022. ISSN 0003-682X. doi: 10.1016/j.apacoust.2022.108660. URL https://www.sciencedirect.com/science/article/pii/S0003682X22000342. [Cited on pages 2, 32, 33, 34, and 122.]