Université de BLIDA 1

Faculté des Sciences

Département d'Informatique

MASTER THESIS

Option : Ingénierie des Logiciels

DEEP NEURAL NETWORKS-BASED SYSTEMS FOR ACOUSTIC SCENE RECOGNITION: COMPARATIVE STUDY BETWEEN DATA AUGMENTATION PARADIGMS

By:

Yousra ZEKRIFA & Zhor DIFFALLAH

In front of a jury composed of:

Ms. BACHA Sihem	President
Ms. GUESSOUM Dalila	Examiner
Ms. YKHLEF Hadjer	Supervisor
Mr. YKHLEF Farid	Supervisor

2019/2020

Abstract

Acoustic scene classification (ASC) refers to the identification of the environment in which audio excerpts have been recorded, which associates a semantic label to each audio recording. This task has drawn lots of attention during the past several years as a result of machines and electronics such as smartphones, autonomous robots, or security systems acquiring the ability to perceive sounds. This work aims to classify 10 common indoor and outdoor locations using environmental sounds. To accomplish the ensuing task, we have performed multiple experiments using a dataset which consists of 14400 sound files. The goal is to explore three different aspects of an ASC system: deep learning architecture, feature extraction technique and data augmentation method. In particular, two deep neural networks have been employed in the construction of our systems namely: Residual Neural Network (ResNet) and Alex Neural Network (AlexNet), along with a combination of feature representations based on signal processing techniques. Specifically, 3 feature sets have been extracted: Log-Mel energies, ΔLog-Mel energies and ΔΔLog-Mel energies. Moreover, this work deeply explores the use of Mixup data augmentation method and the effects of varying its hyperparameters in reducing the chance of overfitting. A series of thorough experimental comparisons and statistical tests have been performed with regards to evaluating our systems. The obtained results indicate that a proper choice of the feature set is crucial in view of the deep learning architecture. Additionally, statistical testing has shown the significant impact of mixup data augmentation technique on the predictive performance of our models, as systems trained on augmented data have a considerably better generalization ability compared to the counterpart systems trained on original data. Moreover, we have found that a well-tuned mixup hyperparameter α significantly improves the classification system performance.

Keywords: Acoustic Scene Classification, Feature Extraction, Data Augmentation, Deep Learning, Mixup, Statistical Tests.

Résumé

La classification de scène sonore consiste à identifier l'environnement dans lequel des extraits audios ont été enregistrés, et à associer un label sémantique à chaque enregistrement audio. Cette tâche a attiré beaucoup d'attention au cours des dernières années en raison de l'acquisition de la capacité de perception des sons par les machines et les appareils électroniques tels que les smartphones, les robots autonomes ou les systèmes de sécurité. Ce travail vise à classer 10 lieux communs intérieurs et extérieurs en utilisant les sons de l'environnement. Pour accomplir la tâche qui s'ensuit, nous avons réalisé de multiples expériences en utilisant une base de données constitué de 14400 fichiers sonores. Le but est d'explorer trois aspects différents d'un système de classification des scènes sonores : architecture d'apprentissage profond, technique d'extraction de caractéristiques et méthode d'augmentation des données. En particulier, deux réseaux neuronaux profonds ont été utilisés dans la construction de nos systèmes, à savoir le réseau neuronal résiduel (ResNet) et le réseau de neurones convolutif Alex (AlexNet), ainsi qu'une combinaison de représentations de caractéristiques basées sur des techniques de traitement du signal. Plus précisément, trois ensembles de caractéristiques ont été extraits : les Log-Mel énergies, ΔLog-Mel énergies et ΔΔLog-Mel énergies. En outre, ce travail explore l'utilisation de la méthode d'augmentation des données Mixup et les effets de la variation de ses hyperparamètres pour réduire le risque de surapprentissage. Une série de comparaisons expérimentales et de tests statistiques approfondis ont été réalisés dans le but d'évaluer nos systèmes. Les résultats obtenus indiquent qu'un choix approprié de l'ensemble des fonctionnalités est crucial en tenant compte de l'architecture d'apprentissage approfondie. En outre, les tests statistiques ont montré l'impact significatif de la technique d'augmentation des données Mixup sur la performance prédictive de nos modèles, car les systèmes formés sur des données augmentées ont une capacité de généralisation considérablement meilleure que les systèmes homologues formés sur des données originales. De plus, nous avons constaté qu'un hyperparamètre du Mixup bien réglé α améliore considérablement les performances du système de classification.

Mots-clés: Classification des scènes sonores, Extraction de caractéristiques, Augmentation des données, Apprentissage approfondi, Mixup, Tests statistiques.

ملخص

يشير تصنيف المشهد الصوتي إلى تعريف البيئة التي تم فيها تسجيل مقتطفات صوتية، والتي تربط تسمية دلالية بكل تسجيل صوتي. حازت هذه المهمة الكثير من الاهتمام خلال السنوات العديدة الماضية نتيجة اكتساب الآلات والإلكترونيات مثل الهواتف الذكية أو الروبوتات المستقلة أو أنظمة الأمان القدرة على إدراك الأصوات. يهدف هذا العمل إلى تصنيف 10 مواقع داخلية وخارجية شائعة باستخدام الأصوات البيئية. لإنجاز هذه المهمة، أجرينا تجارب متعددة لاستكشاف ثلاثة جوانب مختلفة لنظام تصنيف المشهد الصوتي: بنية التعلم العميق، تقنية استخراج الميزات وطريقة زيادة البيانات. على وجه الخصوص، تم استخدام شبكتين عصبيتين عميقتين في بناء أنظمتنا وهما: الشبكة العصبية المتبقية (ResNet) والشبكة العصبية التلافيفية (AlexNet)، جنبًا إلى جنب مع مجموعة من عروض الميزات القائمة على تقنيات معالجة الإشارات. على وجه التحديد، تم استخراج 3 مجموعات ميزات: طاقات Log-Mel وطاقات Log-Me Δ وطاقات ΔΔLog-Mel. علاوة على ذلك، يبحث هذا العمل في طريقة زيادة البيانات Mixup وتأثيرات تغيير المعلمات الفائقة في تقليل فرصة التجهيز الزائد. تم إجراء سلسلة من المقارنات التجريبية الشاملة والاختبارات الإحصائية فيما يتعلق بتقييم أنظمتنا. تشير النتائج التي تم الحصول عليها إلى أن الاختيار الصحيح لمجموعة الميزات أمر بالغ الأهمية في ضوء بنية التعلم العميق. بالإضافة إلى ذلك، أظهر الاختبار الإحصائي التأثير الكبير لتقنية زيادة البيانات المختلطة على الأداء التنبئي لنماذجنا، حيث تتمتع الأنظمة المدربة على البيانات المعززة بقدرة تعميم أفضل بكثير مقارنة بالأنظمة المماثلة المدربة على البيانات الأصلية.

كلمات المفاتيح: تصنيف المشهد الصوتي، استخلاص الميزة، زيادة البيانات، التعلم العميق، Mixup، الاختبارات الإحصائية.

Acknowledgements

First and foremost, praises and thanks to the God, the Almighty, for His showers of blessings throughout our research work.

We would like to express our deepest and most sincere gratitude to our research supervisor, Ms. YKHLEF Hadjer for giving us the opportunity to conduct this research and providing us invaluable guidance throughout our work. Her dynamism, vision and sincerity have deeply inspired and motivated us to complete this work in these trying times of global pandemic. She has taught us the methodology to carry out the research and to present our findings as clearly as possible. It was a great privilege and honor to work and study under her supervision.

We are extending our thanks to Mr. YKHLEF Farid, Professor, University of Blida 1, for his help and guidance during our research work. Special thanks goes to the examiners for the time they spent on carefully reviewing this thesis.

Finally, many thanks goes to all the people who have contributed to the completion of our research work directly or indirectly.

CONTENT

List of Figures

List of Tables

INTRODUCTION

1. Background and problem definition

Environmental sounds are an immensely valuable modality in our day-to-day human lives as they complement the visual information that we acquire through perception [1]. We are constantly surrounded by environmental sounds on which we rely heavily to obtain important information about what is happening around us as a result of these sounds carrying a great number of clues. More broadly, sound contains various information that could indicate the sound sources, surrounding environment, music genres, possible dangers or even the emotions of a speaker; thus, playing a crucial part in our daily communication and interaction with the world. The task of recognizing sounds is not considered difficult for humans since we are able to discern and classify audio signals all the time without conscious effort [2]. However, automating the hearing task by developing robust systems capable of recognizing a wide range of sound events in realistic audio streams i.e. teaching machines to hear, is a complex task and can benefit humans in a broad range of fields. Such systems are known as sound scene recognition systems and have been attracting a continuously growing attention during the past years [3].

In smart cities or in automated surveillance of public spaces, an automated audio recognition system could infer events from audible information using audio sensors that are lower cost, require less networking bandwidth, consume less power, are potentially more robust and less easily obscured by weather, dust or pollution than video sensors [4]. Other relevant applications of such systems include auditory medical information monitoring [5], context-aware services [6] and multimedia content analysis [7].

The general framework for acoustic scene classification is usually categorized into two major steps. First, a 2D time-frequency representation of audio data is obtained via multiple signal processing-based methods. This representation is then used for extracting relevant features. Second, the extracted features are employed to achieve recognition using a classification model. One of the major challenges of acoustic scene classification is the complex and unstructured nature of environmental sounds.

Environmental sounds consist of various non-human and human sounds with a high degree of overlap. Compared with speech, environmental sounds are more diverse and span a wider range of frequencies, they are often less well-defined [3]. For example, there is no standard dictionary for environmental sound events analogous to sub-word dictionary phonemes in speech, and the duration of environmental sounds could vary widely. However, environmental sounds may include strong spectro-temporal signatures. Thus, it is important to consider non-stationary aspects of signal and capture its variation in both time and frequency domains i.e. the choice of proper feature sets during the feature extraction step is crucial [3].

State of the art sound scene classification relies on deep neural network architectures to learn the associations between class labels and audio recordings within a dataset [8]. However, the complexity of these architectures arises another challenge for acoustic scene classification. Deep learning-based architectures are trained using a large number of parameters which makes them more prone to overfitting [9]. The easiest and most widely used approach to reduce overfitting is to employ larger datasets. As an alternative, data augmentation methods are currently used to improve the performance of neural networks by artificially enlarging the dataset using label-preserving transformation [3].

2. Thesis contributions

Although a great number of acoustic scene classification systems have been developed using a plethora of audio processing methods and machine learning paradigms, a review on the design and implementation pipeline of these systems with a comparative study using a recent dataset is required. Moreover, a wide range of data augmentation techniques have been recently invoked in the development of acoustic scene classification systems. However, further research is still needed to properly assess the effects of the latter techniques on the performance of these systems. Motivated by these needs, we have designed and analyzed the behavior of multiple acoustic scene recognition systems. Additionally, we have conducted extensive experimental comparisons among the developed systems, our contribution to the area of acoustic scene classification are 5 folds:

- We provide the proper guidelines to follow for developing and analyzing acoustic scene classification systems, along with a comprehensive description on conducting machine learning experiments.

- We conduct thorough and extensive experiments on acoustic scene recognition systems trained using a large-scale audio dataset.
- We explore various deep neural network architectures and analyze the observed behavior of these architectures when trained on augmented/non-augmented data.
- We analyze the importance of well-tuning the mixup hyperparameter α and its impact on the predictive capacity of the ASC systems.
- We expand our research work by statistically comparing the developed audio classification systems using numerous tests.

3. Thesis organization

This thesis is categorized into two major parts. The first part reports the state-of-the-art notions that are necessary for understanding the ideas developed in this thesis. In **Chapter 1**, we cover some fundamentals behind sound signals. Specifically, we describe the various representations of sound signals and the most frequently used feature extraction techniques in literature. In **Chapter 2**, we review some relevant classification concepts, providing a brief description of machine learning, state-of-the-art deep learning architectures along with the evaluation metrics and statistical tests invoked in this work. The second half of this thesis describes the approach that we have chosen for constructing and evaluating acoustic scene recognition systems. We provide in **Chapter 3**, a detailed description of the experimental setup, hyperparameters tuning and evaluation procedure. In **Chapter 4**, we lay out the obtained results through performance tables and statistics-based plots and discuss these findings while employing robust statistical tests. Finally, we conclude by summarizing the contributions of this thesis, the lines of limitations and future work.

PART 1: PRINCIPLES OF ACOUSTIC SCENE CLASSIFICATION

Throughout this part of the thesis, we present the fundamentals behind the ideas developed in our work. In **Chapter 1**, we introduce the essential notion of signals along with the acoustic features used to represent audio signals. We describe the elements which are required for the development of acoustic scene classification systems and highlight the importance of feature engineering to transform the signal into a better fitting representation for the acoustic scene classification task. Furthermore, we present the different feature extraction techniques that have been extensively invoked in experiments. In **Chapter 2** of our thesis, we provide a comprehensive overview of the notion of machine learning along with the diverse methods and techniques used for the generation of systems specialized in the acoustic scene classification task.

Chapter 1: OVERVIEW OF SOUND SIGNALS

1.1 Introduction

Signal processing is defined as the manipulation of the properties of a specific signal to obtain another signal with more desirable properties. In the later part of the 20th century and along with the introduction of computers and their fast and tremendous growth, a number of researchers resorted to modeling and simulating various concepts of signal processing in their research endeavors using digital computers. These endeavors eventually led to what is known today as digital signal processing (DSP) [10]. During recent years, we have witnessed the increasing availability of audio content via numerous distribution channels both for commercial and non-profit purposes. The resulting wealth of data has inevitably highlighted the need for systems that are capable of analyzing the audio content in order to extract useful knowledge that can be consumed by users or subsequently exploited by other processing systems [11]. With the rapid growth of computing power in terms of speed and memory capacity, researchers aim to develop smart systems that are able to perform various tasks on the basis of the available data [10].

In this Chapter, we introduce a few fundamental concepts behind audio signal processing that will be required to perform our work. We commence by defining audio signals and the multiple characteristics of audio signals in **Section 1.2-1.3**. **Section 1.4** describes the process of acquiring audio data for digital processing. In **Section 1.5** we carry on outlining the various representations of audio signals. Preprocessing and feature extraction techniques are depicted in **Sections 1.6** and **1.7**.

1.2 Audio signals

A signal is defined as the representation of a **quantity** that varies in **time**. One of the concrete examples of a signal is an audio signal [12]. An audio signal is the representation of a sound which is a variation in air pressure. Specifically, it represents variations in **air pressure** over **time**. Producing a sound means creating a vibration, the created vibration results in an air

pressure disturbance, that is to say, the nearest particles to the sound's source will bounce off the particles close to them and so on, generating what is known as a **sound wave**. The vibration can come from a tuning fork, a guitar string, the column of air in an organ pipe, the head (or rim) of a snare drum, steam escaping from a radiator, the reed on a clarinet, the diaphragm of a loudspeaker, the vocal cords, or virtually anything that vibrates in a frequency range that is audible to a human listener [13]. The average human can hear frequencies in the range starting from 20 Hz all the way up to 15,000 Hz, although young children can sometimes hear sounds with frequencies up to 27,000 Hz [2].

1.3 Audio signals characteristics

In order to explain the different elements that characterize an audio signal we will take the sound of a bell strike as an example. Striking a bell generates what is known as a **periodic signal**. After recording the sound generated by striking a bell and plot it, we get the representation shown in Figure 1.1 [12].

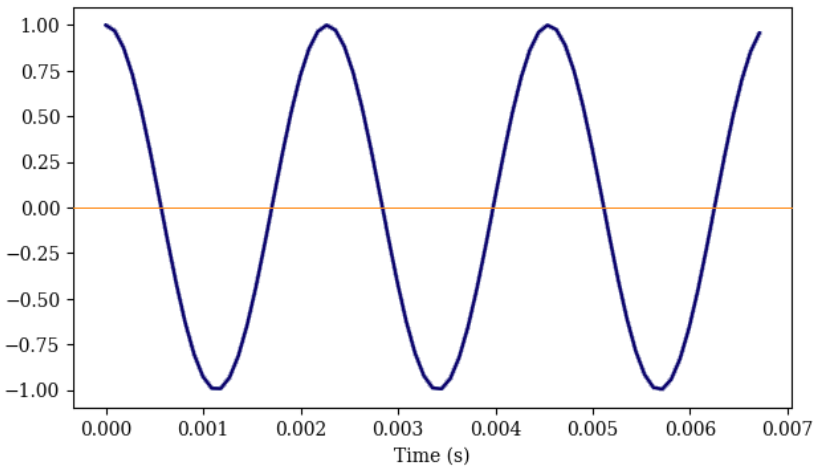


Figure 1.1: Plot of segment from a recording of a bell [3].

The waveform that is shown in the Figure 1.1 is the representation of a periodic vibration, which simply means that there is a pattern that repeats itself over time. A periodic audio signal is characterized by the following components:

CYCLE: a cycle refers to one repetition of the pattern within the sound signal. Figure 1.1 shows three full repetitions of the pattern. The duration of each cycle is called the period.

PERIOD: period is the time required to complete one cycle of vibration. For example, Figure 1.1 shows that 3 cycles are completed in 6.9ms, the period is 6.9$ms/3ms$, or 2.3ms. For speech applications, the most commonly used unit of measurement for period is the millisecond (ms) where:

$$1ms = {}^{1}/_{1{,}000s} = 0.001s. \tag{1.1}$$

FREQUENCY: frequency is defined as the number of cycles completed in one second. The unit of measurement for frequency is **Hertz** (Hz), and it is fully synonymous to the older and more straightforward term **cycles per second** (cps). Conceptually, frequency is simply the rate of vibration. Consequently, frequency is the single most important concept in hearing science. The formula for frequency is:

$$f = {}^{1}/_{t}, \tag{1.2}$$

where: f represents the frequency in Hertz and t represents the period in seconds. So, for our bell striking example, for a period of 0.0023s:

$$f = {}^{1}/_{t} = {}^{1}/_{0.0023} = 434.7\ Hz. \tag{1.3}$$

The frequency of the sound generated by striking a bell is about 435 Hz [2].

AMPLITUDE: the term amplitude refers to the magnitude of displacement of the particles in the air generated by the vibration. It is the magnitude of the air pressure disturbance [2]. On Figure 1.1 the amplitude is presented by the height of the waves which defines whether the volume of the sound is high or low [14].

1.4 Audio data acquisition

As we have mentioned earlier, sound is the result of a vibration that propagates in the form of waves through a medium such as air or water. These vibrations can be recorded by means of an electroacoustic transducer such as a microphone. A microphone is a device that measures the variations in air pressure caused by the vibrations, and generates an electrical signal that represents sound, it is called a transducer because they transduce, or convert, signals from one

form to another [12]. The output generated by a microphone is represented under the form of an electric signal $x(t)$ where t represents the time. In order to process the generated analog signals $x(t)$ with digital processors, the analog signal must be converted to digital. The process whereby analog signals are converted to digital signals involve a **sampling** and **quantization** procedure. It is then stored on a computer before further analysis [14].

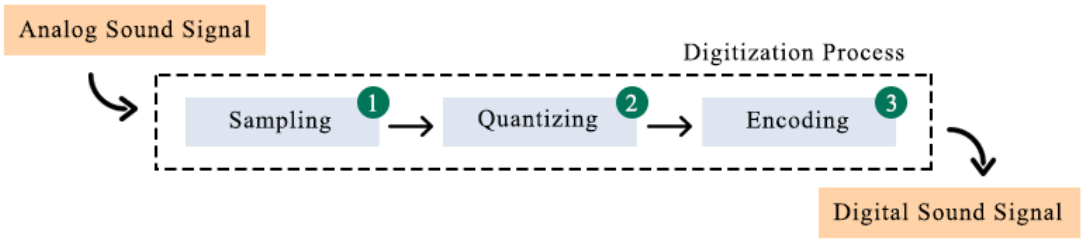


Figure 1.2: Digitization process of sound signals [3].

1.5 Audio signal sampling

Sampling is a process that implies taking snapshots of an analog signal at a specific **sampling rate** or **sampling frequency** [15][16][17]. The sample rate is the number of samples taken per second [18]. Although any sampling frequency above 40 kHz would be adequate to capture the full range of audible frequencies, a widely used sampling rate is 44,100 Hz (or 44.1 kHz), which arose from the historical need to synchronize audio with video data [19]. If the sampling is performed at higher rates, it generates more samples and hence it creates a much larger demand for memory to store the samples [10].

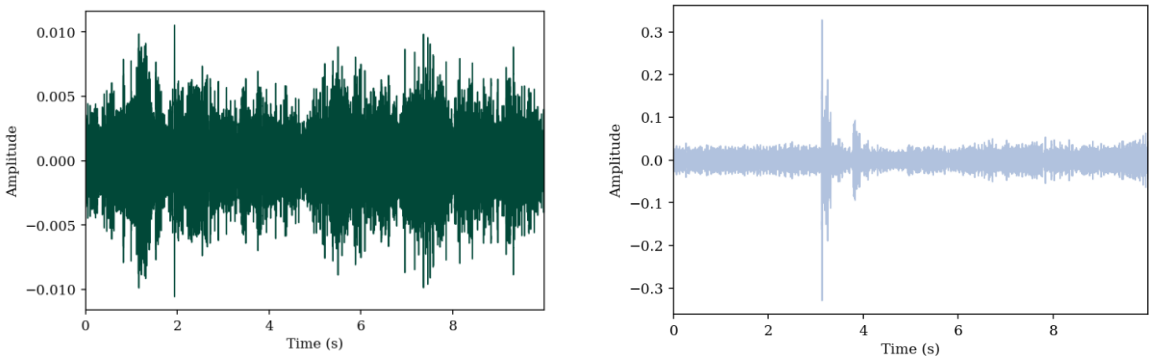
1.6 Audio signal quantization

In order for a signal to be suitable for treatment by numerical circuitry, it must first be represented in a numerical format, or quantized. That is, a continuous range of values is replaced by a limited set of values separated by discrete steps. Usually the number of steps is chosen to be a power of two, for the reason that it yields the most economical representation in binary digital electronics. Naturally, the quality of the approximation depends on the number of steps used to approximate the original signal [20].

1.7 Audio signal representation

The conversion from analog signal to digital signal generates what is known as the representation of sound in the **time domain** as shown in Figures **1.2-1.3**. However, the time

domain representation of a sound signal, or waveform, is not easy to interpret [14]; these representations alone are insufficient to provide comprehensive information about sound signals [21]. Furthermore, it is nearly impossible, from a waveform, to identify or even localize sound events, and to discriminate between sound [14]. As shown in Figures **1.2-1.3**; without any knowledge about the recordings, the distinction between audio from an airport and audio from a bus is beyond the bounds of possibility. For that purpose, **frequency-domain** representations and time-frequency domain representations have been in use for years providing representations of the sound signals that are more in line with the human perception of sounds [22][14]. The sound signals are usually converted to the frequency-domain prior to any analysis. The frequency-domain representation of a signal can be obtained using the Fourier Transform [14].



(a) Airport in Barcelona.

(b) Bus in Prague.

Figure 1.3: Time-domain representation.

1.8 The Fourier transform

A sinusoid is a mathematical function that traces out the simplest repetitive motion in nature [2]. Any sound can be created by adding together an infinite number of these sinusoids [2]. This is the essence of **Fourier synthesis** [2]. To put this into context, any function can be generated from the summation of an infinite number of sinusoids of different frequencies and amplitudes [2]. The opposite of Fourier synthesis, **Fourier analysis** consists of decomposing a function into its component sinusoids. **The Fourier transform** is a mathematical way to go between the functional representation of a signal and its Fourier representation [2]. The Fourier representation of a signal shows the spectral composition of the signal. It contains a list of sinusoidal functions, identified by frequency, and each sinusoid has an associated amplitude

and phase. The phase of a signal is the start location of the sinusoid relative to some specific zero [2].

1.9 The Short-Time Fourier transform

The Fourier transform provides information about how much of each frequency is present in a signal. If the spectral content of the signal does not change much over time -stationarity of the signal-, then this works quite well [2]. However, if the signal changes over time, the Fourier transform will not be able to distinguish between the different changes in the spectral content. The short-time Fourier transform (STFT) is an attempt to fix the lack of time resolution in the classic Fourier transform [2].

The input data is broken into many small sequential pieces, called **frames**, and the Fourier transform is applied to each of these frames in succession. What is produced is a time-dependent representation, showing the changes in the spectrum as the signal progresses [2].

As a consequence, there is often a discontinuity, or break in the signal, at the frame boundaries. This introduces spectral components into the transform that are not present in the original signal i.e. spectral leakage. The solution to this problem is to apply a **windowing function** to the frame, which gently scales the amplitude of the signal to zero at each end, reducing the discontinuity at frame boundaries [2]. When these windowing functions are applied to a signal, it is clear that some information near the frame boundaries is lost. For this reason, a further improvement to the STFT is to **overlap the frames** [2]. When each part of the signal is analyzed in more than one frame, information that is lost at a frame boundary is picked up between the boundaries of the next frame [2].

The STFT allows for defining the linear-frequency spectrogram which is a 2D representation of a sound where energy in each frequency band is given as a function of time [14]. The spectrogram is then the matrix where each column is the modulus of the DFT of a sound signal frame [23][24].

1.10 Audio pre-processing

If needed, the audio data is pre-processed [23]. The role of this step is to enhance certain characteristics of the signal for further analysis. This is achieved by reducing the effects of noise or by emphasizing the target sounds in the signal [14].

Knowledge about the recording conditions and characteristics of target sounds can be utilized in the pre-processing stage to enhance the signal [14]. In the case where the audio data is captured in non-uniform recording settings, down-mixing the audio signal into a fixed number of channels along with re-sampling it into fixed sampling frequency will result in converting the input data into a uniform format for further analysis [14]. After the pre-processing phase, the audio data is now appropriate to be used in the feature extraction phase.

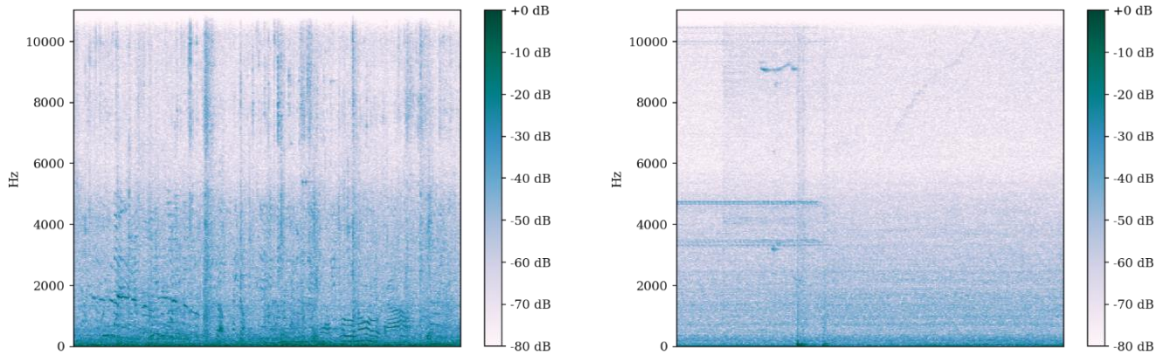
1.11 Feature extraction

Most real-world data, and in particular sound data, is very large and contains much redundancy, and important features are lost in the cacophony of unreduced data [2]. The data reduction stage is often called **feature extraction**, and consists of discovering a few important facts about each data item, or case [25]. The features that are extracted from each case are the same, so that they can be compared. Feature extraction is rarely skipped as a step, unless the data in its original form is already in features, such as temperature read from a thermometer over time [2]. These features can be **physical**, based on measurable characteristics, or **perceptual**, based on characteristics reported to be perceived by humans [2][26].

1.11.1 Physical features

Physical features are low-level signal parameters that capture particular aspects of the temporal or spectral properties of the signal [27][28]. Although some of the features are perceptually motivated, we classify them as physical features since they are computed directly from the audio waveform amplitudes or the corresponding short-time spectral values [29].

SPECTROGRAMS: A **sound spectrum** is a representation of a sound – usually a short sample of a sound – in terms of the amount of vibration at each individual frequency [30]. It is usually presented as a graph of either power or pressure as a function of frequency [30][31][32]. A spectrogram is built from a sequence of spectra by stacking them together in time and by the amplitude axis into a 'contour map' usually drawn in a grey scale. The final graph has time along the horizontal axis, frequency along the vertical axis, and the amplitude of the signal at any given time and frequency is shown as a grey level. Conventionally, black is used to signal the most energy, while white is used to signal the least but the contour map can be of other colors [33]. The figure above shows the spectrogram features of the airport in Barcelona and the bus in Prague audio recordings.



(a) Airport in Barcelona.

(b) Bus in Prague.

Figure 1.4: Spectrogram of a recording from an airport and a bus.

MEL-SCALED SPECTROGRAMS: This representation is derived from the classical spectrogram by weighted averaging of the absolute values squared of the STFT and can undoubtedly be referred to as the most important feature set used in speech and audio processing, together with Mel Frequency Cepstral Coefficients (MFCC) [34]. Mel scale corresponds to an approximation of the psychological sensation of heights of a pure sound. Several analytical expressions exist; however, a common relation between the Mel scale $mel(f)$ and the Hertz scale f was given by **Fant** [15]. Mel Spectrogram, is, rather surprisingly, a Spectrogram with the Mel Scale as its *y* axis. It is the result of a non-linear transformation of the frequency scale. It partitions the Hz scale into bins, and transforms each bin into a corresponding bin in the Mel Scale, using overlapping triangular filters [35].

A classical approximation is to define the frequency-to-Mel transform function for a frequency f as [36]:

$$m = 2595 log_{10}\left(1 + \frac{f}{700}\right). \tag{1.4}$$

The inverse transform can be readily derived as:

$$f = 700\left(10^{\frac{m}{2595}} - 1\right). \tag{1.5}$$

LOG-MEL SPECTROGRAMS: The logarithmic spectrum, on the other hand, is a much more accessible representation. It is not only more visual, but importantly, the logarithm approximates roughly the sensitivity of the ear, such that logarithmic spectra can be used to

assess auditory importance of spectral features [36]. The logarithmic spectrum visualizes spectral content such that the magnitude of values is approximately uniform throughout the spectrum. The only exception is zeros and other very small values in the magnitude spectrum, which give negative infinities or arbitrarily large negative values in the log spectrum. To cope with this problematic, we can use for example an energy bias similar to the **mu-law** rule or integrate energies over frequencies [36]. Specifically, instead of

$$y = \log(|x|^2), \tag{1.6}$$

we can use:

$$y = \log(|x|^2 + e), \tag{1.7}$$

where e is a small positive number. The output y will then never go lower than a threshold [36]:

$$y \geq \log(e). \tag{1.8}$$

DELTAS AND DELTA-DELTAS: A common method for extracting information about such transitions is to determine the **first difference** of signal features, known as the **delta of a feature**. Specifically, for a feature f_k, at time-instant k, the corresponding delta is defined as:

$$\Delta_k = f_k - f_{k-1}. \tag{1.9}$$

The **second difference**, known as the **delta-delta** of a feature, is correspondingly:

$$\Delta\Delta_k = \Delta_k - \Delta_{k-1}. \tag{1.10}$$

Common short-hand notations for the deltas and delta-deltas are, respectively, Δ and ΔΔ-features. Features in a recognition engine are then typically appended by their Δ and ΔΔ -features to triple the number of features with a very small computational overhead [36].

MEL-FREQUENCY CEPSTRAL COEFFICIENTS (MFCC): MFCCs are perceptually motivated features that provide a compact representation of the short-time spectrum envelope. MFCCs have long been applied in speech recognition and, much more recently, to music [28]. To compute the MFCC, the windowed audio data frame is transformed by a Discrete Fourier Transform (DFT). Next, a Mel-scale filter bank is applied in the frequency domain and the power within each sub-band is computed by squaring and summing the spectral magnitudes within bands. The Mel-frequency scale, a perceptual scale like the critical band scale, is linear below 1 kHz and logarithmic above this frequency. Finally, the logarithm of the band wise power values is taken and decorrelated by applying a Discrete Cosine Transform (DCT) to obtain the cepstral coefficients.

The log transformation serves to deconvolve multiplicative components of the spectrum such as the source and filter transfer function. The decorrelation results in most of the energy being concentrated in a few cepstral coefficients. For instance, in 16 kHz sampled speech, 13 low-order MFCCs are adequate to represent the spectral envelope across phonemes [28][23].

1.11.2 Perceptual features

The human recognition of sound is based on the perceptual attributes of the sound. When a good source model is not available, perceptual features provide an alternative basis for segmentation and classification. The psychological sensations evoked by a sound can be broadly categorized as loudness, pitch and timbre [29][27][37].

LOUDNESS: Loudness is a sensation of signal strength; it is a measure of sound wave intensity [38]. As would be expected, it is correlated with the sound intensity, but it is also dependent on the duration and the spectrum of the sound [28].

PITCH: Although pitch is a perceptual attribute, it is closely correlated with the physical attribute of fundamental frequency (F0) [28]. Subjective pitch changes are related to the logarithm of F0 so that a constant pitch change in music refers to a constant ratio of fundamental frequencies [2][29]. Pitch is the frequency of the fundamental component in the sound, that is, the frequency with which the waveform repeats itself [38].

TIMBRE: Pitch and loudness of sound are well-definable perceptual quantities that occur very often when humans discuss sounds, but some perceptible characteristics of a sound are more difficult to quantify. These characteristics are grouped together, and are called "timbre", which has been defined as that quality of sound which allows the distinction of different instruments or voices sounding the same pitch. Many spectral characteristics, as discussed above, can be used as classification features, and many of these correspond to the timbre of the sound [2][13][10][19][16].

1.12 Conclusion

The proper design of the feature set while considering the intended audio categories is crucial to the classification task. Features are chosen based on the knowledge of the salient signal characteristics either in terms of production or perception. It is also possible to select features from a large set of possible features based on exhaustive comparative evaluations in classification experiments. Once the features are extracted, standard machine learning techniques are used to design the classifier. Throughout this chapter, we have reviewed important sound characteristics and the widely used techniques to represent and amplify these characteristics for analysis. We have presented several types of sound features which are key to understand the experiments conducted in our work. In the next chapter, we will present the fundamental notions of machine learning, the definition of classification task in machine learning, as well as an overview of the machine learning pipeline for acoustic scene classification.

Chapter 2: MACHINE LEARNING PIPELINE FOR ACOUSTIC SCENE CLASSIFICATION

2.1 Introduction

In the previous chapter, we have defined audio signals and listed their most substantial characteristics. We have also discussed the fundamental concepts and notions behind sound acquisition, including numerous representations and preprocessing techniques required to prepare the audio signal for the machine learning task. Machine learning (ML) is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence [39][40]. Our main focus in this chapter revolves around the classification task in machine learning with use of the supervised learning method. The rest of this chapter is structured as follows: in **Section 2.2**, we introduce the concept of machine learning and the specific task of classification. In the following section, we present another branch of artificial intelligence: **Deep learning**. In **Section 2.4**, we define the notion of data augmentation, which is a common strategy for handling scarce data situations by synthesizing new data from existing training data, with the objective of improving the performance of the downstream model. This strategy has been a key factor in the performance improvement of various neural network models, mainly in the domains of computer vision and speech recognition [41].

2.2 Generalities on machine learning

Machine learning can be defined as the study of the construction of computer programs that automatically improve and/or adapt their performance through experience; it can be thought of as **"programming by example"** [39]. The main goal of machine learning is to develop learning algorithms that are able to learn automatically without human intervention or assistance. Rather than programming the computer to solve the task directly, in machine learning, we are looking for methods that the computer will create its own program based on examples that we provide [39]. Learning methods fall into three major categories [14], depending on the nature of the

learning “signal” or “feedback” available to a learning system. These categories are: **Supervised learning** , **unsupervised learning** and **semi-supervised learning** [40]**.**

Supervised methods consist of learning a classification model from a set of **labeled training data** that it takes as an input. The training set consists of training samples. Each sample in the training set is a pair consisting of an input object (usually a vector) and a desired output value or label. A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used to map new examples [42]. Unsupervised methods receive **unlabeled input** training data. However, the issue with unlabeled data is that we do not have a correct result to match, which means there is no error or reward signals to evaluate a potential solution. Therefore, the learning algorithm will try to discover persistent patterns and find hidden structures to link results that are close to each other in order to group them into classes using clustering algorithms [42][43]. Semi-supervised learning is more recent when compared with the supervised and unsupervised learning [44], as the name suggests; semi-supervised learning is somewhere between unsupervised and supervised learning [45]. The dataset provided to the semi-supervised learning model is partially labeled [46], and is also provided with some supervision information [47]. The main objective of semi-supervised learning is to overcome the drawbacks of both supervised and unsupervised learning [44].

Another categorization of machine learning tasks arises when one considers the desired **output** of a machine learned system, this categorization gathers two broad tasks in machine learning which are: **Classification** and **Regression** [40]. The final output in the classification task is a label, whereas the final output in the regression task is a quantity; put in other words, the key distinction between classification and regression is that the former has discrete outputs, whereas the latter has continuous outputs [48][49].

2.2.1 Overview of classification task

The term **data science** generally refers to the extraction of knowledge from data [50][51]. This involves a wide range of techniques and theories drawn from many research fields within mathematics, statistics and information technology, including statistical learning, pattern recognition, probability models, high performance computing, signal processing and also machine learning. Classification problems and methods have been considered a key part within the machine learning field, with a huge amount of applications published in the last few years

[52][53]. The concept of classification in ML has been traditionally treated in a broad sense, very often including supervised, unsupervised and semi-supervised learning problems [54].

In the case of supervised learning, each **data input** is preassigned a **class label**. The main task of supervised algorithms is to learn a **model** that ideally produces the same labeling for the provided data and generalizes well on unseen data (i.e., prediction). The general aim of supervised classification algorithms is to separate the classes of the problem -with a margin as wide as possible- using only training data. If the output variable has two possible values, the problem is referred to as **binary classification**. On the other hand, if there are more than two classes, the problem is named **multiclass** or **multinomial classification**. A classification problem can be formally defined as the task of estimating the label y of a K-dimensional input vector x, where $x \in X \subseteq R^K$ and $y \in Y = \{C_1, C_2, \dots, C_Q\}$, where y is the true class label preassigned to the input *x*.

This task is accomplished by using a classification rule or function $g: X \to Y$ able to predict the label of unseen samples. In the supervised setting, we are given a training set of N points, represented by $D = \{(x_i, y_i); i = 1, \dots, N\}$ [54]. The steps involved in the process of the classification task are described in the following subsections.

2.2.2 Classification task pipeline

A. Data acquisition/ data gathering

This step involves understanding the problem at hand and identifying a priori knowledge to create the dataset [54]. When creating an intelligent system which will later be applied in real-life uses, we require a set of realistic data, the data can either be naturally recorded, or artificially created with sufficient realism, and will be used in the development step, as well as the testing step; thus, ensuring better performance of our system [55]. There are different methods of acquiring a data collection such as the following:

CREATING A NEW DATASET FROM SCRATCH: This can be achieved in many ways, such as recording real-world sounds as they are present in real life. The advantage here is the complete control over quality and/or the content of the data [55]. In order for the data to cover a maximum of acoustic variability, the recording phase has to be done several times by changing different factors such as location and time. This may require traveling to

different geographical locations which can subsequently be time consuming and financially draining [55].

AVAILABLE DATASETS: Since the creation of a dataset from scratch can be difficult, complicated, time consuming and financially overwhelming, another option is to search the web for potentially suitable dataset. There are numerous available free datasets for every machine learning task such as : Detection and Classification of Acoustic Scenes and Events (DCASE) datasets which are balanced and dedicated to different audio centered machine learning tasks [56], NYU Urban Sound [57], and Freefield [58]. Other audio data sources are the commercially available audio samples from BBC, Stockmusic, and others. [58]. A list of datasets suitable for research involving the audio-based context recognition and acoustic scene classification is presented in Table 2.1.

Table 2.1: Available audio datasets for scene classification.

Dataset Name	Provider	Classes	Files	Ref
TAU Urban Acoustic Scenes 2019 Mobile Development dataset	TUT/TAU	10	16560	[55]
TAU Urban Acoustic Scenes 2019 Development dataset	TUT/TAU	10	14400	[59]
Scene	IEEE AASP Challenge 2013	10	100	[60]
Rouen audio scene dataset	LITIS	19	3026	[56]
CASA2010	TUT	13	160	[61]
CASR	TUT	27	225	[6]
TUT Urban Acoustic Scenes 2018 Development dataset	TUT	10	8640	[55]
UEA	Noise DB/Series 2	12	35	[62]
AucoDefro07	AucoDefro07	4	16	[63]

DATA SIMULATION: Another method for creating datasets for development is **data simulation**. This involves mixing tokens of isolated sounds, with the desired complexity and overlap and background noise, in order to create examples of complex sound. The

advantage with this technique is the total control over the levels of several sounds in the background, which subsequently makes creating different sound samples with the same combination of sounds possible. However, it remains very limited due to its tendency to easily diverge from realism [64].

B. Data preparation/ preprocessing

Data preprocessing is considered as one of the most important phases in ML [65]. Preprocessing algorithms are usually used for: **data cleaning**, **outlier detection**, **data imputation** and **transformation of features** [54]. There is a hierarchy of problems that are often encountered in data preparation and pre-processing such as outliers, missing values or irrelevant and redundant data. Depending on the type of problem at hand, numerous approaches exist for addressing these hurdles; we can cite feature selection techniques [66][67] and anomaly detection methods [68][69].

C. Algorithm/ model selection

The choice of the learning algorithm is a critical step in the machine learning process. The classifier evaluation is most often based on **predictive accuracy** which is "*the percentage of correct predictions divided by the total number of predictions*", or on the error rate which is "*the percentage of incorrect predictions by the total number of predictions*" [70]. There are at least three techniques which are used to estimate a classifier accuracy/error rate, one technique is to **split the training set by using two-thirds for training set and the other third for testing set**. However, this approach requires a large amount of data in order to obtain a reliable estimate of the performance [66]. An alternative strategy consists of invoking a resampling technique, known as **cross-validation.**

The training set is divided into mutually exclusive and equal-sized subsets and for each subset the classifier is trained on the union of all the other subsets, while the remaining subset is used for testing the learned model, i.e. computing its performance. The average of the error rate of each subset is therefore an estimate of the error rate of the classifier [71]. **Leave-one-out** validation is a special case of cross validation. All test subsets consist of a single instance. This type of validation is a lot more computationally expensive, but useful when the most accurate estimate of a classifier error rate is required [72]. If the error rate evaluation is unsatisfactory, a variety of factors must be examined: perhaps relevant features for the problem are not being used, a larger training set is needed, the dimensionality of the problem is too high

or the selected algorithm is inappropriate [69]. There are also many issues of concern to the would-be classifier. Below is a list of a few of these concerns:

ACCURACY: There is the reliability of the rule, usually represented by the proportion of correct classifications. In some cases, it may be that some errors are more critical than others, and it may be important to control the error rate for some key classes.

SPEED: In Some cases, the speed of the classifier is a major issue. A classifier that is 90% accurate may be preferred over a classifier that is 95% accurate if it is 100 times faster in testing (such differences in time-scales are not uncommon in neural networks for example). Such considerations would be important for the automatic reading postal codes, or automatic fault detection items on a production line.

TRAINING TIME: In a rapidly changing environment, it may be necessary to learn a classification rule quickly, or make adjustments to an existing rule in real time [66].

D. Learning/ training

The classification of new samples becomes possible when the classifier decision boundaries are set. These boundaries have to reflect the considered classes. Relevant decision boundaries are achieved by **learning**, which is the process when a set of samples is used to tune the classifier to the desired task. How the classifier is tuned depends on which algorithm the classifier uses. The learning algorithm has to be given a training set of samples that are used to learn/to construct decision boundaries [73].

E. Classification/ prediction

Once the classifier has been trained, it can be used to **classify** new samples. However, a perfect classification is seldom possible. Therefore, numerous different classification algorithms are used with varying complexity and performance. The main problem for a classifier is to cope with feature value variation for samples belonging to specific classes. This variation may be large due to the complexity of the classification task. To maximize classification accuracy, decision boundaries should be chosen based on combinations of the feature values [73].

F. Evaluation of a classifier performance

The evaluation metric is a crucial element in achieving the optimal classifier during the training process. Thus, a selection of a suitable evaluation metric is an important key for discriminating and obtaining the optimal classifier [70]. For classification problems, the evaluation of the optimal solution during the training stage can be defined based on **confusion matrix** as shown in Table 2.2. The row of the table represents the predicted class, while the column represents the true class. From this confusion matrix, *tp* and *tn* denote the number of positive and negative instances that are correctly classified. Meanwhile, *fp* and *fn* denote the number of misclassified negative and positive instances, respectively [70].

Table 2.2: Confusion Matrix Representation.

	Actual positive class	**Actual negative class**
Predicted positive class	True positive *(tp)*	False negative *(fn)*
Predicted negative class	False positive *(fp)*	True negative *(tn)*

From Table 2.2, numerous commonly used metrics can be generated to evaluate the performance of classifier with different focuses of evaluations as shown in Table 2.3:

Table 2.3: Metrics for Classification Evaluation.

Metrics	**Formula**	**Evaluation Focus**
Accuracy (acc)	$\frac{tp + tn}{tp + fp + tn + fn}$	The accuracy metric measures the ratio of correct predictions over the total number of instances evaluated.
Error Rate (err)	$\frac{fp + fn}{tp + fp + tn + fn}$	Misclassification error measures the ratio of incorrect predictions over the total number of instances evaluated.
Sensitivity (sn)	$\frac{tp}{tp + fn}$	This metric is used to measure the fraction of positive patterns that are correctly classified
Specificity (sp)	$\frac{tn}{tn + fp}$	This metric is used to measure the fraction of negative patterns that are correctly classified

Precision (p)	$\frac{tp}{tp + fp}$	Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class.
Recall (r)	$\frac{tp}{tp + tn}$	Recall is used to measure the fraction of positive patterns that are correctly classified
F-measure (FM)	$2 * \frac{p * r}{p + r}$	This metric represents the harmonic mean between recall and precision values
Averaged Accuracy	$\frac{\sum_{i=1}^{l} \frac{tpi + tni}{tpi + fni + fpi + tni}}{l}$	The average effectiveness of all classes
Averaged Error Rate	$\frac{\sum_{i=1}^{l} \frac{fpi + fni}{tpi + fni + fpi + tni}}{l}$	The average error rate of all classes
Averaged Precision	$\frac{\sum_{i=1}^{l} \frac{tpi}{tpi + fpi}}{l}$	The average of per-class precision
Averaged Recall	$\frac{\sum_{i=1}^{l} \frac{tpi}{tpi + fni}}{l}$	The average of per-class recall

Note: for each class c_i of data where $c \in C = \{c_1, c_2, \dots, c_l\}$; tp_i – true positive for c_i; fp_i – false positive for c_i; fn_i – false negative for c_i; tn_i – true negative for c_i.

2.2.3 Statistical Tests

Comparative classification studies generally focus on a single performance metric such as the accuracy or the error rate. Single point estimates of such measures were often compared directly in order to identify which classifier produces the most accurate classifiers when trained on samples from other domains. However recently, comparisons of point estimates are less frequently used, and null hypothesis significance tests are now gaining increasing popularity in machine learning [74]. A null hypothesis test takes the observed performance measure as input and assesses whether the difference between the classifiers is significant or not [75].

In this regard, Dietterich [76], Demšar [74], García et al. [77], and Japkowicz et al. [78] introduced several statistical tests such as McNemar, Friedman, Nemenyi, Bonferroni-Dunn,

Wilcoxon, and ANOVA for performance comparison. In the following subsections, we briefly review the statistical tests we invoked in our experiments.

A. Friedman test

The Friedman test is useful for comparing several algorithms over multiple domains. It first ranks the techniques for each dataset separately according to the generalization accuracy in descending order. The best performing technique gets the rank 1, the second best gets rank 2 and so on. In case of ties, average ranks are assigned. Let r_i^j be the rank attributed to the j^{th} algorithm on the i^{th} dataset; and $R_j = \frac{1}{N}\sum_{i=1}^{N} r_i^j$ denote the *average rank* of algorithm $j \in \{1, \ldots, t\}$ over *N* datasets. Under the null hypothesis, it is assumed that all techniques are equivalent; hence, their average ranks should be equal.

$$X_F^2 = \frac{12N}{t(t+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{t(t+1)^2}{4}\right]. \quad (2.1)$$

The statistic follows chi-squared distribution with $t - 1$ degrees of freedom for sufficiently large *N* and *t* (usually $N > 10$ and $t > 5$). In their study, Iman and Davenport reported that X_F^2 is conservative and derived a new statistic:

$$F_F = \frac{(N-1)X_F^2}{(N-1)-X_F^2}. \quad (2.2)$$

This test provides only an assessment on whether the observed differences in the performances are statistically significant. In order to have a zoomed-in view of what these differences correspond to precisely i.e. identify pairs of techniques with significant different performances, a post hoc test is usually performed when Friedman test rejects the null hypothesis. Nemenyi, Bonfferoni-Dunn, and Holm are examples of post hoc tests that are widely used in conjunction with Friedman test.

B. Nemenyi Test

This test is invoked when all techniques are compared with each other. The performance of two methods is significantly different if their corresponding average ranks differ by at least the critical difference

$$CD = q_\alpha \sqrt{\frac{t(t+1)}{6N}}, \tag{2.3}$$

where the critical value q_α is defined based on the Studentized range statistic divided by $\sqrt{2}$.

C. Bonferroni-Dunn Test

In general, the Bonferroni-Dunn test is undesirably conservative and has little power; nevertheless, this test is useful when the main interest is the comparison of all techniques against a control algorithm. In this specific case, Bonferroni-Dunn test is more powerful than Nemenyi test because this latter adjusts the critical value for making $t(t-1)$ comparisons, whereas when comparing with a control method, only $t-1$ comparisons are made. This test is basically defined similarly to Nemenyi test except that we estimate the critical value for $\alpha/(t-1)$ significance level.

D. Wilcoxon signed-ranks test

Wilcoxon signed-ranks test is a non-parametric alternative to the paired *t-test* and is considered the best strategy to compare two algorithms over multiple domains. The formulation of this test is the following. We designate by d_i the differences between the performance scores of two techniques on *N* datasets, $i \in \{1, \dots, N\}$. We first rank these differences according to their absolute values; in case of ties average ranks are attributed. Then, we compute the sum of ranks for the positive and the negative differences, which are denoted as R^+ and R^-, respectively. Their formal definitions are given by:

$$R^+ = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \quad , \quad R^- = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i). \tag{2.4}$$

Notice that the ranks of $d_i = 0$ are split evenly between R^+ and R^-. Finally, the statistics T_w is computed as $T_w = \min(R^+, R^-)$. For small *N*, the critical values for T_w can be found in any textbook on general statistics [78], whereas for larger *N*, the statistics

$$z = \frac{T - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}}, \tag{2.5}$$

follows the normal distribution with 1 mean and 0 variance. For instance, the hypothesis which states that two approaches perform equally is rejected if $z \leq -1.96$ at a 5% significance level.

2.3 Outlining of deep learning

Deep learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks [79][80]. Deep learning approaches aim to mimic the way human brains work. Deep learning has already proved useful in many software disciplines, including computer vision [81], speech and audio processing [81], natural language processing [82], robotics [83], bioinformatics [84], chemistry [85], video games [86], search engines [87], online advertising [88] and even finance [89]. It has drawn heavily on our knowledge of the human brain, statistics and applied mathematics as it developed over the past several decades. In recent years, deep learning has seen tremendous growth in its popularity and usefulness, largely as the result of more powerful computers, larger datasets and techniques to train deeper networks [90].

2.3.1 Multi-layer perceptrons

Artificial neurons, which try to mimic the behavior of the human brain, are the principle building blocks of Artificial Neural Networks (ANNs). The basic computational element is called a **node** (or unit) which receives inputs from external sources and possess internal parameters which produce outputs. This unit is the **perceptron** [91].

A perceptron can be formally described as a function f_j of the input $x = (x_1, x_1, \dots, x_N)$ weighted by a vector of connection weights $w_j = (w_{j,1}, \dots, w_{j,N})$, completed by a neuron bias b_j, and associated to an activation function φ, namely:

$$y_j = f_j(x) = \varphi\left(\left(w_j, x\right) + b_j\right). \tag{2.6}$$

Figure 2.1 shows a schematic representation of an artificial neuron where:

$$y_j = \varphi\left[\left(\sum_{i=1}^{N} w_{ij} x_i\right) + b\right]. \tag{2.7}$$

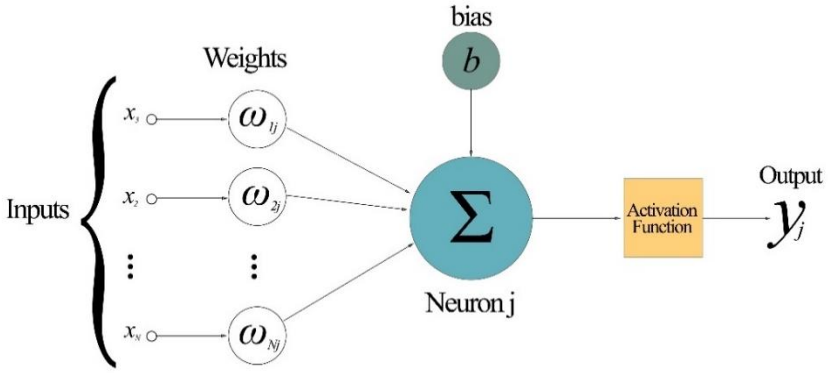


Figure 2.1: Schematic representation of an artificial neuron [91].

Several activation functions can be considered in the classification task. Table 2.4 shows the most commonly used activation functions [92]:

Table 2.4: Activation functions for classification.

Activation Function	**Formula**
Identity	$\varphi(x) = x.$
Sigmoid (logistic)	$\varphi(x) = \frac{1}{1 + exp^{(-x)}}.$
Hyperbolic tangent (tanh)	$\varphi(x) = \frac{exp^{(x)} - exp^{(-x)}}{exp^{(x)} + exp^{(-x)}} = \frac{exp^{(2x)} - 1}{exp^{(2x)} + 1}.$
The Rectified Linear Unit (ReLU)	$\varphi(x) = \max(0, x).$
Softmax	$\varphi(x)_i = \frac{\exp\,(x_i)}{\sum_j \exp\,(x_j)}.$

An Artificial Neural Network or a Multi-layer perceptron (MLP) is a connectionist model which consists of multiple perceptron cells arranged in the form of a directed graph [80]. The basic structure of an ANN can be modelled as shown in Figure 2.2. The input is usually loaded in the form of a multidimensional vector to the **input layer** of which will distribute it to the **hidden layers**. The hidden layers will then make decisions from the previous layer and weigh up how a stochastic change within itself detriments or improves the final output. This is referred to as the process of learning.

A standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations [93]. Having **multiple** hidden layers stacked upon each other creates a deep learning architecture [94]. Within the hidden

layers of an artificial neural network, the output of a perceptron acts as the input to an activation, and then the output of this activation function goes into the next perceptron cell it is connected to. Being descendants of the perceptron, using a neural network, we can obtain non-linear separation boundaries [80].

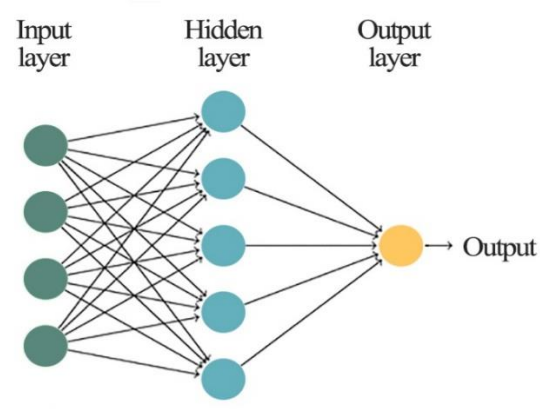


Figure 2.2: A basic neural network with one hidden layer [91].

Multilayer perceptrons have a basic architecture since each neuron of a layer is linked to all the units of the next layer but has no link with the neurons of the same layer. The parameters of the architecture are the number of hidden layers and of neurons in each layer. The activation functions are also to choose by the user. For the **output layer**, the activation function is generally different from the one used on the hidden layers. In the case of binary classification, the output gives a prediction of $P(Y = 1|X)$. Since this value is in [0, 1], the sigmoid activation function is generally considered. For multi-class classification, the output layer contains one neuron per class, giving a prediction of $P(Y = i|X)$. The sum of all these values has to be equal to 1. The multidimensional function Softmax is generally used in multi-class classification [92].

2.3.2 Deep neural network architectures

In recent years, deep artificial neural networks have won numerous contests in pattern recognition and machine learning [95][96][93]. The deep neural network approach have gained significant interest and have fostered major progress in audio processing [97][98]. They have been used on auditory data and evaluated on various audio classification tasks such as acoustic scene classification [99], audio event detection [100] and speech recognition [101]. Traditional neural networks are composed of one input layer, one hidden layer, and one output layer; these

are shallow networks. More than one hidden layer qualifies a network as a deep learning network [73]. In this section of the thesis, we introduce some of the most commonly used deep artificial network architectures used in acoustic scene classification task.

A. Feedforward neural networks

The first type of neural network that has been developed historically is a regular Feedforward Neural Network (FNN). FNNs are used in several ASC algorithms [102]; for instance, they have been used to concatenate features of the audio signal [103], and have been combined with multiple classifiers to model acoustic scenes [104]. This network does not take into account any particular structure that the input data might have. Nevertheless, it is already a very powerful machine learning tool [105]. A FNN is formed by one input layer, one or more hidden layers and one output layer as shown in Figure 2.2. Each layer of the network -except the output layer- is connected to the following layer [105], which grants the FNN its most important feature: *"fully-connected network"* [106]. The learning process in FNNs consists of a forward pass and a backward pass. In the forward pass outputs are calculated and compared with desired outputs. The error is calculated on the basis of the produced output and the real output. In the backward pass i.e. backpropagation, this error is used to alter the weights in the network in order to decrease the error. Forward and backward passes are repeated until convergence [107].

B. Convolutional neural networks

The Convolutional Neural Networks (CNN) introduced by LeCun [108] have revolutionized image processing, and removed the burden of manual extraction of features in addition to being the exclusive architecture used by state-of-the-art ASC algorithms [102]. Convolutional neural networks have been extensively used in the acoustic scene classification task since they usually provide a summarizing classification of longer acoustic scene excerpts [109][110]. A CNN is made up primarily of 3 kinds of layers: **Convolutional layers**, **Pooling layers**, and **Fully Connected layers** [80]. A simplified CNN architecture for classification task is illustrated in Figure 2.3.

The basic functionality of the example CNN above can be broken down into four key areas as follows:

1. As found in other forms of Artificial neural networks, the input layer will hold the pixel values of the image.

2. The convolutional layer will determine the output of neurons of which are connected to local regions of the input through the calculation of the scalar product between their weights and the region connected to the input volume. The rectified linear unit (commonly shortened to ReLU) aims to apply an 'element wise' activation function such as sigmoid to the output of the activation produced by the previous layer.

3. The pooling layer will then simply perform down sampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation.

4. The fully-connected layers will then perform the same duties found in standard ANNs and attempt to produce class scores from the activations, to be used for classification. It is also suggested that ReLU may be used between these layers, as to improve performance.

Through this simple method of transformation, CNNs are able to transform the original input layer by layer using convolutional and down sampling techniques to produce class scores for classification and regression purposes [94].

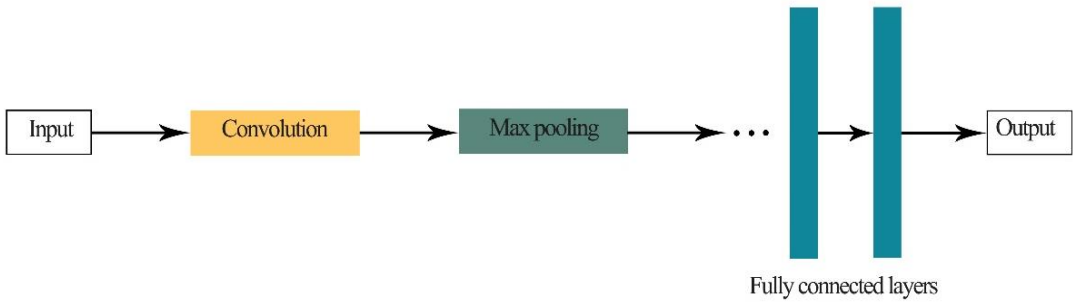


Figure 2.3: Convolutional neural network architecture [80].

THE CONVOLUTIONAL LAYER: The convolution operation that gives its name to the CNN is the fundamental building block of this type of network [105]. The layer parameters focus around the use of learnable kernels. These kernels are usually small in spatial dimensionality, but spread along the entirety of the depth of the input. When the data hits a convolutional layer, the layer convolves each filter across the spatial dimensionality of the

input to produce a 2D activation map. As we glide through the input, the scalar product is calculated for each value in that kernel. From this, the network will learn kernels that *"fire"* when they see a specific feature at a given spatial position of the input. These are commonly known as activations [94]. Every kernel will have a corresponding activation map, of which will be stacked along the depth dimension to form the full output volume from the convolutional layer [94].

THE POOLING LAYER: The pooling operation, less and less used in the current state of the art CNN, is fundamentally a dimension reduction operation [105]. This motivation behind using a pooling layer is to just pass relevant features to the next layer and thereby, summarizing generated features in the previous convolution layer. There can be many types of pooling operations: max pooling (selects the maximum valued cell out of the receptive field of that neuron), average pooling (passes the average of the cell values in the local receptive field of the neuron). Either before or after the pooling layer an additive bias and sigmoidal nonlinearity is applied to each feature map [80].

THE FULLY CONNECTED LAYER: After several convolution and pooling layers, the CNN generally ends with several fully connected layers [92]. The fully-connected layer; also known as the classification layer, contains neurons of which are directly connected to the neurons in the two adjacent layers, without being connected to any layers within them. This is analogous to the way neurons are arranged in traditional forms of ANN [94]. This is the fully connected layer which computes the score of each class from the extracted features from a convolutional layer in the preceding steps. The final layer feature maps are represented as vectors with scalar values which are passed to the fully connected layers [91].

C. Deep residual neural networks

Residual Network (ResNet) is a deep neural network developed by *Kaiming He* for large-scale data analysis, with the intent of designing ultra-deep networks that did not suffer from the **vanishing gradients** problem that predecessors had [91]. The vanishing gradients problem refers to the large decrease in the norm of the gradient during backpropagation. Such events are due to the long term components going exponentially fast to norm 0, making it impossible for the model to learn correlation between temporally distant events [111].

ResNet is a traditional feedforward network with a **residual connection**. It is developed with many different numbers of layers: 34, 50,101, 152, and even 1202 [91]. The ResNet architecture has been used in audio tasks such as speaker spoof detection [112], unsupervised audio representation learning [113] and acoustic scene classification [114].

A building block of the residual learning is shown in Figure 2.4. The sub-blocks in the architecture represent the complete convolutional layers including the activation functions. A deep residual network can consist of many such building blocks stacked together. In one residual building block, the output $H(x)$ of the block is a mapping of the input x. Instead of letting the multiple convolutional layers directly approximate the mapping $H(x)$, the residual mapping $F(x) = H(x) - x$ is to be approximated. A **shortcut connection**; also known as a **skip connection**, from the input to the output adds an identity mapping to the output of the stacked layers [115]. Augmenting neural networks with skip connections surprised the community by enabling the training of networks of more than 1,000 layers with significant performance gains. The skip connections in the residual blocks facilitate preserving the norm of the gradient, avoiding by this manner the vanishing gradient problem and leading to stable back propagation [116].

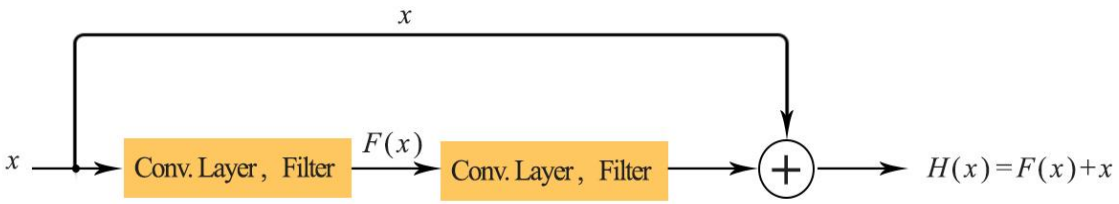


Figure 2.4: Building block of the Residual Neural Network [91].

Although it is known that very deep residual networks can improve accuracy of a model, it is argued that it takes too many extra layers to be rewarded with a small improvement in accuracy [117]. To address this problem, experiments with wider, rather than deeper, networks have been conducted and have led to an interesting finding with great practical importance concerning residual networks: **Wide Residual Networks (WRN)**. This network architecture has proven that shallow networks with increased width are able to provide similar or better results than those obtained with very deep neural networks. The architecture of a wide residual block is shown in Figure 2.5. The widening of a residual network is characterized by the widening factor *k*. Increasing the width of a network refers to increasing the number of filters on the convolutional layers *k* times.

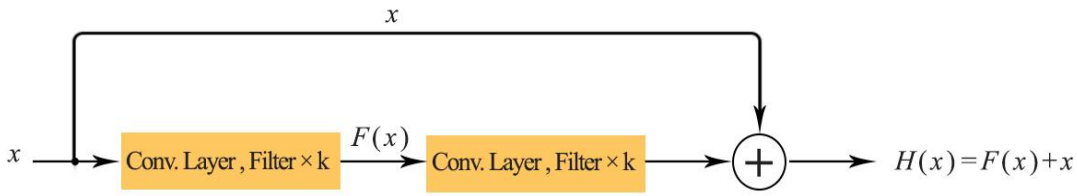


Figure 2.5: Building block of the Wide Residual Neural Network [117].

D. Alex neural networks

The Alex Network (AlexNet) is a fundamental, simple, and effective deep convolutional neural network architecture [118], which was first proposed by *Alex Krizhevsky et al.* in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC-2012) [119][9]. AlexNet has achieved state-of-the-art recognition accuracy against all traditional machine learning and computer vision approaches. It has demonstrated a significant breakthrough in the field of visual recognition and classification tasks and is the point in history where interest in deep learning increased rapidly [119].

The architecture of The Alex Network is shown in Figure 2.6. The network is mainly composed of cascaded stages, namely, convolution layers, pooling layers, rectified linear unit (ReLU) layers and fully connected layers. Specifically, AlexNet consists of five convolutional layers and three fully-connected layers. The first two convolutional layers are followed by normalization and a max-pooling layer, the third and fourth are directly connected whereas the fifth convolutional layer is followed by a max-pooling layer. The output of the fifth layer goes into a series of two fully-connected layers, in which the second fully-connected layer output is fed into a *softmax* classifier [120]. In order to prevent overfitting in the fully-connected layers, a regularization method called *"dropout"* is employed [121], which essentially consists of setting to zero the output of each hidden neuron with a certain probability [9]. The neurons which are *"dropped out"* do not contribute to the forward pass and do not participate in backpropagation. Another feature of the AlexNet model is the use of Rectified Linear Unit (ReLU), which is applied to each of the first seven layers. The use of ReLU non linearity has been shown by the authors to accelerate training time[120].

Although this architecture was designed for image recognition purposes, it has been shown that AlexNet and similar deep convolutional neural networks can be successfully trained to classify spectral images of environmental sounds [122][123][124].

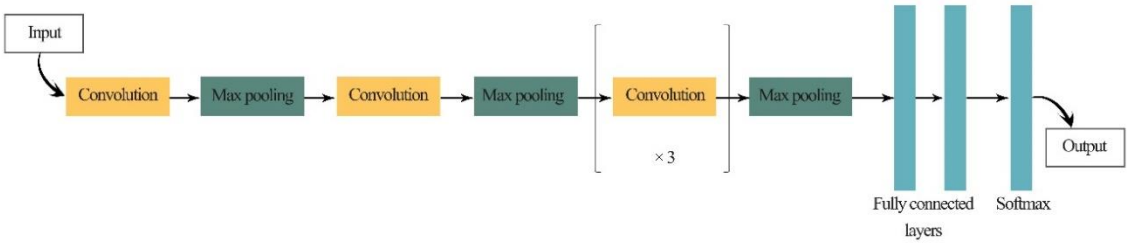


Figure 2.6: Alex Network architecture [119].

2.4 Data augmentation

Data collection is a major bottleneck in machine learning and an active research topic in multiple communities [125]. There are largely two reasons that data collection has recently become a critical issue. First, as machine learning is becoming more widely-used, new applications that do not necessarily have enough labeled data are emerging [125]. Second, unlike traditional machine learning, deep learning techniques automatically generate features, which saves feature engineering costs, but in return may require larger amounts of labeled data [125]. In consequence, the lack of sufficient amount of the training data or uneven class balance within the datasets became the most frequently mentioned problem in the field of machine learning [125]. As a result, there is a pressing need for accurate and scalable data collection techniques in the era of Big data [125], therefore, one of the ways of dealing with the lack of data problem and expanding the size of the data set is **Data Augmentation (DA)** [126].

The idea behind data augmentation is to disrupt the training data by injecting variance in order to have more data as varied as possible to feed the learning model [127]. It is done under the assumption that more information can be extracted from the original dataset through augmentations [128]. Thus, the model becomes more efficient in characterizing the differences between classes and less prone to overfitting i.e. learning a function with very high variance such as to perfectly model the training data.

For acoustic modeling, creating a perturbation in the data can be as little as time stretching, adding background noises, or a constant pitch shifting, and can go as far as deforming the sound with variable speed perturbation. These deformations must not alter the semantics of the labels, hence why, care must be taken to ensure that the deformations applied to the input audio leave the labels unmodified [127].

2.4.1 Online vs offline data augmentation

There are two different approaches to performing data augmentation: **Online data augmentation** and **offline data augmentation** [129]. The online data augmentation approach -data augmentation on the fly- is made during training, which consequently signifies that the augmented data does not exist afterwards [129]. The Offline data augmentation approach means that a new complete dataset is created before the training starts. Both approaches have different advantages: Offline augmentation makes the training process even faster, but is more advantageous if used in the case of smaller datasets considering that it involves an increase in the number of training data which leads to offline augmentation requiring more storage available beforehand [129]. Nonetheless the online augmentation approach can generate a larger number of unique samples if the training set is iterated over multiple times. On the other hand, Offline augmentation has a fixed number of possible augmented samples [129].

2.4.2 Common data augmentation techniques

A. Mixup

Mixup is a novel form of data augmentation that is used in order to combat overfitting [130] and improve the generalization ability of state-of-the-art neural network architectures [131][132]. The mixup data augmentation technique has been frequently used in a plethora of deep learning tasks, ranging from medical image segmentation [133] to natural language processing[134][82] to environmental sound recognition [135]. Dissimilar to traditional augmentation approaches such as rotation, flipping, distorting and deformation which are data-dependent and require the use of expert knowledge [136][100], mixup can be seen as data-agnostic augmentation procedure i.e. mixup is a non-label-preserving procedure [137][131][138]. Moreover, mixup is proven to be a better suited augmentation method for audio-based inputs than previously mentioned transformations, due to the temporal nature of audio signals and the physical meaning of the spectrum over the horizontal and vertical axes [130].

Mixup creates new synthetic training examples by drawing samples from the original dataset and **convexly** combining both in terms of the **input** and the **output** [134]. Combining two data points convexly is equivalent to linearly combining the latter points i.e. given two vectors v and w, any vector of the form $av + bw$ is a linear combination of v and w [139]. In more detail, the new synthetic training example $(\tilde{x}, \tilde{y})$ can be constructed by using the following formula:

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j \,, \tag{2.8}$$

$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j \,, \tag{2.9}$$

where x_i and x_j are two randomly selected feature vectors from the training data (regardless of the provided label of the samples),y_i and $y_j$are their corresponding class labels respectively [131]. The **linear interpolator** $\lambda \in [0,1]$ also known as the mix factor [135] or the mixing ratio [140], is sampled from a probability distribution known as *Be(a, b)*: ***Beta* distribution**[141] with a hyperparameter $\alpha = a = b$ for $\alpha \in (0 \rightarrow \infty)$. It is of paramount importance to note that the hyperparameter α is responsible for determining and controlling the shape of the *Beta* distribution, hence, regularizing the mixing ratio. Figure 2.7 shows the *Beta* probability density function with varying values of the hyperparameter α [142].

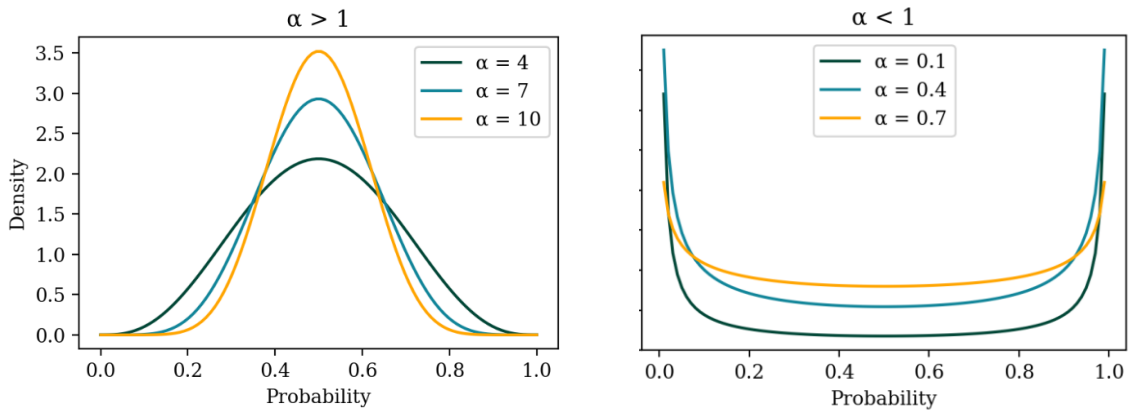


Figure 2.7: Probability density function of the Beta distribution.

As shown in the figure above, given a value of α<1, the beta distribution is U-shaped which signifies that the probability of selecting a value closer to either 0 or 1 is very high, whereas the probability of selecting values that belong between these two extremes is low and constant. Put in context, when the mixup hyperparameter α is smaller than 1, the possibility that the newly constructed virtual training example belongs to one of the given two classes is very high [143]. In contrast, given a value of α >1 results in a very low probability of selecting a value closer to either 0 or 1, i.e. the probability that the synthetic training example belongs to one of the given two classes is very low [143].

One exceptional case of the *Beta* probability distribution is the case where the hyperparameter α = 1. In this particular case, the *Beta* distribution *Be(a, b)* is equivalent to the *Uniform* distribution *U(0,1)* as shown in Figure 2.8 [142].

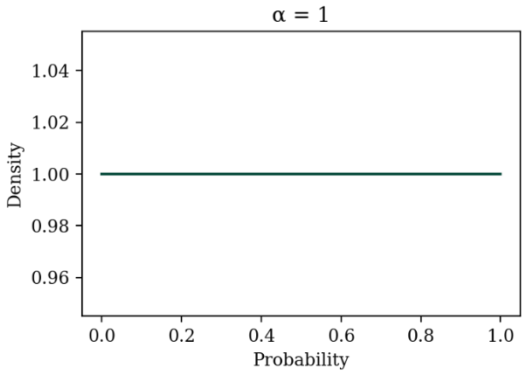
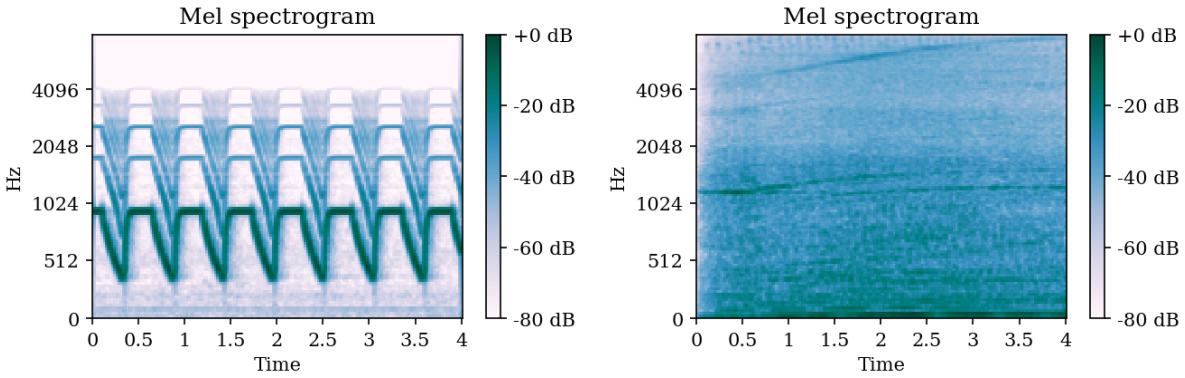


Figure 2.8: Probability density function of the Beta distribution with α=1.

Figure 2.9(a) and Figure 2.9(b) are spectrogram representations of a siren and a bus sound recording respectively.



(a) Siren recording. (b) Bus recording.

Figure 2.9: Spectrogram representations.

Figure 2.10 illustrates the spectrogram representation of a virtual training example constructed by the mixup of the audio samples shown in Figure 2.9, where the mixup hyperparameter alpha is set to $\alpha = 0.4$. Despite its **simplicity** and **minimal computation overhead** [131], the mixup data augmentation methods have provided state-of-the-art performance in many datasets, which include the *CIFAR-10*, *CIFAR-100* image classification datasets along with sound classification datasets namely: *CHIME-HOME* [144]**,** *ESC-10* and *ESC-50* [145].

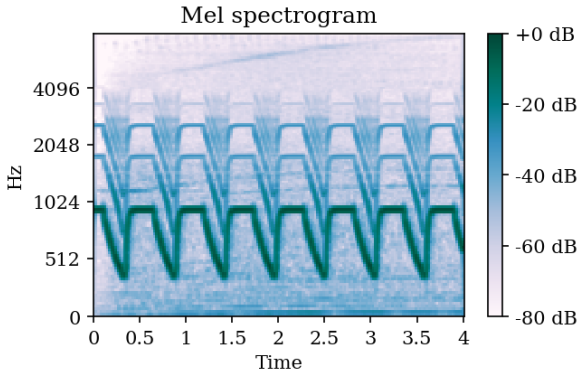
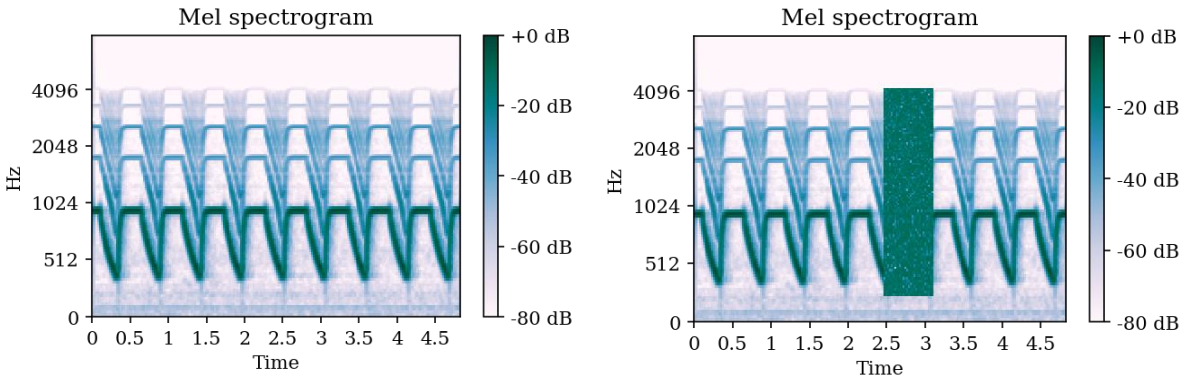


Figure 2.10: Spectrogram representation of the Mixup sample.

B. Random erasing

Random erasing is a data augmentation technique applied by selecting an **arbitrary** region from the input sample and erasing the content of this area [146]. This approach is extensively used in the field of computer vision where the training dataset consists of images [147][148]. Nonetheless, random erasing has been used for augmenting multiple audio datasets [149][150][151], considering that the preprocessing of audio data usually converts a sound signal into a two-dimensional spectrogram i.e. an image representation of the sound signal [152][153][130].

During training, the process of randomly erasing a region S_e, of a spectrogram S is performed using a certain probability P, where the ratio of the erased surface S_e/S is smaller than 1 [152]. Once an erasing region has been selected, each pixel within the area is assigned a random value within the range [0,255] as shown in Figure 2.11. This procedure generates a set of spectrograms with varying levels of occlusion, which helps making the model robust to occlusion and reducing the risk of network overfitting [146]. Random Erasing is parameter learning free, easy to implement, and can be integrated into most of the CNN-based models for both image and audio recognition [150][151].



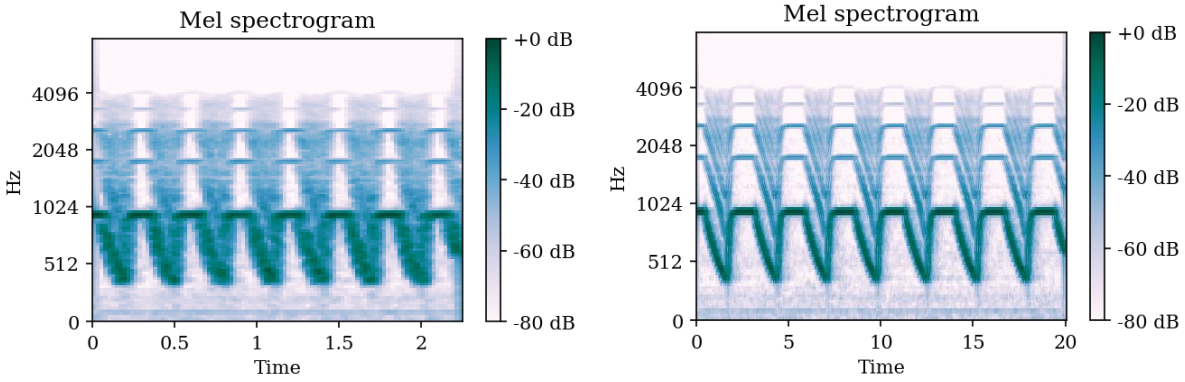
(a) Non-augmented sound. (b) Increased loudness sound.

Figure 2.11: Spectrogram of a siren recording augmented by random erasing.

C. Audio deformation

Audio deformation is a data augmentation method which can be performed by directly applying changes to an audio sample while maintaining the semantic validity of the corresponding label, converting it into a new input example to train a neural network [154][155]. Several methods have been implemented in literature to serve the audio deformation purpose, a few of these methods are presented below.

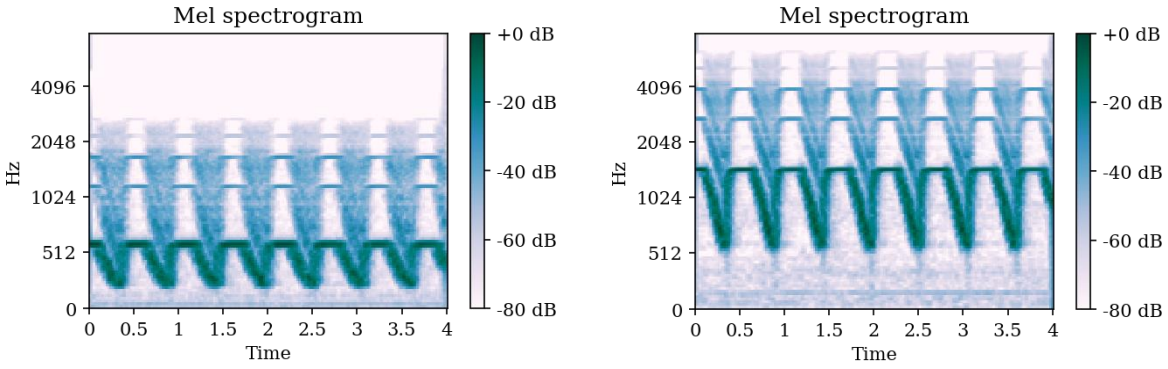
TIME STRETCHING: The basic principle behind time stretching is changing the **speed** of the audio sample, either by accelerating it or slowing it down [154]. Thus, time stretching changes the duration of the sample whereas its frequency content remains unchanged [156]. Figure 2.12 shows the resulting spectrogram representation of the previous siren recording after the application of time stretching augmentation.



(a) Decelerated sound. (c) Accelerated sound.

Figure 2.12: Spectrogram of a siren recording augmented by time stretching.

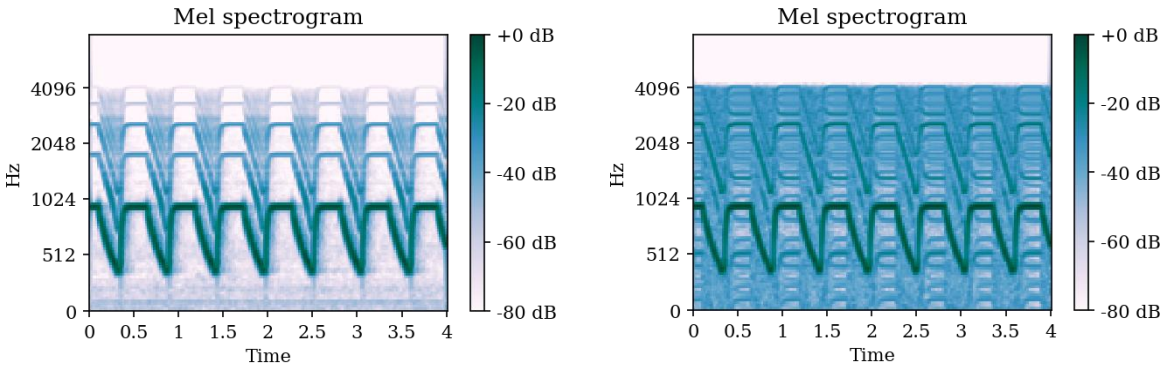
PITCH SHIFTING: Pitch shifting is the process of augmenting audio samples by altering the **pitch** without changing the speed of a sound recording [157]. Figure 2.13 illustrates the effect of pitch shifting on the previous siren recording. The application of pitch shifting augmentation is performed by scaling the frequency content of the audio sample by a constant factor i.e. scaling the linear-frequency spectrogram vertically [154][152][153].



(a) Lower pitched sound. (b) Higher pitched sound.

Figure 2.13: Spectrogram of a siren recording augmented by pitch shifting.

SIMPLE GAIN: Simple gain augmentation technique, also known as volume gain, is used to generate new training examples by increasing the **loudness** of audio samples in the training set [158]. This procedure is implemented through scaling the spectrogram of an audio sample by a specified gain in decibels (dB). Varying the loudness of sound samples in a dataset can be helpful for the deep learning model in cases where the audio input is not loud enough [152][153], the figure below shows how the spectrogram representation is affected after applying simple gain data augmentation on a siren recording.



(a) Non-augmented sound. (b) Increased loudness sound.

Figure 2.14: Spectrogram of a siren recording augmented by simple gain.

2.5 Summary of empirical and theoretical findings

Despite the prominent success of deep learning-based models in both computer vision tasks; such as image classification [147] and object detection [148], and in audio-related tasks, such as speech recognition [153] and audio classification [154], these models are trained using large parameters and are therefore heavily dependent on large-scale training samples [9].

Unfortunately, many tasks lack large amounts of diverse, trustworthy data. The consequences of this is model overfitting and poor generalization. This problem is extremely challenging in both the visual and the audio fields. To mend with this problem, data augmentation techniques have been extensively used in literature and have been proven to effectively enhance the model generalization ability [131][153][146][130].

S. Thulasidasan et al. [159] analyzed the effect of data augmentation-based learning on neural network calibration by performing numerous experiments using mixup augmentation technique. The experiments were conducted on both image and natural language processing datasets, with the use of various deep network architectures. They have shown that mixup-trained networks are better calibrated and provide more reliable estimates both for in-sample and out-of-sample data.

In 2018, *W. Shengyun et al.* [160] trained a convolutional recurrent neural network (CRNN) on mixed time-frequency representation of sound samples for domestic audio tagging. They explored multiple ratios for the mixup approach and established based on their experiments that mixup generalizes better than other mixed form data augmentation methods in the audio tagging task.

Most recently, *W. Shengyun et al.* [130] conducted a thorough comparison of multiple data augmentation techniques for sound classification namely: time stretching, mixup, pitch shifting and time masking, and have proved that data augmentation methods are very helpful for the improvement of audio classification performance, especially directly used on spectrograms.

In addition, *W Shengyun et al.* proposed an effective and easy to implement data augmentation method named Mixed Frequency Masking data augmentation. This method is inspired by mixup data augmentation [131] and *SpecAugment* method [161], and adopts nonlinear combinations to construct new samples and linear combinations to construct labels.

Mixup-based augmentation methods have also been used in a variety of deep learning tasks such as: sentence classification [162] and information processing in medical imaging [163].

It is worth mentioning that various other techniques have been used for augmenting sound data, for instance: *Um et al.* [164] used a variety of augmentations, such as jittering, scaling, rotation, slicing, permutation, magnitude warping, and time warping, to improve wearable sensor data for deep temporal convolutional neural networks.

K. M Rashid and J Louis [83] used jittering, scaling, rotation, and time warping with long short-term memory networks for construction equipment activity recognition. In acoustic recognition, frequency warping has been extensively used to alleviate the problem of overfitting [165][166]. Table 2.5 provides an in-detail summary of the previously mentioned studies along with their achieved performances.

Table 2.5: Summary of related works results.

Ref	Dataset	Data Type	Architecture	Augmentation	Performance
[159]	STL-10	Image	VGG-16	Mixup	82% accurate
[159]	CIFAR-100	Image	ResNet-34	Mixup	80% accurate
[159]	Fashion-MNIST	Image	ResNet-18	Mixup	86% accurate
[142]	ImageNet	Image	ResNet-101	Mixup	93% accurate
[142]	IMDB	Text	CNN	Mixup	83% accurate
[143]	CHIME-HOME	Audio	CRNN	Mixup	0.10 EER[1]
[112]	Freesound Kaggle2018	Audio	ResNet	Mixup	93.60% mAP@3[2]
				Time Stretching	92.59% mAP@3
[145]	TREC	Text	CNN	Mixup	92.1% accurate
			LSTM		89.4 % accurate
[147]	Pakinson's Disease Dataset	Audio	CNN	Rotation	82.62% accurate
				Time warping	82.00% accurate
[148]	2015UCR Series Archive	Audio	VGG	Magnitude Warping	86.00% accurate
			LSTM	Scaling	61.6% accurate

[1] The EER is an accuracy measurement in biometric systems used to predetermine the threshold values for its False Acceptance Rate (FAR) and False Rejection Rate (FRR).
[2] map@3 is an evaluation metric which returns the best 3 classifications for each audio sample.

2.6 Conclusion

Throughout this chapter, we have reviewed the major concepts of classification task as well as the steps to undertake in order to develop a classification model. We have presented a few of the most commonly used deep learning architectures and data augmentation techniques in acoustic scene classification context. We have then summarized a few of the most recent related works while shedding the light on the data augmentation aspects of these works. Furthermore, we have defined the most common evaluation metrics and approaches used in literature, highlighting the importance of statistical tests in assessing the performance of machine learning classifiers as well as comparing between models.

PART 2: EXPERIMENTATIONS

This part of the thesis is dedicated to the procedure that we have adopted for the design and evaluation of sound classification systems. It is composed of two chapters. In the first chapter we present the experimental setup defined to construct and analyze our ASC system, whereas the second chapter is devoted to the examination and discussion of the results of our experiments.

Chapter 3: EXPERIMENTAL DESIGN

3.1 Introduction

This chapter presents the setup defined to conduct our experiments. In **Section 3.2** we present our data acquisition procedure. Next, in **Section 3.3**, we lay out the numerous tools that we have used in the development of our systems. Then in **Section 3.4**, we describe our implemented ASC system, and present the features used to train the system as well as the model topology. Finally, in **Section 3.5**, we explain the procedure that we have followed in order to analyze and discuss our results.

3.2 Data acquisition procedure

DATASET: We have carried out our experiments on the TAU Urban Acoustic Scenes 2019 dataset (35,6 GB) which consists of 40 hours of high-quality binaural audio recordings from 10 acoustic scenes in 12 different cities [59]. The acoustic scene class labels are the following: *Airport, Indoor shopping mall, Metro station, Pedestrian street, Public square, Street with medium level of traffic, travelling by a tram, travelling by a bus, Travelling by an underground metro* and *Urban park*. For each scene class, 5-6 minutes long audio files have been recorded in each location. The original recordings have been split into segments with a length of 10 seconds each and are provided in individual files in wav format. This format is strongly recommended and widely used in various audio processing fields as it encodes data without compression i.e. it is a **lossless** file format [167]. The training/test subsets are created based on the recording location such that the training subset contains approximately 70% of recording locations from each city. The development set contains 14400 segments in total (144 per city per acoustic scene class) divided as follows: there are 9185 segments in the training set and 5215 in the test set.

RECORDING SETTINGS: The audio files were recorded by *Soundman OKM II Klassik/studio* with a power supply *Adapter A3* [59], electret binaural microphone and a *Zoom F8* audio recorder using 48kHz sampling rate and 24-bit resolution which guarantees a high audio quality. This microphone is specifically made to resemble headphones i.e. the microphones

are positioned in the ears. By positioning the microphones in the ears, the recordings are very similar to the subjective perception of sound reaching the user's ear [59].

3.3 Development environment and utility libraries

Deep learning research relies on exhaustive datasets and heavy computations during training which is generally time consuming and resource hungry. Thus, the use of parallel computing is necessary given that it considerably accelerates the training process [168]. For this purpose, Graphics Processing Units (GPU) is considered to be the leading parallel computing device used to conduct deep learning experiments. A GPU is an integrated single chip processor, consisting of a highly parallel structure designed to perform extensive graphical and mathematical computations [169]. The structure of GPUs allows parallel computing through thousands of threads at a time hence giving this category of hardware the upper hand in deep learning executions [170].

However, the use of such hardware resources can be not only costly in terms of purchase and maintenance, but also risky if events such as under/over utilization and/or equipment depreciation occur, making the deep learning project very cost effective [171].

On this account, multiple online solutions have been proposed by various tech companies such as Google, Amazon and Intel which consists in providing on-the-fly hardware i.e. providing pay-by-hour or free GPUs and fully configured runtime sessions for deep learning projects [171]. We have opted for this online solution to carry out our experiments considering the numerous advantages that it provides. The deep learning environment setup that we have used to complete our work is described below.

3.3.1 Google Drive Storage

Regarding the storage of our data, we have chosen *Google Drive*. It is an online storage, synchronization and sharing service that offers 15GB of free space, and several packages for different storage spaces. This way, the dataset is easier to access and load in the chosen runtime environment which is Google Colaboratory.

3.3.2 Google Colaboratory

Google Colaboartory, also referred to as *"Google Colab"*, is a cloud service based on the *Jupyter* environment for machine learning education and research [171][172]. It provides a

fully configured runtime for deep learning using *Python* and free access to Tensor Processing Unit (TPU); which offers up to 35 GB of RAM and 107 GB of disk space, and a TESLA k80 GPU. We have made use of the provided TPU to perform audio feature extraction taking into account that this process is costly in terms of RAM, while we have performed training of our models using the provided GPU. The figure below shows the colab environment setup.

Note that further robust resources can be accessed by upgrading to a professional version of Google Colaboratory. This upgrade guarantees priority access to highly powerful GPUs such as Tesla T4 and Tesla P100 and provides additional disk space and RAM capacity [173].

3.3.3 Utility Libraries

Another bright spot for Google Colaboratory is the availability of all the necessary python libraries used for audio processing and deep learning experiments [171]. These libraries do not require any installation or configuration. The following are a few of the most relevant libraries that we have used in our work.

***LIBROSA*:** *Librosa* is a Python package for audio and music signal analysis and processing. It provides implementations of a variety of common functions that fall into four categories that are audio and time-series operations, spectrogram calculation, time and frequency conversion, and pitch operations [174]. These functions are heavily used throughout our experiments.

***TENSORFLOW*:** *Tensorflow* is an open-source library that implements automatic learning methods based on the principle of deep learning neural networks. We have used *Tensorflow* in our work as it supports a variety of applications, with a focus on training and inference on deep neural networks [175].

***KERAS*:** *Keras* is a high-level API written in python that runs on a Tensorflow backend. It is an approachable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. Its simplicity helps users develop a deep learning model quickly and provides a ton of flexibility while still being a high-level API [176].

***PYTORCH*:** *PyTorch* is an open-source library developed by Facebook that performs instantaneous dynamic tensor computations with automatic differentiation and GPU acceleration, while maintaining performance comparable to the fastest modern libraries for deep learning [177].

In addition to the above mentioned deep learning libraries we have made use of the *Numpy* library to perform manipulation operations on our data [178], the *Matplotlib* library for plotting and graphical representations, the *pickle* module [179] for serialization of python objects for storing purposes and the H5py python package to store our trained models for testing. Table 3.1 provides additional information about the libraries we have used in our work.

Table 3.1: Utility libraries used for deep learning.

Utility Library	Version
Python	3.5
Librosa	0.6.3
Keras	2.3.1
Tensorflow	2.2.0
PyTorch	1.0.0
Matplotlib	2.0.2
Numpy	1.14.0
Pickle	5

3.4 Design and analysis of acoustic scene classification systems

3.4.1 ASC System Characterization

In order to fully exploit our dataset, we aim at building an acoustic scene classification system able to classify audio samples into one of predefined acoustic scene classes in a closed set classification setup. The designed supervised system follows the general framework for acoustic scene classification which usually consists of two stages. First, preprocessing of audio samples i.e. obtaining time-frequency representation of the data and extracting relevant features. Second, employing the extracted features to perform classification. Each of the developed systems is trained using specific feature sets along with different classification paradigms while varying their parameters.

To explore the impact of augmenting the data for training, we have invoked the well-known data augmentation technique: ***mixup*** [131]. All developed systems have been trained and evaluated using mixup augmentation technique. We have utilized the F1-score metric and confusion matrix to perform the evaluation of our systems and have supported our analysis and discussion with numerous statistical tests. The proposed system is illustrated in Figure 3.1.

Additional details on the feature extraction techniques and the machine learning models that were used can be found in **Sections 1.11** and **2.3.2.**

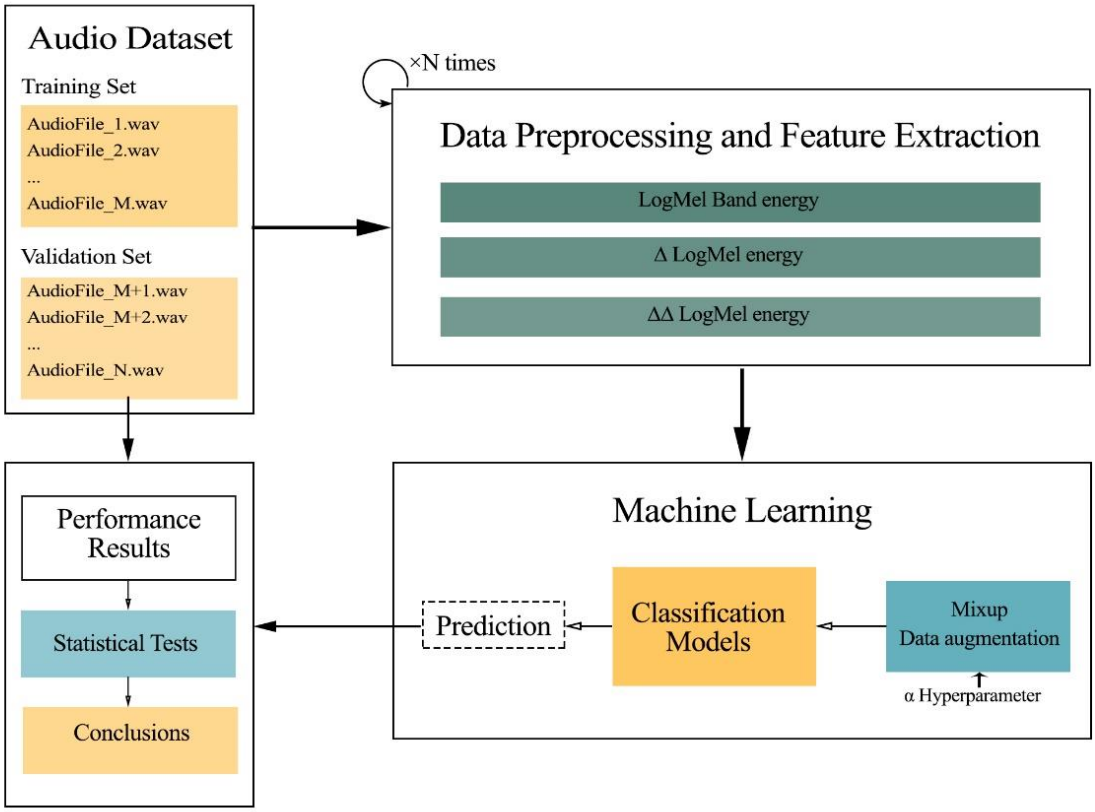


Figure 3.1: Acoustic Scene Classification System pipeline.

3.4.2 Data preprocessing and feature extraction

One important characteristic to take into account when training a machine learning model is imbalanced data [180]. In supervised learning, the data set is said to be imbalanced if the class prior probabilities are highly unequal and prediction models built from imbalanced datasets are most often biased towards the majority concept [181]. Generally, data exploration techniques ranging from data querying and basic statistics to advanced visualization, are used to discover class imbalance and other characteristics of a dataset.

To gain insight on the class distribution of our dataset, we have plotted the number of sound recordings per class in both the training set and the testing set as depicted in Figure 3.2. The below visualization shows that both training and testing sets are balanced, hence, we did not invoke under/over sampling methods or cross-validation in our work.

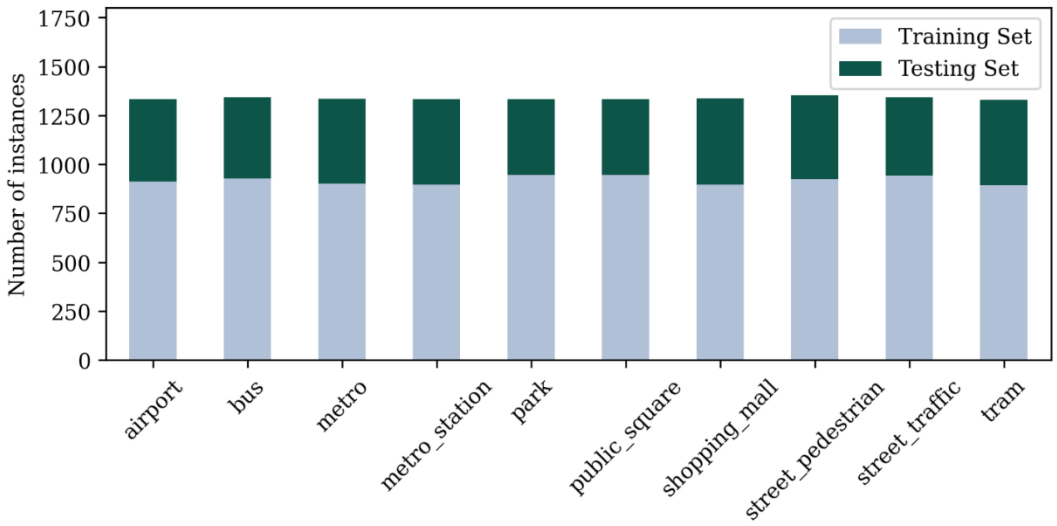


Figure 3.2: Stacked histogram class distribution in the training and testing sets.

Furthermore, we have empirically proved that our dataset is split in a balanced manner with the use of **Shannon's entropy** [182]. A Shannon entropy is a general measure of diversity of objects in a given set i.e. it provides insight on the certainty of drawing an object at random from a given set. On a data set of *n* instances and *k* classes of size c_i, the Shannon entropy is computed as follows:

$$H = -\sum_{i=1}^{k} \frac{c_i}{n} \, log \frac{c_i}{n}, \tag{3.1}$$

where $\frac{c_i}{n}$ is the probability of occurrence of a class c_i. H tends to 0 when the data set is unbalanced and it tends to $logk$ otherwise.

Feature extraction is one of the most crucial stages in the acoustic scene classification task framework. This stage contributes greatly to the effectiveness of an ASC system. One of the most popular features applied in the ASC are representations of Mel-frequency scales such as Log-Mel energies [183]. The main reason for their success is that they provide a reasonably good representation of the spectral properties of the audio signal. In addition, they produce a reasonably high inter-class variability allowing for class discrimination. Besides that, these features can be used as basis to derive higher level features such as ΔLog-Mel energies and ΔΔ*L*og-Mel energies.

Following the above indications, we have extracted the frequency domain features: Log-Mel band energies and their derivatives ΔLog-Mel energies + ΔΔLog-Mel energies, from the audio samples to train our models.

D. Log-Mel band Energies

The provided audio files in our dataset are sampled at a rate of 48 kHz and are in stereo format. As a first feature extraction step, we have applied a **Short-Time Fourier Transform** with 2048 FFT points to each of the 10 second binaural signals in our dataset. Note that the number of FFT points should be a power of 2 for fastest computation of the spectrogram. To prevent spectral leakage, we have applied a **Hamming windowing function** with a 50% overlap. Next, **128 bin *Mel*-filter banks** are applied to the calculated power spectrums.

It is worth noting that it is important to keep a sufficient number of bins for representing the spectral characteristics, while greatly reducing the feature dimensions; hence, 128 frequency bins is an adequate number of bins to use. Finally, the resulting *Mel*-energy values are **logarithmically scaled** to obtain the Log-*Mel* features. We have extracted these features using the *Librosa* version *0.6.3* python package. A summary of the values and variants used in extracting Log-Mel band features is shown in Table 3.2.

Table 3.2: Log-Mel features parameters.

Parameter	Configuration
Sample rate	48 000 Hz
FFT points	2048
Windowing function	Hamming
Overlap	50%
Mel bands	128

E. Delta Log-Mel energies and Delta-Delta Log-Mel energies

It is widely acknowledged that Log-Mel energies only describe the power spectral envelope of a single frame and are referred to as static feature [184]. Nevertheless, an acoustic scene recognition system could benefit from information about the rate of change of these Log-Mel features. In our work, we have extracted information about the temporal dynamics of the audio files by computing first and second derivatives of Log-Mel energies namely: ΔLog-Mel

energies and ΔΔLog-Mel energies, using the procedure presented in **Section 1.11.1**. We then stack the resulting features along with Log-Mel features for the training process. The figures above depict the time-domain, Log-Mel energies and ΔΔLog-Mel energies representations of a sound sample belonging to the *"street traffic"* acoustic scene.

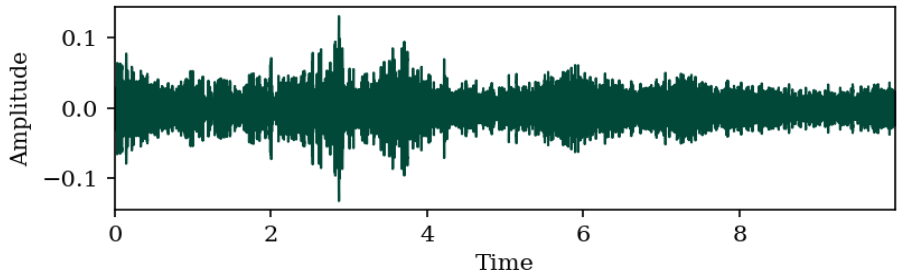


Figure 3.3: Time domain representation of a street traffic audio sample.

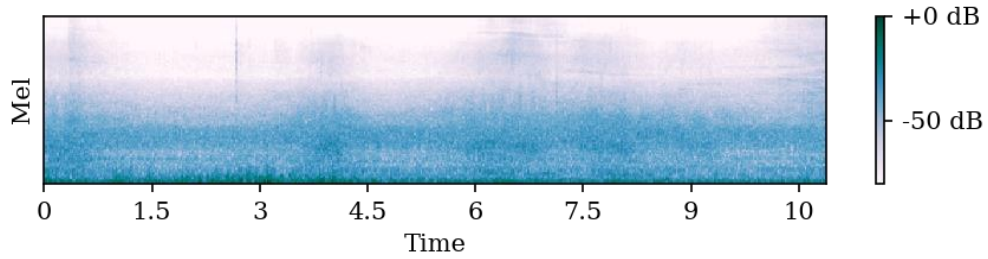


Figure 3.4: Log-Mel energies representation of a street traffic audio sample.

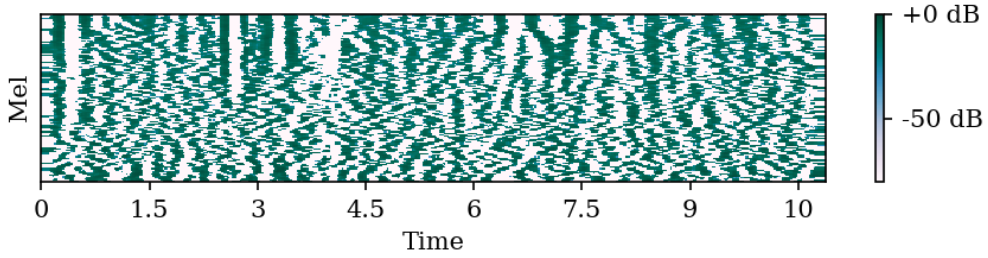


Figure 3.5: ΔΔ Log-Mel energies representation of a street traffic audio sample.

3.4.4 Data Augmentation and Regularization

MIXUP: To prevent overfitting during training and enhance the robustness of our model, we have invoked the mixup data augmentation technique. We have applied mixup augmentation by randomly selecting a pair of audio feature vectors from the training batch

and computing their weighted sum them as introduced in **Section 2.4.2.A**. While the audio feature vectors are computed and stored in advance, we have used on-the-fly data augmentation approach to construct the new samples. This approach has allowed us to thoroughly explore the effect of mixup augmentation technique by virtue of being computationally inexpensive and requiring no additional disk-space while guaranteeing a large amount of diverse training data. For exploration purposes, different values of the mix factor α have been tested.

WEIGHT DECAY: Multiple studies have shown that the generalization ability of a deep neural network depends on a balance between the information in the training set and the complexity of the network [185]. Bad generalization occurs if the input information does not match the complexity of the deep neural network. Thus, the deep neural network requires regularization techniques during the training process in order to achieve better generalization and avoid overfitting. For this purpose, we have opted for a regularization method called **weight decay**.

Weight decay is an explicit way of regularization such that a regularization term is added into the energy in order to penalize large weight values. We can directly control the regularization effect through manually tuning the weight decay coefficient ***wd*.** In our work, we have used a value of 1×10^{-3} for the weight decay coefficient on all convolutional layers.

3.4.5 Neural Network Architectures

Highest accuracy in numerous approaches to scene recognition have arisen from training deep convolutional neural networks on audio spectrogram features [186]. Driven by this, we have also adopted the use of CNNs to build our ASC system namely: Residual Network and Alex Network.

DEEP RESIDUAL NETWORK: The key idea that has enabled a deeper network to be trained effectively was the introduction of so-called **skip-connections** [116] in the network architecture, hence developing Residual networks. Many variations of these networks have since been proposed and widely used as they offer the virtue of simplicity [187]. Motivated by this, we have used a Wide Residual Network architecture (WRN) with a widening factor

k, as these architectures have been demonstrated to produce better accuracy in less training time than deeper networks [187].

Residual networks with a widening factor k = 1 are referred to as "thin" whereas networks with k > 1 are referred to as "wide". Note that, the residual network design used in our work is a pre-activation variety [8, 11] i.e. the order of batch normalization, activation and convolution in a residual block was changed from conv-BN-ReLU to BN-ReLU-conv as the latter was shown to train faster and achieve better results [188]. Table 3.3 shows the ResNet architecture that we have used to conduct our experiments. Overall, our network had approximately *1,487,268* parameters for $k = 1$ and *3,254,468* parameters for $k = 2$.

Table 3.3: Residual Network architecture. BN: Batch Normalization, ReLU: Rectified Linear Unit.

ResNet
Input $1 \times 128 \times 128$
$\begin{bmatrix} \begin{bmatrix} \text{BN-ReLU-3} \times 3 \text{ Convolution-64}\times k \\ \\ \text{BN-ReLU-3} \times 3 \text{ Convolution-64}\times k \end{bmatrix} \times 2 \end{bmatrix} \times 4$
BN-1× 1 Convolution-128
BN-1× 1 Convolution-128
Global Average Pooling
10-way Softmax

ALEX NETWORK: Given the successful application of image-based neural networks for acoustic scene classification tasks, we have employed a CNN-based architecture namely: *AlexNet*. The model topology; including details of each layer, is presented in Table 3.4.

Our system is inspired by the original AlexNet design [96] with a few modifications. Please note that the rectified linear unit activation function *"ReLU"* is applied for each convolutional layer. In addition, we have employed batch normalization for all convolutional layer as experiments have shown that it reduces the training time and improves the performance of CNN-based systems. Overall, our network had approximately *13,143,642* parameters.

Table 3.4: Alex Network architecture. BN: Batch Normalization, ReLU: Rectified Linear Unit.

AlexNet
Input $1 \times 128 \times 128$
3×3 Convolution-48-BN-ReLU
2×2 Max Pooling
3×3 Convolution-96-BN-ReLU
2×2 Max Pooling
3×3 Convolution-192-BN-ReLU
2×2 Max Pooling
3×3 Convolution-192-BN-ReLU
3×3 Convolution-192-BN-ReLU
2×2 Max Pooling
Flattening
Fully Connected (dim-1024)-BN-ReLU
Fully Connected (dim-256)-BN-ReLU
10-way Softmax

3.5 Evaluation procedure

There are different ways of evaluating the performance of learning algorithms and the classifiers that they generate. Classification quality measures are generally constructed from a confusion matrix that records correctly and incorrectly recognized examples for each class as shown in Table 2.3 in **Section 2.2.2.F**.

In order to evaluate the results of our experiments, we have initially opted for the accuracy metric which is extensively used in literature. However, the use of this metric for statistical tests ranking did not provide sufficient information for proper evaluation. Therefore, we have chosen the F1-score metric which; unlike the accuracy, takes both false positives and false negatives into account, making it more reliable in our work.

It is worthwhile emphasizing that relying on a single performance metric namely: averaged F1-score in our analysis is not sufficient for drawing any conclusions. For this purpose, we have supported our analysis and discussion with numerous statistical tests. The choice of specific statistical tests for the results of each evaluation case is not only based on statistical

appropriateness but also on the intended objective. The evaluation procedure that we have followed in order to compare and analyze our trained systems is as follows:

A. For the purpose of comparing **two classifiers or multiple classifiers in a pairwise manner over multiple datasets**, we have opted for the non-parametric *Wilcoxon signed-ranks test* [74]. We have chosen this test over the acclaimed t-test by reason of it being more sensible i.e. outliers (exceptionally good/bad performances on a few data sets) have less effect on the *Wilcoxon* than on the *t-test* [74].

B. We have used the non-parametric *Friedman* test for measuring **the difference between more than two classifiers, i.e. multiple classifiers, over multiple datasets**. The null hypothesis being tested where we made use of the *Friedman* test was that all classifiers perform the same and the observed differences are merely due to chance.

C. If the null-hypothesis was rejected after the use of *Friedman* test, we have proceeded with post-hoc tests namely: *Nemenyi* test [74] when we aimed at comparing **all classifiers against each other,** and *Bonferroni-Dunn* [74] test when the aim was to compare **all classifiers with a control system.**

D. For **graphical representation** of statistical results, we have used critical difference diagrams (CD). The top line in the diagram is the axis on which the average ranks computer during Friedman test are plotted. The axis is turned so that the lowest (best) ranks are to the right since systems on the right side are perceived as better.

E. To gain more insight into the **errors** made by our systems and the **type of these errors** we have computed the confusion matrix, which allowed us to thoroughly observe the model behavior over all the classes.

F. For experiments where results are too numerous and complicated to be described adequately in plain text, we have opted for a graphical representation i.e. graph plot to observe and analyze the progression of the systems.

3.6 Conclusion

In this chapter, we have described the setup used to conduct our explorations and assess our results. We have presented the dataset used to train our systems as well as the features and augmentation technique utilized for the latter aim. In the following chapter, we will present and analyze the results of these experiments in order to derive empirical findings and conclusions.

Chapter 4: EXPERIMENTAL FINDINGS AND DISCUSSION

4.1 Introduction

This chapter describes and discusses the experimental results and findings that we have obtained during our experiments. The main goal of our case study is to compare the performances of several scene recognition systems, varying the mechanism for extracting features and the hyperparameters used for learning the models. Most importantly, we have thoroughly examined the influence of data augmentation on our systems. Specifically, we have conducted our experiments on TAU Urban Acoustic Scenes 2019 dataset, consisting of audio files recorded using one recording device. We have compared two deep neural network architectures namely, ResNet and AlexNet; and have trained our systems on Log-Mel Spectrogram features. Table 4.1 describes the scene classification systems that have been tested in our experiments. For additional information on these models and their parameters, please refer to **Sections 3.5.4.A** and **3.5.4.B**.

For evaluating these systems, we have estimated the F1 score along with the confusion matrix on the provided test set. Furthermore, we have based our discussions and conclusions on various statistical tests. Specifically, our case study consists of 4 experiments:

- **Experiment 1**: investigates the impact of the mixup data augmentation technique.
- **Experiment 2:** examines the impact of the mixup parameter.
- **Experiment 3:** studies and compares several acoustic sound recognition systems obtained by varying hyperparameters and the feature sets.
- **Experiment 4:** explores the change of the prediction scores as a function of the number of epochs.

Table 4.1: List of all classifiers used in the experiments.

System Abbreviation	Feature Set	System Description
$Res_{k=1}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs without mixup
$Res_{k=1} + Mix$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 0.4
$Res_{k=1} + \Delta\Delta$	Log-Mel+ΔLog-Mel+ΔΔLog-Mel	ResNet model of width 1 trained for 250 epochs without mixup
$Res_{k=1} + \Delta\Delta + Mix$	Log-Mel+ΔLog-Mel+ΔΔLog-Mel	ResNet model of width 1 trained for 250 epochs with mixup α = 0.4
$Res_{k=2}$	Log-Mel energies	ResNet model of width 2 trained for 250 epochs without mixup
$Res_{k=2} + Mix$	Log-Mel energies	ResNet model of width 2 trained for 250 epochs with mixup α = 0.4
$Res_{k=2} + \Delta\Delta$	Log-Mel+ΔLog-Mel+ΔΔLog-Mel	ResNet model of width 2 trained for 250 epochs without mixup
$Res_{k=2} + \Delta\Delta + Mix$	Log-Mel+ΔLog-Mel+ΔΔLog-Mel	ResNet model of width 2 trained for 250 epochs with mixup α = 0.4
$Res_{\alpha=0.1}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 0.1
$Res_{\alpha=0.4}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 0.4
$Res_{\alpha=0.7}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 0.7
$Res_{\alpha=1}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 1
$Res_{\alpha=4}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 4
$Res_{\alpha=7}$	Log-Mel energies	ResNet model of width 1 trained for 250 epochs with mixup α = 7
$Alex$	Log-Mel energies	AlexNet model trained for 20 epochs without mixup
$Alex + Mix$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 0.4
$Alex_{\alpha=0.1}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 0.1
$Alex_{\alpha=0.4}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 0.4
$Alex_{\alpha=0.7}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 0.7
$Alex_{\alpha=1}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 1
$Alex_{\alpha=4}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 4
$Alex_{\alpha=7}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 7
$Alex_{\alpha=10}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 10
$Alex_{\alpha=12}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup α = 12
$Alex_{a=4,b=7}$	Log-Mel energies	AlexNet model trained for 20 epochs with mixup a = 4, b = 7

4.2 Experiment 1: Impact of the mixup data augmentation technique

This experiment is devoted to studying the impact of the Mixup data augmentation technique on the performance of ResNet and AlexNet models. To this end, we have trained each model twice: first, on the augmented data; second, on the original data. Note that setting the value of the mixup parameter α is of vital importance and significantly influences the overall performance [131]. Exploratory experiments have indicated that setting α to 0.4 is preferable. Table 4.2 reports the results of this experiment. The last row specifies the averaged performance of each system over all scenes.

Table 4.2: Average F1-score results of $Res_{k=1}$, $Res_{k=1}$+Mix, Alex and Alex+Mix

Classes	$\boldsymbol{Res_{k=1}}$	$\boldsymbol{Res_{k=1} + Mix}$	$\boldsymbol{Alex}$	$\boldsymbol{Alex + Mix}$
Airport	0.61	0.67	0.70	**0.72**
Bus	0.73	**0.79**	0.74	0.77
Metro	0.69	**0.73**	0.66	0.61
Metro Station	0.62	**0.65**	**0.65**	0.61
Park	0.86	0.87	**0.88**	0.86
Public Square	0.56	**0.65**	0.57	0.58
Shopping Mall	0.68	**0.70**	0.61	0.61
Street Pedestrian	0.60	**0.71**	0.63	0.61
Street Traffic	0.81	**0.83**	**0.83**	**0.83**
Tram	0.65	0.67	**0.70**	0.67
Average F1-score	**0.68**	**0.73**	**0.70**	**0.68**

Table 4.2 reveals that $Res_{k=1} + Mix$ yields the highest averaged score followed by $Alex$, whereas, $Alex + Mix$ produces the **worst** performance. Therefore, these initial observations suggest that the mixup technique has a **positive** impact on the ResNet model. However, training on augmented data does not demonstrate a similar effect on the AlexNet-based system.

Note that relying our analysis only on averaged F1-scores does not provide strong evidence on the significance of these conclusions. Moreover, it is widely acknowledged that the analysis of results through statistical tests is of paramount importance and should be conducted properly in order to ensure significance [74]. Dietterich [76], Demšar [74], García et al. [77], and Japkowicz et al. [78] introduced several statistical tests such as McNemar, Friedman, Nemenyi,

Bonferroni-Dunn, Wilcoxon, and ANOVA for performance comparison. Following their recommendations, we have considered using the Wilcoxon signed-ranks test, which fits well with our purpose. Under the null hypothesis, we have assumed that the differences between a system trained on the augmented data and non-augmented data are insignificant and have occurred merely due to chance. A summary of this test statistics is shown in Table 4.3. Row 2 specifies the number of Win/Tie/Loss of the data augmentation-based system (column highlighted in grey in Table 4.2) over the baseline system. Row 3 shows the associated $p-value$. For example, a $p-value$ = 0.05 indicates that the system in the column highlighted in grey is significantly better at 5% significance level than the system trained on the original data i.e. introducing the mixup data augmentation technique significantly improves the predictive performance.

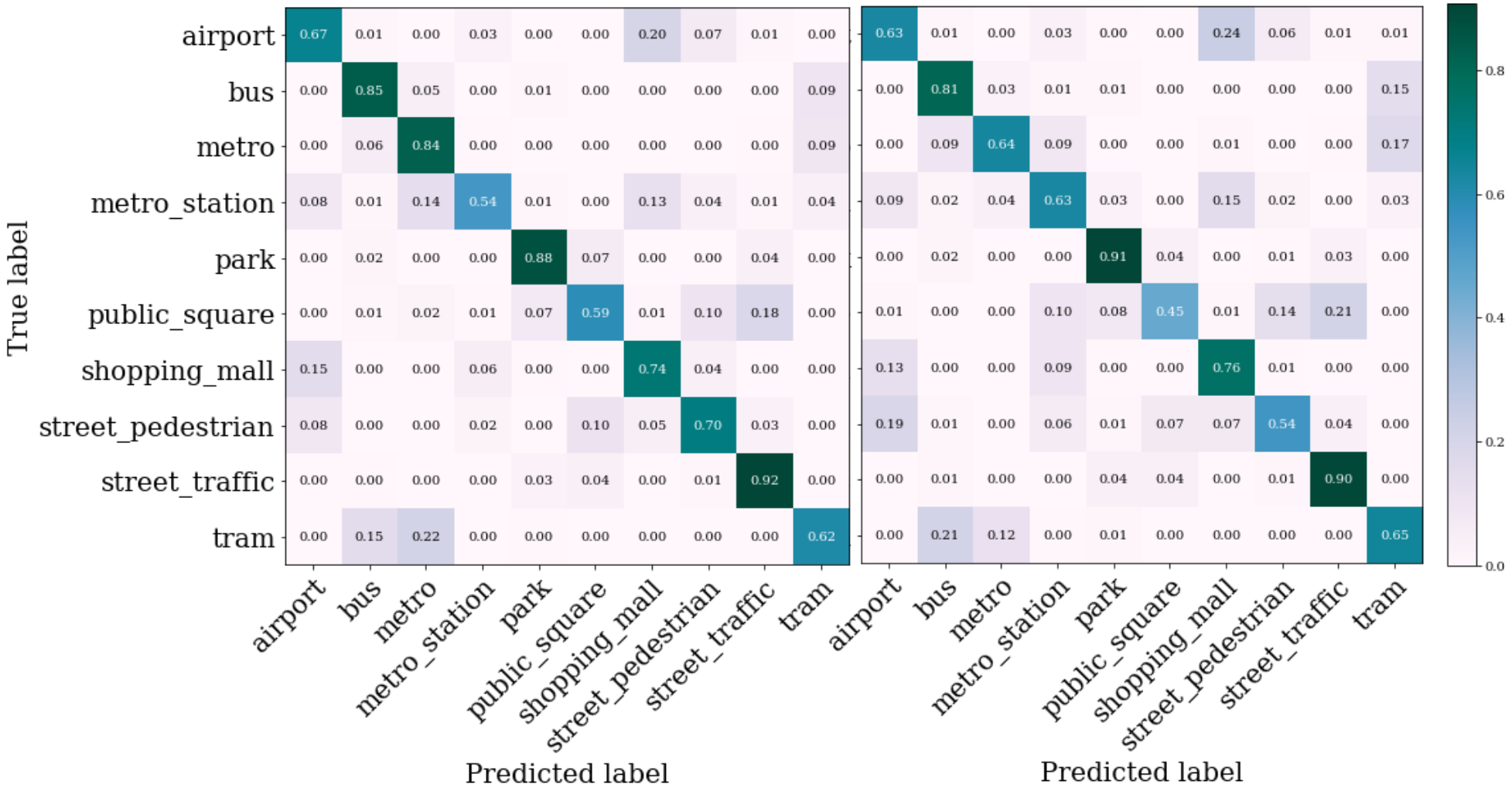
Table 4.3: Wilcoxon signed-rank test results.

		$\boldsymbol{Res_{k=1}+Mix}$			$\boldsymbol{Alex+Mix}$
$\boldsymbol{Res_{k=1}}$	$W/T/L$	10 / 0 / 0	$\boldsymbol{Alex}$	$W/T/L$	3 / 2 / 5
	$p-value$	**0.0051**		$p-value$	0.2845

The results shown above indicate that training ResNet on augmented data significantly improves the generalization ability with $p-value$ = 0.0051. Therefore, we can reject the null hypothesis regarding the ResNet-based system. However, introducing the mixup data augmentation technique does not demonstrate any improvement in the performance of AlexNet as depicted in Table 4.3 ($p-value$ = 0.2845). Hence, the observed differences between $Alex+Mix$ and $Alex$ are not significant. This behavior is expected since the mixup technique works better on models trained for longer runs [131]. Similar results have been reported in [189].

Confusion matrix of Resnet

In order to get a better insight on how mixup augmentation method improves the performance of Resnet-based model, we display in Figure 4.1 (a) and (b) the confusion matrices of $Res_{k=1}$ and $Res_{k=1}$+Mix, respectively, computed on the test set.

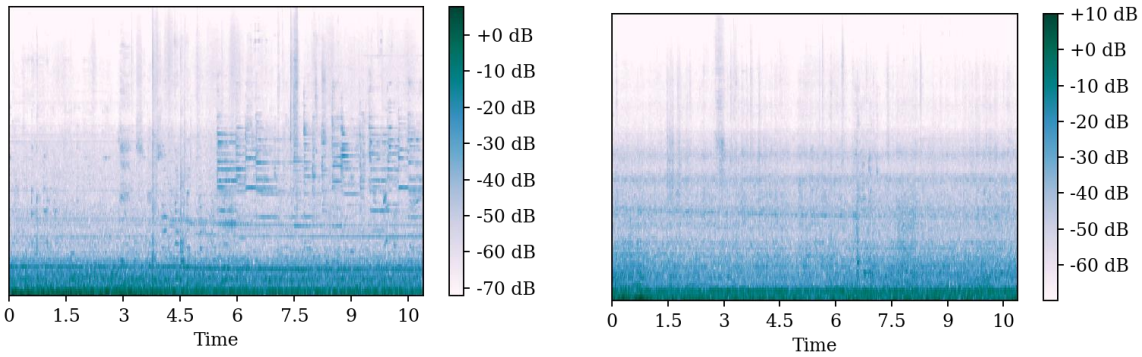


(a) $Res_{k=1}$.

(b) $Res_{k=1}$+Mix.

Figure 4.1: Confusion Matrices of ResNet-based models.

The matrix of $\boldsymbol{Res_{k=1}}$ shows an appreciable diagonal, meaning that several scenes are correctly classified. However, some classes are misrecognized as others; for instance, *"bus"* and *"metro"* are classified as *"tram"* and vice versa. It is worth noting that the mostly misclassified classes are quite **similar**, which makes it difficult even for a human being to distinguish between them. Figure 4.2 shows the Log-Mel spectrogram representation of 2 scenes *"tram"* and *"metro"*.



(a) Tram. (b) Metro.

Figure 4.2: Spectrogram representations.

As reported earlier, training ResNet on the augmented data has boosted its performance. Most importantly, we observe a considerable reduction in the misclassification rate of numerous scenes that sound similar, as shown in Figure 4.2.

The above findings further confirm the efficiency of the ResNet-based model for sound scene recognition. In Experiment 3, we thoroughly investigate the ResNet-based systems, while varying some of its hyperparameters and introducing the derivatives of Log-Mel features.

4.3 Experiment 2: Impact of the Mixup parameter

To further assess the influence of the mixup data augmentation technique, we have built several acoustic scene recognition systems on the augmented data, while varying the mixup parameter α. Recall that the mix factor value used for mixing 2 sound samples λ is randomly drawn following the Beta distribution *Be(a, b)*, with a = b = α.

A. AlexNet: We have performed this experiment with 8 different values of $\alpha \in \{0.1, 0.4, 0.7, 1, 4, 7, 10, 12\}$ (please refer to **Section 2.4.2.A** for additional details on this parameter).

Most importantly, we have tested the effect of setting a and b differently, a = 4, b = 7. Moreover, we have only considered training with the AlexNet model.

Table 4.4 gives the results of this experiment. The last row specifies the average ranks of each model computed using the Friedman test. The ranks of each system have been assigned according to the F1-Score performance (Table 4.4) in descending order. Most importantly, systems with lower ranks are preferred i.e. succeed at recognizing most scenes.

Table 4.4: AlexNet based models ranking according to average F1-score performance.

Classes	$Alex_{\alpha=0.1}$	$Alex_{\alpha=0.4}$	$Alex_{\alpha=0.7}$	$Alex_{\alpha=1}$	$Alex_{\alpha=4}$	$Alex_{\alpha=7}$	$Alex_{\alpha=10}$	$Alex_{\alpha=12}$	$Alex_{\alpha=4,b=7}$
Airport	0.71	0.72	0.69	0.69	**0.75**	0.74	0.69	0.69	0.72
Bus	0.73	**0.77**	0.75	0.74	0.72	0.70	0.69	0.69	**0.77**
Metro	0.61	0.61	**0.71**	0.66	0.66	0.57	0.62	0.62	0.65
Metro Station	0.58	0.61	**0.66**	0.62	0.65	0.64	0.61	0.61	0.61
Park	0.84	0.86	0.84	0.58	**0.87**	0.83	0.86	0.86	0.86
Public Square	0.50	0.58	**0.60**	0.58	0.56	0.53	0.54	0.54	0.52
Shopping Mall	0.63	0.61	0.64	0.63	0.65	0.62	0.59	0.59	**0.68**
Street Pedestrian	0.67	0.61	0.65	0.60	**0.69**	0.66	0.60	0.60	**0.69**
Street Traffic	**0.85**	0.83	0.82	0.80	**0.85**	0.80	0.81	0.81	0.83
Tram	0.65	0.67	0.65	0.68	0.68	0.68	0.64	0.64	**0.69**
Average F1-score	**0.68**	**0.68**	**0.70**	**0.69**	**0.71**	**0.68**	**0.69**	**0.69**	**0.70**
Average Rank	**5.75**	**4.65**	**3.95**	**5.35**	**2.45**	**5.75**	**6.85**	**6.85**	**3.40**

In order to study these results and reveal significant differences, we have first conducted the Friedman test, while assuming that the observed differences are due to random behavior. This test rejects our hypothesis with $FF = 3.97 > F(8,72) = 3.91$ for $\alpha = 0.0007$ (FF is distributed according to the F distribution with $9-1 = 8$ and $(9-1)\times(10-1) = 72$ degrees of freedom), which indicates an existence of at least one pairwise significant difference.

For further analysis of these results, we have compared these scores in a pairwise manner based on the Wilcoxon test in Table 4.5. The first row of each entry specifies the number of Win/Tie/Loss of the technique in the column over the technique in the row; whereas, the second row shows the p-values for the Wilcoxon test. If the entry is bold, this means that the number of wins/losses over 10 is statistically significant using the Wilcoxon test.

Table 4.5: Comparison of AlexNet based models in a pairwise manner based on the Wilcoxon test.

		$Alex_{\alpha=0.4}$	$Alex_{\alpha=0.7}$	$Alex_{\alpha=1}$	$Alex_{\alpha=4}$	$Alex_{\alpha=7}$	$Alex_{\alpha=10}$	$Alex_{\alpha=12}$	$Alex_{\alpha=4,b=7}$
$Alex_{\alpha=0.1}$	W/T/L	6/1/3	5/2/3	5/1/4	**8/1/1**	4/0/6	4/0/6	4/0/6	**9/0/1**
	p-value	0.35	0.35	0.95	**0.01**	0.91	0.30	0.30	**0.01**
$Alex_{\alpha=0.4}$	W/T/L		5/0/5	4/1/5	**8/0/2**	5/0/5	**1/2/7**	**1/2/7**	4/5/1
	p-value		0.26	0.47	**0.10**	0.44	**0.02**	**0.02**	0.22
$Alex_{\alpha=0.7}$	W/T/L			**1/1/8**	6/0/4	3/0/7	**1/1/8**	**1/1/8**	7/0/3
	p-value			**0.03**	0.57	0.20	**0.01**	**0.01**	0.95
$Alex_{\alpha=1}$	W/T/L				**6/2/2**	4/2/4	2/2/6	2/2/6	**7/0/3**
	p-value				**0.05**	0.75	0.20	0.20	**0.11**
$Alex_{\alpha=4}$	W/T/L					**0/1/9**	**0/0/10**	**0/0/10**	3/1/6
	p-value					**0.005**	**0.005**	**0.005**	0.44
$Alex_{\alpha=7}$	W/T/L						4/0/6	4/0/6	**7/0/3**
	p-value						0.33	0.33	**0.07**
$Alex_{\alpha=10}$	W/T/L							0/10/0	**7/2/1**
	p-value							1	**0.02**
$Alex_{\alpha=12}$	W/T/L								**7/0/3**
	p-value								**0.02**

The test results reveals the existence of four categories of models: C1={ $Alex_{\alpha=4}$, $Alex_{a=4,b=7}$}; C2={$Alex_{\alpha=0.7}$ }; C3={$Alex_{\alpha=0.4}$ }; and C4={$Alex_{\alpha=0.1}$, $Alex_{\alpha=1}$, $Alex_{\alpha=7}$, $Alex_{\alpha=10}$, $Alex_{\alpha=12}$}. The system trained with α = 4 achieve the best results with 6 significant wins and the system trained with *a = 4, b = 7* achieve 5 significant wins , followed by $Alex_{\alpha=0.7}$ and $Alex_{\alpha=0.4}$ with 3 and 2 significant wins respectively, whereas, $Alex_{\alpha=0.1}$, $Alex_{\alpha=1}$, $Alex_{\alpha=7}$, $Alex_{\alpha=10}$ and $Alex_{\alpha=12}$ yield the worst scores with 0 significant wins. According to the Wilcoxon test statistics, we can derive the following inferences:

- We find that training with α = 4 significantly outperforms categories C3 and C4 at $0.0051 \leq p - value \leq 0.1$ (bold entry of Table 4.5).
- We could not reject the null hypothesis stating that $Alex_{\alpha=0.4}$ and $Alex_{\alpha=0.7}$ are equivalent i.e. the observed differences are due to chance. Therefore, the values α = 0.4 and α = 0.7 yield systems which have similar performances.
- We observe that systems belonging to C4 are significantly worse than the other alternatives, as indicated by the Wilcoxon test results.

Based on the above discussion, we can conclude that correctly setting the value of α parameter is of vital importance for the success of the mixup technique. Recall that the mix factor used for mixing 2 instances follows a *Be(a, b)* distribution, such that a=b= α.We can distinguish 2 cases:

i. **α < 0.4**: For these values, the beta distribution has the shape of the U-shaped bimodal distribution (Figure 2.5(b)). It samples more values closer to either 0 and 1, generating more redundant samples. This leads to less mixup effect, hence, low generalization ability. Similar results have been reported in [131].

ii. **α > 4**: In this case, the beta distribution is the same as the gaussian distribution (Figure 2.5(a)). It focuses on values that are neither close to 0 nor to 1, creating a grand mixup effect. This latter usually leads to underfitting as reported by several experimental investigations [159].

One should avoid the configurations indicated by (i) and (ii), and set the value of α for AlexNet between these two extreme cases.

B. ResNet: We have performed this experiment with 6 different values of $\alpha \in \{0.1, 0.4, 0.7, 1, 4, 7\}$. Moreover, we have only considered training with the $Res_{k=1}$ model. Table 4.6 gives the results of this experiment. The last row specifies the average ranks of each model computed using the Friedman test.

Table 4.6: ResNet of width 1 based models ranking according to average F1-scores.

Classes	**$Res_{\alpha=0.1}$**	**$Res_{\alpha=0.4}$**	**$Res_{\alpha=0.7}$**	**$Res_{\alpha=1}$**	**$Res_{\alpha=4}$**	**$Res_{\alpha=7}$**
Airport	0.64	**0.67**	0.66	0.51	0	0.12
Bus	0.76	**0.79**	0.76	0.69	0.23	0.28
Metro	**0.73**	**0.73**	0.67	0.58	0	0
Metro Station	0.60	**0.65**	**0.65**	0.56	0	0
Park	0.86	**0.87**	0.83	0.80	0.75	0.55
Public Square	0.61	**0.65**	0.54	0.49	0	0
Shopping Mall	0.68	**0.70**	0.61	0.63	0	0
Street Pedestrian	0.62	**0.71**	0.48	0.54	0	0
Street Traffic	**0.84**	0.83	0.80	0.76	0.05	0.57
Tram	0.67	0.67	**0.69**	0.57	0.04	0.19
Average F1-score	**0.70**	**0.73**	**0.67**	**0.61**	**0.10**	**0.17**
Average Ranks	**2.15**	**1.35**	**2.70**	**3.80**	**5.75**	**5.25**

The results show that systems trained with α < 1 surpasses those trained with α > 1. In addition, Friedman's test rejects the hypothesis that all algorithms perform equally with FF=62.59. With 6 algorithms and 10 classes, FF is distributed according to the F distribution with 6−1 = 5 and (6−1) × (10−1) = 45 degrees of freedom. The critical value of F (5,45) for α = 1×10^{-16} is 46.55, and since FF = 62.59 > F (5,45) = 46.55 we reject the null-hypothesis. This finding confirms the existence of at least one pair of systems with significant different performances. Then, we have proceeded with a post hoc Nemenyi test at a 5% significance level with the critical value $q_{0.05}$ = 2.84 and the critical difference CD = 2.38. This test aims at identifying pairs of algorithms that are significantly different. The results of the Nemenyi test are depicted in Figure 4.3. On the horizontal axis, we represent the average ranks of each method (given in Table 4.6), and join the groups of systems that are not significantly different using thick lines. On the top left, we display the critical difference CD used in our experiments.

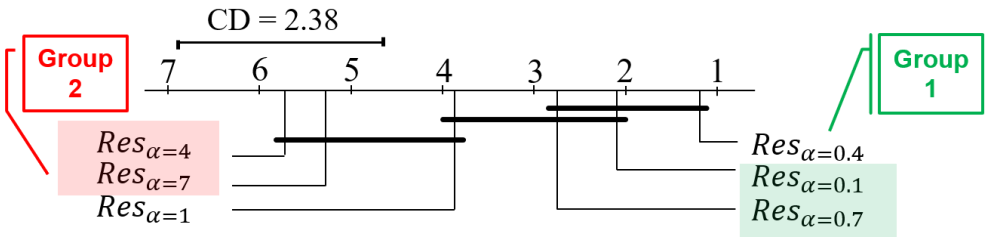


Figure 4.3: Comparison of ResNet of width 1 based systems against each other with the Nemenyi test.

We can identify two groups of systems: the ones trained using small values of α (α < 1) and those built with larger values of α (α > 1). Notice that systems within the same group achieve similar performances, and the observed differences are solely due to chance. Most importantly, the test provides a strong evidence that training models with α < 1 yields significantly better results than α > 1, which confirms our initial assumption. In addition, $Res_{\alpha=1}$ performs significantly worse than $Res_{\alpha=0.4}$. However, it achieves similar performance to $Res_{\alpha=0.1}$, $Res_{\alpha=0.7}$ (group 1), and to $Res_{\alpha=4}$, $Res_{\alpha=7}$ (group 2). Therefore, we cannot tell which group the $Res_{\alpha=1}$ belongs to, i.e. no conclusions can be drawn regarding $Res_{\alpha=1}$ due to the lack of the experimental data.

Based on the above analysis, in the case of the ResNet model, setting α between 0.1 and 1 (excluded) is preferred over the other two extremes.

4.4 Experiment 3: Analysis of ResNet-based system

According to our experiment 1, $Res_{k=1} + Mix$ has provided the best results in terms of F1-scores and architecture complexity. In order to further explore the ResNet-based system efficiency in recognizing sound scenes, we have built numerous ResNet models, obtained by varying some hyperparameters and the mechanism used for extracting features. Because we are only interested in comparing $Res_{k=1}$ against the other alternatives, we have conducted the following experiment, while considering the $Res_{k=1}$ as our **baseline** system. Specifically, we consider two main changes: **(i)** the width of the ResNet model $k \in \{1,2\}$, **(ii)** the feature set used for learning, namely: Log-Mel and Log-Mel+ΔLog-Mel+ΔΔLog-Mel. The resulting four models have been trained twice: with/without the mixup technique. We report in Table 4.7 the F1-score of the 7 variants of the original system. Note that we also give in the last column of Table 4.7 the performance results of our baseline system as a reference. Moreover, the last row of Table 4.7 specifies the average ranks of each model computed using the Friedman test.

Table 4.7: Average Ranks and F1-scores of ResNet based models.

Classes	$\boldsymbol{Res_{k=1} + Mix}$	$\boldsymbol{Res_{k=1}}$ + ΔΔ	$\boldsymbol{Res_{k=1}}$ + ΔΔ + $\boldsymbol{Mix}$	$\boldsymbol{Res_{k=2}}$	$\boldsymbol{Res_{k=2}} + \boldsymbol{Mix}$	$\boldsymbol{Res_{k=2}}$ + ΔΔ	$\boldsymbol{Res_{k=2}}$ + ΔΔ + $\boldsymbol{Mix}$	$\boldsymbol{Res_{k=1}}$
Airport	0.67	0	**0.70**	0.05	0.49	0.16	0.67	0.61
Bus	0.79	0.13	0.75	0.02	0.60	0	**0.81**	0.73
Metro	0.73	0.24	0.72	0.32	0.38	0.41	**0.77**	0.69
Metro Station	0.65	0	0.68	0.02	0.45	0.03	**0.79**	0.62
Park	0.87	0.77	**0.91**	0.64	0.65	0.59	**0.91**	0.86
Public Square	0.65	0.21	0.62	0.01	0.32	0.28	**0.69**	0.56
Shopping Mall	0.70	0	0.72	0.47	0.56	0.09	**0.76**	0.68
Street Pedestrian	**0.71**	0.03	0.69	0.03	0.50	0.07	**0.71**	0.60
Street Traffic	0.83	0.61	0.86	0.38	0.77	0.52	**0.87**	0.81
Tram	0.67	0.27	0.68	0.40	0.12	0.40	**0.74**	0.65
Average F1-score	**0.73**	**0.22**	**0.73**	**0.23**	**0.48**	**0.25**	**0.77**	**0.68**
Average Ranks	**2.49**	**7.05**	**2.25**	**6.99**	**5.50**	**6.44**	**1.25**	**3.99**

Friedman test reject the hypothesis that the 7 variants of our baseline system $Res_{k=1}$ perform similarly, as illustrated in Table 4.6. The rejection of this hypothesis confirms

the existence of **at least one pair** of systems with significantly different performances. Because we are only interested in testing whether the variants of $Res_{k=1}$ significantly improve the performance of the baseline system we have conducted a Bonferroni-Dunn test at a 10% significance level with the critical value $q_{0.10}$ = 2.45 and the critical difference CD = 2.68. The results of this test are depicted by Figure 4.1. On the horizontal axis, we represent the averaged rank of each system given in the last row of Table 4.7, and mark using a thick line an interval of 2×CD one on the right and the other to the left of $Res_{k=1}$ mean rank.

Friedman test reject the hypothesis that the 7 variants of our baseline system $Res_{k=1}$ perform similarly, as illustrated in Table 4.7. The rejection of this hypothesis confirms the existence of **at least one pair** of systems with significantly different performances. Because we are only interested in testing whether the variants of $Res_{k=1}$ significantly improve the performance of the baseline system we have conducted a Bonferroni-Dunn test at a 10% significance level with the critical value $q_{0.10}$ = 2.45 and the critical difference CD = 2.68. The results of this test are depicted by Figure 4.4. On the horizontal axis, we represent the averaged rank of each system given in the column 1 of Table 4.7, and mark using a thick line an interval of 2×CD one on the right and the other to the left of $Res_{k=1}$ mean rank.

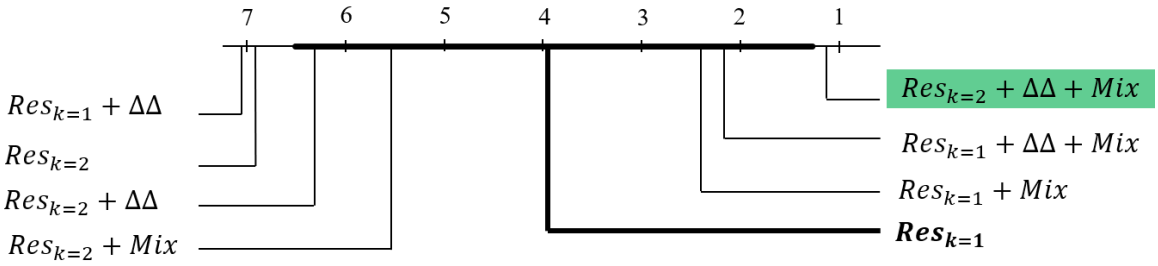


Figure 4.4: Comparison of the baseline system against the 7 other systems with the Bonferroni-Dunn test.

From the analysis of the Bonferroni-Dunn test results shown above, we can derive the following insights:

- $Res_{k=2} + \Delta\Delta + Mix$ falls outside the marked interval and has the lowest rank; hence, it significantly improves the baseline system and outperforms the other alternatives. We can conclude that training a wide ResNet on additional features with the mixup technique-activated significantly boosts the pure ResNet model. However, introducing ΔLog-Mel + ΔΔLog-Mel features without data augmentation ($Res_{k=2} + \Delta\Delta$) does not demonstrate any positive impact on our baseline system. This is rather an expected result, considering that wider networks

require more data to decrease the error rates [190]. In addition, $Res_{k=2}$ and $Res_{k=2} + Mix$ are significantly worse than $Res_{k=2} + \Delta\Delta + Mix$. To put it simply, $Res_{k=2}$ works well when trained on Log-Mel+ΔLog-Mel + ΔΔLog-Mel augmented features.

- Training $Res_{k=1}$ on augmented data with or without the derivative features ($Res_{k=1} + \Delta\Delta + Mix$ and $Res_{k=1} + Mix$) considerably improves the performance of the pure model. However, learning from the additional features without augmentation ($Res_{k=1} + \Delta\Delta$) **significantly** deteriorates the generalization ability. We believe that this behavior occurs because adding ΔLog-Mel + ΔΔLog-Mel features increases the dimension of our dataset, which leads to overfitting, hence, a decrease in the performance. It is widely acknowledged that, in case of high dimensional data with low number of samples, deep neural networks surfers from overfitting [191].

4.5 Experiment 4: Effect of the number of epochs on AlexNet performance

Based on the results of the first experiment, we have noticed that training AlexNet on augmented data did not demonstrate any improvement in the performance. Note that several studies have indicated that the mixup technique works better on models trained for longer runs [131]. To further investigate this matter, we have carried out the following experiment. We have varied the number of epochs from 1 to 100 and measured the F1-Score on the test set. We have performed this experiment with two different mixup values: α = 0.4 and α = 4. Figure 4.5 illustrates the results of this experiment.

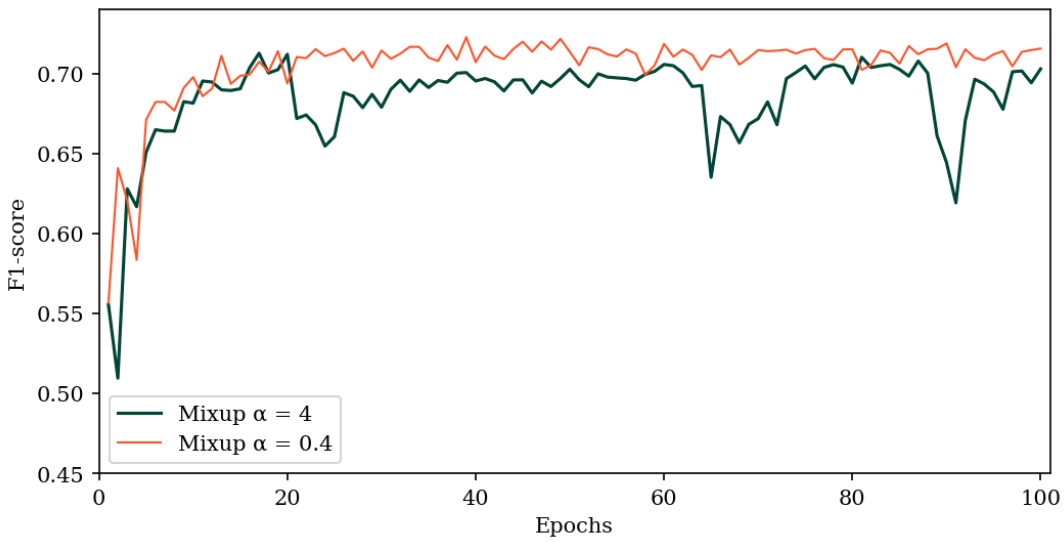


Figure 4.5: Effect of mixup technique on the Alexnet-based model.

The analysis of the curve results reported by Figure 4.5 can be summarized as follows. The performance improves as the number of epochs increases; then, it settles after a certain value (~ 20 epochs), and keeps this F1-score with some variations. This behavior coincides with our initial observation: the effect of mixup appears after longer runs i.e. higher number of epochs. We also notice that $Alex_{\alpha=4}$ is less stable than its counterpart $Alex_{\alpha=0.4}$. This latter confirms our results reported in Experiment 2. As indicated earlier, setting α between 0.4 and 4 yields better scores. Specifically, the higher boundary of this interval (values close to 4) creates more mixup effect than the lower boundary (values close to 0.4). Therefore, values of α that are close to 4 are more prone to underfitting than those that are close to 0.4. Combining these findings with the results of Experiment 2 enables us to refine the interval of α values: α ~ 0.4 are preferable.

4.6 Summary of Empirical Findings

In this chapter, we have presented the results of our experimental enquiries. Several lessons can be derived from our analysis:

- Systems trained on the augmented data have demonstrated superiority over those trained on non-augmented data.
- Mixup technique works better when the models are trained for longer runs, i.e. higher number of epochs.
- The mixup parameter α is of paramount importance and should be defined carefully. As α increases (α > 1), the trained models are prone to underfit, leading to a degradation of the generalization ability. Therefore, values larger than 1 should be avoided. We recommend choosing values of α smaller than 1, while taking into consideration the number of epochs for the model training.
- Systems trained on additional features (Δ Log-Mel + ΔΔ Log-Mel) without data augmentation considerably deteriorates the performance of the ResNet model. Therefore, the mixup demonstrates a positive impact when learning from high dimensional data with low number of samples, which frequently occurs in the case of Acoustic Scene Recognition.
- Statistical testing is a powerful mechanism for unraveling existing differences among acoustic scene classification systems. Therefore, it can be used for the analysis and the selection of the best model for the problem at hand.

CONCLUSION

1. Summary of results

In this thesis, we tackled the area of acoustic scene recognition of environmental sounds. The main objective behind our endeavor was to design and analyze acoustic scene classification systems able to recognize and classify acoustic scenes. To this end, we built several sound recognition systems and conducted multiple experiments to analyze the behavior of the latter systems.

The experiments have been conducted on the TAU Urban Acoustic Scenes 2019 dataset; an audio database collected from real life contexts which contains 10 diverse indoor and outdoor locations (labels), totaling in 40 hours of recording and 35.6GB in wav format. In this work, we made use of two deep learning-based models: Residual Neural Network (ResNet) and Alex Neural Network (AlexNet). We trained the latter models on 3 different feature sets based on signal processing methods namely: Log-Mel energies, ΔLog-Mel energies and ΔΔLog-Mel energies. To help address overfitting, we opted for artificially enlarging the dataset through creating new training sample by making small changes to the original data while keeping its characteristics i.e. data augmentation. Concretely, we used a sample mixed-based data augmentation method named mixup. It is implemented by creating virtual feature-target pairs through linearly combining two randomly chosen feature vectors. To evaluate each developed system, we favored the F1-score metric over the accuracy metric as the former takes into account both false positives and false negatives. Finally, we endorsed our discussion with various powerful statistical tests namely: Friedman test, Wilcoxon signed rank test, Nemenyi test and Bonferroni Dunn test. The analysis of our experiments indicate the following:

- Getting good results from a recognition system is fundamentally based on the selection of relevant features and the suitable classifier.
- Data augmentation effectively improves the generalization ability of both ResNet and AlexNet-based systems and greatly reduces the chances of overfitting. Concretely, Mixup data augmentation technique provided interesting results despite its simplicity and minimal computation overhead.

- The mixup parameter α is of critical importance to the performance of the trained model and should be defined carefully. We found that as α increases (α > 1), the trained models are prone to underfit, leading to a degradation of the generalization ability. Therefore, values larger than 1 should be avoided. We highly suggest choosing values of α smaller than 1, while taking into consideration the number of epochs for the model training. Moreover, systems utilizing mixup-augmented data should be trained for longer runs i.e. higher number of epochs in order to obtain favorable results.
- Training deep learning models on additional features (ΔLog-Mel + ΔΔLog-Mel) without the use of data augmentation considerably deteriorates the performance of the ResNet model. Therefore, the mixup demonstrates a positive impact when learning from high dimensional data with low number of samples, which frequently occurs in the case of Acoustic Scene Recognition.
- Statistically testing the obtained scores is a powerful mechanism for comparing and unraveling existing differences among acoustic scene classification systems.

2. Potential directions and future work

The work presented in this thesis adds to a growing corpus of research showing the vital importance of wisely choosing the various hyperparameters in a machine learning project. Based on the insights gained from the experimental findings, we have concluded that the proper choice of feature sets for training can greatly affect the performance of ASC systems. Regardless, future investigations are necessary to validate the kinds of conclusions that can be drawn from this study. In addition, our findings on the use of mixup provide additional information about the capability of this data augmentation method in alleviating the problem of overfitting during training. Future research should further develop and confirm these initial findings by exploring the effects of various other values of the hyperparameter α.

Another appealing work direction would be to combine the mixup data augmentation technique with multiple other data augmentation techniques and analyze how it affects the performance of an ASC system. During this work we have used a high-quality audio dataset specifically collected for machine learning purposes. However, deployment of machine learning models on smart devices or systems are expected to correctly classify sounds detected via sensors or microphones which do not necessarily detect sounds in high quality. Therefore, future research should be conducted in more realistic settings by making use of audio data sets which contain sounds recorded using different devices.

Appendix A

Design and Implementation of On-line Audio Recognition Demonstration

A.1 Context and intentions

Online demonstrations primarily act as a bridge between two parties: the party who wishes to share information and the party who wants to consume it. In order to bring forward the findings uncovered throughout our work, we have decided to create a static web interface. A static website can be defined as a collection of several web pages that are all related to each other and can be accessed by visiting a homepage via a browser.

A.2 Implementation process and setup

Following a discovery and planning step we have opted to design a one-page website, also known as a brochure website, to display the different aspects of our research work. Our web interface consists of a collection of items like text, images and multimedia elements created with Hyper Text Markup Language (HTML) and styled using Cascading Style Sheets (CSS). We have implemented and tested the web interface on a windows operating system using Google Chrome as a navigator. All code for the creation of our demonstration has been written and edited using Atom text editor.

A.3 Web interface walkthrough

Our demonstration is a one-page web interface containing several full-screen sections. The **first** section of our demonstration is a landing page consisting of a svg image on the right side of the screen and a text section on the left. We have included logos of the various programming languages used to create the interface. The textual part on the left side of the page describes the acoustic scene classification task and its applications. To move to the next section of the web page, a click on the “Continue Reading” button is required. The following image represents the landing page.

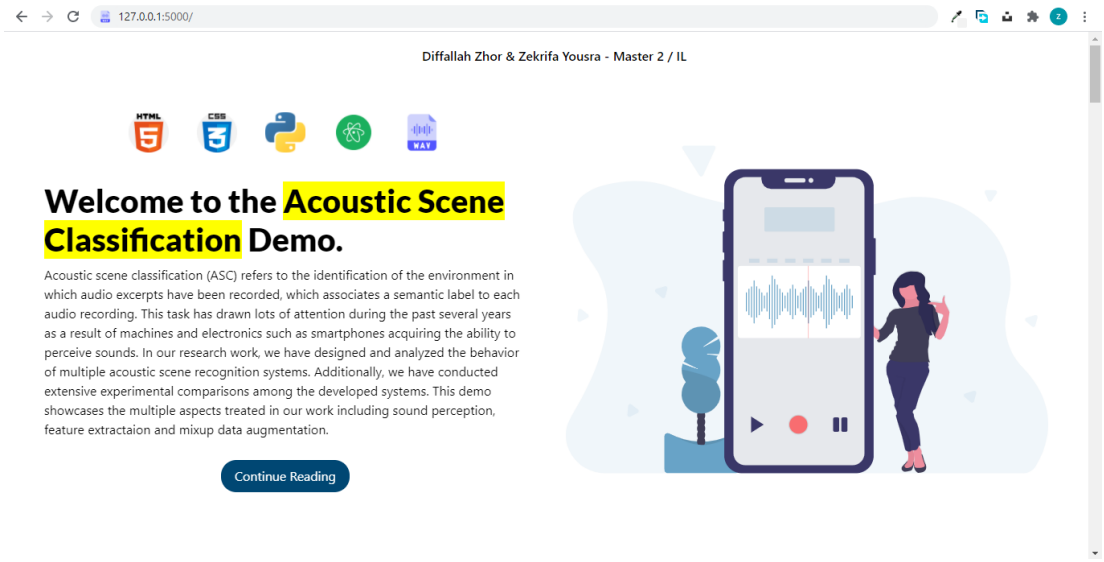


Figure A.1: Section 1 of acoustic scene classification demo.

The **second** section of our demonstration provides a brief description of the major steps usually followed in the design of an acoustic scene classification system as shown in Figure A.2. It includes 3 boxes consisting of mainly text which highlights each step. Clicking on the "Continue Button" moves the user to the next section of the web page.

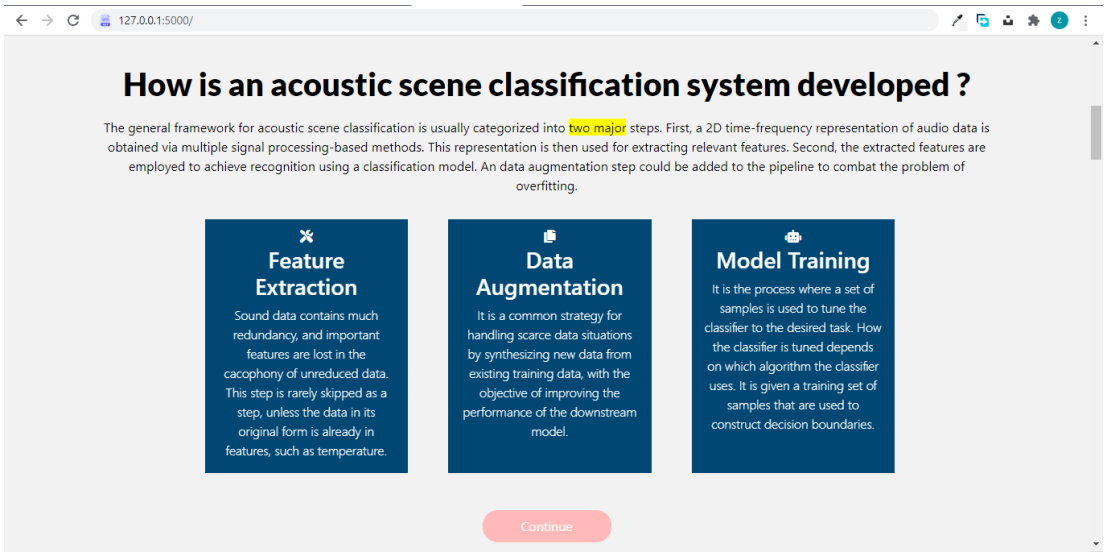


Figure A.2: Section 2 of acoustic scene classification demo.

The following section is an in-depth description of the way a human auditory system perceives sounds on a daily basis. The text area is enhanced with the use of an audio player element, which links to a 3 second recording of an ocean wave. This multimedia element is used to play the stored sound by clicking on the start button.

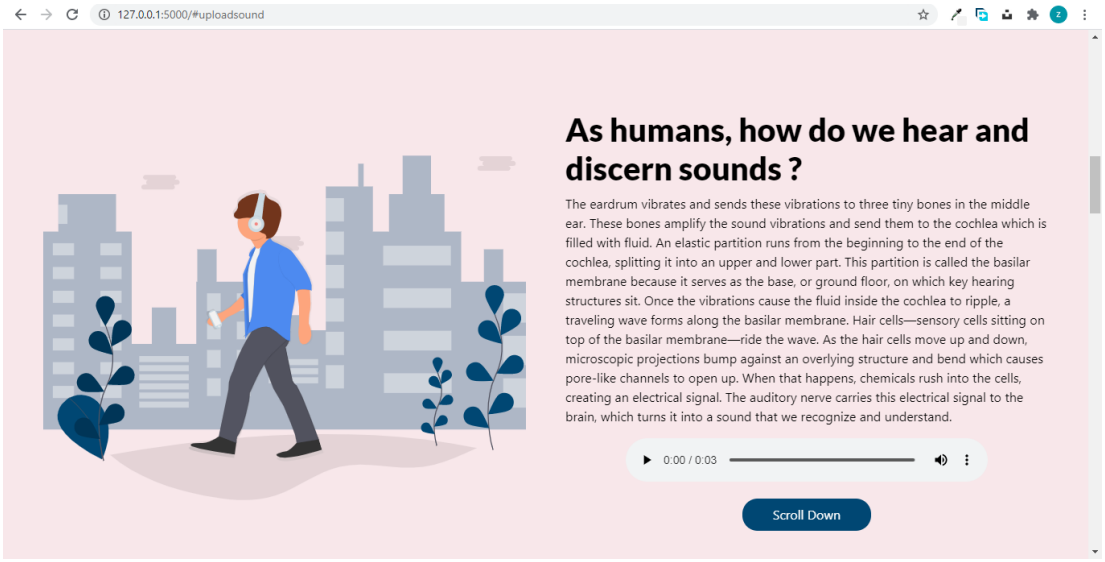


Figure A.3: Section 3 of acoustic scene classification demo.

The fourth section of our one-page website demonstration includes another audio player which links back to a ***travelling by tram*** sound recording retrieved from the dataset used to conduct our experiments (please refer to **Section 3.2** for further details on the dataset).

We then portray on this section various feature representations of the audio recording namely: time-domain representation, Mel-spectrogram and $\Delta\Delta$ Mel-spectrogram (as shown in Figure A.4).

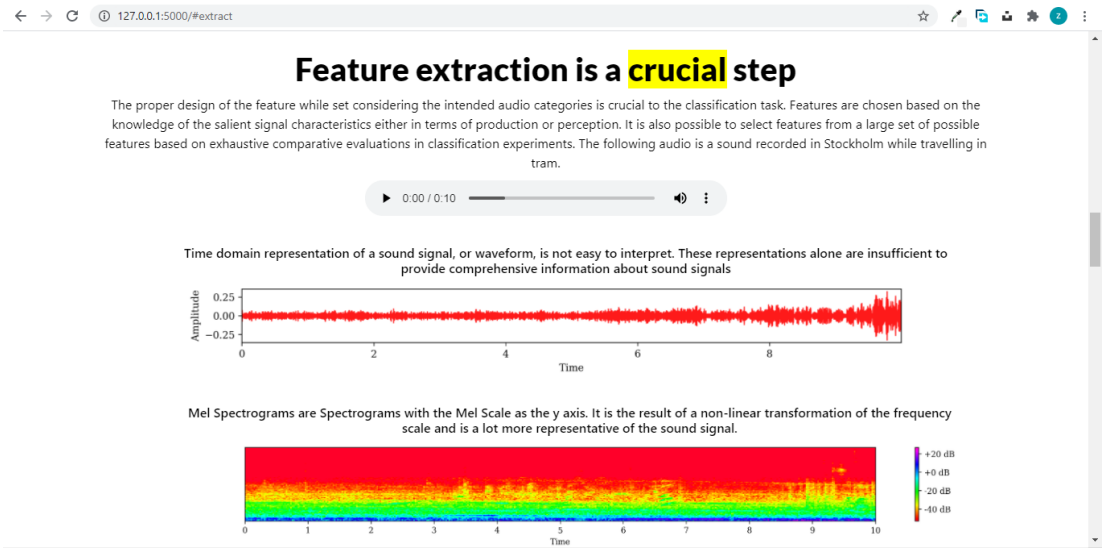


Figure A.4: Section 4 of acoustic scene classification demo.

In the following section, we provide the results of the prediction of two different models trained during our experiments. We have fed the ***travelling by tram*** sound recording from the previous section through two saved ResNet-based models: one model has been trained on Log-Mel energies features only **($Res_{k=1}$)** and the second one has been trained on Log-Mel energies + ΔLog-Mel energies + $\Delta\Delta$Log-Mel energies **($Res_{k=1} + \Delta\Delta$)**. The results of these predictions are indicated in Figure A.5: the ResNet-based model trained only using Log-Mel energies predicted that the recording is the sound of a ***metro station*** (prediction on the left of the web page). However, the ResNet system trained using Log-Mel energies + ΔLog-Mel energies + $\Delta\Delta$Log-Mel energies predicted that the recording is indeed the sound of ***travelling by tram*** (prediction on the right of the web page).

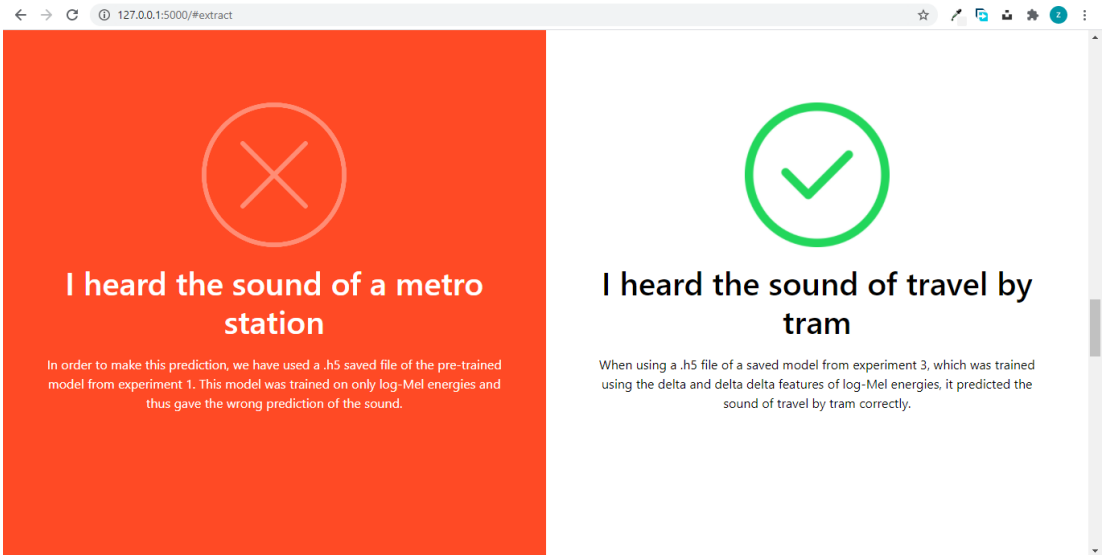


Figure A.5: Section 5 of acoustic scene classification demo.

Following this section, we showcase the auditory effects of the mixup data augmentation. We have chosen two different sound recordings: an ocean wave sound and a siren sound. Next, we have applied mixup by linearly combining the previous sounds with a mix factor $\lambda = 0.07$. The following equation describes how the obtained sound has been created:

$$New\ Sample = 0.07(Siren) + (1 - \ 0.07)(Ocean\ wave).$$

We have implemented a small program which allows us to create the new sample and generate a readable .wav file. When playing the generated sound file using the multimedia player we can hear two sounds at different ratios: the sound of the ocean wave is dominant as it has been mixed at 93% with the sound of the siren at 7%.

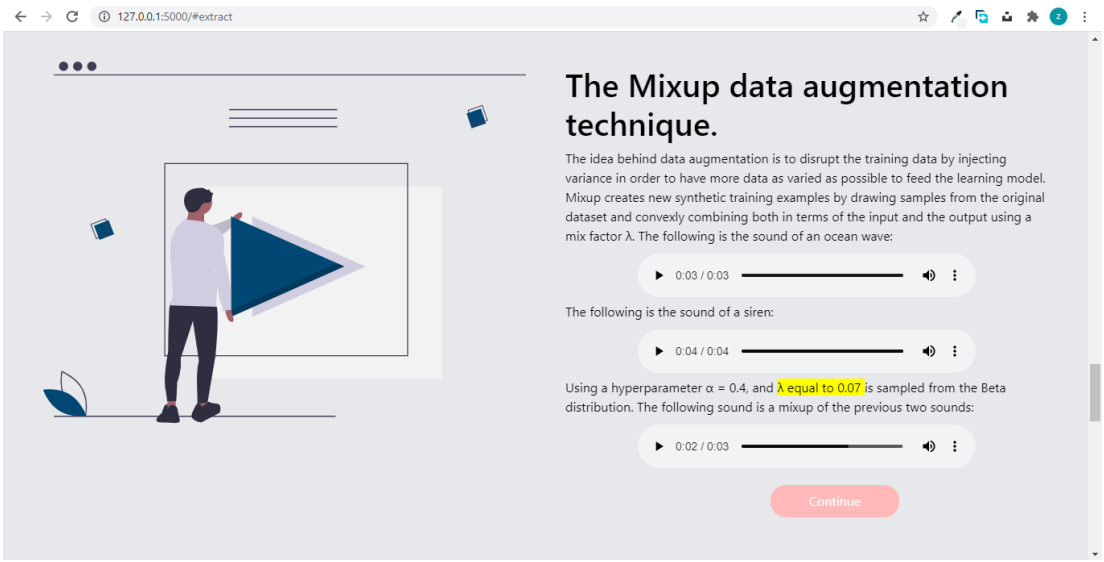


Figure A.6: Section 6 of acoustic scene classification demo.

Finally, we have chosen one of the most performant ResNet-based models from our case study: $Res_{k=1} + Mix$ and have tested it using a sound which does not belong to the dataset used in our experiments. This was achieved after training the system with the same settings using the entire dataset. The results of the prediction as well as details of the external sound are represented on this section of the webpage as illustrated in Figure A.7.

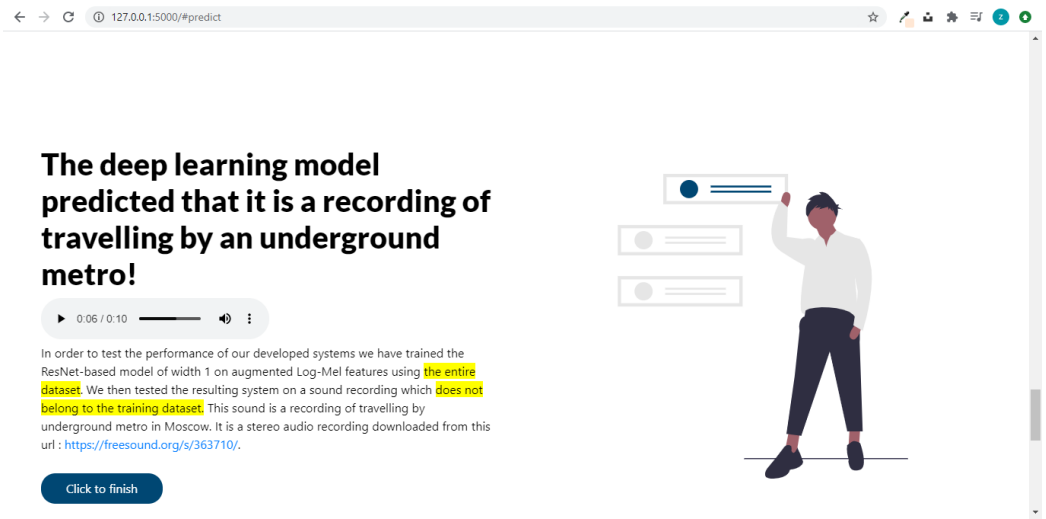


Figure A.7: Section 7 of acoustic scene classification demo.

REFERENCES

[1] W. Dai, “Acoustic Scene Recognition with Deep Learning,” pp. 1–18, 2016.

[2] D. Gerhard, “Audio Signal Classification : History and Current Techniques,” *Dep. Comput. Sci. Univ. Regina Regina, Saskatchewan, CANADA*, pp. 0–37, 2003.

[3] I. Lezhenin, N. Bogach, and E. Pyshkin, *Urban Sound Classification using Long Short-Term Memory Neural Network*. 2019.

[4] I. Mcloughlin, H. Zhang, Z. Xie, Y. Song, W. Xiao, and H. Phan, “Continuous robust sound event classification using time-frequency features and deep learning,” *PLoS One*, vol. 12, p. e0182309, Sep. 2017, doi: 10.1371/journal.pone.0182309.

[5] J. P. Dominguez-Morales, A. Jiménez-Fernandez, M. Dominguez-Morales, and G. Jimenez, “Deep Neural Networks for the Recognition and Classification of Heart Murmurs Using Neuromorphic Auditory Sensors,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, pp. 24–34, Feb. 2018, doi: 10.1109/TBCAS.2017.2751545.

[6] A. J. Eronen *et al.*, “Audio-based context recognition,” *IEEE Trans. Audio, Speech Lang. Process.*, vol. 14, no. 1, pp. 321–329, 2006, doi: 10.1109/TSA.2005.854103.

[7] N. Dey, A. S. Ashour, and N. Nhu, “Deep Learning for Multimedia Content Analysis,” 2016.

[8] A. Jiménez, B. Elizalde, and B. Raj, “Sound event classification using ontology-based neural networks,” *Conf. Neural Inf. Process. Syst.*, no. NeurIPS, 2018, [Online]. Available: https://openreview.net/pdf?id=HkGv2NMTjQ.

[9] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Neural Inf. Process. Syst.*, vol. 25, Jan. 2012, doi: 10.1145/3065386.

[10] S. H. Mneney, *An Introduction to Digital Signal Processing: A Focus on Implementation*, vol. 1. 2009.

[11] T. Giannakopoulos and A. Pikrakis, *Introduction to Audio Analysis: A MATLAB Approach*. Elsevier Ltd, 2014.

[12] A. B. Downey, *Think DSP: Digital Signal Processing in Python*, Green Tea. Needham, Massachusetts, 2014.

[13] J. M. Hillenbrand, “The Physics of Sound Sound and Vibration,” pp. 1–44, 2016.

[14] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer International Publishing, 2017.

[15] P. Christensson, "Sampling Definition," *january 8*, 2016. https://techterms.com (accessed Jun. 11, 2020).

[16] R. N. Mutagi, "Understanding the sampling process," *RF Des.*, no. September, pp. 38–48, 2004.

[17] E. H. Mamdani, "Science of Information," *J. Inf. Technol.*, vol. 1, no. 4, pp. 6–32, 1986, doi: 10.1057/jit.1986.34.

[18] M. Handley, "2 : Audio Basics Simple Analog-to-Digital Converter," *Audio*, pp. 1–17.

[19] J. S. G. Ronald J. Compesi, *Introduction to Video Production: Studio, Field, and Beyond*, 2nd, illustr ed. 2017.

[20] J. Belleman, "From analog to digital," *CERN*, 2008, doi: 10.5170/CERN-2008-003.131.

[21] E. Sejdić, I. Djurović, and J. Jiang, "Time-frequency feature representation using energy concentration: An overview of recent advances," *Digit. Signal Process. A Rev. J.*, vol. 19, no. 1, pp. 153–183, 2009, doi: 10.1016/j.dsp.2007.12.004.

[22] J. Schr *et al.*, "ON THE USE OF SPECTRO-TEMPORAL FEATURES FOR THE IEEE AASP CHALLENGE ' DETECTION AND CLASSIFICATION OF ACOUSTIC SCENES AND EVENTS ' e Sch ¨ J ¨ orn Anem ¨ Fraunhofer IDMT , Project Group Hearing , Speech and Audio Technology , Oldenburg , Germany Universit," pp. 2–5, 2013.

[23] R. Serizel *et al.*, "Acoustic Features for Environmental Sound Analysis To cite this version : HAL Id : hal-01575619 Chapter 4 : Acoustic Features for Environmental Sound Analysis .," 2017.

[24] R. B. H. S. S. Troncy, *Multimedia semantics: metadata, analysis and interaction*, 1st ed. Wiley-Blackwell, 2011.

[25] D. Grissa, M. Pétéra, M. Brandolini, A. Napoli, B. Comte, and E. Pujos-Guillot, "Feature selection methods for early predictive biomarker discovery using untargeted metabolomic data," *Front. Mol. Biosci.*, vol. 3, no. JUL, 2016, doi: 10.3389/fmolb.2016.00030.

[26] F. Alías, J. C. Socoró, and X. Sevillano, "A review of physical and perceptual feature extraction techniques for speech, music and environmental sounds," *Appl. Sci.*, vol. 6, no. 5, pp. 0–44, 2016, doi: 10.3390/app6050143.

[27] J. Chaki, "Pattern analysis based acoustic signal processing: a survey of the state-of-art,"

Int J Speech Technol, 2020, doi: https://doi.org/10.1007/s10772-020-09681-3.

[28] *Speech , Audio , Image and Biomedical Signal Processing using Neural Networks Studies in Computational Intelligence , Volume 83*. 2008.

[29] R. M. Aarts, "Audio signal processing device," *J. Acoust. Soc. Am.*, vol. 120, no. 6, p. 3445, 2006, doi: 10.1121/1.2409418.

[30] R. N. Rajiv Padhye, *Acoustic Textiles Textile Science and Clothing Technology*, Illustrate. 2016.

[31] J. Wolfe, "What is a Sound Spectrum?," *UNSW*, 2005. https://newt.phys.unsw.edu.au/ (accessed Jun. 13, 2020).

[32] Susan Maclagan, *A Dictionary for the Modern Flutist*. Scarecrow Press, 2009.

[33] M. M. Morgan *et al.*, "Spectrograms," *Encycl. Psychopharmacol.*, pp. 1270–1270, 2010, doi: 10.1007/978-3-540-68706-1_1219.

[34] M. Dörfler, T. Grill, R. Bammer, and A. Flexer, "Basic filters for convolutional neural networks applied to music: Training or design?," *Neural Comput. Appl.*, vol. 32, no. 4, pp. 941–954, 2020, doi: 10.1007/s00521-018-3704-x.

[35] T. Virtanen *et al.*, "Time-frequency processing - Spectral properties To cite this version : HAL Id : hal-01881426," 2018.

[36] Tom Bäckström, "Introduction to Speech Processing," *Alto University Wiki*, 2019. .

[37] G. Richard, S. Sundaram, and S. Narayanan, "An overview on perceptually motivated audio indexing and classification," *Proc. IEEE*, vol. 101, no. 9, pp. 1939–1954, 2013, doi: 10.1109/JPROC.2013.2251591.

[38] S. W. Smith, "Chp 22 Audio Processing," in *Digital Signal Processing*, no. 1, 2003, pp. 351–372.

[39] D. Zekrifa, *Intelligent Systems: Current Progress*. 2017.

[40] P. Dönmez, "Introduction to Machine Learning, 2nd ed., by Ethem Alpaydın. Cambridge, MA: The MIT Press2010. ISBN: 978-0-262-01243-0. $54/£ 39.95 + 584 pages.," *Nat. Lang. Eng.*, vol. 19, no. 2, pp. 285–288, 2013, doi: 10.1017/s1351324912000290.

[41] J. Ramírez Sánchez, M. Bereau, and J. Lara, "A Survey of the Effects of Data Augmentation for Automatic Speech Recognition Systems," pp. 669–678, Oct. 2019, doi: 10.1007/978-3-030-33904-3_63.

[42] S. Das, A. dey, A. Pal, and N. Roy, "Applications of Artificial Intelligence in Machine Learning: Review and Prospect," *Int. J. Comput. Appl.*, vol. 115, pp. 31–41, Apr. 2015,

doi: 10.5120/20182-2402.

[43] D. Sharma and N. Kumar, “A Review on Machine Learning Algorithms, Tasks and Applications,” *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 6, no. 10, pp. 1323–2278, Oct. 2017.

[44] Y. C A Padmanabha Reddy, P. Viswanath, and B. Eswara Reddy, “Semi-supervised learning: a brief review,” *Int. J. Eng. Technol.*, vol. 7, no. 1.8, p. 81, 2018, doi: 10.14419/ijet.v7i1.8.9977.

[45] X. Goldberg, *Introduction to semi-supervised learning*, vol. 6. 2009.

[46] S. Vluymans, “Multi-label Learning,” *Stud. Comput. Intell.*, vol. 807, pp. 189–218, 2019, doi: 10.1007/978-3-030-04663-7_7.

[47] T. Dietterich, C. Bishop, D. Heckerman, M. Jordan, and M. Kearns, *Semi-Supervised Learning*. .

[48] I. Uysal and H. A. Güvenir, “An overview of regression techniques for knowledge discovery,” *Knowl. Eng. Rev.*, vol. 14, no. 4, pp. 319–340, 1999, doi: 10.1017/S026988899900404X.

[49] P. Liang, “Lecture 2: Machine learning I,” 2018. [Online]. Available: https://web.stanford.edu/class/cs221/lectures/learning1.pdf.

[50] Y. Zhu and Y. Xiong, “Defining Data Science,” 2015, [Online]. Available: http://arxiv.org/abs/1501.05039.

[51] A. Liu, “Data Science and Data Scientist,” p. 11, 2015, [Online]. Available: http://www.researchmethods.org/DataScienceDataScientists.pdf.

[52] J. C. V. Guerra, Z. Khanam, S. Ehsan, R. Stolkin, and K. McDonald-Maier, “Weather Classification: A new multi-class dataset, data augmentation approach and comprehensive evaluations of Convolutional Neural Networks,” in *2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, 2018, pp. 305–310.

[53] P. F. Christ *et al.*, “Automatic Liver and Tumor Segmentation of CT and MRI Volumes using Cascaded Fully Convolutional Neural Networks,” pp. 1–20, 2017, [Online]. Available: http://arxiv.org/abs/1702.05970.

[54] M. Pérez-Ortiz, S. Jiménez-Fernández, P. A. Gutiérrez, E. Alexandre, C. Hervás-Martínez, and S. Salcedo-Sanz, “A review of classification problems and algorithms in renewable energy applications,” *Energies*, vol. 9, no. 8, pp. 1–27, 2016, doi: 10.3390/en9080607.

[55] A. Mesaros, T. Heittola, and D. Ellis, “Datasets and Evaluation,” in *Computational*

Analysis of Sound Scenes and Events, 2018, pp. 147–179.

[56] A. Rakotomamonjy and G. Gasso, "Histogram of Gradients of Time–Frequency Representations for Audio Scene Classification," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 1, pp. 142–153, 2015, doi: 10.1109/TASLP.2014.2375575.

[57] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," *2014 ACM Conference on Multimedia, MM 2014*. Association for Computing Machinery, Inc, pp. 1041-1044 BT-MM 2014-Proceedings of the 2014, Nov. 03, 2014, doi: 10.1145/2647868.2655045.

[58] D. Stowell and M. D. Plumbley, "An open dataset for research on audio field recording archives: freefield1010," *CoRR*, vol. abs/1309.5, 2013, [Online]. Available: http://arxiv.org/abs/1309.5275.

[59] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic Scene Classification in DCASE 2019 Challenge: Closed and Open Set Classification and Data Mismatch Setups," no. October, pp. 164–168, 2019, doi: 10.33682/m5kp-fa97.

[60] D. Giannoulis, E. Benetos, D. Stowell, M. Rossignol, M. Lagrange, and M. Plumbley, *Detection and classification of acoustic scenes and events: An IEEE AASP challenge*. 2013.

[61] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Audio context recognition using audio event histograms," *Eur. Signal Process. Conf.*, pp. 1272–1276, 2010.

[62] T. C. Area, "Conservation & Development Strategy for the Maloti Drakensberg," vol. 2, no. 2, 2008.

[63] M. Lagrange, G. Lafay, B. Défréville, and J.-J. Aucouturier, "The bag-of-frames approach: A not so sufficient model for urban soundscapes," *J. Acoust. Soc. Am.*, vol. 138, no. 5, pp. EL487–EL492, 2015, doi: 10.1121/1.4935350.

[64] G. Lafay, M. Lagrange, E. Benetos, M. Rossignol, and A. Roebel, "A Morphological Model for Simulating Acoustic Scenes and Its Application to Sound Event Detection," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, pp. 1854–1864, Oct. 2016, doi: 10.1109/TASLP.2016.2587218.

[65] D. Pyle, S. Editor, and D. D. Cerra, *Data Preparation for Data Mining*, Editorial. San Francisco, CA 94104-3205 USA: Morgan Kaufmann Publishers, Inc., 1999.

[66] E. D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine Learning , Neural and Statistical Classification*. 1994.

[67] V. Bachu and J. Anuradha, "A Review of Feature Selection and Its Methods," no. March,

2019, doi: 10.2478/cait-2019-0001.

[68] S. Omar and H. H. Jebur, “Machine Learning Techniques for Anomaly Detection : An Overview Machine Learning Techniques for Anomaly Detection : An Overview,” no. October, 2013, doi: 10.5120/13715-1478.

[69] S. B. K. I. D. Z. P. E. Pintelas, “Machine learning : a review of classification and combining techniques,” no. 2006, pp. 159–190, 2007, doi: 10.1007/s10462-007-9052-3.

[70] M. Hossin, “A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS,” *Int. J. Data Min. Knowl. Manag. Process*, vol. 5, no. August, pp. 1–4, 2015, doi: 10.5121/ijdkp.2015.5201.

[71] D. Berrar, “Cross-Validation Cross-validation,” no. January 2018, pp. 0–8, 2019, doi: 10.1016/B978-0-12-809633-8.20349-X.

[72] H. L. PAYAM REFAEILZADEH, LEI TANG, “Cross Validation,” Arizona, 2008.

[73] T. Andersson, “MASTER ’ S THESIS Audio Classification and Content Description,” Lule˚a University of Technology, 2004.

[74] J. Demsar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[75] D. Berrar and J. A. Lozano, “Significance tests or confidence intervals: Which are preferable for the comparison of classifiers?,” *J. Exp. Theor. Artif. Intell.*, vol. 25, no. 2, pp. 189–206, 2013, doi: 10.1080/0952813X.2012.680252.

[76] T. G. Dietterich, “Statistical Tests for Comparing Supervised Classi cation Learning Algorithms 1 Introduction,” *Science (80-.).*, vol. 10, no. 7, pp. 1–24, 1997, [Online]. Available: http://dx.doi.org/10.1162/089976698300017197.

[77] S. i and F. Herrera, “An Extension on ‘Statistical Comparisons of Classifiers over Multiple Data Sets’ for all Pairwise Comparisons,” *J. Mach. Learn. Res. - JMLR*, vol. 9, Dec. 2008.

[78] N. Japkowicz and M. Shah, “Evaluating Learning Algorithms: A Classification Perspective,” *Eval. Learn. Algorithms. A Classif. Perspect.*, Jan. 2011, doi: 10.1017/CBO9780511921803.

[79] G. Nguyen *et al.*, “Machine Learning and Deep Learning frameworks and libraries for large-scale data mining : a survey Machine Learning and Deep Learning frameworks and libraries for large-scale data mining : a survey,” *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 77–124, 2019, doi: 10.1007/s10462-018-09679-z.

[80] A. Kumar, “Deep Learning Methods for Classification with Limited Training Data

Seminar Report : Spring 2017 Aviral Kumar," Bombay, 2017.

[81] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Muller, "Deep learning for time series classification: a review," *Data Min. Knowl. Discov.*, vol. 33, no. 4, pp. 917–963, 2019, doi: 10.1007/s10618-019-00619-1.

[82] P. K. B. Giridhara, C. Mishra, R. K. M. Venkataramana, S. S. Bukhari, and A. Dengel, "A study of various text augmentation techniques for relation classification in free text," in *ICPRAM 2019 - Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods*, 2019, pp. 360–367.

[83] K. M. Rashid and J. Louis, "Window-warping: A time series data augmentation of IMU data for construction equipment activity identification," *Proc. 36th Int. Symp. Autom. Robot. Constr. ISARC 2019*, no. Isarc, pp. 651–657, 2019.

[84] Y. Baştanlar and M. Ozuysal, *Introduction to Machine Learning Second Edition*, vol. 1107. 2014.

[85] S. Min, B. Lee, and S. Yoon, "Deep Learning in Bioinformatics," p. 46.

[86] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, "Deep Learning for Video Game Playing," pp. 1–20, 2019.

[87] P. Rai, S. Prabhumoye, P. Khattri, L. Sandhu, and S. Kamath S, *A Prototype of an Intelligent Search Engine Using Machine Learning Based Training for Learning to Rank*, vol. 27. 2014.

[88] M. Fire and J. Schler, "Exploring Online Ad Images Using a Deep Convolutional Neural Network Approach," no. July 2017, 2015.

[89] V. Sokolov, "Discussion of 'Deep learning for finance: deep portfolios,'" *Appl. Stoch. Model. Bus. Ind.*, vol. 33, pp. 16–18, Feb. 2017, doi: 10.1002/asmb.2228.

[90] I. Goodfellow and Y. B. and A. Courville, *Deep Learning*. MIT Press, 2016.

[91] Z. Alom *et al.*, "A State-of-the-Art Survey on Deep Learning Theory and Architectures," *Electronics*, vol. 8, no. 292, pp. 1–67, 2019, doi: 10.3390/electronics8030292.

[92] WikiStat, "Neural Networks and Introduction to Deep Learning," Toulouse, 2015. [Online]. Available: http://wikistat.fr/pdf/st-m-explo-classif.pd.

[93] S. The, S. Ai, I. Dalle, and S. Galleria, "Deep Learning in Neural Networks : An Overview," Lugano,Switzerland, 2014.

[94] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," no. December, 2015, [Online]. Available: http://arxiv.org/abs/1511.08458.

[95] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer International

Publishing.

[96] K. Yun, A. Huyen, and T. Lu, “Deep neural networks for pattern recognition,” in *Advances in Pattern Recognition Research*, Nova Science Publishers, Inc., 2018, pp. 49–79.

[97] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, *Very deep convolutional neural networks for raw waveforms*. 2017.

[98] G. Hinton *et al.*, “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups,” *Signal Process. Mag. IEEE*, vol. 29, pp. 82–97, Nov. 2012, doi: 10.1109/MSP.2012.2205597.

[99] J. Huang *et al.*, “ACOUSTIC SCENE CLASSIFICATION USING DEEP LEARNING-BASED ENSEMBLE AVERAGING Technical Report Intel Corp , Intel Labs , 2200 Mission College Blvd ., Santa Clara , CA 95054 , USA ,” *Dcase 2019,声音场景识别*, pp. 1–5, 2019.

[100] X. Xia, R. Togneri, F. Sohel, Y. Zhao, and D. Huang, “A Survey: Neural Network-Based Deep Learning for Acoustic Event Detection,” *Circuits Syst. Signal Process.*, Mar. 2019, doi: 10.1007/s00034-019-01094-1.

[101] H. Lee, P. Pham, Y. Largman, and A. Ng, “Unsupervised feature learning for audio classification using convolutional deep belief networks,” in *NIPS*, Jan. 2009, pp. 1096–1104.

[102] J. Abeßer, “A review of deep learning based methods for acoustic scene classification,” *Appl. Sci.*, vol. 10, no. 6, 2020, doi: 10.3390/app10062020.

[103] V. Bisot, R. Serizel, S. Essid, and G. Richard, “Nonnegative Feature Learning Methods for Acoustic Scene Classification,” *DCASE 2017 - Work. Detect. Classif. Acoust. Scenes Events*, no. November, pp. 2–6, 2017.

[104] G. Takahashi, T. Yamada, N. Ono, and S. Makino, *Performance evaluation of acoustic scene classification using DNN-GMM and frame-concatenated acoustic features*. 2017.

[105] T. Epelbaum, “Deep learning : Technical introduction,” 2017.

[106] M. Sazli, “A brief review of feed-forward neural networks,” *Commun. Fac. Sci. Univ. Ankara*, vol. 50, pp. 11–17, Jan. 2006, doi: 10.1501/0003168.

[107] M. Cilimkovic, “Neural Networks and Back Propagation Algorithm,” *Fett.Tu-Sofia.Bg*, pp. 3–7, 2010, [Online]. Available: http://fett.tu-sofia.bg/et/2006/ET2006 BOOK 1/Circuits and Systems/173 Paper-V_Skorpil.pdf.

[108] Y. L. C. and L. D. J. and B. E. B. and J. D. and H. P. G. and I. G. and D. R. H. and R. H.

and W. Hubbard, “Handwritten digit recognition: applications of neural network chips and automatic learning,” *IEEE Commun. Mag.*, vol. 27, no. 11, pp. 41–46, 1989.

[109] B. Lehner, H. Eghbal-Zadeh, M. Dorfer, F. Korzeniowski, K. Koutini, and G. Widmer, “Classifying short acoustic scenes with i-vectors and cnns: challenges and optimisations for the 2017 dcase asc task,” *DCASE 2017-Workshop Detect. Classif. Acoust. Scenes Events*, no. November, 2017, [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2017/documents/challenge_technical_reports/DCASE2017_Lehner_142.pdf.

[110] Y. Han, J. Park, and K. Lee, *Convolutional Neural Networks with Binaural Represntations and Background Subtraction for Acoustic Scene Classification*. 2017.

[111] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” *30th Int. Conf. Mach. Learn. ICML 2013*, no. PART 3, pp. 2347–2355, 2013.

[112] Z. Chen, Z. Xie, W. Zhang, and X. Xu, “ResNet and model fusion for automatic spoofing detection,” *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2017-Augus, pp. 102–106, 2017, doi: 10.21437/Interspeech.2017-1085.

[113] A. Jansen *et al.*, “Unsupervised Learning of Semantic Audio Representations,” *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2018-April, pp. 126–130, 2018, doi: 10.1109/ICASSP.2018.8461684.

[114] M. D. McDonnell and W. Gao, “Acoustic Scene Classification Using Deep Residual Networks with Late Fusion of Separated High and Low Frequency Paths,” pp. 141–145, 2020, doi: 10.1109/icassp40776.2020.9053274.

[115] A. Scenes, “Adsc Submission for Dcase 2017 : Acoustic Scene Classification Using Deep,” *Dcase 2017*, no. November, pp. 2–6, 2017.

[116] A. Zaeemzadeh, N. Rahnavard, and M. Shah, “Norm-Preservation: Why Residual Networks Can Become Extremely Deep?,” *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2020, doi: 10.1109/tpami.2020.2990339.

[117] L. Cordeiro, “Wide Residual Network for the Tiny ImageNet Challenge,” pp. 2–5.

[118] X. Han, Y. Zhong, L. Cao, and L. Zhang, “Pre-trained alexnet architecture with pyramid pooling and supervision for high spatial resolution remote sensing image scene classification,” *Remote Sens.*, vol. 9, no. 8, 2017, doi: 10.3390/rs9080848.

[119] M. Z. Alom *et al.*, “The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches,” Mar. 2018.

[120] V. Maeda-Gutiérrez *et al.*, “Comparison of convolutional neural network architectures

for classification of tomato plant diseases," *Appl. Sci.*, vol. 10, no. 4, 2020, doi: 10.3390/app10041245.

[121] A. Labach, H. Salehinejad, and S. Valaee, *Survey of Dropout Methods for Deep Neural Networks*. 2019.

[122] V. Boddapati, A. Petef, J. Rasmusson, and L. Lundberg, "Classifying environmental sounds using image recognition networks," *Procedia Comput. Sci.*, vol. 112, pp. 2048–2056, 2017, doi: 10.1016/j.procs.2017.08.250.

[123] M. Lederle and B. Wilhelm, *Combining High-Level Features of Raw Audio Waves and Mel-Spectrograms for Audio Tagging*. 2018.

[124] Z. Ren, Q. Kong, K. Qian, M. Plumbley, and B. Schuller, *Attention-based Convolutional Neural Networks for Acoustic Scene Classification*. 2018.

[125] Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2019, doi: 10.1109/tkde.2019.2946162.

[126] A. Mikołajczyk and M. Grochowski, *Data augmentation for improving deep learning in image classification problem*. 2018.

[127] B. McFee, E. J. Humphrey, and J. P. Bello, "A software framework for musical data augmentation," *16th International Society for Music Information Retrieval Conference, ISMIR 2015*. International Society for Music Information Retrieval, pp. 248-254 BT-Proceedings of the 16th Internationa, Jan. 01, 2015, [Online]. Available: http://www.scopus.com/inward/record.url?scp=84996516893&partnerID=8YFLogxK.

[128] C. Shorten and T. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *J. Big Data*, vol. 6, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[129] E. Andersson and R. Berglund, "Evaluation of Data Augmentation of MR Images for Deep Learning," Lund University, 2018.

[130] S. Wei, S. Zou, F. Liao, and W. Lang, "A Comparison on Data Augmentation Methods Based on Deep Learning for Audio Classification," *J. Phys. Conf. Ser.*, vol. 1453, no. 1, 2020, doi: 10.1088/1742-6596/1453/1/012085.

[131] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "MixUp: Beyond empirical risk minimization," in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.

[132] V. Sarnatskyi, V. Ovcharenko, M. Tkachenko, S. Stirenko, Y. Gordienko, and A. Rojbi, "Music Transcription by Deep Learning with Data and 'Artificial Semantic'

Augmentation," Dec. 2017.

[133] Z. Eaton-Rosen, F. Bragman, S. Ourselin, and M. J. Cardoso, "Improving Data Augmentation for Medical Image Segmentation," *1st Conf. Med. Imaging with Deep Learn.*, vol. 10670 LNCS, no. Midl, pp. 450–462, 2018, doi: 10.1007/978-3-319-75238-9_38.

[134] V. Marivate and T. Sefara, "Improving short text classification through global augmentation methods," pp. 1–15, 2019, [Online]. Available: http://arxiv.org/abs/1907.03752.

[135] Z. Zhang, S. Xu, S. Cao, and S. Zhang, "Deep convolutional neural network with mixup for environmental sound classification," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11257 LNCS, pp. 356–367, 2018, doi: 10.1007/978-3-030-03335-4_31.

[136] K. Xu *et al.*, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11166 LNCS, pp. 14–23, 2018, doi: 10.1007/978-3-030-00764-5_2.

[137] H. Zhang, M. Cisse, Y. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," Oct. 2017.

[138] C. Summers and M. J. Dinneen, "Improved mixed-example data augmentation," *Proc. - 2019 IEEE Winter Conf. Appl. Comput. Vision, WACV 2019*, pp. 1262–1270, 2019, doi: 10.1109/WACV.2019.00139.

[139] D. Joyce, "Span and independence Math 130 Linear Algebra," 2015.

[140] H. Guo, Y. Mao, and R. Zhang, "Augmenting Data with Mixup for Sentence Classification: An Empirical Study," May 2019.

[141] J. B. Jeremy Orloff, "Beta distribution," 2014.

[142] A. Bessi *et al.*, *Everyday the Same Picture: Popularity and Content Diversity*. 2015.

[143] Ş. Çelik and M. Korkmaz, "BETA DISTRIBUTION AND INFERENCES ABOUT THE BETA FUNCTIONS," *Asian J. Sci. Technol.*, vol. 7, no. 5, pp. 2960–2970, May 2016.

[144] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "CHIME-HOME : A DATASET FOR SOUND SOURCE RECOGNITION IN A DOMESTIC ENVIRONMENT School of Electronic Engineering and Computer Science , Queen Mary University of London , UK Audio Analytic , Cambridge , UK Department of Computer Science , University of Sheff," pp. 4–8, 2015.

[145] K. J. Piczak, “ESC: Dataset for environmental sound classification,” *MM 2015 - Proc. 2015 ACM Multimed. Conf.*, pp. 1015–1018, 2015, doi: 10.1145/2733373.2806390.

[146] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, “Random Erasing Data Augmentation,” *Proc. AAAI Conf. Artif. Intell.*, vol. 34, Aug. 2017, doi: 10.1609/aaai.v34i07.7000.

[147] R. Takahashi, T. Matsubara, and K. Uehara, “RICAP: Random Image Cropping and Patching Data Augmentation for Deep CNNs,” *Proc. 10th Asian Conf. Mach. Learn.*, no. 2012, p. PMLR 95:786-798, 2018, [Online]. Available: http://arxiv.org/abs/1811.09030.

[148] Z. Yang, Z. Wang, W. Xu, X. He, Z. Wang, and Z. Yin, “Region-aware Random Erasing,” in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, 2019, pp. 1699–1703.

[149] S. Adapa, “Ust,Cnn,” pp. 8–10, 2019.

[150] B. Pantic, “ENSEMBLE OF CONVOLUTIONAL NEURAL NETWORKS FOR GENERAL PURPOSE,” Belgrad, 2018.

[151] X. R. Qingkai WEI , Yanfang LIU, “A REPORT ON AUDIO TAGGING WITH DEEPER CNN , 1D-CONVNET AND 2D-CONVNET,” Beijing, 2018.

[152] S. Bhardwaj, “Audio Data Augmentation with respect to Musical Instrument Recognition,” no. August, 2017, doi: https://doi.org/10.5281/zenodo.1066137.

[153] J. Schlüter and T. Grill, “Exploring data augmentation for improved singing voice detection with neural networks,” *Proc. 16th Int. Soc. Music Inf. Retr. Conf. ISMIR 2015*, pp. 121–126, 2015.

[154] J. Salamon and J. P. Bello, “Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification,” *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 279–283, 2017, doi: 10.1109/LSP.2017.2657381.

[155] M. Mdesa, *Audio Deformation based Data Augmentation for Convolution Neural Network in Vibration Analysis*. 2019.

[156] E.-P. Damskägg and V. Välimäki, “Audio Time Stretching Using Fuzzy Classification of Spectral Bins,” *Appl. Sci.*, vol. 7, p. 1293, Dec. 2017, doi: 10.3390/app7121293.

[157] N. Juillerat and B. Hirsbrunner, *Low latency audio pitch shifting in the frequency domain*. 2010.

[158] G. Maguolo, M. Paci, L. Nanni, and L. Bonan, “Audiogmenter : a MATLAB Toolbox for Audio Data Augmentation,” pp. 2–8, 2019, [Online]. Available:

https://arxiv.org/abs/1912.05472%0Ahttps://github.com/LorisNanni/Audiogmenter.

[159] S. Thulasidasan, G. Chennupati, J. Bilmes, T. Bhattacharya, and S. Michalak, "On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks," no. NeurIPS, pp. 1–12, 2019, [Online]. Available: http://arxiv.org/abs/1905.11001.

[160] S. Wei, K. Xu, D. Wang, F. Liao, H. Wang, and Q. Kong, "Sample Mixed-Based Data Augmentation for Domestic Audio Tagging," 2018. [Online]. Available: http://arxiv.org/abs/1808.03883.

[161] D. S. Park *et al.*, "Specaugment: A simple data augmentation method for automatic speech recognition," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2019-Septe, pp. 2613–2617, 2019, doi: 10.21437/Interspeech.2019-2680.

[162] H. Guo, Y. Mao, and R. Zhang, "Augmenting Data with Mixup for Sentence Classification: An Empirical Study," 2019. [Online]. Available: http://arxiv.org/abs/1905.08941.

[163] S. Gao, G. Mishne, and D. Scheinost, *Framework for High-Dimensional Neuroimaging Data*, vol. 1. 2019.

[164] T. T. Um *et al.*, "Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks," *ICMI 2017 - Proc. 19th ACM Int. Conf. Multimodal Interact.*, vol. 2017-Janua, pp. 216–220, 2017, doi: 10.1145/3136755.3136817.

[165] N. Jaitly and G. Hinton, "Vocal Tract Length Perturbation for Speech Recognition with DNN-HMMs," 2013.

[166] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep convolutional neural network acoustic modeling," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2015-Augus, pp. 4545–4549, 2015, doi: 10.1109/ICASSP.2015.7178831.

[167] S. Wilson and M. Bosi, "WAVE PCM Soundfile Format," pp. 2–5, 2003, [Online]. Available: https://ccrma.stanford.edu/courses/422/projects/WaveFormat/.

[168] A. R. Brodtkorb, C. Dyken, T. R. Hagen, J. M. Hjelmervik, and O. O. Storaasli, "State-of-the-art in Heterogeneous Computing," *Sci. Program.*, vol. 18, no. 1, pp. 1–33, 2010, doi: 10.1155/2010/540159.

[169] B. S. H. (Scott) Michel, "GPU computing---General-purpose GPU computing," p. 233, 2006, doi: 10.1145/1188455.1188698.

[170] A. Kayid, Y. Khaled, and M. Elmahdy, *Performance of CPUs/GPUs for Deep Learning*

workloads. 2018.

[171] T. Carneiro, R. V. M. Da Nobrega, T. Nepomuceno, G. Bin Bian, V. H. C. De Albuquerque, and P. P. R. Filho, "Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications," *IEEE Access*, vol. 6, pp. 61677–61685, 2018, doi: 10.1109/ACCESS.2018.2874767.

[172] E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. 2019.

[173] L. von Chamier *et al.*, "ZeroCostDL4Mic: an open platform to simplify access and use of Deep-Learning in Microscopy," *bioRxiv*, p. 2020.03.20.000133, 2020, doi: 10.1101/2020.03.20.000133.

[174] B. Mcfee *et al.*, "Librosa - audio processing Python library," *PROC. 14th PYTHON Sci. CONF*, no. Scipy, pp. 18–25, 2015, [Online]. Available: http://conference.scipy.org/proceedings/scipy2015/pdfs/brian_mcfee.pdf.

[175] J. D. Dignam, P. L. Martin, B. S. Shastry, and R. G. Roeder, "Eukaryotic gene transcription with purified components," *Methods Enzymol.*, vol. 101, no. C, pp. 582–598, 1983, doi: 10.1016/0076-6879(83)01039-3.

[176] J. Moolayil, *Learn Keras for Deep Neural Networks*. 2019.

[177] T. Viehmann, *Eli Stevens Luca Antiga*. .

[178] "Numpy documentation— NUMPY V1.19 MANUAL," 2020. .

[179] "pickle — Python object serialization — Python 3.8.4 documentation," 2020. https://docs.python.org/3/library/pickle.html (accessed Jul. 14, 2020).

[180] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, and J. Santos, "Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches [Research Frontier]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 4, pp. 59–76, 2018, doi: 10.1109/MCI.2018.2866730.

[181] A. Vellido, J. D. Martín, F. Rossi, and P. J. G. Lisboa, "Seeing is believing: The importance of visualization in real-world machine learning applications," *ESANN 2011 proceedings, 19th Eur. Symp. Artif. Neural Networks, Comput. Intell. Mach. Learn.*, no. April, pp. 219–226, 2010.

[182] K. Gajowniczek, T. Ząbkowski, and A. Orłowski, "Comparison of decision trees with Rényi and Tsallis entropy applied for imbalanced churn dataset," *Proc. 2015 Fed. Conf. Comput. Sci. Inf. Syst. FedCSIS 2015*, vol. 5, pp. 39–44, 2015, doi: 10.15439/2015F121.

[183] S. Majeed, H. HUSAIN, S. Samad, and T. Idbeaa, "Mel frequency cepstral coefficients

(Mfcc) feature extraction enhancement in the application of speech recognition: A comparison study," *J. Theor. Appl. Inf. Technol.*, vol. 79, pp. 38–56, Sep. 2015.

[184] A. Wendemuth, B. Vlasenko, I. Siegert, R. Böck, F. Schwenker, and G. Palm, "Emotion recognition from speech," *Cogn. Technol.*, no. 9783319436647, pp. 409–428, 2017, doi: 10.1007/978-3-319-43665-4_20.

[185] M. Henini, "Corrigendum to 'Molecular beam epitaxy: From research to manufacturing' [Thin Solid Films 306 (1997) 331-337] (DOI:10.1016/S0040-6090(97)00242-3)," *Thin Solid Films*, vol. 517, no. 16, p. 4698, 2009, doi: 10.1016/j.tsf.2008.10.035.

[186] Y. Han and K. Lee, "Acoustic scene classification using convolutional neural network and multiple-width frequency-delta data augmentation," vol. 14, no. 8, pp. 1–11, 2016, [Online]. Available: http://arxiv.org/abs/1607.02383.

[187] M. D. McDonnell, "Training wide residual networks for deployment using a single bit for each weight," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, pp. 1–16, 2018.

[188] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," *Br. Mach. Vis. Conf. 2016, BMVC 2016*, vol. 2016-Septe, pp. 87.1-87.12, 2016, doi: 10.5244/C.30.87.

[189] S. Verma, A. Chug, and A. Singh, "Impact of Hyperparameter Tuning on Deep Learning Based Estimation of Disease Severity in Grape Plant," 2020, pp. 161–171.

[190] L. Zhang, D. Wang, C. Bao, Y. Wang, and K. Xu, "Large-scale whale-call classification by transfer learning on multi-scale waveforms and time-frequency features," *Appl. Sci.*, vol. 9, no. 5, pp. 1–11, 2019, doi: 10.3390/app9051020.

[191] B. Liu, Y. Wei, Y. Zhang, and Q. Yang, "Deep neural networks for high dimension, low sample size data," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 0, pp. 2287–2293, 2017, doi: 10.24963/ijcai.2017/318.