TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2023 150

Pablo Gimeno Jordán

Advances in Binary and Multiclass Audio Segmentation with Deep Learning Techniques

Director/es
Ortega Giménez, Alfonso

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

ISSN 2254-7606

Universidad Zaragoza
1542

Tesis Doctoral

ADVANCES IN BINARY AND MULTICLASS AUDIO SEGMENTATION WITH DEEP LEARNING TECHNIQUES

Autor

Pablo Gimeno Jordán

Director/es

Ortega Giménez, Alfonso

UNIVERSIDAD DE ZARAGOZA
Escuela de Doctorado

Programa de Doctorado en Tecnologías de la Información y Comunicaciones en Redes Móviles

2023

Universidad de Zaragoza

Programa de Doctorado en Tecnologías de la Información y Comunicaciones en Redes Móviles

Ph.D. Thesis

Advances in Binary and Multiclass Audio Segmentation with Deep Learning Techniques

Author

Pablo Gimeno Jordán

Supervisor

Dr. Alfonso Ortega Giménez

Departamento de ingeniería electrónica y comunicaciones

Escuela de Ingeniería y Arquitectura

March 2023

Universidad Zaragoza

A mis padres

Agradecimientos

Resulta complicado resumir todos esos momentos en los que me he sentido agradecido durante los casi cinco años que han pasado desde que tome la decisión de comenzar mis estudios de doctorado. Ahora que echo la vista atrás me siento especialmente afortunado al comprobar que en este tiempo he podido colaborar con gente estupenda tanto en lo personal como en lo profesional. Estas líneas van dedicadas a todas esas personas que a lo largo de la realización de esta tesis me han dedicado una parte de su valioso tiempo.

En primer lugar, me gustaría darle las gracias a mi director de tesis, Alfonso Ortega, por su inestimable apoyo durante el desarrollo de esta tesis. Fue, en cierta manera, quien hizo que eligiera comenzar mis estudios en ingeniería de telecomunicación y quien me ha guiado en estos primeros pasos de mi carrera investigadora. Gracias por tu tiempo durante todos estos años, por todas esas innumerables conversaciones que hemos mantenido, y sobre todo gracias por la confianza que has depositado en mi durante estos años.

Me gustaría también agradecer a todos los miembros del grupo ViVoLab y a mis compañeros de laboratorio, ya que sin ellos elaborar esta tesis habría sido mucho mas difícil. Son varios nombres los que me vienen a la cabeza cuando pienso en compañeros con los que he compartido muchos momentos y que, de alguna manera, han contribuido a que este trabajo salga adelante: Eduardo, Antonio, Victoria, Dayana, Luis, Nacho, Jorge, Julia. Mi agradecimiento también para todos los compañeros del Departamento de Ingeniería Electrónica y Comunicaciones, en especial a mis compañeros del programa de doctorado.

De la misma manera, debo dar las gracias a mi amigo Miguel Vela por haber podido compartir experiencias durante estos años y por haberme permitido acercarme a otras áreas de investigación fuera de la ingeniería. Colaborar contigo ha resultado especialmente enriquecedor. Ojalá podamos encontrar más maneras de conectar en el futuro.

A mi familia, y en especial a mis padres. Gracias por haber estado siempre a mi lado y por enseñarme las cosas que son importantes en la vida. Si soy quien soy hoy en día es gracias a vuestra educación. Gracias por vuestra compresión y vuestro apoyo constante.

A Darian, que siempre ha creído en mí y me ayuda día a día a sacar la mejor versión de mí mismo. Este camino no habría sido lo mismo sin ti. No se me ocurre una mejor compañera de viaje que tú.

A todos los que habeis hecho esta tesis posible, muchas gracias.

Abstract

Advances in technology over the last decade have reshaped the way population interact with multimedia content. This fact aroused a significant rise both in generation and consumption of these data in recent years. Manual analysis and annotation of this information is unfeasible given the current volume, revealing the necessity of automatic tools that can help advance from manual working pipelines to assisted or partially automatic practices. Over the last few years, most of these tools for multimedia information retrieval are based on the deep learning paradigm. In this context, work presented in this thesis focuses on the audio information retrieval domain. In particular, this dissertation studies the audio segmentation task, whose main goal is to provide a sequence of labels that isolates different regions in an input audio signal according to the characteristics described in a predefined set of classes, e.g., speech, music or noise.

The first part of this thesis dissertation is devoted to the speech activity detection task. In the last few years, a number of international evaluation campaigns have been proposing this task as one of their challenges. Among them, Fearless steps challenge deals with digitised audio coming from the original recordings of the NASA's Apollo space missions. For this challenge, we introduce a deep learning solution that is based on a convolutional recurrent neural network classifier. As the main contribution, a novel approach combining information from 1D and 2D filters to be processed by a recurrent neural networks is presented. Motivated by the introduction of the Fearless steps data, an evaluation of different domain adaptation methods is presented to test how well a speech activity detection system trained on common datasets can perform on this new domain. Adaptation techniques described do not require labels in the target domain, facilitating its use in practical applications. In general terms, methods seeking to minimise the statistical distribution shift between source and target domains provide the most promising results. Recent advances in self-supervised representation learning have shown significant improvements in performance in several speech processing tasks under different domains. Driven by these advances, we aim to incorporate them in the speech activity detection task. With last editions of the Fearless steps challenge shifting its focus to test system generalisation capabilities, the goal is to benefit from large amounts of unlabelled data to improve robustness. Experimental findings suggest that self-supervised feature learning allows to build systems that are significantly less sensitive to domain mismatch.

The second part of this manuscript analyses a more general audio segmentation task that seeks to simultaneously classify audio signals as being speech, music, noise or a

combination of these. In the context of Albayzín audio segmentation challenge corpora, an approach based on the combination of recurrent neural networks as the main classifier and a hidden Markov model postprocessing module is proposed. Furthermore, a new block is introduced in the neural architecture with the objective of removing redundant temporal information, improving the performance and reducing the number of operations per second at the same time. This proposal outperformed previous solutions in the literature and similar approaches based on deep neural networks in both 2010 and 2012 challenge data. While results using self-supervised learning were promising in a binary segmentation task such as speech activity detection, if applied on multiclass segmentation tasks some issues arise. Common augmentation techniques applied in training time force the model to compensate background noises or music. Under these conditions, features obtained do not represent accurately classes that are generated in a similar way as the augmented versions seen in training. This fact limits the overall improvement observed when using self-supervised features in tasks such as the one proposed by the Albayzín 2010 evaluation, where both speech and music must be separated.

Last part of this thesis investigated the application of new training objectives in audio segmentation with the main goal of mitigating the issues derived from a limited training dataset. New optimisation techniques based on AUC and partial AUC metrics have been demonstrated to outperform traditional training objectives such as cross entropy in several detection tasks. Therefore, this dissertation introduces recently proposed AUC optimisation techniques into the music detection task. Providing that labelled data for music detection is limited when compared to other tasks such as speech activity detection, these techniques are applied with the main goal of improving the performance when using a relatively small training dataset. Most systems utilising AUC optimisation techniques are limited to binary tasks, as this is the typical scenario to apply AUC metrics. Furthermore, labelling audios with detailed taxonomies involving multiples options is significantly more complex, resulting in a limited amount of labelled audio for some multiclass audio segmentation tasks. Therefore, as a natural extension, we propose a generalisation of binary AUC optimisation techniques so that they can be applied to an arbitrary number of classes. Two different loss functions are introduced, using as the starting point for the formulation multiclass variations of the AUC metric proposed in the literature: one based on a one-versus-one approach, and one based on a one-versus-rest approach.

Resumen

Los avances tecnológicos acaecidos en la última década han cambiado completamente la forma en la que la población interactúa con el contenido multimedia. Esto ha propiciado un aumento significativo tanto en la generación como el consumo de dicho contenido. El análisis y la anotación manual de toda esta información no son factibles dado el gran volumen actual, lo que releva la necesidad de herramientas automáticas que ayuden en la transición hacia flujos de trabajo asistidos o parcialmente automáticos. En los últimos años, la mayoría de estas herramientas están basadas en el uso de redes neuronales y *deep learning*. En este contexto, el trabajo que se describe en esta tesis se centra en el ámbito de la extracción de información a partir de señales de audio. Particularmente, se estudia la tarea de segmentación de audio, cuyo principal objetivo es obtener una secuencia de etiquetas que aíslen diferentes regiones en una señal de entrada de acuerdo con una serie de características descritas en un conjunto predefinido de clases, como por ejemplo voz, música o ruido.

La primera parte de esta memoria esta centrada en la tarea de detección de actividad de voz. Recientemente, diferentes campañas de evaluación internacionales han propuesto esta tarea como uno de sus retos. Entre ellas se encuentra el reto *Fearless steps*, que trabaja con audios de las grabaciones de las misiones *Apollo* de la NASA. Para este reto, se propone una solución basada en aprendizaje supervisado usando una red convolucional recurrente como clasificador. La principal contribución es un método que combina información de filtros de 1D y 2D en la etapa convolucional para que sea procesada posteriormente por la etapa recurrente. Motivado por la introducción de los datos del reto *Fearless steps*, se plantea una evaluación de diferentes técnicas de adaptación de dominio, con el objetivo de comprobar las prestaciones de un sistema entrenado con datos de dominios habituales y evaluado en este nuevo dominio presentado en el reto. Los métodos descritos no requieren de etiquetas en el dominio objetivo, lo que facilita su uso en aplicaciones prácticas. En términos generales, se observa que los métodos que buscan minimizar el cambio en las distribuciones estadísticas entre los dominios fuente y objetivo obtienen los resultados mas prometedores. Los avances recientes en técnicas de representación obtenidas mediante aprendizaje auto-supervisado han demostrado grandes mejoras en prestaciones en varias tareas relacionadas con el procesado de voz. Siguiendo esta línea, se plantea la incorporación de dichas representaciones en la tarea de detección de actividad de voz. Las ediciones más recientes del reto *Fearless steps* modificaron su propósito, buscando ahora evaluar las capacidades de generalización de los sistemas. El objetivo entonces con las técnicas introducidas es poder beneficiarse de grandes can-

tidades de datos no etiquetados para mejorar la robustez del sistema. Los resultados experimentales sugieren que el aprendizaje auto-supervisado de representaciones permite obtener sistemas que son mucho menos sensibles al cambio de dominio.

En la segunda parte de este documento se analiza una tarea de segmentación de audio más genérica que busca clasificar de manera simultanea una señal de audio como voz, música, ruido o una combinación de estas. En el contexto de los datos propuesto para el reto de segmentación de audio Albayzín 2010, se presenta un enfoque basado en el uso de redes neuronales recurrentes como clasificador principal, y un modelo de postprocesado integrado por modelos ocultos de Markov. Se introduce un nuevo bloque en la arquitectura neuronal con el objetivo de eliminar la información temporal redundante, mejorando las prestaciones y reduciendo el numero de operaciones por segundo al mismo tiempo. Esta propuesta obtuvo mejores prestaciones que soluciones presentadas anteriormente en la literatura, y que aproximaciones similares basades en redes neuronales profundas. Mientras que los resultados con aprendizaje auto-supervisado de representaciones eran prometedores en tareas de segmentación binaria, si se aplican en tareas de segmentación multiclase surgen una serie de cuestiones. Las técnicas habituales de aumento de datos que se aplican en el entrenamiento fuerzan al modelo a compensar el ruido de fondo o la música. En estas condiciones las características obtenidas podrían no representar de manera precisa aquellas clases generadas de manera similar a las versiones aumentadas vistas en el entrenamiento. Este hecho limita la mejora global de prestaciones observada al aplicar estas técnicas en tareas como la propuesta en la evaluación Albayzín 2010.

La última parte de este trabajo ha investigado la aplicación de nuevas funciones de coste en la tarea de segmentación de audio, con el principal objetivo de mitigar los problemas que se derivan de utilizar un conjunto de datos de entrenamiento limitado. Se ha demostrado que nuevas técnicas de optimización basadas en las métricas AUC y AUC parcial pueden mejorar objetivos de entrenamiento tradicionales como la entropía cruzada en varias tareas de detección. Con esta idea en mente, en esta tesis se introducen dichas técnicas en la tarea de detección de música. Considerando que la cantidad de datos etiquetados para esta tarea es limitada comparado con otras tareas, las funciones de coste basadas en la métrica AUC se aplican con el objetivo de mejorar las prestaciones cuando el conjunto de datos de entrenamiento es relativamente pequeño. La mayoría de los sistemas que utilizan las técnicas de optimización basadas en métricas AUC se limitan a tareas binarias ya que ese el ámbito de aplicación habitual de la métrica AUC. Además, el etiquetado de audios con taxonomías más detalladas en las que hay múltiples opciones posibles es más complejo, por lo que la cantidad de audio etiquetada en algunas tareas de segmentación multiclase es limitada. Como una extensión natural, se propone una generalización de las técnicas de optimización basadas en la métrica AUC binaria, de tal manera que se puedan aplicar con un número arbitrario de clases. Dos funciones de coste distintas se introducen, usando como base para su formulación las variaciones multiclase de la métrica AUC propuestas en la literatura: una basada en un enfoque uno contra uno, y otra basada en un enfoque uno contra el resto.

Contents

List of Figures

List of Tables

List of Acronyms

Adam	Adaptative moment estimation
AMI	Augmented multi-pary interaction
AUC	Area under the ROC curve
ASR	Automatic speech recognition
BBC	British broadcast corporation
BERT	Bidirectional encoder representations from transformers
BGRU	Bidirectional GRU
BIC	Bayesian information criterion
BLSTM	Bidirectional LSTM
BTT	Backpropagation through time
CARTV	Corporación aragonesa de radio y televisión
CE	Cross entropy
CNN	Convolutional neural network
CORAL	Correlation alignment
CRNN	Convolutional recurrent neural network
DCASE	Detection and classification of acoustic scenes and events
DCF	Detection cost function
DCT	Discrete cosine transform
DET	Detection error trade-off
DER	Diarisation error rate
DNN	Deep neural network
EER	Equal error rate
FA	Factor analysis
FFT	Fast Fourier transform
FIR	Finite impulse response
FN	False negative
FNR	False negative rate
FP	False positive
FPR	False positive rate
FS	Fearless steps
GAN	Generative adversarial networks
GELU	Gaussian error linear unit

GLR	Generalised likelihood ratio
GMM	Gaussian mixture model
GPU	Graphics processing unit
GRU	Gated recurrent unit
HEAR	Holistic evaluation of audio representations
HMM	Hidden Markov model
ICSI	International computer science institute
IP	Internet protocol
JFA	Joint factor analysis
KD	Knowledge distillation
KLD	Kullback-Leibler divergence
LDC	Linguistic data consortium
LSTM	Long short-term memory
MAP	Maximum a posteriori
MFCC	Mel frequency cepstrum coefficient
MGB	Multi-genre broadcast
MIR	Music information retrieval
MIREX	Music information retrieval evaluation exchange
MLP	Multilayer perceptron
MSA	Multihead self attention
NASA	National aeronautics and space administration
NIST	National institute of standards and technology
OVO	one-versus-one
OVR	one-versus-rest
pAUC	partial AUC
PL	pseudo-labelling
PLP	Perceptual linear prediction
ReLU	Rectified linear unit
RIR	Room impulse response
RNN	Recurrent neural network
ROC	Receiver operating characteristics
RT	Rich transcription
RTVE	Radio televisión española
SAD	Speech activity detection
SED	Sound event detection
SER	Segmentation error rate
SNR	Signal to noise ratio
SRE	Speaker recognition evaluation
SVM	Support vector machine
TCN	Temporal convolution network
TN	True negative
TP	True positive
TSNE	T distributed stochastic neighbour embedding
UBM	Universal background model

Part I

Introduction and state-of-the-art

"Data are becoming the new raw material of business."

Craig Mundie

1 Introduction and motivation

1.1 Motivation & context

Technological advances over the last few years have completely modified the way that general public interacts with multimedia content. Technology is currently present in all of our daily interactions. The evolution of electronic devices such as mobile phones, cameras or laptops has played a key role in the huge rise in consumption and generation of multimedia resources observed in recent years. At the same time, these resources are being made available to the general public through different platforms, with audio and video on demand systems being extremely popular nowadays. These large multimedia repositories represent a significant part of the Internet data traffic. In 2017, Cisco annual internet report estimated multimedia streaming traffic to be the 75% of the IP traffic in the world. According to YouTube most recent copyright transparency report [1], 500 hours of video per minute are uploaded to YouTube video streaming platform. This is equivalent to 82 years of video content uploaded in a single hour. As of 2022 statistics, 2.6 billion users worldwide interact monthly with the platform generating a total of one billion hours watched daily. In the audio streaming domain, as of 2022 statistics [2], Spotify platform featured 433 million active monthly listeners. These users

could choose from a catalogue of around 82 million songs and more than 4 million podcasts, with around 2 millions songs more being uploaded every month and 1.2 million podcasts added in year 2022. In addition to the problem of a dramatically rising amount of multimedia data volume, the amount of different information that shall be extracted from a multimedia stream is now bigger than in previous years. A multimedia signal is intrinsically rich in information, and recent technological advances have also allowed an increase in the number of information streams that are considered relevant in this kind of signals. Focusing on the two main information sources present in a multimedia signal, audio and video, these are some examples of information extraction that can be cited:

- **Audio**: transcription, speaker recognition, speech activity detection, music detection, speech pathology detection, etc, ...
- **Video**: face recognition, object detection, movement detection, scene description, sentiment analysis, etc, ...

Currently, one the main efforts from multimedia content providers is to increase value for its clients by analysing and cataloguing the available content. Manual annotation of this content in all the possible dimensions mentioned previously has become a really expensive and time-consuming task. For example, manually transcribing a document may take up from 4 to 6 times the original length of the document, with its associated cost in terms of staff. Under these circumstances, there is a need for the advancement of automatic systems that can analyse, index and retrieve information from a multimedia stream in a fast and accurate way. Research community is specially aware of this issue, devoting huge efforts for several decades in building solutions towards process automation in multimedia information extraction. The main idea in all these systems is to advance from manual working pipelines to assisted or partially automatic practices. Over the last few years a trend can be observed, with most of the current multimedia information retrieval systems based on the deep learning paradigm. Deep learning is a subset of machine learning algorithms where several neural networks processing layers are stacked in order to extract high level features from data [3]. These systems have demonstrated great performance in different applications dealing with large amounts of data.

Driven by the context that has just been introduced, the work presented in this thesis dissertation focuses on the development of automatic systems that can facilitate the labelling and indexing of multimedia documents. Namely, this thesis is articulated around the audio information retrieval domain, which seeks to extract meaningful information from audio streams. This field includes several disciplines that aim to automatically obtain different kinds of information from an audio signal. For example, automatic speech recognition (ASR) seeks to convert an speech signal into its textual transcription. Speaker identification aims to determine the speaker of an audio fragment. In the field of music information retrieval (MIR), applications can be find that recognise the genre of a musical piece, or that estimate the chord being played in a music segment. Among all these possible applications, this dissertation is devoted to audio segmentation. The goal of a generic audio segmentation system is to provide a sequence of labels that isolates different regions in an input audio signal according to the characteristics described in a

predefined set of classes, e.g., speech, music or noise. This definition is really wide, and it includes several kinds of systems depending on the classes taken into account in the classification. For example, a speech activity detection (SAD) task is considered if a binary speech/non-speech segmentation is performed. Another example of audio segmentation is the speaker diarisation task, that aims to separate different speakers in an audio stream defining one class per speaker. A more detailed description of the audio segmentation systems will be provided in Chapter 2.

From an overall perspective, an audio signal can be classified as speech, music or noise. Several works can be cited that adopt a similar taxonomy [4] [5] [6]. Determining the segments of an audio stream where speech is being uttered may be interesting for different reasons. An initial segmentation marking speech fragments can serve as a guide for other automatic systems that should work only on speech fragments. In fact, this preprocessing is an essential step before applying other speech information retrieval applications such as ASR or speaker recognition. The separation of music content in an audio signal is also relevant from a document information retrieval perspective. Furthermore, in broadcast content, music detection plays a key role in order to monitor copyright infringement. Concerning noise separation, it is true that noise may not be as informative as speech or music but its accurate detection could be relevant seeking to reduce it by running some enhancement algorithm on the audio signal.

Research work developed in this thesis dissertation has been performed in the Voice Input Voice Output Laboratory (ViVoLab)[1] research group within the Aragón Institute for Engineering Research (I3A) of the University of Zaragoza, and under the supervision of Dr. Alfonso Ortega Giménez.

1.2 Thesis objectives

The main goal of this dissertation is the development of new solutions based on deep neural networks for audio segmentation, evaluating different binary and multiclass tasks and involving different acoustic classes that need to be separated. Technical developments in the last decade have significantly improved the performance of automatic information extraction systems, with deep learning applications being specially relevant nowadays. In fact, the use of deep learning for automatic labelling and information extraction has become a de facto standard in the audio processing community. Additionally, current technology capabilities allow its use in industry and other environments outside research discipline. Despite recent advances in deep learning research, there are still several issues that need further research.

Generally, deep learning algorithms require large amounts of manually labelled training examples in order to obtain a reliable model. Under some circumstances, this labelled data may not available, or its collection may be an arduous task. Research effort in this dissertation seeks to overcome the problems derived from labelled data availability by exploring different techniques that can improve audio segmentation performance even

[1] https://vivolab.i3a.es

when these systems are trained with a limited size dataset. Furthermore, the generalisation issue is also considered in this dissertation. Some of the experiments presented in this dissertation aim to obtain robust audio segmentation models that can operate under different domain conditions.

Self-supervised representation learning has recently attracted attention in the research community due to its demonstrated capabilities on several speech and audio processing tasks. So far, few works have evaluated the performance of these models in audio segmentation problems. Part of the work presented in this dissertation aims to benefit from the knowledge acquired from huge amounts of unlabelled data in order to be applied to different audio segmentation tasks.

The broad objective of this thesis, consisting of developing new deep learning applications for audio segmentation tasks, can be divided into the following specific objectives:

- The application of deep learning systems on audio segmentation tasks in order to evaluate the current limitations of this technology for different scenarios.
- The development and evaluation of techniques that reduce the loss in performance observed when audio segmentation technologies operate in a data domain different from the one they were trained on, seeking to efficiently adapt current systems to new unseen domains.
- The adaptation of self-supervised feature representation models to audio segmentation systems as a way to alleviate the need for large amounts of labelled data.
- The introduction of new training objectives in the audio segmentation task that can improve the results of these systems when dealing with limited training data.

1.3 Thesis organisation

A conceptual map describing the content of this thesis dissertation is presented in Figure 1.1. As shown, this dissertation is divided into five main parts. In the first part (Chapters 1, 2, 3 and 4), an introduction to the context and motivation of this dissertation is presented initially. Then, a review on audio segmentation solutions is provided, as well as a description of the most relevant methods and techniques involved in this dissertation. Finally, this part describes the datasets used in this dissertation and the metrics commonly applied in the evaluation of audio segmentation systems. The second part (Chapters 5, 6 and 7) is devoted to the speech activity detection task, presenting a set of experiments performed in a challenging acoustic environment such as Apollo space missions recordings. The third part (Chapters 8 and 9), describes the advances obtained using a recurrent neural network based approach in a multiclass audio segmentation task that aims to separate speech, music, noise or a combinations of these. In the fourth part (Chapters 10 and 11), a new optimisation framework based on the area under the ROC curve is applied to the music detection task. This framework is then extended in order to be used in a multiclass environment. Lastly, the fifth part, consisting of Chapter 12, summarises the main

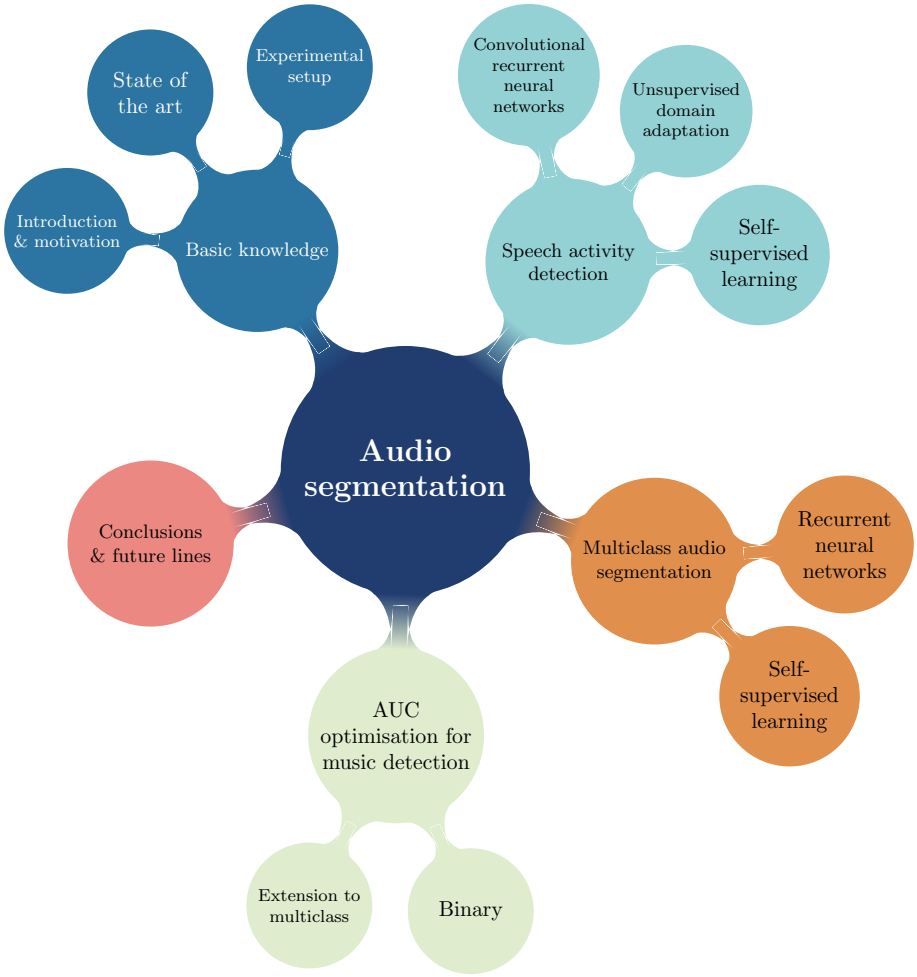


Figure 1.1: Conceptual map of the work presented in this thesis dissertation.

conclusions obtained from this thesis dissertation and introduces some future research lines. In the following lines a description of the organisation for the different chapters in this document is provided:

- **Chapter 1. Introduction and motivation**: in this present chapter, the work described in this dissertation is contextualised. In addition, the motivation and objectives of the thesis are presented.
- **Chapter 2. State-of-the-art in audio segmentation**: this chapter introduces the concept of audio segmentation and performs a review of the state-of-the-art for these systems. This review describes the different approaches used in the audio segmentation community, from traditional methods based on distance metrics or statistical classifiers to more recent solutions based on deep learning. An spe-

cial focus is made on this last kind of systems being the ones mainly used in this dissertation.

- **Chapter 3. Methods review**: this chapter aims to provide an outline of the most relevant methods and techniques applied in this dissertation. Namely, the focus is set on two topics: signal representation methods, describing feature extraction techniques, and sequence modelling techniques, introducing an overview of neural network classifiers and hidden Markov models.

- **Chapter 4. Experimental framework**: this chapter characterises the general experimental framework used in this dissertation. First, a description is presented for the different datasets used in the experiments of this PhD thesis. Then, an overview of the possible evaluation methods and metrics for audio segmentation is provided.

- **Chapter 5. Convolutional recurrent neural networks for speech activity detection**: the work presented in this chapter explores the use of convolutional recurrent neural networks (CRNN) in the SAD task. Several alternatives are evaluated using a variety of convolutional processing stages. A novel architecture is proposed in order to fuse information captured by 1D and 2D convolutional filters. Models presented were used to participate in the Fearless steps challenge 2020, featuring a dataset coming from Apollo space missions.

- **Chapter 6. Unsupervised domain adaptation of speech activity detection models**: deep learning solutions usually show a significant drop in performance when test data are different from training data due to the domain shift observed. Furthermore, machine learning algorithms require large amounts of labelled data, which may be hard to obtain in real applications. Considering both ideas, in this chapter we consider different unsupervised domain adaptation techniques in order to be applied to the SAD task. A baseline system is trained on a combination of data from different domains and then adapted to a new unseen domain, namely, data from Apollo space missions coming from the Fearless steps challenge 2020.

- **Chapter 7. Self-supervised representation learning for speech activity detection**: influenced by the great results obtained in several speech processing tasks by self-supervised representation learning systems, and building upon our previous work in Chapter 5, in this chapter we propose the introduction of the self-supervised representation learning paradigm in the SAD task in order to obtain new features from audio signals more discriminative than traditional perceptual features. This approach is evaluated in the datasets released for the 2021 and 2022 versions of the Fearless steps challenge, focused on testing system generalisation capabilities to varying data conditions.

- **Chapter 8. Recurrent neural networks for multiclass audio segmentation**: in this chapter, we introduce an approach based on the use of recurrent neural networks to the multiclass audio segmentation task whose goal is to separate audio fragments containing speech, music, noise or a combination of those. We explore

different neural architectures introducing temporal pooling layers to reduce the neural network output sampling rate. This new architecture is also combined with mixup augmentation, a data-agnostic data augmentation technique. The described proposal is evaluated in both the Albayzín 2010 and 2012 audio segmentation evaluation datasets

- **Chapter 9. Wav2vec representation learning for multiclass audio segmentation**: Considering the previous experiments presented in Chapter 7 as starting point, the work presented in this chapter describes a study on the use of new self-supervised representation architectures seeking to jointly model speech and music fragments of audio signals in the multiclass audio segmentation task introduced in the previous chapter. Different data conditions and data augmentation policies are considered in order to maximise the performance of the obtained representations to describe both speech and music fragments simultaneously.

- **Chapter 10. Binary AUC optimisation for audio segmentation**: motivated by recent advances in metric learning, in this chapter we introduce the area under the receiver operating characteristic (ROC) curve (AUC) and partial AUC (pAUC) loss functions into the audio segmentation field to improve its performance under limited training data conditions. The work presented explores different threshold-independent metric training objectives in a binary audio segmentation task. Furthermore, we propose a novel training objective based on the decomposition of the AUC as the sum of two pAUCs.

- **Chapter 11. Generalising AUC optimisation to multiclass classification**: as a natural extension of the work presented in the previous chapter, in this chapter we propose a generalisation of the AUC optimisation techniques so that they can be applied to an arbitrary number of classes. This is done using the multiclass variations of the AUC metric proposed on the literature as the starting point for the formulation of the training objectives. Our proposal is validated experimentally in a 3 class segmentation task, seeking to overcome the issues derived from a limited training data scenario.

- **Chapter 12. Conclusions & future lines**: This final chapter describes the main conclusions extracted from this thesis dissertation and a proposal of future research lines.

"The science of today is the technology of tomorrow."

Edward Teller

2

State-of-the-art in audio segmentation

2.1 What is audio segmentation?

According to the Cambridge English dictionary, segmentation can be defined as the division of something into smaller parts. Adapting this definition to the the context of this dissertation, audio segmentation could be defined as the division of an audio signal into smaller fragments according to a predefined set of rules so that each fragment contains only information from an specific audio typology. This definition is really wide and it includes a variety of systems depending on the set of rules used to separate the audio signal. The specifications provided in the set of rules will define the different acoustic classes that the audio signal will be divided into. Furthermore, the information provided by each acoustic class will also depend on the set of rules applied.

Considering the description provided, a generic audio segmentation system receives as input an audio signal and, after performing the needed computations, it generates a set of labels representing the obtained fragments as output. These labels indicate start time,

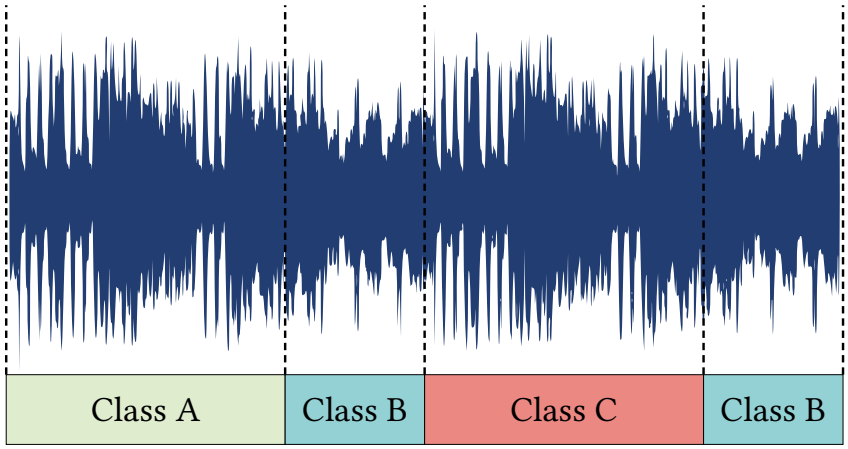


Figure 2.1: Schematic representation of a generic audio segmentation system output on top of a waveform.

end time and the corresponding acoustic class for each fragment. Figure 2.1 represents graphically this idea, by showing the output of a generic audio segmentation with three classes (A, B and C) synchronised with the input audio signal.

Suppose an example where a system is needed to separate male and female speech in an audio stream. An audio segmentation system working under this set of rules will need to differentiate three acoustic classes: male speech, female speech and non-speech. Suppose now that the system also needs to differentiate adult speech and children speech. In this scenario, an audio segmentation system will operate with five acoustic classes: adult male speech, children male speech, adult female speech, children female speech and non-speech. According to the example provided, it becomes apparent that one of the main differences between audio segmentation systems comes from the acoustic classes that must be differentiated. By increasing the number of classes in the system, acoustic classes become more specific but audio segmentation is generally harder, as it is the case with all multiclass classification problems [7].

The final purpose of an audio segmentation system depends then on the acoustic classes considered. Due to its relevance, some audio segmentation tasks that deal with certain acoustic classes are named under an specific denomination:

- **Speech activity detection (SAD)** [8]: SAD systems perform a binary segmentation whose target is to discriminate audio segments containing human speech.
- **Music detection** [9]: In a similar task to SAD but considering music instead of speech, music detection systems aim to separate music and non-music audio fragments.
- **Speaker diarisation** [10]: the aim of speaker diarisation is to segment an audio signal so that different speakers can be separated. In this case, one acoustic class per speaker is needed.

In addition, in this work we present an audio segmentation task that could be seen as a combination of SAD and music detection with a third class modelling noise. This task, proposed originally in the Albayzín 2010 challenge [11], aims to detect segments containing music, noise, speech or a combination of the three. A more detailed description of this task will be provided in Chapter 4.

In this dissertation, we use the concept "audio segmentation" to refer to the described task. However, some other notations can be found in the literature for similar tasks. For example, the concept "sound event detection" (SED) [12] is also used to describe tasks that have a similar goal of recognising at what temporal instances different sounds are active within an audio signal. Unlike audio segmentation, in general terms, SED systems tend to deal with non-speech and non-music sounds [13]. This makes these systems differentiate from more established speech and music analysis tasks, towards the field of environmental sound analysis.

Other terminologies associated with audio information extraction tasks such as "acoustic scene classification" [14] or "audio tagging" [15] strongly differ with the task we refer to as audio segmentation due to the use of weak labels as output. Weak labels only inform about the presence of an acoustic class in the whole input audio, without providing start or end times for that acoustic class. Labels used in audio segmentation are considered as strong labels because the annotation contains temporal information for each acoustic class, namely its start and end times.

2.2 Paradigms in audio segmentation

Most literature reviews on audio segmentation tend to differentiate two paradigms according to how the segmentation is performed: the segmentation & classification approach and the segmentation by classification approach [16]. The first approach can also be described as distance based segmentation due to the use of a distance metric to compute class boundaries. Obtained segments are then classified in a posterior stage. On the other hand, the second approach obtains its segmentation hypothesis directly as a sequence of decisions over the input audio. In the following lines a more detailed description for both paradigms is provided:

2.2.1 Segmentation & classification

Segmentation & classification algorithms compute a metric between two audio fragments with the objective of deciding if an acoustic class change occurs between both fragments. That is the reason why they are also known as distance based audio segmentation methods. These kind of systems separate fragments sharing a common acoustic class without providing an explicit labelling for them. Suppose two audio segments i and j with feature sequences X_i, X_j of length N_i and N_j respectively. In some approaches, a third audio segment can be generated by concatenating i and j, with a feature sequence $X_{ij} = \{X_i \cup X_j\}$.

According to the provided definitions, a distance based segmentation system can generate two different hypotheses:

- **Hypothesis $\mathbf{H_0}$**, or null hypothesis, affirming that segments i and j belong to the same acoustic class.
- **Hypothesis $\mathbf{H_1}$**, or alternative hypothesis, affirming that a change in acoustic class exists between segments i and j.

Therefore, the main goal of segmentation & classification methods is finding a distance $D(i, j)$ between segments i and j in order to determine wich one is the correct hypothesis H_0 or H_1. The metric obtained will be compared with a threshold ϵ to select one of the two hypotheses, as shown in Equation (2.1)

$$D(i, j) \underset{H_0}{\overset{H_1}{\gtrless}} \epsilon \,. \tag{2.1}$$

The application of this strategy over consecutive audio frames may lead to a noisy estimation of changes in the acoustic class. In order to improve the accuracy, distance is usually computed using a window considering L frames from the audio signal. As a final remark, it is worth mentioning that distance based segmentation systems provide as output a set of segments that do not belong to any specific acoustic class. An additional processing step will be needed in order to assign a class to each one of the segments.

Distance metrics

A number of distance criteria can be found in the audio segmentation literature. In the following lines we describe the most widespread metrics applied in segmentation & classification approaches:

- **Bayesian information criterion (BIC)** [17]: This metric assesses the goodness of fit of a given statistical model considering its complexity measured in number of parameters. Given a set of samples X obtained from a random process and a model θ describing these samples, BIC can be defined according to Equation (2.2)

$$\mathrm{BIC}(X|\theta) = \log\left(\mathfrak{L}(X|\theta)\right) - \lambda\frac{1}{2}\#(\theta)\log N \,. \tag{2.2}$$

  The first term of the equation represents the log likelihood of the data X given the model θ. The second term aims to penalise the considered model θ according to its complexity. $\#(\theta)$ represents the number of free parameters of the model θ and λ is a modifiable weight for the penalisation term.

  BIC can be easily applied to determine if a change in acoustic class occurs between two segments i and j. Two BIC values need to be computed: a first value considering hypothesis H_0, where both segments can be correctly described by a single model θ_{ij}, and a second value that assumes hypothesis H_1, where data from each

segment should be explained by a different model, θ_i and θ_j, respectively. To obtain a distance metric, the difference between both values is computed, obtaining what is usually known as ΔBIC [18], as described in Equation (2.3)

$$\Delta\text{BIC}(X) = \text{BIC}(X|H_1) - \text{BIC}(X|H_0)\,. \tag{2.3}$$

As some examples of application of this metric, the work described in [19] applies a ΔBIC segmentation to separate languages in a multilingual audio stream. Authors in [20] use BIC applied to the speaker change detection task.

- **Generalised likelihood ratio (GLR)** [21]: Given the already presented two audio segments X_i, X_j and the concatenation of both X_{ij}, this metric is calculated as the likelihood ratio between hypothesis H_0, with both segments belonging to the same class, and hypothesis H_1 where segments are assumed to belong to different classes. Equation (2.4) shows the computation performed to obtain the GLR metric:

$$\text{GLR}(X) = \frac{\mathfrak{L}(X_{ij}|H_0)}{\mathfrak{L}(X_{ij}|H_1)} = \frac{\mathfrak{L}(X_{ij}|\theta_{ij})}{\mathfrak{L}(X_{ij}|\theta_i)\mathfrak{L}(X_{ij}|\theta_j)}\,, \tag{2.4}$$

  where $\mathfrak{L}(X|\theta)$ represents the likelihood of data X given a model θ.

  GLR distance has been applied in different works. For example, the use of GLR is proposed in [22] to separate speakers in an audio stream. A variation on the GLR metric is used to perform online speaker segmentation and polyphonic music segmentation in [23].

- **Others**: some other examples of distance metrics can be found in the literature. The use of self-similarity matrices is proposed in [24] to perform media segmentation. The work presented in [25] uses cosine distance in combination with self-similarity analysis to detect music and speech transitions. Authors in [26] use the symmetric Kullback-Liebler distance as a first preprocessing step in order to evaluate acoustic similarity. More recent work has introduced the use of neural distance metrics, such as the one proposed in [27] for speaker change detection, or the audio novelty measure described in [28] using an autoencoder architecture.

2.2.2 Segmentation by classification

In contrast to distance based audio segmentation systems where only boundaries between segments are detected, the segmentation by classification approach classifies each audio frame as belonging to one specific acoustic class. This classification is performed according to a model that contains a priori information representing each acoustic class. Due to the use of these models, the segmentation by classification paradigm can be also referred to as model based segmentation.

One of the main characteristics of model based segmentation systems is that they have two different working phases. First, in the training phase, the system adjusts its parameters in order to describe the a priori distribution of the available data. Once this

process is over, the system can operate in inference mode. In this phase, new unseen data is presented to the system with the objective of obtaining an estimation of the most likely acoustic class using the information learned in the training phase. Several modelling approaches have been applied in the context of audio segmentation. In the following lines a description is provided for some of the most common modelling techniques used in audio segmentation tasks:

- **Gaussian mixture model (GMM)**: a Gaussian mixture model represents a probability distribution obtained through the weighted sum of different Gaussian distributions. GMMs were widely adopted as an efficient way of modelling multimodal distributions. Given a feature vector x, the probability distribution for a GMM can be defined according to Equation (2.5):

$$P(x|\lambda) = \sum_{k=1}^{K} \omega_k \mathcal{N}(x|\mu_k, \Sigma_k)\,, \tag{2.5}$$

  where K is the number of components in the mixture and $\lambda = \{\omega, \mu, \Sigma\}$ is the set of parameters of the model. For these parameters, $\omega_k, \mu_k, \Sigma_k$ represent, respectively, the weight, the mean and the covariance matrix of component k from the mixture.

  A number of examples of the use of GMMs on audio segmentation tasks can be cited. In [29], a GMM classifier is used to perform real time speech and music segmentation. GMMs are used together with a maximum entropy classifier in [30] to separate speech and non-speech frames in YouTube videos.

- **Hidden Markov models (HMM)**: a Markov chain models a state sequence where the state transition probability depends exclusively on the previous state. In the context of HMMs, this idea is further evolved in order to model stochastic sequences as Markov chains. In this case, the sequence of states is not directly observable, only the output of the stochastic processes for each state can be observed. Due to the use of these models done in part of the experiments described in this dissertation, a more detailed explanation of HMMs is provided in Section 3.3.

  HMMs have been applied to audio segmentation for different purposes. Authors in [31] presented a hierarchical approach to the separation of speech, music and different environmental sounds. The system presented in [32] applied an HMM classifier to discriminate speech and music excerpts. Something similar is done in [33], that proposed the use of wavelet features as input to an HMM system.

- **Joint factor analysis (JFA)** [34]: JFA is a statistical technique used in signal processing and machine learning for analysing multivariate data sets. The JFA model assumes that each observation can be represented as a combination of factors, where the factors are shared across multiple datasets. By jointly modelling the distribution of the datasets, JFA is able to capture the underlying structure and relationships between the variables. The choice of the basic underlying statistical distribution depends on the specific characteristics of the data and the requirements of the application, but it is often based on GMMs. Under this consideration,

a GMM is estimated for each target class. These models are usually obtained by maximum a posteriori (MAP) adaptation of a universal background model (UBM) that represents the distribution of all possible feature vectors that might be encountered in a given application. The means for all the components of the adapted GMMs are concatenated to obtain high dimensionality supervectors that can be decomposed into a number of different parts that represent the different sources of variability in the data under analysis. JFA uses a number of hidden variables that tie together the means of the Gaussians in the adaptation process [35]. Then, the supervector M_{is} for an audio segment belonging to class i and with session conditions s is decomposed according to Equation (2.6):

$$M_{is} = m + Vy_i + Ux_s + Dz\,, \tag{2.6}$$

where m is the UBM means supervector, Vy_i is the term modelling the interclass variability through a low rank matrix V and the vector y_i associated with the class identity i. Term Ux_s is used to model the session variability through a low rank matrix U and the vector x_s associated with session conditions s. Finally, term Dz aims to model residual variability that could not be captured by the class and session terms. This is usually done through a diagonal matrix D and the residual variability vector z.

JFA models were initially developed for the speaker identification and speaker verification tasks [36] [37], yet they have also been applied successfully in audio segmentation tasks in broadcast data environments [38] [39].

- **Neural networks**: A neural network is a computational learning system that uses a network of functions to understand and convert a set of inputs of one form into a desired set of outputs, usually in another form. All the audio segmentation experiments described in this dissertation have been performed using neural networks as the main element of the system. Due to its relevance for this work, a more detailed explanation on the different neural network architectures and the different learning methods will be provided on Section 3.2. Furthermore, an extensive review of the state-of-the-art of neural network systems and deep learning methods in audio segmentation tasks is presented in Section 2.3.

- **Others**: More algorithms and pattern recognition techniques have been applied to the audio segmentation task. Some examples can be cited such as support vector machines (SVM) [40], decision trees [41], or fuzzy logic [42].

2.3 Deep learning for audio segmentation

Since the early 2010s, scientific community has experienced the exponential growth of neural networks and deep leaning applications due to the advances in hardware and the higher availability of data for training. These models have been increasingly applied in audio and speech processing tasks, gradually replacing previous approaches based on

statistical methods. The audio segmentation field has also been influenced by the irruption of deep learning, with most segmentation by classification systems moving towards these kind of models. Intial neural network models for audio segmentation were based on feed forwards neural networks. Research in [26] presents a multilayer perceptron classifier integrated in a system that combines SAD, speaker segmentation and clustering. Authors in [43] use a multilayer perceptron to detect speech segments seeking to improve the performance in a speaker verification task. A multilayer perceptron model is implemented to discriminate speech and music fragments in [44].

Recurrent neural networks and convolutional neural networks

Feed forward neural networks show strong limitations to capture temporal dependencies. Recurrent neural networks solve this issue by introducing a feedback loop between the input and the output of the system. These networks have been applied successfully in several audio processing tasks including audio segmentation. In particular, several examples applying LSTM networks can be found in the literature. The first approach to SAD using LSTM networks was presented in [45], where authors demonstrated the feasibility of this approach in a real-life noisy speech from Hollywood movies. A similar neural architecture is used in [46] to implement a noise-robust vowel based SAD. Furthermore, GRU networks, a lightweight version of the LSTM network, have also been applied to audio segmentation. Authors in [47] analyse the results of a phoneme segmentation system based on GRU cells. The work presented in [48] applied a GRU model to perform singing voice detection.

Convolutional neural networks (CNN), commonly related to the image processing field, were also progressively introduced in different audio processing task with significant results. These kind of models usually rely on time versus frequency representations of the audio signal that are treated as channels in an analogy with image processing systems. Audio segmentation literature also relates several segmentation by classification approaches that are based on these architectures. A fully convolutional architecture is presented in [49] for simultaneous speech and music segmentation. A CNN model featuring kernels based on the Mel scale is used for the music detection task in [50]. Similarly, the work described in [51] separates speech and music in audio streams using a CNN trained in a semi-supervised way. Concerning the SAD task, authors in [52] evaluate a CNN system in mismatched acoustic conditions. CNNs are also typically used to automatically learn and identify important features from audio signals [53] [54]. In this context, an observed common practice in the literature is to combine convolutional blocks that can learn relevant information from data, with the capability of RNNs to deal with temporal series. These models are usually known as convolutional recurrent neural networks (CRNN) and have also been applied in several audio segmentation tasks. Authors in [55] compared different deep learning models for audio segmentation, demonstrating the good performance of CRNN models in a speech versus music separation task. Results described in [56] show a similar trend, with CRNNs outperforming other models for both speech and music detection. The system employing CRNNs presented in [57] ranked first among all submissions to the 2019 Fearless steps challenge SAD task.

A unified approach seeking to combine low level features from CNNs and high level temporal representations from RNNs was presented with the temporal convolution networks (TCN) [58]. These models were initially proposed for activity segmentation in video processing applications, but they have been widely adopted in audio processing and, specifically, in audio segmentation. The study presented in [59] extends TCN models to also capture non causal dependencies within the framework of speech and music detection for broadcast content. A residual TCN model is applied to estimate music relative loudness in [60]. Research described in [61] uses a TCN model to jointly perform sound event detection and direction of arrival estimation.

Attention and self attention models

The introduction of attention in sequence modelling architectures [62], originally proposed for machine translation, solved the problem of capturing alignments between input and output sequences in the common encoder-decoder architecture. RNNs were combined with these attention models seeking to let the network infer how each output token is influenced by some specific parts of the input sequence. Some examples can be cited of the use of attention models in audio segmentation. For instance, attention is applied in [63] in combination with an LSTM network for the SAD task. Heart sounds are segmented by means of a bidirectional LSTM network with an attention module in [64]. Both a temporal and a frequential attention model are implemented in [65] to segment rare occurring sounds in audio recordings. As a further step, the presentation of the transformer architecture [66] in the deep learning community shifted research efforts in audio processing towards purely self attention based models. These kind of models are also being introduced in audio segmentation tasks. In most audio segmentation works using self attention mechanisms, it is usual to discard the decoder block from the original transformer architecture, finding one or more transformer encoder blocks. Concerning the SAD task, the multihead self attention mechanism is used in [67] to detect speech fragments in noisy conditions. Work in [68] splits acoustic features into patches to further introduce locality to the transformer architecture. Transformers have also been applied in the speaker diarisation task, with special relevance in end-to-end approaches [69].

Unsupervised feature representation and self-supervised learning

In recent years, audio and speech research landscape has been disrupted by the advances in unsupervised and self-supervised representation learning [70] [71]. These models benefit from huge amounts of unlabelled data in order to obtain new audio representation that can later be applied in different specific applications. Motivated by the good performance observed across several speech processing tasks, some researchers are starting to apply these models to audio segmentation applications. WavLM features are considered in [72] to detect overlapped speech fragments and separate male and female speech. Authors in [73] tested self-supervised representations for the SAD task under domain shift conditions. A representation learning algorithm is combined with a clustering algorithm to improve diarisation performance in [74]. Even tough, the amount of work considering

these new representations is still limited in the audio segmentation field. Furthermore, most works are limited to speech related tasks, with few systems using them in more general audio segmentation tasks.

Multimodality in deep learning models

In addition to unsupervised and self-supervised learning, multimodality is also playing a key role in the development of recent deep learning models. A number of works aim to integrate different modalities apart from audio in a single unified model so that it can benefit from this additional information. Considering those that combine audio with image or video information, authors in [75] introduced the audiovisual segmentation task, where the goal is to generate a pixel segmentation of the object that produces the sound. Similarly, audiovisual diarisation seeks to separate an stream according to different identities by marked by face and speech information. Several research works have been developed recently targeting this task [76] [77]. Concerning the combination of audio and text modalities, SAMU-XLSR model [78] seeks to describe audio and text in a single multimodal and semantically aligned space. Other works have also developed methods to unify representations from speech and text modalities [79] [80]. Some works have also explored more complex combinations, bringing together audio, text and image modalities [81].

"The human brain is an incredible pattern-matching machine."

Jeff Bezos

3 Methods review

Designing an audio segmentation system requires considering two main issues: deciding how to represent the raw audio signal in the most appropriate way, and then, given the provided representation, finding a way to properly model the information contained in the feature sequence. According to the information extracted from that model, a decision on the segmentation is given. The first idea presented is usually known as feature extraction and it is the first block in most conventional audio segmentation systems. For the second issue presented, audio segmentation systems based on the segmentation by classification approach rely on a classifier. These kind of systems need a training phase where the most relevant information from the feature sequences is extracted and used to adjust the parameters of the classifier underlying model.

In this section, we provide a review on some of of the most well-known methods for feature extraction and sequence modelling that are relevant for this dissertation. For feature extraction, we mainly focus on the description of short-term analysis and some sets of features used in this work such as those based on Mel filter-banks and chroma fea-

tures. Concerning sequence modelling, we provide an overview of the neural network classifiers, ranging from the different architectures employed, to some of the most relevant techniques. This family of classifiers have been the main element for all the audio segmentation systems developed in this work. Furthermore, we also provide a description of another sequence modelling method, the hidden Markov models, that is used as a smoothing technique in some parts of this dissertation.

3.1 Feature extraction

Feature extraction is usually defined as the set of transformations applied to a dataset in order to maximise its discriminative properties, seeking to adapt this data in the most suitable format for the task that is going to be performed. Audio signals are usually considered to be non stationary signals. That is the reason why all the common feature extraction techniques applied in the audio processing field rely on the use of short-term analysis. The use of short temporal windows (25 - 50 ms) allows to assume a quasi-stationary behaviour. The general description for short-term analysis is shown in Equation (3.1):

$$Q_n = \sum_{k=-\infty}^{\infty} T(x[n]w[n-k]), \tag{3.1}$$

where $x[n]$ is the audio signal being analysed, $w[n]$ is the K samples length window applied to the signal, and $T(\cdot)$ is the transformation performed. Some of the most common transformations applied are the Fourier transform, Mel filter-bank energies, Mel frequency cepstrum coefficients (MFCC) [82], or perceptual linear prediction (PLP) [83].

3.1.1 Perceptual features: MFCCs and Mel filter-bank energies

In the following lines we provide a more detailed description for MFCCs and Mel filter-bank energies. These two transformations have been used in several task among the speech and audio processing community, becoming one of the most common choices for feature extraction in these kind of signals. Both transformations are derived from the concept of cepstral analysis. A brief introduction to this topic is also provided.

Cepstral analysis is a common processing technique for speech signal representation. The most interesting property of cepstral analysis is the fact that it transforms a convolution in the temporal domain into a sum in the cepstral domain, as shown in Equation (3.2):

$$x_1[n] * x_2[n] \xrightarrow{\text{cepstrum}} \hat{x_1}[n] + \hat{x_2}[n]. \tag{3.2}$$

A common framework for modelling a speech signal is considering a signal, $v[n]$, as an excitation $e[n]$ being convolved with a filter, $h[n]$, that represents the vocal tract, resulting in $v[n] = e[n] * h[n]$. By applying the mentioned cepstral analysis techniques, this becomes $\hat{v}[n] = \hat{e}[n] + \hat{h}[n]$ in the cepstral domain, resulting in a representation that can be manipulated in a simpler way.

Let $x[n]$ be the audio signal being analysed, the complex cepstrum of $x[n]$ can be defined according to Equation (3.3):

$$\hat{x}[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln\left(X(e^{j\omega n})\right) e^{j\omega n} d\omega\,, \tag{3.3}$$

where $X(e^{j\omega n})$ is the Fourier transform of $x[n]$. Therefore, the complex cepstrum of $x[n]$ is defined as the inverse Fourier transform of the natural logarithm of the Fourier transform of $x[n]$. In audio signals, most relevant information is usually in the modulus of its Fourier transform, that is why phase information is sometimes discarded. In order to consider this fact, Equation (3.4) describes the expression of the real cepstrum that only considers the magnitude information of the Fourier transform.

$$c[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln\left(|X(e^{j\omega n})|\right) e^{j\omega n} d\omega\,. \tag{3.4}$$

MFCCs are based on the Mel frequency scale. This scale represents the frequency axis linearly up to 1000 Hz and in a logarithmic way from that point on. This scale is derived from the human audition perception model, whose frequency discrimination is bigger for low frequencies than for high frequencies [84]. Conversion from the linear frequency scale obtained by the Fourier transform is performed by appling a filter-bank with a narrow bandwidth for low frequencies and a wider bandwidth for high frequencies. An example of a typical Mel scale filter-bank is depicted in Figure 3.1. This particular

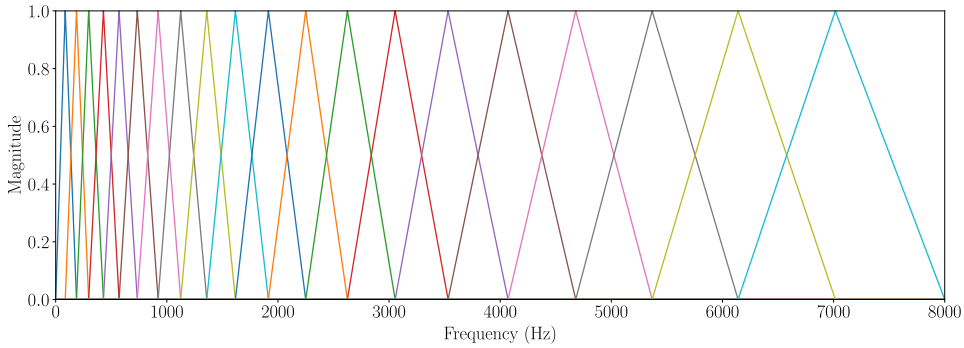


Figure 3.1: Example of a typical filter-bank used for conversion to Mel scale spectrogram.

configuration uses 20 Mel filters and a sampling rate of 16 kHz. In order to obtain the Mel scaled spectrogram the Fourier transform magnitude is forwarded through each one of the filters. Then the average energy is computed for the output of each filter. As a last step to obtain the MFCC features, the discrete cosine transform (DCT) is applied. This transformation is used mainly seeking to obtain a decorrelated set of features. According to the provided explanations, Figure 3.2 shows the block diagram for MFCCs feature extraction. By applying a similar processing pipeline but considering the output one step before applying the DCT, another set of features based on the Mel perceptual scale can be obtained. These features are usually known as Mel filter-bank energies. The main difference observed with MFCCs is that, by not applying discrete cosine transform,

Figure 3.2: Block diagram for Mel frequency cepstrum coefficients feature extraction.

the correlation among coefficients is preserved. The block diagram associated with Mel filter-bank coefficients extraction is depicted in Figure 3.3.

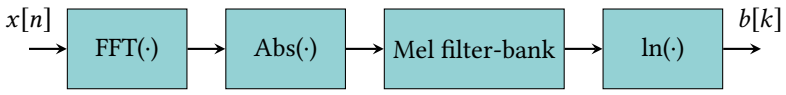


Figure 3.3: Block diagram for Mel filter-bank coefficients feature extraction.

Seeking to represent in an accurate way temporal variations on the input signal, first and second derivatives of the cepstral parameters are usually computed and concatenated to the static coefficients. In a similar way, input signal energy is also useful for the feature extraction pipeline and can also be concatenated to the original features to provide more information.

3.1.2 Chroma features

Part of the work described in this manuscript is based on the detection and segmentation of music or a combination of music with different acoustic classes. Mel based features have also been successfully applied in a number of music related task, but when considering the representation of audio signals with musical content there exist other options that can capture its intrinsic properties in a better way. One of the most common representation it is the chromagram [85] [86], describing the temporal evolution of the chroma features in an audio signal.

The theoretical basis of chroma features is that human perception of pitch is periodic. This means that two pitches are perceived as similar if they differ by an octave. According to this idea, a pitch can be then separated in two components, which we refer to as tonal height and chromatic component [87]. Assuming the equal-tempered scale as used in Western music notation, chroma components are usually associated to the set {C, $C^{\sharp}$, D, $D^{\sharp}$, E, F, $F^{\sharp}$, G, $G^{\sharp}$, A, $A^{\sharp}$, B} that consists of twelve different pitch spelling attributes[1]. Under this representation, chroma and height can be described according to the frequency helix model, shown in Figure 3.4.

Chroma features combine the information belonging to a chroma component for all the possible tonal heights. Therefore, a chroma feature is a 12-dimensional vector, one for each chroma component. By performing this process for all the audio signal, through

[1] In the equal-tempered scale, different spellings of the same pitch such as $F^{\sharp}$ or $G^{\flat}$ identify the same chroma component.

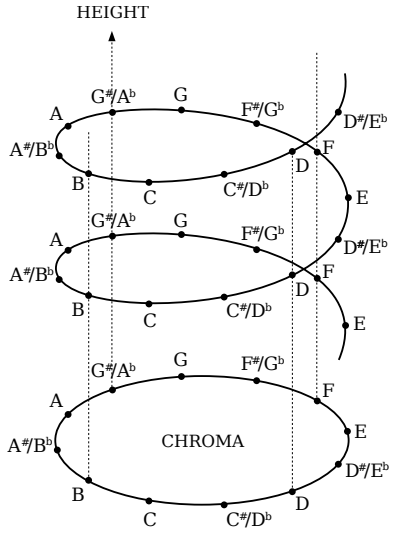


Figure 3.4: Height and chroma components representation in the frequency helix model.

short-time analysis, a sequence of coefficients is obtained that expresses how the energy is spread across the 12 chroma components defined.

The property of identifying pitches differing by an octave is what makes chroma features highly robust to variations in timbre, showing a strong correlation with the musical aspect of harmony. That is the reason why chroma features have been applied in several musical information retrieval tasks such as chord recognition [88] or synchronisation of musical sequences [89].

3.2 Neural networks

Neural networks describe a computation paradigm based on learning non linear transformations of the input through examples. Neural network parameters are then modified consequently according to the information provided by those examples. The concept of neural network is not novel, in fact, first models were introduced in the forties [90]. Most algorithms and techniques related with this field were developed in the seventies and in the eighties. However, scientific community had to wail until around year 2010 in order to be able to benefit from the full potential of neural networks. This was mainly due to important advances in hardware and the bigger availability of data needed for training. Neural networks became state-of-the-art in many pattern matching fields such as as computer vision [91] or automatic speech recognition [92].

3.2.1 Neural network architectures

Feed forward neural network

The term feed forward is normally used to refer to the most basic neural network architecture. Unlike more complex architectures, in this kind of systems there is no feedback loop between the input and the output. Feed forward networks are also called perceptrons or multilayer perceptrons (MLP) in case the system has multiple layers stacked. Some other notations based on its connectivity structure refer to them as dense or fully connected networks.

Feed forward neural networks are built by combining a number of basic processing units called neurons. Figure 3.5 represents one of these basic neurons. As it can be seen, it

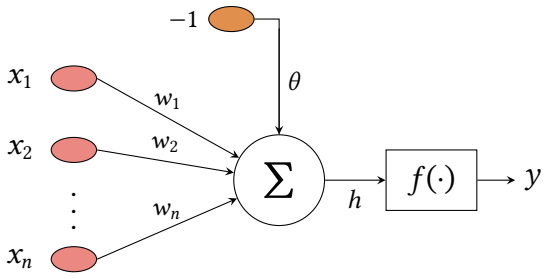


Figure 3.5: Basic neuron processing unit in a neural network.

receives n different inputs, x_j, that may come from an external source or from the output of a previous neuron. Then, a linear combination of the inputs is performed with a set of weights, w_j, considering also a constant term θ in this combination. This term is usually known as bias. This leads to the expression for the term h described in Equation (3.5):

$$h(x, w, \theta) = \sum_{i=1}^{n} w_i x_i - \theta \,. \tag{3.5}$$

The final output of the neuron, y, is obtained by applying the function $f(\cdot)$ to the term h, as expressed in Equation (3.6):

$$y = f(h) = f\left(\sum_{i=1}^{n} w_i x_i - \theta\right) . \tag{3.6}$$

For simplicity, matrix notation is commonly used to describe this expression. Assuming a vector $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$ representing the inputs, and a vector $\mathbf{W} = \{w_1, w_2, \dots, w_n\}$ representing the weights, a compact expression of the basic neuron in matrix format is shown in the following equation:

$$y = f\left(\mathbf{X}\mathbf{W}^T - \theta\right) . \tag{3.7}$$

The function $f(\cdot)$ applied over the weighted average of the input is defined as activation function. Several implementations for this function can be found in the literature, all

of them sharing the non linear behaviour. Two of the most widespread options are the sigmoid function and the hyperbolic tangent function.

The union of several basic neurons builds a neural network. Figure 3.6 shows an example multilayer perceptron architecture with a single hidden layer. As it can be seen,

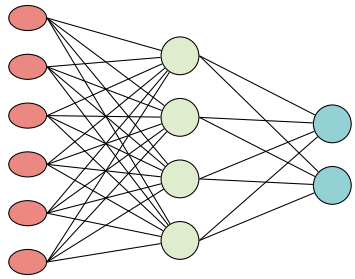


Figure 3.6: Multilayer perceptron with a single hidden layer.

neurons work in parallel forming layers and computing its outputs. These outputs may serve as input to the next layer, showing a dense interconnection where the output of a neuron in the layer i serves as input for all the neurons in the layer $i + 1$. Furthermore, Figure 3.6 serves also to illustrate the different kinds of layer forming a neural network. Neurons in red represent the input layer, while neurons in blue are associated with the output layer. Neurons in green form a hidden layer. The amount of hidden layers in a neural network defines its modelling capabilities and its complexity.

The introduction of supervised learning techniques allowed for the effective application of neural networks in real solutions. Two main components need to be introduced: gradient descent optimisation techniques that iteratively update the model weights according to an objective function, and the backpropagation algorithm to efficiently calculate the gradients in complex architectures. Due to the relevance of this topic for all the neural network architectures explained in this section, a more detailed explanation for supervised learning techniques is provided in Section 3.2.2.

Recurrent neural networks

Recurrent neural networks (RNN) are a special kind of neural network that are able to model temporal dependencies by introducing feedback loops between the input and the output of the network. Figure 3.7 shows a general representation of a recurrent neural network, describing the two possible ways to interpret its way of working. On the left, we can see the already mentioned concept of an RNN being described using a feedback loop between the input and the output. On the right part of the figure, an equivalent version can be observed with the RNN being described as multiple copies of a neural network, each one of them passing a message to the next one. This idea is based on one of the commonly used algorithms for training RNNs, known as backpropagation trough time (BTT) [93].

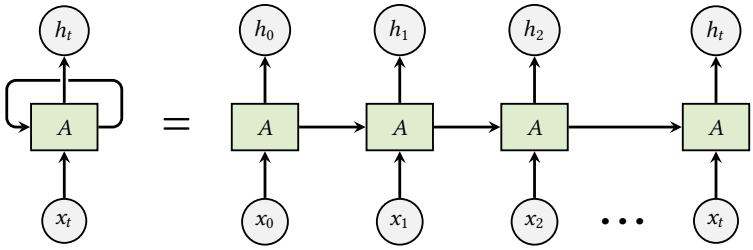


Figure 3.7: General representation of a recurrent neural network. Adapted from [94]

Despite recurrent neural networks being able to capture long-term dependencies theoretically, this does not occur in practice. It has been demonstrated that they suffer from a number of problems associated to gradient computation such as vanishing gradient and exploding gradient [95]. That is the reason why several improved versions of the basic RNN model were developed to address this issue. One of the most widespread variants is the long short-term memory (LSTM) network, whose main difference is the introduction of a new structure known as memory cell.

Long short-term memory networks LSTM networks [96] are an evolution over the basic RNN model that are able to capture simultaneously long and short term dependencies on an input sequence. Compared to the basic RNN, this kind of model shows a significant increase in computational complexity. LSTM networks compute several interconnected intermediate steps in an specific structure, the LSTM memory cell. Figure 3.8 shows the definition of the mentioned LSTM memory cell.

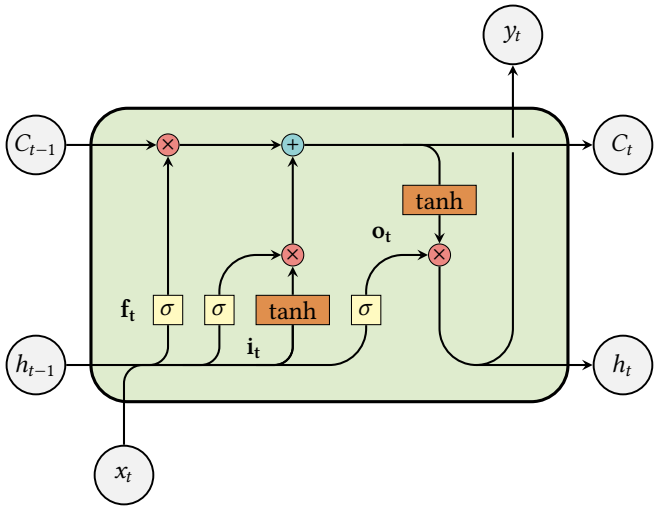


Figure 3.8: Overview of an LSTM memory cell structure. Adapted from [94]

The key element of LSTM networks is the cell state (C_t), a memory unit that is propagated through time. LSTM networks are able to modify this cell state, adding or removing

information, by applying an specific structure usually named as gate. These gates are built using a neural layer with sigmoid activation (output between 0 and 1). An LSTM memory cell is made of three different gates:

- **Forget gate** ($\mathbf{f_t}$): this gate combines information from the current input, x_t, and the previous output, h_{t-1}, in order to decide the amount of information from the previous cell state, C_{t-1}, that must be removed.
- **Input gate** ($\mathbf{i_t}$): this gate updates the current cell state, C_t, deciding the amount of information from the current input, x_t, that is included in the memory cell.
- **Output gate** ($\mathbf{o_t}$): this gate generates the current output of the memory cell by calculating a filtered version of the cell state, C_t, guided by the current input, x_t.

A further evolution on the idea of the LSTM network is the bidirectional LSTM network (BLSTM). In this architecture two LSTM networks operate on the same sequence. The first one considers the sequence as it is and the second one works on a reversed version of the sequence. This allows to capture simultaneously causal and anticausal dependencies at the expense of increasing the complexity of the model and not being able to work in real time. In addition to LSTM networks, it is also worth mentioning the gated recurrent units (GRU) [97], a lightweight variant of the LSTM architecture that uses only two gates instead of three.

Convolutional neural networks

The theoretical foundations of convolutional neural networks (CNN) [98] are similar to those described for feed forward networks. Both consist of a set of trainable parameters and perform a non linear transformation of its inputs. The main difference comes from the fact that, in convolutional neural networks, each neuron is connected only to a small region of the input, performing a localised analysis of the input. This is done applying a convolution operation instead of a matrix multiplication as done in feed forward networks. By sharing the weights among all the small regions of the input, this kind of architectures can significantly reduce the number of trainable parameters and the risk of training overfitting. CNNs were originally designed to work with 2D matrices as input, namely images. There are also other variants that can work with 3D or 1D data, these last ones being specially relevant in the audio processing community.

In order to build these networks, an architecture like the one shown in Figure 3.9 is usually followed, where convolution layers and pooling layers are combined. Final classification is usually performed using one or more dense layers as it is also shown in the figure. Convolution layers are the core element of CNNs. They are made of a number of different trainable filters that process the input signal and extract a feature map from it. An schematic overview describing the working mechanism of a convolution layer is presented in Figure 3.10. As it can be observed, the filter (depicted in gray) is overlapped with the input signal and then the corresponding part of the output is generated considering only the region delimited by the filter size and applying a convolution operation.

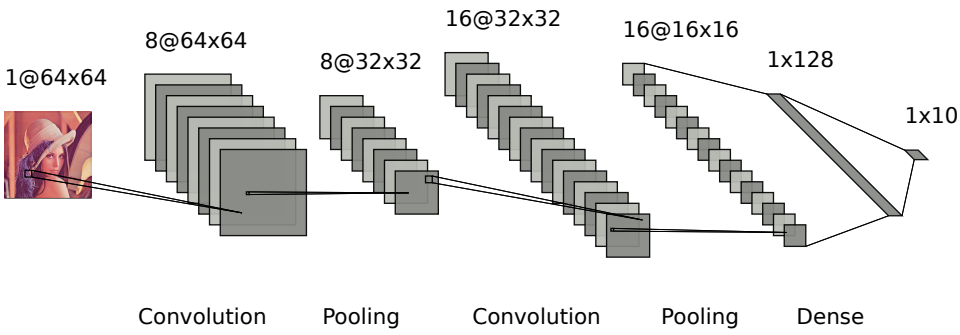


Figure 3.9: Convolutional neural network architecture example.

The filter is then moved along the signal dimensions using a sliding window to generate the full output feature map. Dashed squares represent the padding added to signal in order to maintain the same size in the output.

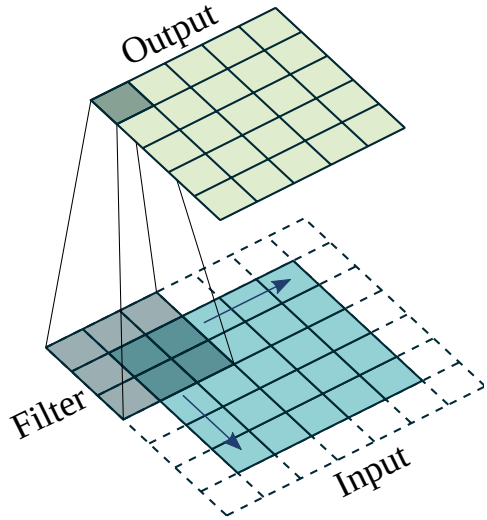


Figure 3.10: Overview of a convolution layer. Adapted from [99]

The application of the convolution operator is a linear procedure. In order to be able to capture complex relationship in data, a non linear activation function is usually implemented after each convolution. One of the most commonly applied non linearities in the context of CNNs is the rectified linear unit (ReLU) [100]. This function is linear if the input is positive or zero, and is zero for negative inputs.

Pooling layers perform a subsampling of the input data in order to reduce its dimensionality. The purpose of this reduction its twofold. By reducing the dimensionality of the intermediate representations, a lower computational load and processing time is obtained in the overall model. Also, this mechanism reduces the probability of the neural network overfitting to training data. The most common pooling mechanisms are max pooling and mean pooling. An example of the application of both of them is presented in

Figure 3.11. As it can be observed, both provide an output that is smaller than the input. The output feature map aggregates the most relevant statistics of the input by computing maximums or means, respectively, from $N \times M$ regions from the input. An interesting

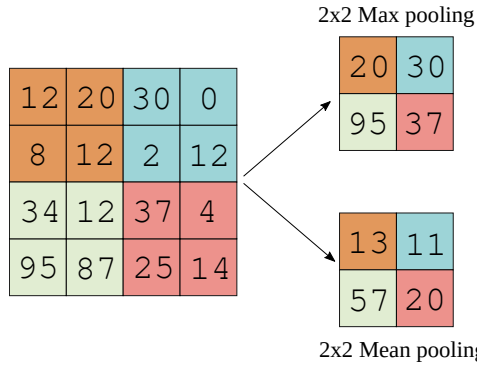


Figure 3.11: 2x2 max pooling and 2x2 mean pooling example.

property of CNNs that is a result of pooling operations is the translational invariance, meaning that representations obtained are not sensitive to positions in the input signal.

Self attention mechanisms and transformer architectures

While RNNs have demonstrated its great performance in sequence modelling tasks, one of the main drawbacks of this kind of structure comes from its sequential nature, that strongly limits the parallelisation. Attention models emerged in order to overcome this issue [62]. The transformer architecture [66], originally proposed in 2017, was the first system to entirely rely on attention for sequence modelling. The original paper reached a new state-of-the-art on the text translation task while allowing for more parallelisation, significantly reducing the training time. In recent years, transformers are being successfully applied to several sequence modelling problems such as ASR [101], language modelling [102] or video recognition [103].

An attention model is characterised as the mapping process of a query and a set of pairs, key and value, to an output. This output is obtained by computing a weighted average of the values, where the weight for each value is estimated by a function involving the query and the corresponding key. In general terms, if queries are generated from a different sequence than keys and values then we refer to this mechanism as cross attention. If keys, values and queries come from the same sequence, then it is called self attention.

The key element of the transformer architecture is the multihead self attention (MSA) mechanism. The main idea is to perform a number h of self attention mechanisms in parallel. In the following lines, we first provide an explanation of the dot self attention mechanism, used as building block of the MSA algorithm. The overview of a single self attention mechanism is presented in Figure 3.12. As observed, the input $\mathbf{x}$ is projected through 3 different linear layers to obtain the queries $\mathbf{Q}$, keys $\mathbf{K}$ and values $\mathbf{V}$. This

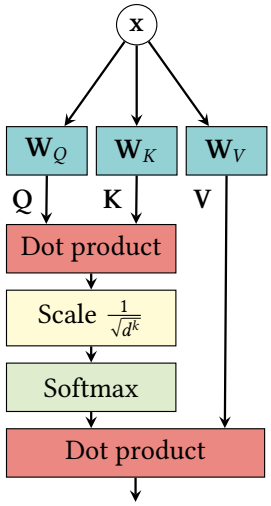


Figure 3.12: Overview of the dot product self attention mechanism.

projection is described in Equation (3.8)

$$\mathbf{Q} = \mathbf{x} \cdot \mathbf{W}_Q\,, \quad \mathbf{K} = \mathbf{x} \cdot \mathbf{W}_K\,, \quad \mathbf{V} = \mathbf{x} \cdot \mathbf{W}_V\,, \tag{3.8}$$

where $\mathbf{W}_Q$, $\mathbf{W}_K$ and $\mathbf{W}_V$ represent the learnable weight matrices used to project the input and obtain $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ respectively. Then a softmax activation is applied on the sequential axis, combining information from $\mathbf{Q}$ and $\mathbf{K}$ using a dot product. The result of this softmax is usually known as self attention matrix and is computed as described in Equation (3.9):

$$\mathbf{A} = \text{softmax}\Big(\frac{\mathbf{Q} \cdot \mathbf{K}^T}{\sqrt{d^k}}\Big)\,, \tag{3.9}$$

where d^k is the dimension of the key and query vectors. Final output is then computed with another dot product between the self attention matrix and the values, as follows:

$$\mathbf{O} = \mathbf{V} \cdot \mathbf{A}\,. \tag{3.10}$$

In the MSA mechanism, the process just explained is repeated in parallel h times. Results obtained for each head are concatenated and then projected once again to obtain the final results. The expression describing the final results for the MSA mechanism is introduced in Equation (3.11):

$$\text{MSA} = [\mathbf{O}_1\,, \mathbf{O}_2\,, \ldots, \mathbf{O}_h] \cdot \mathbf{W}_{\text{MSA}}\,, \tag{3.11}$$

where $\mathbf{O}_h$ is the output of the self attention head h, and $\mathbf{W}_{\text{MSA}}$ is the learnable weight matrix that projects the concatenation of all the heads outputs to obtain the final MSA representation. The intuition behind MSA is that multiple attention heads will allow the system to attend to parts of the sequence in different ways, e.g., long term dependencies versus short term dependencies.

3.2.2 Supervised learning techniques

The most commonly used learning strategy when using neural networks is the supervised learning paradigm. In this scenario, we assume a dataset consisting of a set of features $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$ and its associated labels $\mathbf{Y} = \{y_1, y_2, \dots, y_n\}$. As explained previously, all neural network architectures consist of a set of parameters that needs to be learned. This learning process can also be expressed as an optimisation problem according to Equation (3.12):

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmin}}\, J(\mathbf{W})\,. \tag{3.12}$$

As observed, the goal of this optimisation is to find the parameters $\hat{\mathbf{W}}$ that minimise the function $J(\mathbf{W})$. This function is usually known as loss function or cost function. It serves to measure the difference between the output of the network, $\hat{y}_i$, and the expected value described by the labels, y_i. In classification tasks such as audio segmentation, cross entropy (CE) loss is the most widely adopted cost function to optimise a neural network. A common practice is to feed the last layer of the neural networks to a softmax activation layer in order to obtain a class probability distribution. These probabilities are then compared to the actual class desired output, which is 1 for the labelled class and 0 for the rest. CE loss penalises the probability based on how far it is from the actual expected value.

There exist several algorithms that could solve this optimisation problem, however, the preferred choice to optimise the parameters of a neural network are the gradient descent solutions. These family of algorithms seeks to minimise in an iterative way an objective function by updating the parameters in the opposite direction of the gradient of that objective function with respect to the parameters. The general expression for parameters update through a gradient descent algorithm is described in Equation (3.13)

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \alpha\, \nabla_{\mathbf{W}} J(\mathbf{W})\,, \tag{3.13}$$

where $\mathbf{W}_n$ are the parameters in the current iteration, and $\nabla_{\mathbf{W}} J(\mathbf{W})$ is the gradient of the loss function with respect to the parameters $\mathbf{W}$. The value α that multiplies the gradient term is an hyperparameter known as learning rate used to control the convergence speed of the optimisation task. As explained, the descent direction is extracted from the gradient of the loss function, but the learning rate determines how large the step in that direction is taken. Gradient descent methods need to compute the gradient of the loss function with respect to all the parameters of the neural network. Current deep learning setups with neural networks featuring millions of parameters, reveal the necessity of efficient algorithms to compute those gradients. The backpropagation algorithm [104] is able to solve this issue, facilitating the learning process in deep neural architectures. This algorithm computes the gradient of the loss function with respect to each parameter of the network by using the chain rule. Gradients are computed layer by layer, performing an iteration in backward direction from the last layer. A usual training loop for a neural network involves then two main steps: the forward step, where the outputs of each layer are computed, and the backward step where the gradients for each layer are obtained applying the backpropagation algorithm.

A large number of optimisation methods can be found in the literature, each one proposing improvements over the basic gradient descent optimisation described. One of the most commonly extended nowadays is the adaptative moment estimation (Adam) [105], that uses per parameter learning rates adapted according to first and second order moment estimations of the gradients.

3.2.3 Self-supervised representation learning

In the last decade, deep learning solutions have allowed the advance of several signal processing applications. While supervised learning algorithms have been essential for the development of these deep learning applications, labelled data is not always easy to obtain. This problem is becoming more and more relevant as models grow faster in size and computational requirements. In this context, self-supervised representation learning solutions [106] emerged in order to alleviate the need for labels. Representation learning is defined as a set of algorithms that aim to extract features that describe the underlying explanatory factors for the observed input. The obtained features can then be used for the desired application using a small amount of labelled data. In the case of self-supervised representation leaning, current trends expose a model to huge amounts of unlabelled data, with this objective of understanding the data source by learning to make predictions related to it.

Self-supervised representation learning has been an active research topic since the 1990s. Early work relied on clustering algorithms such as k-means to extract patterns from data [107]. In recent years, self-supervised representation learning shifted towards neural networks models, showing increasing modelling capabilities. First approximations were presented for discrete sources such as text, forcing the network to predict the next items [96]. When dealing with real valued signals, the idea was initially approached by minimising the reconstruction error of the signal [108]. Some evolution on this idea were proposed, such as the reconstruction of missing or corrupted fragments [109]. However, greater gains have been obtained by constructing pretext tasks, where the objective of the system is to solve a prediction as a classification. In many works, this objective is to select an unseen fragment of the signal among other randomly selected distractor fragments [110].

This last idea is usually referred to as contrastive learning. In contrastive approaches, the pretext task is to maximise latent space similarity between an anchor sample and positive samples and, simultaneously, minimise the similarity between the anchor sample and negative samples. This concept was first introduced in a metric learning framework applied to a face verification task [111]. Since then, contrastive loss has been formulated in several ways in the literature, such as in simCLR [112] or in infoNCE [110]. In the following lines we describe the infoNCE formulation due to its relevance for posterior work. Supposing an observed input x_t, the proposed model in [110] maps this input into a sequential latent representation $z_t = g_{\text{enc}}(x_t)$. Then, an autoregressive model gathers the information from z_t into a contextual latent representation $c_t = g_{\text{ar}}(z_t)$. Seeking to predict a future fragment of the signal x_{t+k}, the infoNCE contrastive loss can be defined

according to Equation (3.14):

$$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right], \tag{3.14}$$

where $\mathbf{X} = [x_1, \ldots, x_N]$ is a set of N samples with 1 positive sample and $N - 1$ negative samples, and $f_k(x_{t+k}, c_t)$ represents a density ratio between x_{t+k} and c_t, as follows:

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})}, \tag{3.15}$$

where $\propto$ means "proportional to". Several options could be possible that yield a positive real score, but the implementation in [110] uses a log-bilinear model:

$$f_k(x_{t+k}, c_t) = e^{z_{t+k}^T W_k c_t}. \tag{3.16}$$

As it can be seen, prediction from c_t is performed using a linear transformation that is different for every step k. More complex methods such as recurrent neural networks could also be used for this prediction. Constrastive learning mechanisms have been successfully implemented for large scale tasks with good results for image [113] and speech recognition [114] [115].

With the increasing interest of contrastive self-supervised representation learning, new approaches have been observed recently. Novel solutions, such as the ones proposed in the HuBERT [116] or w2v-BERT [117] models, predict discrete targets of masked regions. Particularly, HuBERT uses an unsupervised clustering step to obtain a prediction loss over a masked region of the input signal. A step further is proposed in the wavLM approach [118], where some inputs are corrupted with noisy or overlapped speech and the target is to predict a pseudo-label on the masked region such as done in HuBERT. WavLM models have been tested in a variety of speech processing tasks with competitive results, achieving state-of-the art performance on the SUPERB benchmark [119].

3.3 Hidden Markov models

Hidden Markov models (HMM) are a powerful statistical method to describe discrete temporal information sequences. The basic theory detailed in this section was initially published by Baum et al. in a series of classic papers [120] [121].

HMMs became one of the most common methods for processing speech signals. Its principles have been applied in several speech processing tasks such as speech recognition, pitch tracking or speech synthesis. In this dissertation, HMMs have been used as a post processing method for smoothing the output of audio segmentation systems. Results obtained were specially relevant in the experiments that consider simultaneous speech, music and noise segmentation.

3.3.1 The Markov chain

Let $\mathbf{Z} = [Z_1, Z_2, \dots, Z_n]$ be a sequence of random variables. Based on the Bayes's rule, the joint probability of observing the sequence $\mathbf{Z}$ can be written according to Equation (3.17):

$$P(Z_1, Z_2, \dots, Z_n) = P(Z_1) \prod_{i=2}^{n} P(Z_i \,|\, Z_1^{i-1}) \,, \tag{3.17}$$

where $Z_1^{i-1} = [Z_1, Z_2, \dots, Z_{i-1}]$. Random variables Z are said to form a first-order Markov chain if they satisfy the following condition:

$$P(Z_i \,|\, Z_1^{i-1}) = P(Z \,|\, Z_{i-1}) \,. \tag{3.18}$$

Equation (3.18) is usually defined as the Markov assumption. According to it, the conditional probability of a random variable at a given time depends only on the value at the preceding time. This way, Equation (3.17) can be rewritten in the form of a first-order Markov chain.

$$P(Z_1, Z_2, \dots, Z_n) = P(Z_1) \prod_{i=2}^{n} P(Z_i \,|\, Z_{i-1}) \,. \tag{3.19}$$

By discarding the time index i, the Markov chain can be used to model stationary events.

$$P(Z_i = s \,|\, Z_{i-1} = s') = P(s \,|\, s') \,. \tag{3.20}$$

Therefore, by associating an state s to Z_i, the Markov chain can be represented through an stochastic finite state machine. Transitions between states are associated with the probability $P(s|s')$. By considering this description, from Equation (3.19) it can be inferred that the probability of the Markov chain being in an specific state at a given time only depends on the state in the preceding time.

Given a Markov chain with N states, let s_t be the state of the markov chain at time t. Parameters defining this chain are expressed in Equations (3.21) and (3.22):

$$a_{ij} = P(s_t = j \,|\, s_{t-1} = i) \quad 1 \leq i, j \leq N \,, \tag{3.21}$$

$$\pi_i = P(s_1 = i) \quad 1 \leq i \leq N \,, \tag{3.22}$$

where a_{ij} is the transition probability between state i and state j, and π_i is the probability of the chain initial state being the state i. Both probabilities verify the following restrictions:

$$\sum_{j=1}^{N} a_{ij} = 1 \quad 1 \leq i \leq N \,, \tag{3.23}$$

$$\sum_{j=1}^{N} \pi_j = 1 \,. \tag{3.24}$$

According to the provided description this Markov chain can be referred to as Markov observable model, because the outcome of the process is the set of states at each time t, with each state being associated with an observable event Z_i. In all cases there is a one-to-one correspondence between the observable events and the state sequence of the Markov chain.

3.3.2 Definition of the hidden Markov model

Let's suppose now a Markov chain where the observation is no longer a deterministic value but a random variable X, that is generated according to a probability distribution associated with each state. Under this condition there is no unique correspondence between the output sequence and the state sequence. In other words, the state sequence is hidden. That is the reason why these kind of models are usually called hidden Markov models. Formally, a hidden Markov model is defined by the following parameters:

- $\mathrm{S} = [1, 2, \dots, N]$: a set of states that characterise the model.
- $\mathbf{A} = \{a_{ij}\}$: a transition probability matrix, where a_{ij} is the probability of transitioning from state i to state j.
- $B = \{b_i\}$: a set of probability distributions, where b_i is the observation probability distribution at state i.
- $\Pi = \{\pi_i\}$: A initial state probability, where π_i is the probability of the state i being active at the initial time $t = 1$.

In the first-order hidden Markov model just discussed there are two assumptions:

1. The Markov assumption for the Markov chain:

$$P(s_t \mid s_1^{t-1}) = P(s_t \mid s_{t-1}), \tag{3.25}$$

   where s_1^{t-1} represents the state sequence $[s_1, s_2, \dots, s_{t-1}]$. This assumption implies that the probability of finding the system in a state s at a given time t only depends on the previous state at a time $t - 1$.

2. Output independence assumption:

$$P(X_t \mid X_1^{t-1}, s_1^t) = P(X_t \mid s_t), \tag{3.26}$$

   where X_1^{t-1} is the output sequence $[X_1, X_2, \dots, X_{t-1}]$. Under this assumption the emission probability of a determined symbol at a given time t depends only on the state s_t and is independent of past observations.

These two assumptions limit the memory of hidden Markov models. However, in practice, they make working with these kind of models feasible and efficient, without strongly affecting its modelling capabilities. Furthermore, due to the mentioned assumptions, the number of parameters that needs to be estimated is significantly reduced.

"In the spirit of science, there really is no such thing as a failed experiment. Any test that yields valid data is a valid test."

Adam Savage

4

Experimental framework

Once an audio segmentation system has been designed and its main components have been chosen, there are yet two key elements that need to be considered in order to properly describe a complete experiment: the data used in the training and test phases of the system, and the evaluation metrics that need to be analysed to describe the performance of the system. This chapter presents a description of the overall experimental framework of this thesis dissertation by providing first a review of the datasets used across the three parts of this work: speech activity detection, multiclass segmentation, and music detection. Then, this chapter introduces some key aspects of the evaluation protocol for audio segmentation and the most relevant evaluation metrics for audio segmentation systems that have been used in our experiments.

4.1 Databases description

4.1.1 Databases for speech activity detection

In the last few years, a number of international evaluation campaigns have been proposing the speech activity detection task as one of their challenges, seeking to advance this kind of systems in a variety of challenging domains. In this context, aiming to motivate the research effort on a demanding domain such as audio from Apollo space missions, a series of annual challenges is being held since 2019 [122] proposing the SAD task among other speech related tasks. This initiative has resulted in the digitisation of the original analog recordings from the space missions. Part of this data has been made available through the Fearless steps (FS) corpus, consisting of a cumulative 19,000 hours of conversational speech coming from the Apollo 11 mission [123]. Audio data belongs to 30 different communication channels, with multiple speakers in diverse locations. Most channels show a strong degradation with transmission noise or noise due to tape ageing.

Experiments performed in this work for the speech activity detection task rely on the data released for the different editions of the Fearless steps challenge. For each annual version of the challenge, the organisation released new manually labelled datasets. The work described in this thesis is developed mainly using the datasets released for the Fearless steps challenge phase II (2020), Fearless steps challenge phase III (2021), and for the Fearless steps challenge phase IV (2021). Fearless steps data was made available to the challenge participants. The three partitions used in this work are not publicly available. However, they may be requested to their respective authors through the challenge official contact email: `fearlessssteps@utdallas.com`.

Fearless steps phase II dataset

The inaugural version of the Fearless steps challenge, that was held in 2019, encouraged its participants to develop speech processing systems with limited resource availability. A total of 80 hours of audio were provided, from which only a subset of 20 hours were human verified ground truth labels. As a natural progression and following the successful first edition of the challenge, the Fearless steps challenge phase II [124] changed its focus to the development of supervised systems, releasing around 80 hours of human labelled data through the training and development datasets. The fact that a larger amount of in-domain annotated data is available in this version opens a new possibility for supervised approaches such as some of the ones proposed in this work.

Released data consist of 3 different partitions, all of them manually labelled: a train subset with 62.5 hours of audio, a development subset with 15 hours of audio and an evaluation subset with 20 hours of audio. Note that evaluation labels were not released and the scoring was performed by the organisers of the challenge.

Fearless steps phase III and phase IV datasets

As a continuation over the the previous versions of the challenge, the 3rd and 4th phases of the challenge (Fearless steps challenge phase III and phase IV), held in 2021 and 2022, focused again on the development of single channel supervised learning strategies. In these mentioned editions, the novelty introduced was in the new aim to test system generalisation to varying channel and mission data. While train and development subsets remained similar as the ones released in previous editions, new data was included in the evaluation subsets. Besides the 20 hours of Apollo-11 evaluation data already released, evaluation data for the phase III of the challenge was complemented with 5 hours of unseen channel evaluation data and 5 additional hours of blind-set Apollo-13 mission. In the case of phase IV, more data from Apollo-8 mission was also incorporated. The new evaluation susbsets for these editions featured, as described, more than 30 hours of audio in total from different channels and missions. This fact indicates that a larger mismatch between development and evaluation results could be observed. Again, evaluation labels were not released and the scoring was performed by the organisers of the challenge.

Additional datasets for unsupervised domain adaptation

In addition to the use of the Fearless steps data, experiments reported in Chapter 6 used different datasets coming from three specific domains: broadcast, telephone channel and meetings. For the broadcast domain data was combined from two different sources. First, the system included data coming from previous Albayzín evaluation campaigns, mainly from Albayzín 2010 dataset [11] (explained in more detail in subsection 4.1.2) and Albayzín 2018 dataset [125], with audio from the main Spanish public service broadcaster. Furthermore, audio from the Multi-Genre broadcast (MGB) challenge [126] was also used. The MGB challenge was organised by the British Broadcast Corporation (BBC) releasing data across four television channels for different speech processing tasks. For the meetings domain, the AMI [127] and ICSI meetings [128] corpora are used. Finally, in order to represent the telephonic domain, the summed partition of NIST 2018 speaker recognition evaluation (SRE) is used.

One more extra dataset for each domain is reserved for evaluation purposes. These dataset are the Albayzín 2020 dataset [129] for the broadcast domain, CALLHOME dataset [130] for telephonic domain, and the dataset originally released for the NIST rich transcription (RT) 2009 evaluation for the meetings domain. A more detailed explanation on the use of the different datasets introduced will be provided in Chapter 6.

Concerning the availability of these datasets, Albayzín 2018 and 2020 datasets are available upon request through the challenge organisation website[1]. AMI corpus is publicly available[2]. ICSI Meetings corpus is available at the Linguistic Data Consortium (LDC) under catalogue numbers LDC2004S02 and LDC2004T04 for audio and transcripts respectively. CALLHOME dataset can also be found on the LDC under the catalogue numbers LDC97S42 and LDC97T14 for audio and transcripts respectively.

[1] http://catedrartve.unizar.es/albayzindatabases.html
[2] https://groups.inf.ed.ac.uk/ami/download/

4.1.2 Databases for multiclass segmentation

The Albayzín evaluation campaigns have proposed several speech and audio processing related tasks in the last decade. Audio segmentation is one of the proposed tasks from 2010 to 2014, focusing mainly on separating speech, music and noise. For this purpose, two datasets were released: the 3/24 TV dataset [11], released for the 2010 evaluation and coming from broadcast television domain, and the CARTV dataset [131], used in the 2012 evaluation and obtained from radio recordings. They share a set of common characteristics and some minor differences that are explained in this subsection. Both datasets are available from their respective authors on reasonable request.

Albayzín 2010 database

The Albayzín 2010 database is part of a set of broadcast news recordings televised originally in 2009 by the Catalan TV channel 3/24 TV. Data was originally collected by the TALP Research Centre from the Universitat Politècnica de Catalunya. The full database includes around 87 hours of manually annotated audio sampled at 16 kHz and it is divided in 24 files of around 4 hour length each. Two thirds of the database are available for training, making a total of 58 hours in 16 different sessions, while the remaining third, 29 hours in 8 sessions, is used for test purposes. The evaluation plan defines five different acoustic classes distributed as follows: 37% for clean speech (sp), 5% for music (mu), 15% for speech over music (sm), 40% for speech over noise (sn) and 3% for others (ot). The class "others" is not evaluated in the final scoring. In the following lines a description is provided for each of the acoustic classes defined: the class "speech" contains speech under studio recording conditions using a close microphone. The "music" class contains music understood in a general sense. The "speech over music" class is defined as the overlap of classes "speech" and "music". The "speech over noise" class contains all the speech that is not recorded under studio conditions or overlapped with any kind of noise. Two voices overlapping are also defined as "speech over noise". Finally, "others" contains any audio that does not match the four previously defined classes.

As it can be appreciated, there is a clear unbalance in the class distribution because most of the data contains speech (92% combining speech, speech over noise and speech over music). However, classes that contain music are underrepresented (only 20% combining music and speech over music). The main language of the 3/24 TV channel is Catalan, with the 87% of the speech segments coming from Catalan speakers and the remaining 13% coming from Spanish speakers. Concerning the gender distribution, 63% of speech fragments are from male speakers and 37% are from female speakers.

Albayzín 2012 database

Additionally, a new dataset was released for the Albayzín 2012 audio segmentation evaluation. The new audio introduced in this version is taken from Aragón radio archive, separated in 3 different subsets: two developments sets of 2 hours each (dev1 and dev2), and a test set consisting of 18 hours. All the audio is sampled at 16 kHz. The use of

previous data released in the 2010 version was allowed in order to train segmentation systems for the 2012 Albayzín evaluation. As in the 2010 Albayzín evaluation, the main goal is segmenting an audio document indicating where speech, music and/or noise are present. However, in the 2012 version no prior classes are defined and a multiple labelling format is proposed, allowing 3 possible overlapped classes, speech, music and noise, to be present at any time in the audio document. This format slightly differs with the experimental setup presented for the Albayzín 2010 evaluation, but it is equivalent to a multiclass segmentation task if a set of non-overlapped labels are generated considering the different combinations of speech, music and noise. This would results in 8 different classes: "speech", "music", "noise", "speech and music", "speech and noise", "speech and music and noise", "music and noise", and "silence". Due to the similarity with the class defined as "others" in the Albayzín 2010 evaluation, we decided to combine "music and noise", "noise" and "silence" (they represent only the 3% of the total time in the database) in a single class that we also name as "others". Therefore, we can see this problem in a similar way to the task in the Albayzín 2010 dataset, but including a new class "speech and music and noise" that was not present in the 2010 version of the evaluation.

4.1.3 Databases for music detection

When compared to SAD datasets, the amount of publicly available data labelled for music detection is relatively low. Most music datasets come from the music information retrieval community and aim to extract relevant information from music excerpts such as genre, chords, or tempo. In these cases, music is not usually mixed with any other acoustic classes, limiting its use in the music detection task. Some datasets suitable for music detection such as the one proposed by Scheirer et al. [132] are too small to be used for training audio segmentation systems in the context of deep learning applications. Furthermore, other released datasets such as MUSAN [133] do not provide strong labels, being annotated with a single label for the whole audio file. In the following lines a description is provided for the two datasets used in this dissertation. both of them belong to the broadcast domain, where it is common to find music in combination with speech or other acoustic classes.

OpenBMAT

The OpenBMAT dataset [134] contains 27 hours of television broadcast audio from different countries labelled for the music detection task under two different scenarios: a binary condition where audio is annotated as containing music or not containing music, and a multiclass setup where audio is separated into foreground, background and no music fragments. All audios were cross-annotated by 3 different annotators providing high inter-annotator agreements ratios. In average, labels contain a 51% of music and a 49% of non-music in the binary annotation. For the 3 class annotation schema, 16.60% of the audio belongs to foreground music, 34.45% belongs to background music and 48.94%

belongs to no music. The OpenBMAT dataset can be downloaded for non-profit purposes upon request to their respective authors [3].

DAFX'07 music detection dataset

This dataset was originally presented in the paper "Automatic Music Detection in Television Productions" [135]. It consists of around 9 hours of television recordings from the Austrian national broadcasting corporation that were manually labelled as music or non-music. There is a great variety of genres, ranging from soap operas to documentaries or talk shows. The data distribution is close to be equally distributed, with 42% of the time being music and 58% of the time being non-music. This dataset can be downloaded for research purposes upon request to their respective authors. [4]

4.2 Evaluation metrics

A really important part of the working pipeline for any machine learning model is the evaluation of its results. In the context of the experimental framework, the use of the appropriate evaluation metrics allows to correctly asses the performance of the evaluated systems. A number of evaluation metrics can be found in the literature aiming for specific purposes and showing benefits and drawbacks. In this subsection, we first describe the collar-aware evaluation protocol, a commonly used technique for the evaluation of audio segmentation tasks that considers time margins around changes in the reference for evaluation. Then, a description is provided for those metrics that are more relevant in the different audio segmentation tasks considered in this work.

4.2.1 Collar-aware evaluation

In the evaluation of audio segmentation systems, the use of time margins around changes in the reference segmentation has become a common practice among different tasks. This idea is usually known as collar-aware evaluation, with the word collar referring to that small amount of time that is not evaluated. Collar-aware evaluation is done in order to consider possible inconsistencies in the human annotation of databases and in order to consider the uncertainty about when an acoustic class begins or ends.

Figure 4.1 presents a graphical representation of the evaluation protocol performed when using collars in an audio segmentation task. As it can be observed, both the segmentation system hypothesis and the reference labels are presented. The reference shows a change between class A and class B. For that time stamp T_{change}, defined as the change between class A and class B, a region of the hypothesis defined by the interval $[T_{\text{change}} - \text{collar}, T_{\text{change}} + \text{collar}]$ will not be considered in the evaluation of the audio segmentation system. The size of the collar may vary among tasks and evaluation protocols.

[3] https://zenodo.org/record/3381249
[4] http://www.seyerlehner.info/joomla/index.php/datasets

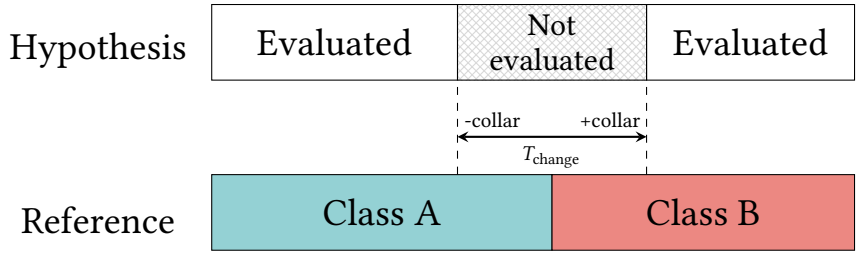


Figure 4.1: Schematic representation of collar-aware evaluation and regions excluded for evaluation.

For example, the collar used in the multiclass audio segmentation task from the Albayzín 2010 evaluation was of 1 second, while the collar used in the Fearless steps challenge for speech activity detection was of 0.5 seconds. The collar-aware evaluation protocol described in the previous lines can be applied to all the evaluation metrics described in this document.

4.2.2 Evaluation metrics for audio segmentation tasks

Some of the audio segmentation tasks described in this document deal only with two different acoustic classes so they can be considered binary problems. Under these circumstances, the goal of any segmentation system is to separate two different sets: targets, also named as positive examples, and non-targets, also named as negative examples. Table 4.1 shows the four possible outcomes of a binary segmentation system considering the predicted values by the system and the real values according to the labels.

		Predicted value	
		Positive	**Negative**
Real value	**Positive**	True positive (TP)	False negative (FN)
	Negative	False positive (FP)	True negative (TN)

Table 4.1: Possible system outcomes in a binary problem considering predicted and real values.

In the mentioned table, the green positive diagonal represents correct predictions, while the red negative diagonal shows incorrect predictions. Focusing first on the correct cases, a true positive (TP) is a positive example that is correctly classified as a positive example, also referred to as a hit. Alternatively, a true negative (TN) is a negative example that is correctly classified as a negative example, making it a correct rejection. Concerning the two possible errors, a false positive (FP) happens when a real negative value is wrongly labelled as a positive value. This scenario is also defined as a type I error or a false alarm. If a real positive value is incorrectly classified as a negative value, then that is considered a false negative (FN). This condition is usually described as a type II error or a miss.

Precision, recall and F_1-measure

Considering the possible outcomes described in Table 4.1 a number of metrics can be defined. Some of the most widespread metrics to evaluate segmentation systems is the set formed by precision, recall and F_1-measure. On the one hand, precision aims to provide an intuition on the proportion of positive classifications that was actually correct. Precision is defined as the number of true positives over the total number of predicted positive examples (true positives plus false positives), as expressed in Equation (4.1):

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \,. \tag{4.1}$$

On the other hand, recall tries to describe the proportion of actual positive examples that was identified correctly. It is formally defined as the number of true positives over the total number of real positives examples (true positives plus false negatives), as shown in Equation (4.2):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \,. \tag{4.2}$$

The F_1-measure combines both metrics by computing the harmonic mean of precision and recall, as shown in Equation (4.3):

$$\text{F}_1 = 2\,\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \,. \tag{4.3}$$

Furthermore, the F_1-measure can also be seen as an specific case of a more general evaluation metric, the F_β-measure, described according to Equation (4.4):

$$\text{F}_\beta = (1 + \beta^2)\,\frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \,, \tag{4.4}$$

where a positive real factor β is used so that recall is considered β times as important as precision. The four metrics described can take values in the interval $[0, 1]$, with a higher value being associated with a better performance.

False positive rate, false negative rate and detection cost function

Another common set of metrics in binary segmentation tasks that can be inferred directly from Table 4.1 is the pair formed by false negative and false positive rates. These two metrics have been commonly used in binary tasks such as speech activity detection in order to measure the two possible type of errors, misses and false alarms.

The false positive rate (FPR) is calculated according to Equation (4.5):

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \,. \tag{4.5}$$

As it can be seen, it is defined as the ratio between the number of negative examples wrongly classified (false positives) and the total number of real negative events (false positives plus true negatives). This metric can be interpreted as the probability of wrongly

classifying a negative example as positive. That is the reason why false positive rate is also usually referred to simply as false alarm rate.

Similarly, the false negative rate (FNR) is calculated according to Equation (4.6):

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}}\,. \tag{4.6}$$

As in the false positive rate case, this metric is calculated as the ratio of two elements: false negative examples and the sum of true positive and false negatives, namely, the total number of real positive examples. Therefore, it can be interpreted as the probability of incorrectly assigning a positive label to a negative example. Usually, false negative rate is also mentioned as miss rate. FPR and FNR can take values in the interval $[0, 1]$. As these metric are measuring error rates, a lower value is usually associated with a better performance.

In addition, the detection cost function (DCF) is a metric that combines both false positive and false negative rates according to Equation (4.7):

$$\mathrm{DCF} = C_{\mathrm{FA}}\,\mathrm{FPR} + C_{\mathrm{MISS}}\,\mathrm{FNR}\,, \tag{4.7}$$

where C_{FA} and C_{MISS} are the associated costs to the false alarm errors and miss errors respectively. These two values can be used in order to give more importance to each one of the two possible errors. As an example, the DCF metric is used in the Fearless steps challenge, where the organisation decided that miss errors are more important than false alarms in the speech activity detection task. That is why in this evaluation C_{FA} takes a value of 0.25 and C_{MISS} takes a value of 0.75.

Confusion matrix

So far all the metrics presented have been derived from Table 4.1, that describes the possible outcome of a binary system. If the system being evaluated considers more than two classes for segmentation, a similar analysis of the different outputs of the system can be performed for the classes present in the task. This analysis is usually summarised using a confusion matrix. Figure 4.2 shows an example of a confusion matrix for a problem with 4 classes. As it can be seen, each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class. This representation simplifies the detection of commonly mistaken classes in a segmentation system, making explicit the amount of instances from one class that are being incorrectly labelled as a different one.

Area under the ROC curve and DET curve

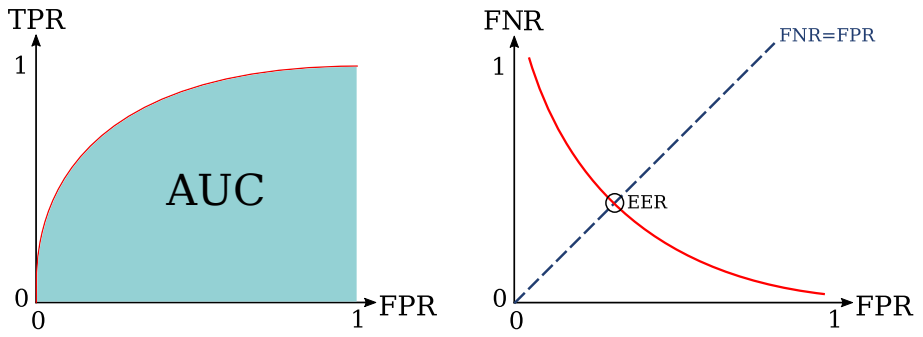
Metrics described up to this point operate under the assumption that an specific threshold has been chosen in a previous stage to the evaluation, fixing the operating point of the system. In some scenarios, it may be interesting to evaluate an audio segmentation system for different thresholds, and even to summarise the performance of the system for all

Real value \ Predicted value	A	B	C	D
A	1620 94%	30 2%	60 5%	0 0%
B	0 0%	1390 77%	110 9%	80 7%
C	100 6%	40 2%	1090 84%	40 4%
D	0 0%	350 19%	30 2%	1020 89%

Figure 4.2: Example of a confusion matrix for a multiclass problem with four classes.

possible thresholds with a single measure. For that purposes, in this section we introduce two metrics widely adopted in the literature: the receiver operating characteristic (ROC) curve and the detection error trade-off (DET) curve. In order to illustrate this description an example of both of them is presented in Figure 4.3.

First, concerning the ROC curve, this graphical representation describes the performance of a system for different thresholds by comparing false positive rates on the x axis versus true positive rates on the y axis. While this curve allows to show the general capabilities for various operating points, in some conditions it is needed to summarise the performance in a single number. A widely used metric is the area under the ROC curve (AUC) [136]. Since the AUC is obtained as a part of the unit square, its value is always between 0 and 1. Furthermore, no realistic classifier should report an AUC below 0.5, which is the case associated with random guessing. One of the most interesting properties of AUC is that the AUC value of any classifier is equivalent to the probability that



(a) ROC curve and area under the ROC curve.

(b) DET curve and EER point.

Figure 4.3: Example illustration of threshold dependent curves: from left to right: (a) ROC curve and area under the ROC curve. (b) DET curve and EER point.

this classifier ranks a randomly chosen positive example higher than a randomly chosen negative example. According to this the AUC metric can be formulated as:

$$\text{AUC} = \frac{1}{N^+N^-}\sum_{i=1}^{N^+}\sum_{j=1}^{N^-}\mathbb{1}\left(s_i^+ > s_j^-\right), \tag{4.8}$$

where s_i^+ represents the classifier scores for positive examples, s_j^- represents the classifier scores for negatives examples, N^+ and N^- represent the number of positive samples and the number of negative samples respectively, and $\mathbb{1}(\cdot)$ is equal to '1' whenever $s_i^+ > s_j^-$ and '0' otherwise. Despite AUC metric is designed for binary classification tasks, a number of extensions of this metric have been proposed in the literature in order to deal with problems with more than two classes. Two of the most common approaches are the one-versus-one, where all possible pairs of classes are considered to compute an average AUC metric, and the one-versus-rest, where a binarisation solution is applied to compute an average AUC metric. A more detailed explanation on these multiclass extensions of AUC will be provided in Chapter 11.

Alternatively to the ROC curve, the DET curve shows the two possible error rates, FNR associated with misses and FPR associated with false alarms, for different thresholds, describing the variation observed for different operating points. In the case of the DET curve a normal deviate scale is applied usually. By doing this, if both targets and non-targets were sampled from a Gaussian distribution, DET obtained would be a line perpendicular to the bisector of the represented quadrant. In this case, a generally used metric to summarise the findings of the DET curve is the equal error rate (EER) point [137]. Mathematically, this point is defined as the intersection of the DET curve with the FNR = FNR line and it represents the point where the false alarm rate is equal to the miss rate.

Segmentation error rate

The segmentation error rate (SER) is a metric strongly inspired by the diarisation error rate (DER), a metric used in NIST speaker diarisation evaluations [138]. It can be seen as the ratio of the total length of the incorrectly classified audio to the total length of the audio in the reference. Given a dataset to evaluate Ω, each document is divided into continuous segments and the segmentation error time for each segment n, $\Xi(n)$, is defined as:

$$\Xi(n) = T(n)\left[\max(N_{\text{ref}}(n), N_{\text{sys}}(n)) - N_{\text{correct}}(n)\right], \tag{4.9}$$

where $T(n)$ is the duration of the segment n, $N_{\text{ref}}(n)$ is the number of reference classes that are present in segment n, $N_{\text{sys}}(n)$ is the number of classes predicted by the system that are present in segment n and $N_{\text{correct}}(n)$ is the number of reference classes that are present in segment n and were correctly assigned by the segmentation system. This way, the SER is computed as follows:

$$\text{SER} = \frac{\sum_{n\in\Omega}\Xi(n)}{\sum_{n\in\Omega}(T(n)N_{\text{ref}}(n))}. \tag{4.10}$$

This metric could be applied to both binary and multiclass setups, but in this work it has been used mainly to evaluate the output of multiclass segmentation systems.

Part II

Speech activity detection in challenging environments

"Any sufficiently advanced technology is indistinguishable from magic."

Arthur C. Clarke

5

Convolutional recurrent neural networks for speech activity detection

5.1 Introduction and motivation

Speech activity detection (SAD) aims to determine the exact location of speech samples in an audio signal. This constitutes an essential preprocessing step in several speech related applications such as speech or speaker recognition, as well as speech enhancement. In many cases, SAD is used as a preliminary block to separate the segments of the signal that contain speech from those that do not. This way, enabling the overall system to process only the speech segments. A large number of approaches have been

proposed for the SAD task. Traditionally, statistical approaches have been used with relevant results under the assumption of quasi-stationary noise. Several works rely on the extraction of specific acoustic features [139] [140]. Conversely, other methods are model-based [141] [142], aiming to estimate a statistical model for the noisy signal. Additionally, some unsupervised approaches can also be cited: based on energy [143], or based on the estimation of the signal spectral characteristics such as long-term spectral divergence [144]. Recently, deep learning approaches are becoming more and more relevant in the SAD task. The research presented in [145] implements a SAD system based on a multilayer perceptron with energy efficiency as the main concern. A deep neural network (DNN) approach is used in [146] to perform SAD in a multi-room environment. In [147], new optimisation techniques based on the area under the ROC curve are explored in the framework of a deep learning SAD system. Recently, convolutional recurrent neural networks have been proposed in the SAD task providing relevant results, with approaches such as the one presented in [57], based on the use of 2D convolutional layers.

In the last few years, a number of international evaluation campaigns have been proposing the SAD task as one of their challenges, seeking to advance this kind of systems in a variety of challenging domains. Some examples can be cited such as the OpenSAT challenge [148] organised by the NIST, or the DARPA RATS evaluation [149]. In this context, the Fearless steps annual series of challenges has proposed the SAD task among other speech related tasks since its first edition in 2019. This initiative deals with digitised audio coming from the original recording of the NASA's Apollo space missions. Such a challenging domain features strongly degraded channels, with a number of transmission noises or noise due to tape ageing. Most recordings were made with head-mounted microphones, but some in-spacecraft recordings were made using fixed far-field microphones which also picked up the presence of environmental noise. All those characteristics are likely to degrade the performance of speech technology applications.

The work described in this chapter is part of our submission to the SAD task proposed for the Fearless steps challenge 2020. In the context of this challenge, we introduce a supervised deep learning solution based on a CRNN that is fed with Mel filter-bank energies as input. Several alternatives for the convolutional layers are explored, namely 1D and 2D filters. Furthermore, a novel approach based on the fusion of two convolutional layers is presented. Under this configuration, information obtained from 1D and 2D filters is combined to be processed by the RNN. The version of the challenge released in 2020 [124] changed its focus to the development of supervised systems, releasing around 80 hours of human labelled data through the training and development datasets. The fact that a larger amount of in-domain annotated data is available in this version opens new possibilities for supervised approaches such as the one described in this chapter. The use of out-of-domain data in these specific conditions, namely naturalistic audio and strongly degraded channels, could lead to poor results.

5.2 Convolutional recurrent neural networks

5.2.1 Neural architectures description

In our submission to 2020 Fearless steps challenge for the SAD task we experimented with different neural architectures. In the following lines we briefly describe the different proposals evaluated. As our baseline model, we choose a solution that is inspired by the SAD system proposed in our previous work in the diarisation framework [150]. It consists of an RNN block generated by stacking three BLSTM layers with 128 neurons each. This block is then followed by a linear layer that generates the final representation as a two neurons output. The following architectures proposed are built on top of the RNN block from the baseline system, incorporating a set of convolutional layers working as a processing stage previous to the RNN block. The schematic representation of the proposed alternatives for the CRNN model is described in Figure 5.1. The RNN block

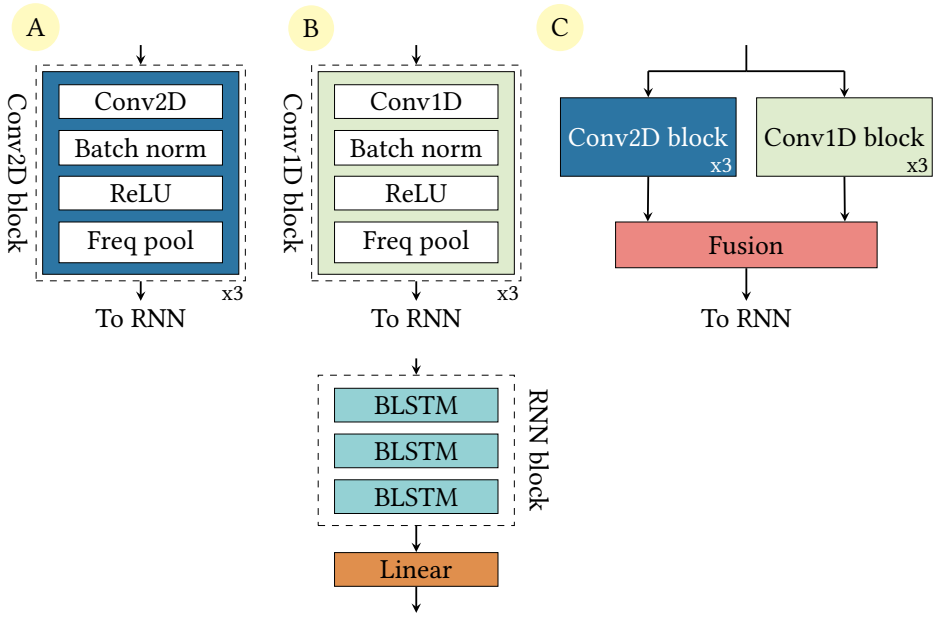


Figure 5.1: Schematic representation of the different variations on the proposed convolutional recurrent neural network used for the SAD task.

followed by a linear layer is shared by the three architectures. The difference comes then from the convolutional stage, that is implemented in three different ways:

- **Architecture A**: This model uses three 2D convolutional blocks processing the input features. Each of these blocks is integrated by a 2D convolutional layer with 3x3 or 5x5 kernel size and 64 filters. Then it is followed by a batch normalisation [151] and the application of a rectified linear unit (ReLU) [100] activation function. Finally, a max-pooling mechanism is applied considering a 4x1 stride, so that only the frequency axis is downsampled.

- **Architecture B**: This model similarly uses three 1D convolutional blocks. Even though, in this case, we experiment with different variations for the 1D convolutional layer. The first approach uses a kernel size of 3 in all the convolutions with no dilation. In the second implementation, each of the three layers uses kernel sizes of 5, 3 and 3 with dilations 1, 2 and 4 respectively. For the third approach, we experiment with the concept of group convolution, which has recently demonstrated its effectiveness in models such as ResNeXt [152]. This alternative employs a kernel size of 3 in all the convolutions but, in this case, these are implemented as 5 independent groups. This change does not affect the dimension of the input and the output feature maps but it reduces computational complexity and the number of parameters in the model. Finally, to obtain a comparable representation to the 2D setup, the convolutional layers have 256 output filters and a max-pooling mechanism is applied on the frequency axis using a 4x1 stride after the batch normalisation layer and a ReLU activation function.
- **Architecture C**: Some previous work has already shown the combined capabilities of 1D and 2D convolutions applying system level fusion techniques [153]. This alternative proposes a novel approach to the CRNN model in the SAD task where we aim to combine the information extracted by two different convolutional branches. One consisting of three Conv2D blocks and other consisting of three Conv1D blocks, both implemented as described in the previous architectures. This fusion is done in an intermediate feature space, where both branches are then combined to be processed by the RNN block. The fusion block (depicted in red) could be implemented in many different ways. In our experiments we test three different options: first, a bilinear layer is used to combine both convolutional branches. As second option, the fusion performs the sum of the output of both branches, and as a final option, the fusion is considered as the concatenation of the output of both branches.

A common characteristic among all the models evaluated in this set of experiments is that training and evaluation are performed using finite-length sequences. The input audio is separated in overlapping fragments of 3 second length and 2.5 second advance in order to limit the delay of the dependencies that the network may take into account. In all cases, the neural networks emit a SAD label for each frame processed at the input, one each 10ms in this case.

5.2.2 Training strategies

Described models are trained using Adam optimiser [105] with a learning rate that decays exponentially from 10^{-3} to 10^{-4} during the 20 epochs that data is presented to the neural network, with a minibatch size of 64. Cross entropy criterion is chosen as loss function, as usually done in classification tasks. Model selection is done choosing the best performing model in terms of frame classification accuracy using the validation subset. Models have been developed using the PyTorch toolkit [154].

5.3 Experimental setup

5.3.1 Data description

The Fearless steps challenge follows open training conditions. Participants can use any available data in addition to the provided challenge data to train and tune their systems. However, in this work we have not used any additional datasets. Considering the specific domain of the audio, namely quite degraded channels and several kinds of transmission noises, we opted to use for training and development only the labelled data provided by the organisation. These data consist of 3 different partitions. In the following lines, we describe them and explain how they have been used in our submission:

- **Train**: A total of 62.5 hours of audio were provided for training. In our experiments we used 10% of these data for training validation. This way, all the proposed systems were trained with around 56 hours of audio from the train partition.
- **Development**: The 15 hours of audio from the development subset were used to obtain an empirical threshold, in order to minimise the detection cost function (DCF) metric. We also report our results on this subset.
- **Evaluation**: Evaluation results are reported as provided by the challenge organisation. The DCF metric obtained in the evaluation subset is the one used to rank the participants.

5.3.2 Feature extraction

As input features for our proposed SAD system, we consider log Mel filter-bank energies. Namely, we use 64 log Mel coefficients concatenated with the log energy of the frame. With an input audio that is sampled at f_s = 8 kHz, Mel filters span across the frequency range between 64 Hz and 4kHz. Features are computed every 10ms using a 25 ms Hamming window. As a final step, mean and variance normalisation is applied over each file in the database. All the neural architectures described for the SAD system proposal share the same set of features.

5.3.3 Evaluation metric

In the SAD task of the Fearless steps challenge false negative errors are considered more important than false positive errors. This is shown in the primary evaluation metric for the challenge, the DCF, which is calculated as follows:

$$\mathrm{DCF}(\theta) = 0.75 \cdot \mathrm{FNR}(\theta) + 0.25 \cdot \mathrm{FPR}(\theta)\,, \tag{5.1}$$

where FNR is the false negative rate and FPR is the false positive rate. Participants are responsible to choose a threshold (θ) that minimises the DCF.

Table 5.1: SAD results in terms of DCF metric on the development and evaluation partition, and number of trainable parameters for different systems considered for submission.

System	# Param	DCF(%)	
		Dev	Eval
Organisation baseline [155]	-	12.50	13.60
RNN baseline	266K	2.02	2.54
A1 - CRNN 2D (3x3)	340K	1.65	2.07
A2 - CRNN 2D (5x5)	473K	1.67	2.28
B1 - CRNN 1D	421K	1.76	2.33
B2 - CRNN 1D dilation	455K	1.86	2.30
B3 - CRNN 1D groups	300K	1.76	2.46
C1 - CRNN fusion bilinear	641K	1.46	**1.78**
C2 - CRNN fusion sum	377K	1.60	1.89
C3 - CRNN fusion concat	411K	**1.43**	1.82

5.4 Results

Table 5.1 presents the obtained results for the different systems submitted. We compare our RNN baseline system and the three proposed architectures to the baseline provided by the organisation [155]. Note that the organisation's baseline is based on a statistical approach. Concerning the fusion architectures, they use the best configurations achieved with the development set: A1 for the 2D setup, and B3 for the 1D setup, as it obtains similar results to B1 with a significantly smaller number of parameters. Results are reported in terms of DCF metric for both, development and evaluation partitions. To measure the level of complexity of the models, we present the number of trainable parameters for all submissions.

Regarding the results reported on the development partition, all our presented systems significantly outperform the baseline algorithm proposed by the organisation. Furthermore, all the systems that include a convolutional processing stage improve the performance compared to the RNN baseline. Our experimental findings are in line with the ones presented in [57], where 2D CRNN models provided better performance than 1D CRNN based SAD systems. In our case, using the 3x3 filter configuration we were able to obtain a DCF of 1.65%, which is better than all the 1D based systems evaluated. For the 1D convolution setup, it is interesting to mention the configuration using groups. A significant relative improvement of 12.77% compared to the RNN baseline is obtained being the CRNN model with the lowest number of parameters. These experimental results indicate that the combination of 1D and 2D feature maps is beneficial for our SAD system. Best overall results are obtained using the proposed fusion architecture. The most relevant result is the one that concatenates both convolutional outputs, obtaining the best result in the development set while keeping a lower number of parameters when compared to

the fusion using a bilinear layer. With this setup we were able to achieve a competitive result for the development set with a 1.43% DCF metric.

Concerning the evaluation partition results, similar trends to the ones described in the development partition can be observed. In general terms, the behaviour for the three different kinds of architectures is consistent. Again, the 2D convolution setup outperforms its 1D equivalent. However, a difference can be observed in the 1D setup between development and evaluation results. While the groups configuration is the best performing in the development set, in the evaluation set this is done by the dilation setup. The fusion architectures show the best overall results, with a DCF metric below 2% for all the variations proposed. This solution allows our best submission to achieve a DCF metric of 1.78% using a bilinear layer as fusion method. Unlike it was observed in the development set, the fusion based on concatenation offers a slight degradation in performance compared to the bilinear fusion, while keeping the number of parameters significantly smaller.

Additionally, Figure 5.2 presents a qualitative visualisation of the SAD performance for the best performing architectures in the development partition. It can be observed

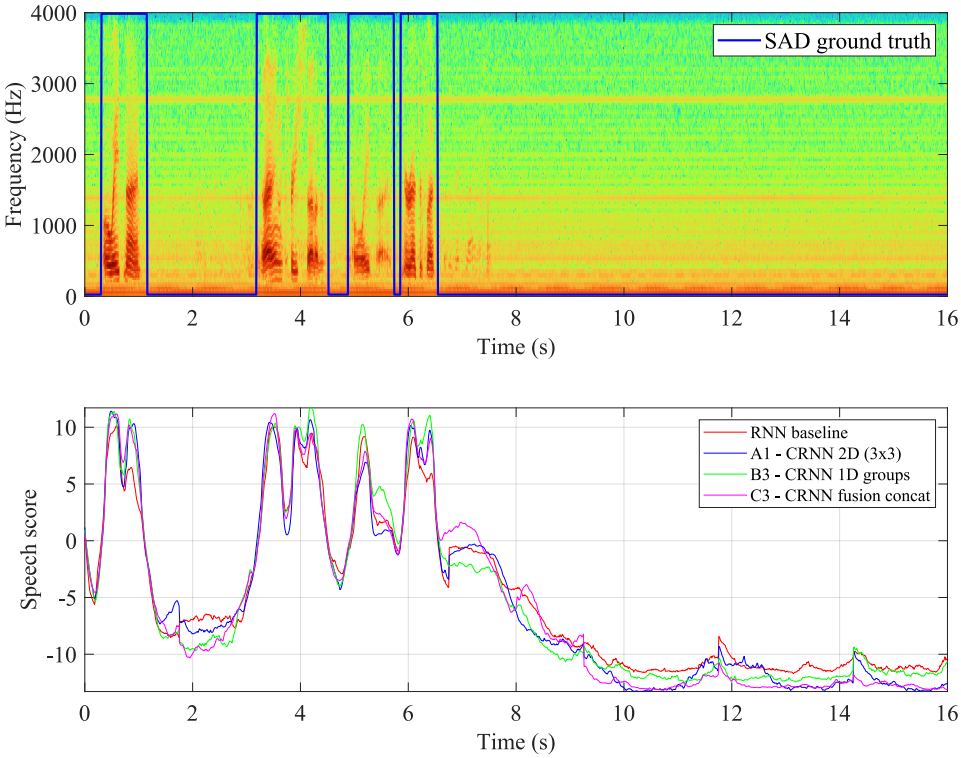


Figure 5.2: Qualitative visualisation of SAD scores for the model alternatives described in a 16 seconds audio fragment extracted from file "FS02_dev_001". From top to bottom: audio spectrogram with the SAD ground truth overlapped and speech score for different SAD systems proposed.

that, as it was expected, a high positive value in the neural network speech score is associated with a strong evidence of speech in the audio signal. In general terms, we can

see that all the systems shown in Figure 5.2 can accurately capture the speech and non-speech segments in the audio fragment with the empirical threshold minimising the DCF being $\theta = -2$. Anyway, some inconsistencies can be observed between the ground truth and the speech scores on some points. This is probably due to labelling conditions, where a few non-speech segments in between two speech segments are labelled as speech. Proposed systems are able to capture this effect by showing a local minimum in the speech score for the mentioned fragments (see between seconds three to five). It can also be observed that around second seven, the score provided by the different systems would result in a positive decision in an area with no actual speech, marking a false alarm error fragment.

Focusing on the individual performance of the proposed systems, we can observe that the 2D and fusion systems show a lower score when a long fragment of non-speech is processed. On the other hand, the system based on 1D tends to output a higher score for speech fragments. In the case of transitions, no significant difference is observed among the systems presented, indicating a similar response between speech and non-speech fragments and vice-versa. All systems presented in this set of experiments achieve competitive results without introducing post-processing or smoothing techniques on the neural network output. As it can be observed from the depicted scores of Figure 5.2, the BLSTM layers are able to impose a certain amount of inertia on the output so that the speech class score is smooth enough to be used by itself.

To summarise, Table 5.2 compares the best results obtained in the challenge by other participant teams with our best submissions and the baseline provided by the challenge organisation. As it can be seen, the first and second best performing teams achieved a

Table 5.2: SAD results in terms of DCF metric on the evaluation partition for the best performing teams in the Fearless steps challenge 2020 compared to our best submission and the baseline provided by the organisation.

System	Eval DCF(%)
Challenge winner team [156]	1.07
Challenge 2nd place team [157]	1.41
CRNN fusion bilinear (ours)	1.78
RNN baseline (ours)	2.54
Organisation baseline [155]	13.60

DCF metric of 1.07% and 1.41% respectively. Both systems are based on deep learning solutions. The proposal of the winner team was based on a CRNN architecture combined with an HMM based smoothing [156]. The team in second place also participated with a CRNN model featuring a ResNet frontend combined with an LSTM layer [157]. This team included additional data from meetings in the training process. Our best results using a CRNN model with bilinear fusion of 1D and 2D branches were ranked in seventh position among the 28 challenge submissions to the SAD task, and fourth among the 7 participant teams[1].

[1] https://fearless-steps.github.io/ChallengePhase2/Final.html

5.5 Conclusions

This chapter presented the work developed in order to participate in the SAD task of the Fearless steps challenge 2020. The mentioned challenge introduced a new demanding audio domain, featuring audio streams from communication channels with different speakers in different locations working for NASA's Apollo space missions. This data is characterised by multiple classes of noise and degradation, posing a significant test for audio and speech processing systems. New data released in this second phase of the challenge facilitated the development of supervised deep learning applications. Considering this, our participation in the challenge is built around the use of CRNN models, seeking to merge the capability of CNNs to capture frequency and time dependencies simultaneously, and the capability of RNNs to deal with temporal series.

For the different submissions described, several CRNN models were explored using 1D and 2D filters in the convolutional layers. These models were also compared with a baseline model using an RNN block only. We proposed a novel architecture that combines information coming from 1D and 2D filters in an intermediate feature space, which then is processed by the recurrent neural network. Obtained results largely outperform the baseline provided by the challenge organisation. Our experimental achievements are in line with previous publications where 2D convolutions obtained better performance than equivalent 1D convolutions. Additionally, experimental results showed that the combination of the information provided by 1D and 2D filters is beneficial for the SAD system, performing with the best results in the development and evaluation sets. Our best submission achieved a DCF metric of 1.46% and 1.78% respectively in the development and evaluation sets, ranking seventh among the 28 submissions to the challenge SAD task, and fourth among the 7 participant teams.

"The measure of intelligence is the ability to change."

Albert Einstein

6

Unsupervised domain adaptation of speech activity detection models

6.1 Introduction and motivation

Whereas current SAD state-of-the-art solutions rely on the use of deep learning techniques, these applications depend strongly on the amount of labelled data available. In some specific scenarios, obtaining labelled data can be significantly expens-

ive or even impossible, that is the reason why unsupervised domain adaptation techniques are an active research topic [158] [159]. Domain adaptation techniques aim to transfer the knowledge obtained from a source domain to a different target domain. In addition, unsupervised domain adaptation techniques work under the constraint that no labels are available in this new target domain.

Motivated by the experiences participating in the Fearless steps challenge described in the previous chapter, that introduced a new audio domain in the research community, in this chapter we aim to explore unsupervised domain adaptation techniques in the context of the SAD task. Data from Apollo space missions introduced in the mentioned challenge is a completely different domain from some of the most commons scenarios where SAD is applied. The research described in this chapter aims to evaluate how well a SAD system trained on common datasets can perform on this new domain, while at the same time it seeks to exploit data from these common domains, by adapting a baseline model to the new domain in a fully unsupervised way. Considering a SAD model trained on different well-known domains in the SAD task, such as broadcast or meetings, with huge amounts of labelled data available, we evaluate three possible ways to adapt the SAD model to a new unseen domain in an unsupervised way. Results are presented using the data provided in the Fearless steps challenge, however the techniques and methods introduced are described in a general way so that domain adaptation could be done on any possible scenario. Unlike the work presented in [160], where the key idea is to perform a pretraining process on DNN models using unlabelled data, seeking to obtain a shared representation, this work aims to perform unsupervised domain adaptation directly on the model space, with the objective of fine tuning a given model trained previously with labelled out of domain data.

6.2 Domain adaptation

Usually, machine learning algorithms require large amounts of manually labelled training examples in order to train a reliable model. In real applications, however, obtaining labelled data requires huge efforts and, in some cases, it is even impossible. A simple and straightforward solution is to train a model on a labelled dataset which is somehow related to the target data, and then apply it to the data being considered. This approach is likely to lead to substantial drops in performance caused by the domain shift, observed in the different feature and label distributions [161]. In order to solve this problem, several domain adaptation techniques have arisen, aiming to learn from a source dataset and transfer that knowledge obtained to a target dataset. Domain adaptation is currently an active research topic in the machine learning community [162] [163]. Focusing on speech technologies applications, several works have also investigated the domain adaptation problem for speech recognition [164], speaker recognition [165] or speech enhancement [166].

6.2.1 Problem formulation

In the following lines a formal introduction to the domain adaptation problem and some definitions relevant to the topic are provided. In order to formulate the domain adaptation problem two domains need to be defined: the source and the target domain. The source domain, D_s, represented by a source dataset $\Pi_s = \{\mathbf{X}_s, \mathbf{Y}_s\}$ where $\mathbf{X}_s = \{x_{s_1}, ..., x_{s_N}\}$ is the set of acoustic features with N examples, and $\mathbf{Y}_s = \{y_{s_1}, ..., y_{s_N}\}$ the speech labels defining each of the elements in $\mathbf{X}_s$ as speech or non-speech examples. Similarly, the target domain, D_t, is represented by a target dataset $\Pi_t = \{\mathbf{X}_t\}$ where $\mathbf{X}_t = \{x_{t_1}, ..., x_{t_M}\}$ is the set of acoustic features with M examples. In the case of unsupervised domain adaptation problems, such as the ones described in this work, no ground-truth labels $\mathbf{Y}_t$ are available for the target dataset.

Traditionally, in supervised learning problems training samples are assumed to be available. This is the case for the source domain. Accordingly, the learning problem is to determine a classifier $f_s(\Pi_s, \theta_s)$ that allows to obtain high classification accuracy for test samples by exploiting the available training set Π_s. The classifier is described by a set of parameters θ_s specific for each family of classifiers.

In the domain adaptation framework, the problem becomes more complex as test samples are drawn for a target domain distribution different from the source domain distribution of training samples. Considering the ideas described, the goal of domain adaptation techniques should be to develop a new classifier $f_t(\Pi_s, \theta_s, \Pi_t, \theta_t)$ that obtains an accurate prediction of test samples coming from the target domain by exploiting labelled training samples Π_s from the source domain D_s and unlabelled samples Π_t from the target domain D_t. As for supervised classifiers, this new model adopted for classification is described by a set of parameters θ_s specific for each family of classifiers, and by a set of parameters θ_t which is specific to each domain adaptation technique.

6.2.2 Approaches to domain adaptation

In order to better understand the domain adaptation problem, a variety of works over the years have tried to categorise the diverse conditions found for the problem. We refer the reader to the following survey for a more detailed description of this categorisation [167]. The first level of categorisation refers to the relation between source and target domains. Under the assumption that the source and target domain are directly related, transferring knowledge can be performed in a single step. This is usually called one-step domain adaptation. In case that assumption fails, one-step domain adaptation may not be effective. Multi-step domain adaptation [168] aims to connect two unrelated domains via a series of intermediate bridges, and then perform one-step domain adaptation.

In this work, as human speech has characteristics that may not vary among domains, we assume that source and target domains are related. Because of that, we focus on one-step domain adaptation solutions. In this scenario, domain adaptation approaches can be summarised intro three big cases according to [169]:

- **Discrepancy-based**: this family of solutions work under the assumption that fine tuning a model using labelled or unlabelled data can reduce the shift between source and target domain. Under this idea several criteria can be used to perform domain adaptation: some authors use class information to transfer knowledge between two domains [170]. Authors in [171] or [172] seek to align the statistical distribution shift between source and target domain. Other approaches also aim to improve generalisation by adjusting the architectures of DNNs, such as the work presented in [173].

- **Adversarial-based**: in this case a domain discriminator tries to identify whether a data point belongs to the source or the target domain. This is used to encourage domain confusion through an adversarial objective that minimises the distance between source and target domain distributions. Two main groups can be observed when implementing this idea: those relying on generative models such as generative adversarial networks (GAN) [174], or those that rely on non-generative models that aim to obtain domain invariant representation through a domain confusion loss [175].

- **Reconstruction-based**: This approach is based on the idea that data reconstruction of the source or target samples may be helpful in order to improve the domain adaptation process. This way the reconstructor is able to ensure both specificity of intra-domain representations and indistinguishability of inter-domain representations. Some examples of these methods can be cited, such as the use of an encoder-decoder reconstruction process [176], or an adversarial reconstruction obtained via a cycle GAN model [177].

6.2.3 Unsupervised domain adaptation techniques

Following the categorisation previously explained, in this work we focus our efforts on the evaluation of one-step, discrepancy-based domain adaptation techniques. Additionally, the three methods presented are fully unsupervised, meaning that no labels for the target domain are needed in order to obtain an adapted model. Descriptions of these methods are presented in the following lines.

Pseudo-labelling

The goal of pseudo-labelling (PL) [178] is to generate a set of predicted labels for unlabelled samples with a model trained on labelled data. This idea is an intuitive and straightforward application that can help overcome the challenge of collecting large labelled datasets. Several works in the literature have explored different algorithms for creating pseudo-labels. In [179] pseudo-labels are assigned to unlabelled samples using neighbourhood graphs. The idea of pseudo-labelling is extended in [180] by incorporating confidence scores for unlabelled samples. Authors in [181] present a new optimisation framework to iteratively update the obtained pseudo-labels. Our approach is inspired

by [178] and [182], where pseudo-labels are generated directly as the predictions of a trained neural network.

The solution we adopt for pseudo-labelling domain adaptation can be described according to the three following steps:

1. **Train source model**: first, an initial model is trained in a supervised way on the source domain.
2. **Predict target labels**: the initial model is then used to predict speech presence or absence for the unlabelled target domain.
3. **Adapt using predicted labels**: finally, the initial model is retrained in a supervised way using the pseudo-labels as if they were true labels.

Furthermore, besides performing a fine tuning of the initial source model to obtain the target model, this solution could also be used to train a target model from scratch using the obtained pseudo-labels or a combination of the source labelled data and the obtained pseudo-labels. Pseudo-labelling techniques have been used in several audio processing applications ranging from acoustic classification problems [183], diarisation [184] or speech recognition [185] with relevant results. In this work, we aim to extend the pseudo-labelling techniques to the SAD task and evaluate its performance in the framework of domain adaptation.

Knowledge distillation

The knowledge distillation (KD) [186] framework was originally proposed as a model compression method in which two DNNs are involved. These two models are usually known as teacher and student models in an analogy to the education process. The main idea of this philosophy is that the teacher model produces soft labels which are used to train the student model. Consequently, the student model should imitate the predictions of the teacher model. In order to do so Kullback-Leibler Divergence (KLD) loss between student and teacher distributions is minimised. KLD loss can formulated according to the following expression:

$$\mathrm{KLD} = -\sum_{i=1}^{I} p_t(y_i|x_i) \log\left(\frac{p_s(y_i|x_i)}{p_t(y_i|x_i)}\right), \tag{6.1}$$

where i is the example index, x_i is the input example, $p_t(y_i|x_i)$ is the output posterior probability of the label y_i from teacher model, and $p_s(y_i|x_i)$ is the output posterior probability of the label y_i from student model for the same example. As done in most KD methods [187] [188], the teacher model is usually frozen, relying on a pretrained model, in order to reduce complexity. In this case, where only the parameters of the student network need to be optimised, minimising KLD loss expressed in Equation (6.1) is equivalent to minimising the expression shown in the following equation:

$$\mathcal{L}_{\mathrm{KLD}} = -\sum_{i=1}^{I} p_t(y_i|x_i) \log(p_s(y_i|x_i)) + const\,, \tag{6.2}$$

where *const* is a constant term as defined in [189]. As it has just been explained, knowledge is transferred via the minimisation of a loss function whose target is the distribution of class probabilities predicted by the teacher model. This is the output of a softmax function applied on the teacher model logits. However, in most cases, this distribution provides a high probability for the correct class, with the other class probabilities close to zero. In order to address this issue, [186] introduced the concept of softmax temperature. The probability p_i of the class i is calculated from the neural network logits z according to the following equation:

$$p_i = \frac{e^{\left(\frac{z_i}{K}\right)}}{\sum_i e^{\left(\frac{z_i}{K}\right)}}, \tag{6.3}$$

where K is the temperature parameter. For the case of $K = 1$ the standard softmax function is obtained. As K grows, the probability distribution generated becomes softer, providing more information as to which classes the teacher found more similar to the predicted class.

The proposed experimental setup for KD based domain adaptation is shown schematically in Figure 6.1. It can be seen that both models, teacher and student, receive target data examples $\mathbf{X}_t$ as input. Predictions from both models are obtained via softmax activations using the same temperature parameter k. Soft predictions from the teacher network are used to transfer knowledge to the student network, aiming at mirroring those predictions using the mentioned KLD loss. In this process, teacher model is frozen and only parameters of the student model are updated. In order to test the system, teacher model is discarded, and final predictions are obtained using the student model with a standard softmax activation with $K = 1$. As it is implemented, this version of KD can be interpreted as a soft version of the pseudo-labelling method previously described.

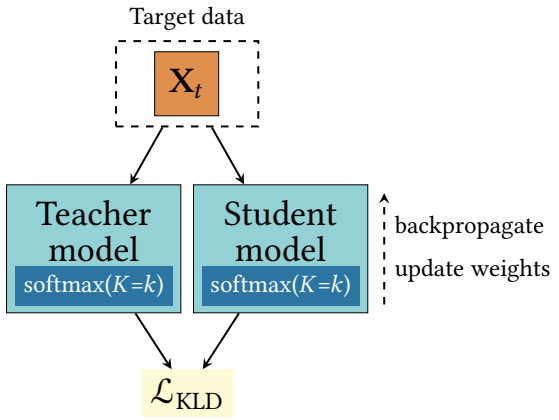


Figure 6.1: Schematic description of the proposal for the knowledge distillation domain adaptation technique.

Several examples of the use of KD techniques for solving the domain adaptation task can be found in the literature. Authors in [190] apply KD to improve acoustic models for ASR in application-specific conditions. Something similar is done in [191], that uses KD algorithms to improve ASR performance in noisy environments. Concerning the SAD

task, we can also see several examples of the teacher student architecture being used in domain adaptation solutions [192] [193].

Deep CORAL

The recently proposed Correlation Alignment (CORAL) method [172] is an unsupervised adaptation technique that is performed by aligning the second-order statistics of the source and the target distributions. However, this technique relies on a linear transformation and is not end-to-end. In order to address those issues, an extension on the CORAL method named Deep CORAL was proposed [194] with the idea of incorporating the CORAL technique directly into deep neural networks by constructing a differentiable loss function that minimises the difference between source and target correlations.

The CORAL loss between two domains for a single feature layer is described in the following lines. Suppose a set of training examples coming from the labelled source domain, D_s, described by $\mathbf{U}_s = \{u_{s_1}, \dots, u_{s_N}\}$ with $u \in \mathbb{R}^d$, and unlabelled target data $\mathbf{V}_t = \{v_{t_1}, \dots, v_{t_M}\}$, with $v \in \mathbb{R}^d$. The number of source and target data are N and M respectively. As described in the original paper, $\mathbf{U}_s$ and $\mathbf{V}_t$ represent the d-dimensional deep layer activations of a deep neural network model. Considering the provided definitions, the covariance matrices of the source and target data are given by the following equations:

$$\mathbf{C_s} = \frac{1}{N-1}(\mathbf{U}_s^T\mathbf{U}_s - \frac{1}{N}(\mathbf{1}^T\mathbf{U}_s)^T(\mathbf{1}^T\mathbf{U}_s))\,, \tag{6.4}$$

$$\mathbf{C_t} = \frac{1}{M-1}(\mathbf{V}_t^T\mathbf{V}_t - \frac{1}{M}(\mathbf{1}^T\mathbf{V}_t)^T(\mathbf{1}^T\mathbf{V}_t))\,, \tag{6.5}$$

where $\mathbf{1}$ is a column vector with all elements equal to 1. The CORAL loss is then defined as the distance between the second-order statistics of the source and target features. This is shown in Equation (6.6):

$$\mathcal{L}_{\text{CORAL}} = \frac{1}{4d^2}\| \mathbf{C_s} - \mathbf{C_t} \|_F^2\,, \tag{6.6}$$

where $\| \cdot \|_F^2$ denotes the squared matrix Frobenius norm.

The idea behind Deep CORAL adaptation technique is to obtain a set of deep features that are both discriminative enough to train a strong classifier and invariant to the change observed between source and target domains. Minimising a classification loss by itself, as usually done in supervised learning approaches, may lead to overfitting on the source domain and a reduced performance on the target domain. Contrastingly, minimising the CORAL loss alone may lead to degenerated features. In order to match the conditions stated above, the final loss to be optimised is a combination of both a classification loss and the CORAL loss. The representation of the neural architecture needed to implement Deep CORAL adaptation technique is shown in Figure 6.2. As shown, source features are forwarded through the DNN model and then a classification loss is computed using source labels. Similarly, target features are used in combination with source features to compute CORAL loss. Network parameters are shared among the two DNN models.

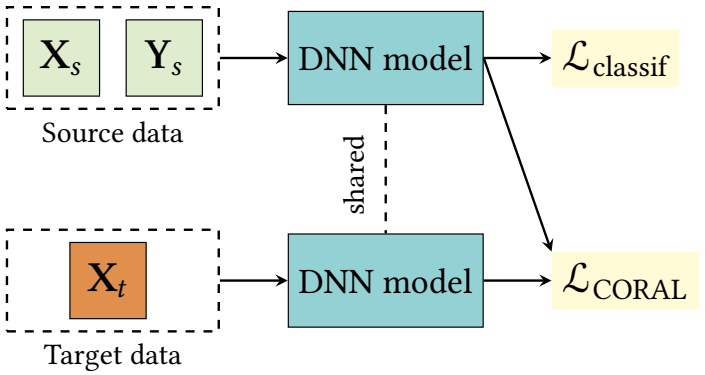


Figure 6.2: Schematic representation of Deep CORAL adaptation technique.

Considering the described architecture, the joint optimisation target of classification loss and CORAL loss is described in Equation (6.7):

$$\mathcal{L} = \mathcal{L}_{\text{classif}} + \sum_{i=1}^{z} \lambda_i \mathcal{L}_{\text{CORAL}_i} \,, \tag{6.7}$$

where L_{classif} is any traditional classification loss function such as cross entropy, z is the number of CORAL loss layers in a deep network and λ_i is a weight that trades off adaptation and classification accuracy on the source domain. The sum term depicted aims to represent the possibility of incorporating the CORAL loss on additional layers of the DNN architecture. However, as described in the original paper, authors apply the CORAL loss only to the last classification layer in the DNN architecture. In our experiments we apply CORAL loss in the same way, simplifying Equation (6.7) to become Equation (6.8) in the case where $z = 1$:

$$\mathcal{L} = \mathcal{L}_{\text{classif}} + \lambda \mathcal{L}_{\text{CORAL}} \,. \tag{6.8}$$

More recently, further work has proposed a new approach built upon the Deep CORAL method. Authors in [195] argue that the Euclidean distance used in the original Deep CORAL proposal may not be the most appropriate way to measure the distance between the source and target domains. Knowing that covariance matrices are positive semi-definite, they can be seen as two points lying on a Riemman manifold, and the metrics defined therein should consider its non-Euclidean structure [196]. Therefore, the Euclidean distance as defined in Equation (6.6) may be seen as only suboptimal in such a space. Considering this information, the log-Euclidean distance is instead a Riemannian metric that better captures the manifold structure. This metric is defined according to the following equation:

$$d_{\log}(\mathbf{X}, \mathbf{Y}) = \| \log(\mathbf{X}) - \log(\mathbf{Y}) \|_F \,, \tag{6.9}$$

where $\log(\mathbf{X})$ is the matrix logarithm of $\mathbf{X}$ [197]. Similarly as the CORAL loss, the log CORAL loss can be obtained by replacing the Euclidean distance in equation (6.6) with the log-Euclidean distance. This is shown in equation (6.10):

$$\mathcal{L}_{\log \text{CORAL}} = \frac{1}{4d^2} \| \log(\mathbf{C_s}) - \log(\mathbf{C_t}) \|_F^2 \,. \tag{6.10}$$

Through the eigenvalue decomposition of matrices $\mathbf{C_s}$ and $\mathbf{C_t}$ in Equation (6.10) we obtain the final expression for Log Deep CORAL loss:

$$\mathcal{L}_{\log \text{CORAL}} = \frac{1}{4d^2} \| \mathbf{S} \operatorname{diag}(\log(s_1), \cdots, \log(s_d)) \mathbf{S}^{\mathrm{T}} - \mathbf{T} \operatorname{diag}(\log(t_1), \cdots, \log(t_d)) \mathbf{T}^{\mathrm{T}} \|_F^2, \quad (6.11)$$

where d denotes the dimension of the features whose covariances are intended to align, as previously explained, $\mathbf{S}$ and $\mathbf{T}$ are the matrices that diagonalise $\mathbf{C_s}$ and $\mathbf{C_t}$ respectively, and s_i and t_i are the corresponding eigenvalues. The final setup for the log CORAL loss is the same as the one explained for the original CORAL loss, being described in similar terms as the ones presented in Equation (6.7): a classification loss is combined with the log CORAL loss to obtain the global loss term.

6.3 Experimental setup

6.3.1 Data description

The idea behind the baseline model training is to obtain a generic model exposed to a huge variety of data, so that an adaptation to new unseen domains can be performed later by transferring that general knowledge to a target dataset. Table 6.1 summarises datasets used for training and evaluating the baseline SAD system. As it can be observed, the

Table 6.1: Data description for baseline SAD training in the unsupervised domain adaptation set of experiments.

	Domain		
	Broadcast	**Telephone**	**Meetings**
Train data	Albayzín 2010 - TV3 [11] Albayzín 2018 - RTVE [125] MGB [126]	SRE08 Summed	AMI [127] ICSI Meet. [128]
Test data	Albayzín 2020 - RTVE [129]	CALLHOME [130]	RT09

baseline SAD is trained on a combination of data coming from three domains: broadcast, telephone channel and meetings. For the broadcast domain, the system is trained on a combination of data from previous Albayzín evaluation campaigns (2010 and 2018) and data from Multi-Genre broadcast (MGB) challenge. For the meetings domain, AMI and ICSI meetings corpora are used. Finally, in order to represent the telephonic domain, the summed partition of NIST 2008 speaker recognition evaluation (SRE) is incorporated into training. In addition, 10% of each training dataset considered is reserved to generate a validation subset containing data across the three domains considered. As described in Table 6.1, we also reserve an additional dataset from each domain to evaluate the obtained results in that domain with the baseline SAD system. These datasets are: Albayzín 2020 evaluation partition for broadcast domain, CALLHOME dataset for telephonic domain, and the dataset originally released for NIST Rich Transcription (RT) 2009 evaluation for meetings domain.

The main goal of this work is to adapt SAD models trained on a variety of domains to a new unseen domain. This new domain is the one introduced in the Fearless steps challenge, with audio from Apollo space missions featuring quite degraded channels and several kinds of transmission noises. In the following lines, we describe partitions provided originally in the second phase of the challenge (FS-02) and explain how they have been used in this work. Concerning the training subset, despite the ground truth labels for this partition are available, in this set of experiments we consider the target domain unlabelled, so we make no use of those labels in our systems. The training subset audio is used then as target data in order to adapt SAD models to this new unseen domain. The development subset is used in this work to evaluate our systems, in terms of the particular metrics described previously. The evaluation partition provided by the organisation was not used in this work. The scoring of this subset was performed by organisers while running the challenge and labels have not been released publicly, making it impossible to obtain comparable results on this subset at the moment of developing this work. Furthermore, all the obtained results in this work for the Fearless steps data are under the challenge conditions because participants were allowed to use any available data in addition to the data provided by the organisation to train and tune their systems.

6.3.2 Neural network classifier

As the main element for the SAD system we opt for a CRNN based classifier. Particularly, we use the variant applying 2D convolutions from the models already presented in our previous work [198]. This specific model was chosen because it was the simplest model obtaining the best performance when compared to its variant using 1D convolutions. The schematic representation of the proposed CRNN model is described in Figure 6.3. As

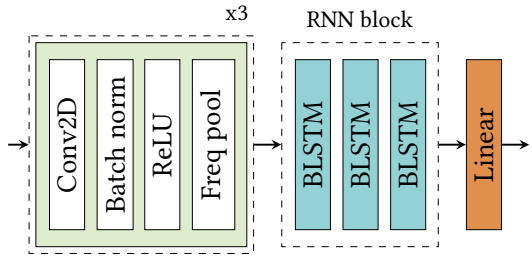


Figure 6.3: Convolutional recurrent neural network model proposed for the SAD domain adaptation experiments.

it can be observed in the figure, the model is mainly composed of two elements. First, three 2D convolutional blocks are used to process input features. Each of these blocks is integrated by a 2D convolutional layer with 3x3 kernel size and 64 filters. Then it is followed by a batch normalisation layer [151] and the application of a rectified linear unit (ReLU) [100] activation function. Finally, a max-pooling mechanism is applied considering a 4x1 stride, so that only the frequency axis is downsampled. Then the output of the last convolutional block is fed to the RNN block, generated by stacking three bidirectional LSTM layers with 128 neurons each. This block is then followed by a linear

layer that generates the final representation as a two neurons output. Raw speech score is computed then as the difference between the logit value from both neurons.

6.3.3 Feature extraction

As a first preprocessing step, all the audios considered for this work were downsampled to 8 kHz and converted to a single channel input. As input features for the proposed neural network based SAD system we consider log Mel filter-bank energies. The approach followed is the same as the one described in Section 5.3.2, considering 64 log Mel coefficients concatenated with the log energy of the frame. Features are computed every 10ms using a 25 ms Hamming window. As a final step, mean and variance normalisation is performed at file level. This set of features described is shared among all experiments described in this chapter.

6.3.4 Evaluation metrics

In order to evaluate our results, the evaluation protocol originally proposed in the Fearless steps challenge [124] is followed. Results are reported according to the DCF metric, previously described in Equation (5.1). In addition to DCF, which depends on the chosen threshold, results of the different systems are also reported using the AUC metric and the EER. Furthermore, the desegregated performance for all possible operating points is described using the DET curve, showing FPR values versus FNR values.

6.4 Results

6.4.1 Baseline system

As starting point for the experimentation, the main objective is to obtain a baseline system so that further experiments could be compared against. This baseline model is the one considered as the unadapted model, trained only with out of domain data. This model is fine tuned in the following experiments using the methods previously explained in order to obtain a model adapted to the unseen domain. The experimental setup is built upon the description provided in Section 6.3. Concerning the details of the optimisation process, Adam optimiser is used with a learning rate that decays exponentially from 10^{-3} to 10^{-4} during 20 epochs. Minibatch size is chosen to maximise the GPU memory usage. Model selection is performed by choosing the best performing model in terms of frame classification accuracy on the validation subset. Unless stated otherwise, these optimisation details are common among all the following experiments described in this chapter.

Results for the baseline system trained on broadcast, telephonic and meetings domains in terms of AUC and EER can be observed in Table 6.2. Additionally, we also report the results of one of our submissions to the original FS02 challenge [198] that

Table 6.2: AUC and EER on three in domain datasets and FS02 development partition compared to a system submitted to the challenge trained using only data from FS challenge.

Model	Train domain	Test domain	Dataset	AUC(%)	EER(%)
Baseline	Broad. + Tele. + Meet.	Broadcast	Albayzín 2020	98.12	6.68
		Telephonic	CALLHOME	96.70	7.62
		Meetings	RT09	97.07	6.98
Baseline	Broad. + Tele. + Meet.	Apollo missions	FS02 - Dev	97.57	6.55
Challenge submission	Apollo missions	Apollo missions		99.56	3.28

was trained using the same experimental setup but with data coming from the training partition provided for the challenge. This result is presented in order to provide an upper bound for the neural architecture performance in case it was trained with in domain data. First, we report results for the three domains that the SAD baseline system has seen in the training process. In general terms, it can be observed that competitive results are obtained on the three domains shown, with EER values around 7%. Focusing on the obtained results for FS02 development partition, it can be observed that, as expected, the baseline system underperforms when compared to the challenge submission trained with in domain data. Particularly, a drop in performance close to 50% can be observed in terms of EER. In the following subsections we aim to fill the gap between the baseline model and the upper bound provided by the system submitted to the challenge by using unsupervised domain adaptation techniques.

Results shown in Table 6.2 are complemented with those presented in Figure 6.4. In this figure we show the DET curve and EER for the baseline system and the challenge submission system on FS02 development partition. As it can be observed, a similar drop in performance measured by the EER metric is applicable to all points in DET curve. Furthermore, the baseline system tends to provide higher FNR values, whereas the challenge submission curve shows a relatively constant slope in all the displayed range. As a point for comparison, for FPR = 1%, the challenge submission provides FNR = 10% while the baseline unadapted system yields FNR greater than 50%.

Now that the baseline system has been appropriately characterised and set in context, in the following subsections we present the results for the three domain adaptation techniques described in this work.

6.4.2 Pseudo-label domain adaptation

As described in Section 6.2.3, pseudo-labelling is a method traditionally used to alleviate the need for labelled data. In the following experiments, we use it in order to adapt a model to a new unseen domain. The first step needed to perform this technique is to obtain a new set of pseudo-labels for the target data. To do so, we run the FS02 train-

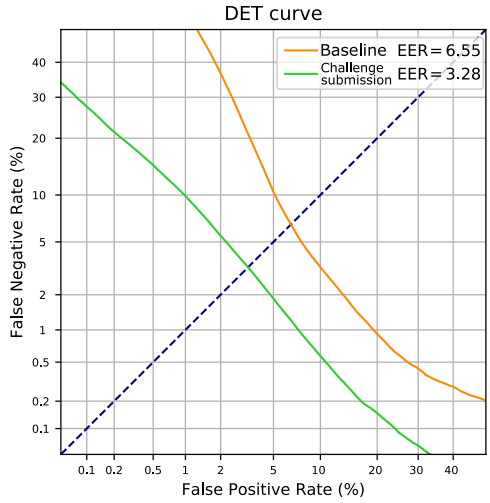


Figure 6.4: DET curve and EER on FS02 development partition comparing a baseline system trained using out of domain data and our submission to the challenge trained using in domain data.

ing partition through the previously obtained baseline SAD model and store the speech scores, that are then thresholded accordingly to obtain the final pseudo-labels. In the next step, those pseudo-labels are considered as the ground truth for the target data and used to train a new model in a supervised way.

Even though labels for FS02 training partition data are not used in this work to train any model, those labels are still available and can be used to obtain an objective evaluation of the pseudo-labels obtained via the baseline model. This evaluation is shown in Table 6.3 in terms of AUC and EER. Furthermore, as this method relies heavily on the operating point chosen for the pseudo-labels, we also report FPR and FNR for three operating points, one with low FPR, one with balanced FPR and FNR, and one with low FNR. By using these operating points, three sets of pseudo-labels are obtained. Experiments are performed separately for each set of pseudo-labels in order to observe the influence of the operating point in this domain adaptation strategy. As it can be observed, the values

Table 6.3: AUC, EER, FPR and FNR on three operating points for the pseudo-labels obtained using the baseline model on FS02 training partition.

Dataset	AUC(%)	EER(%)	Operating point	FPR(%)	FNR(%)
FS02 - Train	97.36	7.28	Low FPR	5.78	10.00
			Balanced	7.37	7.15
			Low FNR	11.04	3.56

for AUC and EER are in line with those obtained with the baseline model on the development partition, yet the EER is slightly greater for the training partition. Concerning the operating points considered, it should be noted that, in general terms, by using these

pseudo-labels the neural network is dealing with an approximately 15% of wrong labels in the adaptation process.

Once pseudo-labels have been obtained using the baseline model, several alternatives can be used to obtain a new adapted model. In this work, we explore two of those alternatives. On the one hand, we train a new model from scratch using the same experimental setup as the one described for the baseline model but using FS02 training partition audio and the obtained pseudo-labels as ground-truth. On the other hand, we also evaluate the possibility of fine tuning the baseline model via the obtained pseudo-labels for the FS02 training partition, using a learning rate ten times smaller than the one used in the original training process. Table 6.4 describes the obtained results in terms of AUC and EER for each possible operating point and for both training strategies. When compared

Table 6.4: AUC and EER for FS02 development partition using the pseudo-label domain transfer setup training a new neural network from scratch and fine tuning the baseline neural network.

Pseudo-labels	**From scratch**		**Fine tuning**	
	AUC(%)	**EER(%)**	**AUC(%)**	**EER(%)**
Low FPR	98.06	6.43	98.03	6.20
Balanced	98.21	6.47	98.09	6.44
Low FNR	98.26	6.66	98.19	6.50

to the unadapted baseline system, it can be observed that the results presented using the pseudo-labelling method share two common characteristics: a minor improvement in terms of AUC metrics, while the EER remains similar to the one reported in the baseline system. No significant difference can be observed between the two training strategies presented. Concerning the operating points for the pseudo-labels, the low FPR operating point yields the lowest EER for both training strategies, while at the same time it also reports the lowest AUC values.

In order to further understand the behaviour of the pseudo-labelling strategy, Figure 6.5 shows the DET curve and EER on the FS02 development partition for the baseline system and the systems trained using pseudo-labelling domain adaptation methods. From Figure 6.5, it can be seen that the pseudo-labelling technique provides no significant improvement in EER values. On the other hand, we can see that the improvement in AUC metric observed previously comes from the improvement compared to the baseline system in DET curve in the areas with high FPR and FNR values. In general terms, experimental results suggest that pseudo-labelling techniques can help obtaining a DET curve with constant slope, reducing error in areas with high FPR and FNR values, while not significantly modifying the EER value of the system used to obtain pseudo-labels.

6.4.3 Knowledge distillation domain adaptation

According to the theoretical explanation provided in Section 6.2.3, we aim to perform domain adaptation using the knowledge distillation framework applied to the SAD task. In the following, we describe the training process. First, teacher and student models

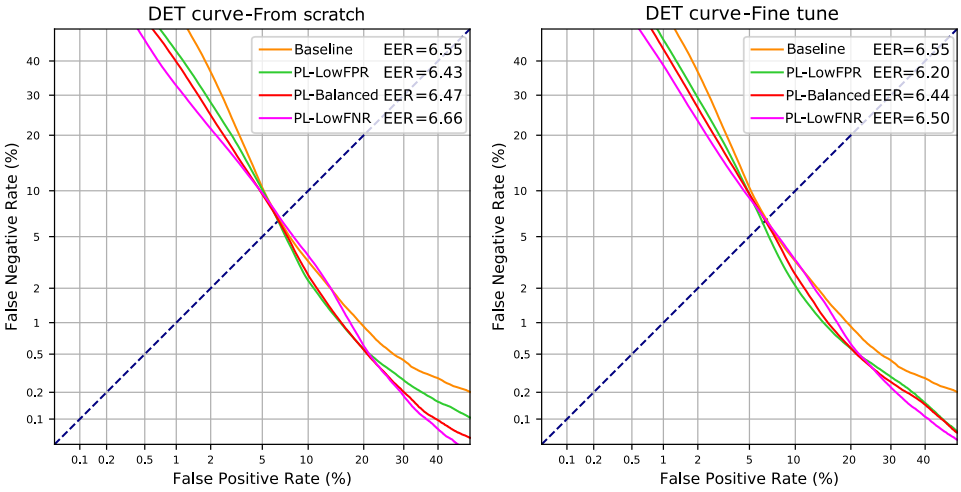


Figure 6.5: DET curve and EER on the FS02 development partition using the pseudo-label domain transfer setup training a new neural network from scratch (left) and fine tuning the baseline neural network (right), both compared to the baseline system.

are initialised using the unadapted baseline model. Teacher model weights are frozen during the entire training, and only student model weights are updated. The output of both models is compared using KLD loss after going through softmax activation with a temperature parameter K, shared for teacher and student models. At inference, the softmax layer is used in its standard form with temperature $K = 1$. Obtained results in terms of AUC and EER using knowledge distillation are shown in Table 6.5 for various values of the temperature parameter K. A decreasing tendency for EER can be observed

Table 6.5: AUC and EER on the FS02 development partition using knowledge distillation domain adaptation setup with various softmax temperature values compared to the baseline system.

Softmax temperature	**AUC(%)**	**EER(%)**
$K = 1$	97.79	6.41
$K = 10$	97.72	6.32
$K = 20$	97.80	6.27
$K = 30$	97.80	6.25
$K = 40$	97.72	6.23
K = 50	**97.86**	**6.02**
$K = 60$	97.70	6.33

when increasing the temperature parameter up to $K = 50$, achieving a best EER value of 6.05%. However, this tendency is not consistent for AUC metric, showing values in between 97.80 and 97.86. When compared to the baseline system, best temperature configuration reports 8.10% relative improvement on the EER value, yet this improvement in EER only leads to 0.29% relative improvement on the AUC metric. In general terms, an

improvement in performance can be observed using the temperature softmax activation as argued by [186], however this improvement is limited.

Results in Table 6.5 are complemented with those presented in Figure 6.6. In this Figure we present the DET curve and EER for some of the best performing knowledge distillation systems compared to the unadapted baseline system. As observed in Figure

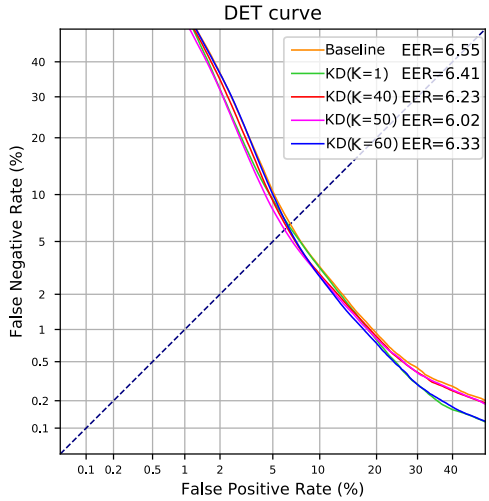


Figure 6.6: DET curve and EER on the FS02 development partition using the knowledge distillation domain adaptation setup with various softmax temperature values compared to the baseline system.

6.6, unlike the pseudo-labelling strategy, knowledge distillation seems to be able to decrease the EER point on the DET curve by using a large temperature parameter. However, it can also be seen that all curves are really close to each other, this being translated in AUC values very similar to those obtained by the baseline system. In general terms, it can also be observed that knowledge distillation does not correct the baseline system tendency to provide high FNR values. This may be due to the fact that KLD loss makes the student network mimic the predictions of the teacher network, so probability distributions and the relations between speech and non-speech observations should remain similar.

6.4.4 Deep CORAL domain adaptation

As an additional third alternative to the previously evaluated unsupervised domain adaptation techniques, in the following lines we evaluate experimentally the feasibility of Deep CORAL and its variations for the SAD task. Following the theoretical explanation provided in Section 6.2.3, we train a new model using the Deep CORAL and Log Deep CORAL techniques using the same experimental setup: the baseline model is used to initialise the new adapted model, that is then fine tuned for 10 epochs using a learning rate decaying exponentially from 10^{-4} to 10^{-5} (10 times smaller than the one used to train the baseline model). CORAL and log CORAL losses are applied only on the final linear layer of the DNN classifier. Final loss term is then computed using cross entropy loss

considering the source labels, and the respective CORAL loss weighted by a factor λ. As described in the original paper, λ value was chosen so that, at the end of the training, the classification loss and the CORAL loss are in the same order of magnitude. Obtained results using both, Deep and Log Deep CORAL methods, are shown in Table 6.6 in terms of AUC and EER for three λ values in the same order of magnitude. As observed in Table

Table 6.6: AUC and EER for FS02 development partition using Deep CORAL and Log Deep CORAL domain adaptation setups using various λ weights.

Method	CORAL weight	AUC(%)	EER(%)
Deep CORAL	λ= 0.9	98.19	5.53
	λ= 1.0	98.18	5.43
	λ= 1.1	98.25	5.53
Log Deep CORAL	λ= 0.9	98.26	5.36
	λ= 1.0	98.26	5.48
	λ= 1.1	98.24	5.37

6.6, both Deep CORAL and Log Deep CORAL, provide the lowest EER values obtained in this work so far through model adaptation. Best result in terms of EER is obtained using Log Deep CORAL method, with an EER of 5.36%, which results in 18.17% relative improvement when compared to the unadapted baseline system. Concerning the AUC values observed, obtained results are in line with the best performing system using the pseudo-labelling techniques, showing also better values than the knowledge distillation method. As done in previous experiments, we also report the DET curves in order to observe the behaviour of the adapted systems in all possible operating points. This curve is shown for the Deep CORAL (left) and Log Deep CORAL (right) systems in Figure 6.7 for multiple values of λ compared to the DET curve of the baseline system.

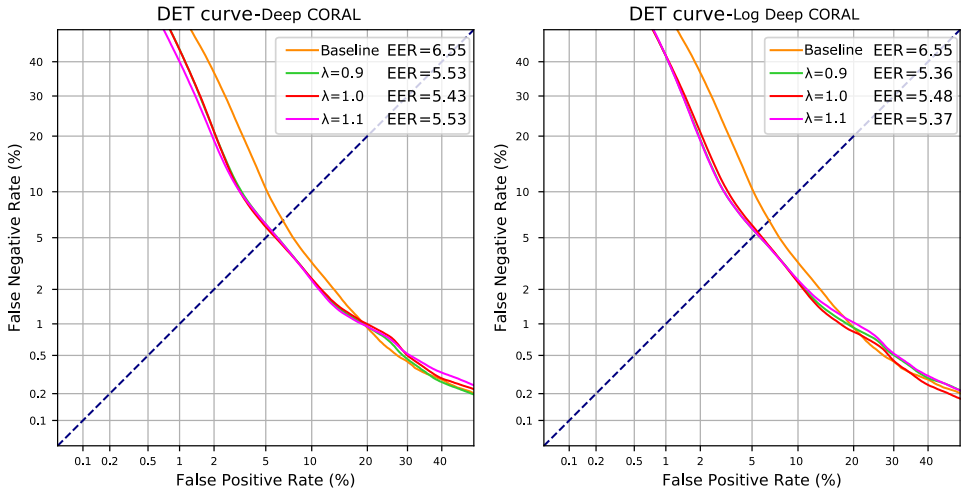


Figure 6.7: DET curve and EER on the FS02 development partition using Deep CORAL and Log Deep CORAL domain adaptation setups using multiple λ weights compared to the baseline system.

As observed, CORAL based domain adaptation techniques provides the best improvement in the DET curve, so far in this set of experiments. When compared to the baseline system, besides significantly decreasing the EER, an overall improvement can be seen in the DET curve for the areas reporting high FNR values. That is the reason why the AUC value reported increased when compared to the baseline system. Both, Deep CORAL and Log Deep CORAL, seem to be insensitive to λ value, showing a similar performance as long as λ remains in the same order of magnitude.

By observing the behaviour of the Log Deep Coral method and the pseudo-label strategies previously characterised, it becomes apparent that both solutions may be complementary. While the Log Deep CORAL DET curve shows no improvement over the baseline for high FPR values, the DET curve for the pseudo-labelling method obtains its best results in that area, making it the one with best performance for high FPR values over the three methods evaluated. This fact suggest that combining both methods, applying them in cascade, might provide even further improvements to the SAD neural network. This idea is evaluated experimentally in the following subsection.

6.4.5 Cascaded application of domain adaptation methods

In view of the results presented in previous subsections, this final experiment evaluates the possibility of applying two domain adaptation methods in a cascaded setup in order to improve even further the results on the new unseen domain. By combining the capabilities of CORAL based adaptation to obtain an overall boost in performance and the pseudo-label adaptation to obtain a DET curve with a constant slope, we use both of them in a cascaded setup. The baseline model is first adapted using the Log Deep CORAL method. Then the adapted model is used to extract a new set of pseudo-labels, which are later used to obtain a final model using both training strategies described in previous experiments, either training a new model from scratch, or fine tuning the previous model. Table 6.7 shows an objective evaluation of the pseudo-labels obtained via the Log Deep CORAL model in terms of AUC, EER and FPR and FNR for the same three possible operating points considered in previous experiments. As expected, we can observe that the

Table 6.7: AUC, EER, FPR and FNR on three operating points for the pseudo-labels obtained using the best performing model adapted through Log Deep CORAL method on FS02 training partition.

Dataset	AUC(%)	EER(%)	Operating point	FPR(%)	FNR(%)
FS02 - Train	98.26	5.63	Low FPR	4.55	7.55
			Balanced	5.70	5.54
			Low FNR	7.27	3.78

overall quality of the pseudo-labels has improved when compared to those obtained using the baseline model (see Table 6.3). Improvements obtained are in line with the ones presented on the FS Development subset when using the Log Deep CORAL method, with the EER metric decreasing from 7.28 to 5.63%. By using this new set of pseudo-labels the adaptation process is performed feeding the neural network with an approximately 11%

of wrong labels distributed according to the three operating points shown. Obtained results using this new set of pseudo-labels are described in Table 6.8 in terms of AUC and EER for each possible operating point and for both training strategies described. The

Table 6.8: AUC and EER for FS02 development partition using pseudo-label domain transfer setup training a new neural network from scratch and fine tuning the best performing model adapted using Log Deep CORAL method.

Pseudo-labels	From scratch		Fine tuning	
	AUC(%)	EER(%)	AUC(%)	EER(%)
Low FPR	98.86	5.21	98.85	5.34
Balanced	98.81	5.50	98.83	5.49
Low FNR	98.75	5.67	98.85	5.51

previously observed behaviour of the pseudo-labelling method is repeated in this new experiment. In general terms, compared to Log Deep CORAL model (AUC = 98.25%, EER = 5.48%), it can be seen that pseudo-labelling strategies provide an improvement on the AUC metric while maintaining a similar EER value. As a matter of fact, all the AUC values reported outperform those from previous experiments, with the best case scenario of AUC = 98.86% obtained by training a new model from scratch using low FPR pseudo-labels. Concerning the training strategies considered, experimental results suggest that no significant difference can be found between training a new model from scratch or fine tuning the previous stage model. In terms of operating point, the EER value obtained using low FPR is slightly lower for both training strategies, however this difference becomes insignificant when considering the AUC metric.

6.4.6 Discussion

Once all the results for the unsupervised domain adaptation methods have been described, in this section we aim to provide a brief discussion on its behaviour, setting them in the context of the original FS challenge and using the original DCF metric in order to obtain a performance comparison. Table 6.9 presents a summary of the best obtained results using all methods explored (pseudo-labels, KD, Deep CORAL), and the cascaded application of both of them in terms of AUC, EER, and DCF metric as used in the FS challenge. We also report the relative improvement obtained in DCF metric compared to the unadapted baseline system. For comparison, we present the challenge baseline result provided by the FS challenge organisation [122], and our submission to the challenge trained using in domain data.

Concerning the results of the pseudo-labelling strategy, we can observe a relative improvement in DCF metric between 11% and 12% when compared to the unadapted baseline system. This improvement comes mainly from the correction made to the DET curve, with those systems showing a behaviour with more constant slope. This means that systems adapted using this method show a better performance in areas with high FPR or high FNR while, at the same time, EER remains very similar to that of the baseline

Table 6.9: AUC, EER, DCF and DCF relative improvement over the unadapted baseline model on the FS02 development partition using the three evaluated domain adaptation methods, and the cascaded application of two of them with the best performing hyperparameter configuration in terms of DCF metric compared to the original challenge baseline and our submission to the challenge trained using in domain data.

Model	**AUC(%)**	**EER(%)**	**DCF(%)**	**Rel. improv.(%)**
Baseline	97.57	6.55	4.84	-
Pseudo-labels (fine tune)	98.03	6.20	4.25	12.19
Pseudo-labels (scratch)	98.21	6.46	4.31	10.95
Knowledge distillation ($K = 1$)	97.79	6.41	4.78	1.24
Knowledge distillation ($K = 50$)	97.86	6.02	4.56	5.79
Deep CORAL ($\lambda = 1.1$)	98.25	5.53	4.26	11.98
Log Deep CORAL ($\lambda = 1.0$)	**98.25**	**5.48**	**4.20**	**13.23**
Log CORAL + PL (fine tune)	98.85	5.34	3.67	24.17
Log CORAL + PL (scratch)	**98.86**	**5.21**	**3.65**	**24.59**
Challenge baseline	-	-	12.50	-
Challenge submission	99.56	3.28	2.56	-

system. The knowledge distillation systems offer the lowest improvement in DCF of all methods evaluated. Even though its configuration with high softmax temperature still shows a non-neglectable 5.79% relative improvement compared to the baseline system, this solution shows limited applications. DET curve is really similar to the one obtained by the baseline system, with limited improvement in AUC. Finally, CORAL based methods show one of the most promising results of this study. The best hyperparameter configuration using Log Deep CORAL achieves a DCF relative improvement of 13.23% compared to the baseline system. These results shows the lowest EER value in this work in the case of the application of a single technique, while also reporting an overall improvement for the full DET curve.

Best results in terms of DCF are obtained by applying Log Deep CORAL and pseudo-labelling in a cascaded setup. In the case of training a new model from scratch using pseudo-labels from the previous step, DCF is 3.65%, which is equivalent to 24.59% relative improvement compared to the unadapted baseline system. Furthermore, experimental results confirm that both methods are complementary. As shown, the total relative improvement with both techniques is equivalent to the sum of relative improvements when used separately.

As a final point in this discussion, and in order to provide a condensed view of the best obtained results in this work, Figure 6.8 presents DET curve and EER of the three evaluated methods by themselves (left), and the cascaded application of Log Deep CORAL and pseudo-labelling (right) using the best hyperparameter configuration in terms of DCF metric compared to the unadapted baseline and our submission to the challenge trained using in domain data.

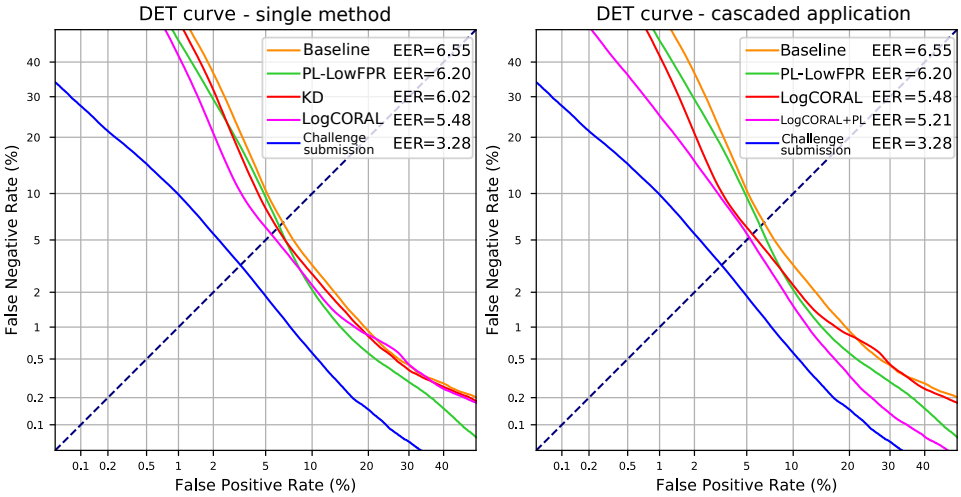


Figure 6.8: DET curve and EER on the FS02 development partition using the three evaluated domain adaptation methods (left), and the cascaded application of two of them (right) with the best performing hyperparameter configuration in terms of DCF metric compared to the unadapted baseline system and our submission to the challenge trained using in domain data.

Observing the comparative of the single method application (left), it can be seen again that the most promising result in terms of improving the DET curve is obtained by the Log Deep CORAL method, while the improvement of the knowledge distillation method is limited. Focusing on the comparison on the cascaded application of Deep CORAL and pseudo-labelling (right), it can be observed that the DET curve obtained for the cascaded application of both methods (in pink) supports again the hypothesis that both techniques are complementary. Applying the pseudo-labelling strategy on top of the Log Deep CORAL model results in a DET curve with a similar EER value but a performance significantly improved in areas with high FPR and high FNR. With this method we achieve the best DET curve in this work that, as already presented, allows to decrease even further the DCF metric, which is influenced in a higher way by false negative errors. As an example operating point for comparison, focusing on a FPR value of 1%, the unadapted baseline system shows a FNR value greater than 50%. While the Log Deep CORAL model reduces this value to be around 40%, the improvement is much more significant with the combination of both adaptation methods, that reports a FNR value of 25%.

Even though there is still a gap between the best obtained results and a model trained using in domain data, in practical scenarios, where no labels are available, experimental results have proved that unsupervised domain adaptation can improve significantly the performance of SAD systems. Best results are obtained using an approach that combines two methods. The application of this solution increases the computational complexity of the adaptation process in training time. However, the increase in complexity introduced by pseudo-labelling is minimal compared to the one already introduced by Log Deep CORAL. The latter implies a training process that computes two covariance matrices for CORAL loss and a classification loss, while the former only requires inference computa-

tion on the unlabelled data and then a simple training process with a classification loss. Furthermore, inference complexity remains the same in all cases as there is only need to compute the final adapted model to obtain SAD predictions.

6.5 Conclusions

In this chapter the use of unsupervised domain adaptation techniques in the context of the SAD task has been explored. An initial baseline model was trained on a variety of well-known domains with large amounts of labelled data available. Then, a study was performed on three methods that allow to perform adaptation directly on the model space with the objective of fine tuning the mentioned baseline model using only unlabelled data. We have used the data provided in the FS challenge, coming from a singular domain such as Apollo space missions, to experimentally validate in the SAD task the methods presented. Yet the methods are general enough so that they can be easily applied to other datasets. Furthermore, no labels are required for them to be used, significantly reducing the constraints for choosing them in practical applications.

Through the application of Deep CORAL based domain adaptation methods, results show a 13% relative improvement in the DCF metric of the original challenge. Furthermore, the cascaded application of Deep CORAL and pseudo-labelling techniques provides the best results in this study, with a significant 24% relative improvement compared to the baseline system. These experimental results suggest that Deep CORAL and pseudo-labelling techniques are complementary. The first one providing an overall improvement in the DET curve and reducing the EER. The second one improves the AUC value by modifying the DET curve so that its slope becomes constant, specially in areas with high FPR and FNR values. The improvements in performance observed allow to substantially reduce the gap for the SAD task between a system trained using in domain data and an approach based on fully unsupervised adaptation. Even if the knowledge distillation method shows an improvement in performance compared to the unadapated baseline model, this improvement is limited compared to the one observed by CORAL based techniques. This kind of domain adaptation methods, seeking to minimise the statistical distribution shift between source and target domains, provide one of the most promising results in this set of experiments.

"That's one small step for a man;
one giant leap for mankind."

Neil Armstrong

7

Self-supervised representation learning for speech activity detection

7.1 Introduction and motivation

While current state-of-the-art SAD systems rely on supervised learning techniques, experiments described in Chapter 6 allowed to quantify the loss in performance observed when SAD systems need to operate in a domain different from the one they were

trained on. This case was tested using a traditional set of perceptual features, namely log Mel filter-bank energies. The experiments described showed that the performance of SAD systems under these circumstances could be improved by applying unsupervised domain adaptation techniques. However, this scenario still requires to gather unlabelled audio from the target domain in order to benefit from those techniques.

Recent advances in self-supervised representation learning have shown significant improvements in performance in several speech processing tasks under different domains. Motivated by this issue, we aim to incorporate recent advances in self-supervised representation learning in audio segmentation systems, specifically in the SAD task, as a way to solve possible domain mismatches observed in test data. Self-supervised representation learning could be used as a way to extract knowledge from huge amounts of unlabelled data, obtaining more robust representations even without any information from the target domain being available. The amount of works that have tested these new representations in the audio segmentation field is still limited, and it is currently an active research line within the speech processing community.

The experimental setup defined in this chapter is built again around the Fearless steps challenge, focusing now on phase III and phase IV of the challenge (in years 2021 and 2022, respectively). As a progression, these two phases were devoted to the development of single channel supervised learning strategies with the aim of testing system generalisation to varying channel and mission data. More than 10 hours of new evaluation data were included as a novelty, featuring unseen channels and additional data from Apollo-13 mission. In the following lines, we describe our submissions to these two editions, that build upon the neural architectures evaluated in the previous edition of this challenge described in Chapter 5, where CRNN models yielded competitive results. The novelty introduced is the use of the mentioned self-supervised feature learning techniques to obtain new representations of audio signals more discriminative than traditional perceptual features. In the experiments performed for phase III, we compare log Mel filter-bank energies with a wav2vec approach, while experiments in phase IV also incorporated recent wavLM models to obtain new audio features.

7.2 Self-supervised representation learning

7.2.1 wav2vec approach

Our first proposed representation learning approach is inspired by the one presented in [110], with some variations. As shown in Figure 7.1, two stages are combined to learn a feature extractor: first, an strided CNN encoder runs directly on the 8 kHz waveform mapping the input sequence X_t, corresponding to a set of samples from the audio signal, to a latent space Z_t. The total downsampling factor of the network is 80, resulting in a feature vector every 10 ms of audio. The second part is implemented as a single layer bidirectional GRU recurrent neural network [97] with 512-dimensional hidden state per direction. The output of the GRU layer is used as the context C_t from which we predict 8 values from the latent space sequence using a contrastive loss seeking to minimise the

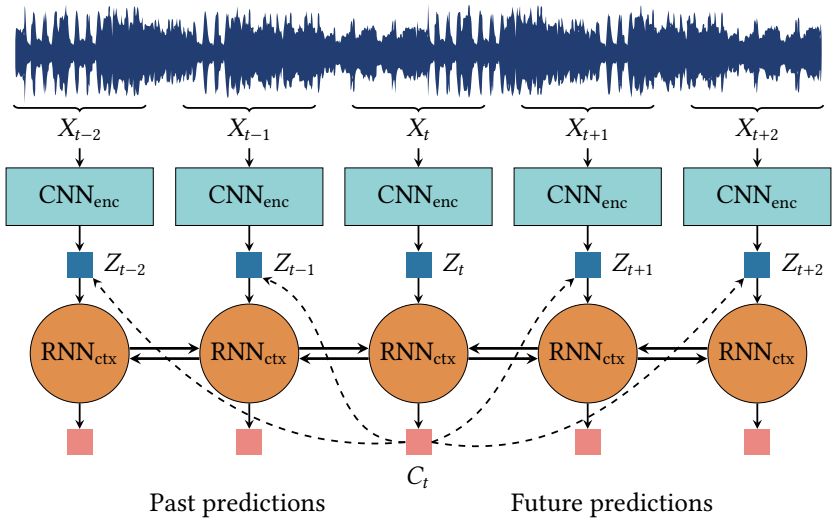


Figure 7.1: Schematic overview of the wav2vec approach for self-supervised representation learning.

InfoNCE metric [110]. Similarly to Decoar [199] or BERT text models [102], the left and right context embeddings of the bidirectional GRU are used to predict future and past values of the latent space sequence respectively, providing a loss term from future and past frame prediction tasks. The final loss function is the sum of both directional losses. This results in a final representation of 1024 dimensions being used as the context embedding. In the training stage of neural networks classifiers, the self-supervised representation learning model is frozen and these context embeddings are extracted to be used as our proposed learned features.

Concerning the prediction process of our system, unlike [114], our approach uses a single head for predicting future and past timesteps respectively, with an architecture consisting of a single hidden layer [112]. Furthermore, predictions for a clean reference Z_t are obtained using augmented data from context C_t. Noises from MUSAN database [133] are added with a signal to noise ratio (SNR) that is sampled from an uniform distribution in the range (3, 15) dB. We also simulated different room impulse responses (RIR) using the gpuRIR toolkit [200] to incorporate reverberated conditions in training time.

7.2.2 wavLM approach

As an alternative to our implementation of the wav2vec approach, we also propose to evaluate recent wavLM models [118] in the SAD task. In this case, we use the publicly available pretrained models from Microsoft's Github repository[1]. Unlike the wav2vec approach that gathers temporal information trough an RNN, the core of the wavLM approach is a transformer model. Figure 7.2 shows an schematic overview of the wavLM neural architecture. As it can be seen, two main blocks are needed for the implementation

[1] `https://github.com/microsoft/unilm/tree/master/wavlm`

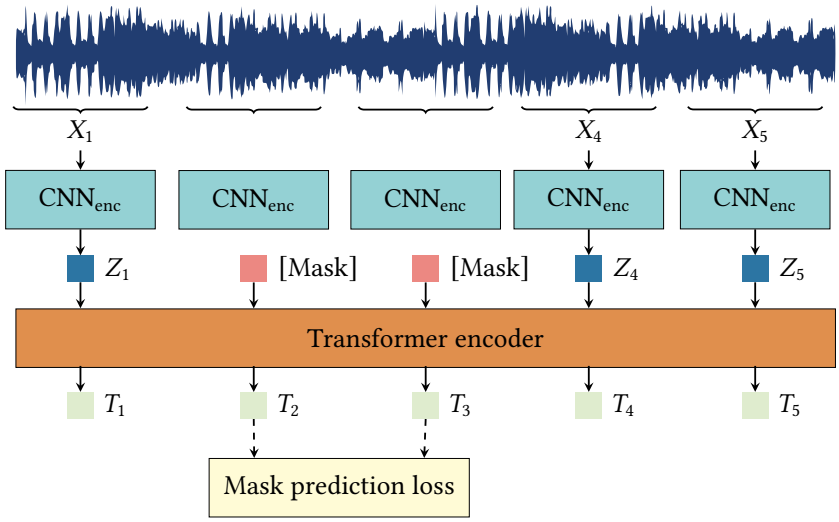


Figure 7.2: Schematic overview of the wavLM approach for self-supervised representation learning.

of this approach: first, a convolutional feature encoder that represents 25ms of audio with a stride of 20ms. This encoder is made of seven temporal convolution blocks followed by a layer normalisation and a GELU activation. Then, a transformer encoder is used to model sequential information. During the training process, the transformer consumes the masked acoustic features coming from the convolutional encoder and outputs hidden states. The training objective forces the network to predict a discrete target sequence. As done in HuBERT, those targets are obtained by applying the k-means clustering algorithm on the training data in an iterative process where MFFC features are used in the first step, and the latent representations learned are used in the following iterations. With the goal of improving model robustness in challenging acoustic environments and preserving speaker information, training process for wavLM models simulated noisy and overlapped speech at the input. Then, target sequence prediction is performed using the original speech on the masked region. A more detailed explanation of the wavLM model and training strategy can be found in [118].

In our experiments we use two variants of the wavLM models, as described in the original paper: wavLM Base+ and wavLM Large. The main difference between both is the number of parameters. The wavLM Base+ model contains 12 encoder layers and uses a hidden state with 768 dimensions, while wavLM Large has 24 encoder layers and uses a hidden state of 1024 dimensions. In both cases we use only the last hidden layer of the model as input to our SAD neural network.

Baseline perceptual features

In order to obtain a baseline result, features obtained through self-supervised learning are compared with a traditional set of perceptual features, considering log Mel filter-bank energies. Namely, we use the same configuration as the one described in Section 5.3.2 from

previous participations in the challenge, 64 log Mel filter-bank energies concatenated with the log energy of the frame.

7.3 Experimental setup

7.3.1 Data description for self-supervised learning

As it was already mentioned in previous chapters, the Fearless steps challenge follows open training conditions. This means that participants can use any available data in addition to the provided challenge data to train and tune their systems. Considering this, our proposed self-supervised representation systems can benefit from huge amounts of unlabelled training data. In the case of the wav2vec approach, the system was trained including several datasets in English: Librispeech, RSR2015, Tedlium release 1, Voxforge, Librilight, Voxceleb 1 and 2, Commonvoice (English only) and MLS (English only). This results in a total of around 130 thousands hours of unlabelled audio used in training for the wav2vec approach. Concerning wavLM models, WavLM Base+ and WavLM Large were trained using the same three datasets: Librilight, GigaSpeech and VoxPopuli, summing more than 90 thousands hours of unlabelled audio.

7.3.2 Data description for SAD classifier

In a similar way as done in Chapter 5, SAD neural network classifiers in this set of experiments are trained using exclusively data provided by the challenge organisation. The training subset provided consisted of 62.5 hours of audio, from which 10% was reserved for training validation. Results are then reported on the development subset consisting of the same 15 hours as in previous editions of the challenge, and in the new evaluations subsets including new channel and mission data. These subsets feature 34 hours of audio in the 2021 edition and 35 hours in the 2022 edition.

7.3.3 Neural network classifier

The different architectures considered for the neural network classifier are taken from our previous experience in the Fearless steps challenge. Namely, we evaluate the same systems described in Chapter 5, combining a set of convolutional layers working as a processing stage and an RNN block made of 3 BLSTM layers. Three variants are implemented for the convolutional layers, with 1D convolutions, 2D convolutions and a third approach that fuses information from both 1D and 2D convolutions in different ways. The exact details of these models can be found in Section 5.2.1.

The neural network output may result in a noisy estimation of class boundaries. Aiming to avoid high-frequency transitions, neural network outputs are smoothed using an L order averaging filter. This filter is implemented as a zero-phase FIR filter to avoid phase distortions in the output signal. The optimal value for L was empirically found to be in

the range between 50 and 60, which is equivalent to considering a moving average of 500 to 600 ms

7.3.4 Evaluation metric

The evaluation protocol for the Fearless steps challenge has remained similar in all the editions of the challenge. Results are evaluated applying the same DCF metric used in Chapters 5 and 6, that is described in Equation (5.1) considering false negative errors more important than false positive errors.

7.4 Results

Results described in this chapter are derived mainly from our submissions to the SAD task of two consecutive editions of the Fearless steps challenge, namely 2021 (phase III) and 2022 (phase IV) editions. In the following subsections we describe results obtained in each one of the challenge editions, with different approaches to the self-supervised representation learning paradigm. The main difference between both editions is that submissions to the 2022 edition also incorporated the recently released wavLM models, while results for 2021 edition only evaluate the wav2vec approach described in Section 7.2.1.

7.4.1 Results for Fearless steps challenge phase III

TSNE [201] is a well-known dimensionality reduction technique useful for visualisation of high dimensional datasets. Seeking to obtain a deeper understanding of the information provided by the new set of features learned through wav2vec methods, we compare them to perceptual log Mel filter-bank energies through a TSNE projection. Figure 7.3 shows this projection on a 2D plane of the validation susbet for both 64 log Mel filter-bank energies and the features obtained with the wav2vec system. It can be observed that log

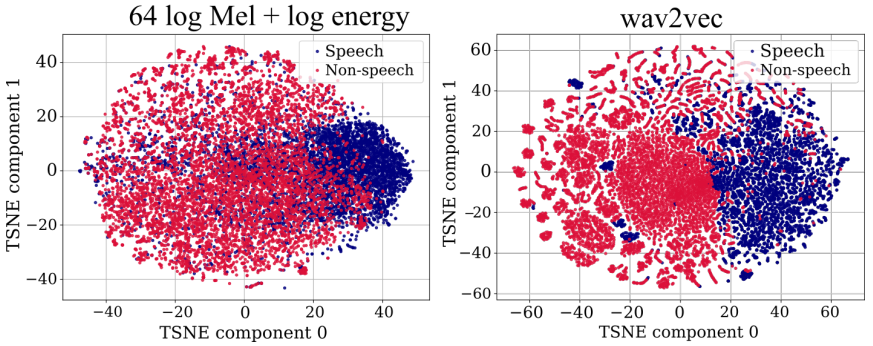


Figure 7.3: TSNE 2D projection for the validation subset of both set of features considered in this work: 64 log Mel filter-bank energies + log energy (left) and features obtained through the wav2vec system (right).

Mel filter-bank energies show a significant overlap between the speech and non-speech

classes. On the other hand, self-supervised features provide a much more separable representation of both classes, with a minimum amount of overlap between speech and non-speech when compared to log Mel filter-bank energies. It is also to interesting to observe how the self-supervised features tend to assemble on small sub-clusters. This fact may come motivated by the sequentiality introduced by the RNN from our self-supervised representation learning system, grouping together features that are temporally close.

Once we have experimentally validated the separability provided by the new self-supervised features, we evaluate them on our SAD system. Table 7.1 presents the obtained results for the different systems submitted. We compare the performance of all

Table 7.1: SAD results in terms of DCF metric on the development and evaluation partition for the CRNN trained using log Mel filter-bank energies and the proposed self-supervised features.

System	log Mel		wav2vec	
	DCF(%)		DCF(%)	
	Dev	Eval	Dev	Eval
CRNN 2D (3x3)	**1.27**	7.82	0.92	3.63
CRNN 1D	1.44	8.49	**0.65**	**2.98**
CRNN 1D dilation	1.59	**7.13**	0.91	3.66
CRNN 1D groups	1.37	8.46	0.96	3.13
CRNN fusion bilinear	1.30	7.55	0.87	3.86
CRNN fusion sum	1.28	8.64	0.97	3.11
CRNN fusion concat	1.48	9.06	0.84	3.36

the presented CRNN architectures using log Mel filter-bank energies and the representation obtained with wav2vec as input. Results are reported in terms of DCF metric for both, development and evaluation partitions. As a comparison, we also report the baseline provided by the organisation, which is based on a statistical approach [155]. This system yielded a DCF value of 12.50% and 15.61% respectively for the development and evaluation partitions. As it can be observed, all our submissions largely outperform the baseline provided by the organisation. Concerning the results using log Mel filter-bank energies as input, competitive results are obtained on the development partition, with a DCF metric of 1.27% obtained in the best case with the 2D convolutional setup. However, a strong degradation in results can be observed when comparing to the metric reported in the evaluation partition, with a best case DCF of 7.13% for the 1D convolutional setup using dilation. This fact could be expected as new data from unseen channels has been included in this new edition of the challenge, so unlike results described in Chapter 5, a bigger mismatch is observed between development and evaluation results.

An overall improvement can be observed on all the neural architectures evaluated when using the new learned features. Best results are obtained with the 1D convolutional setup, yielding a DCF metric of 0.65% and 2.98% on the development and evaluation partitions respectively. Even though the boost in performance observed using the wav2vec features is significant in the development and evaluation partitions, it should

be noted that this improvement is more relevant in the case of the evaluation dataset. While the average relative improvement observed comparing log Mel filter-bank energies and wav2vec features among all the architectures evaluated is around 37% in the development data, this percentage increases to 58% in the case of the evaluation data. Given the composition of the evaluation data in this edition of the challenge, with new unseen channels being present, these results suggest that learned features show a robust behaviour discriminating speech and non-speech classes even in possibly shifted acoustic conditions.

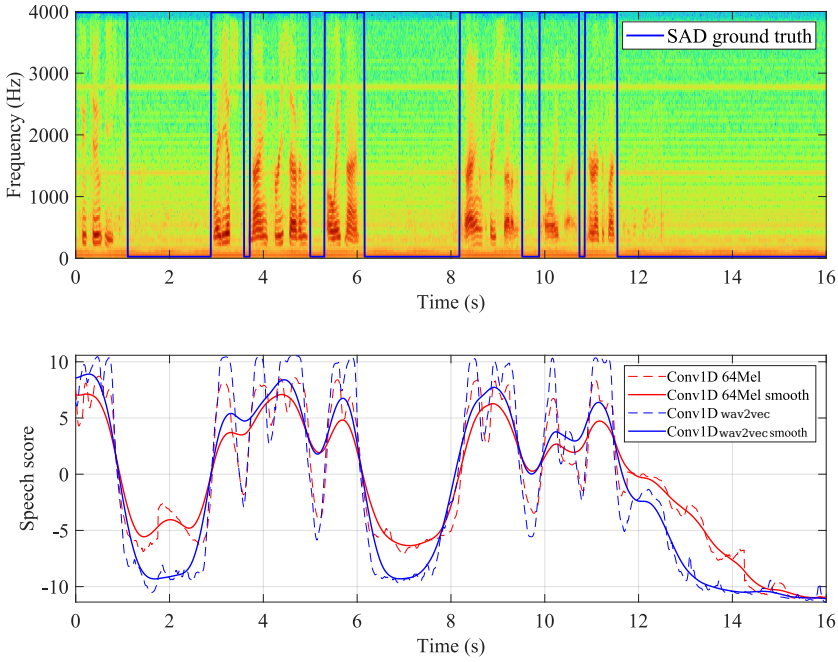


Figure 7.4: Qualitative visualisation of SAD scores in a 16 seconds audio fragment extracted from file "`fsc_p3_dev_001`". From top to bottom: spectrogram with SAD ground truth overlapped, and speech score for different SAD systems proposed.

Additionally, in the following lines a qualitative visualisation of our SAD system performance on different audio excerpts from development partition is shown, comparing the use of log Mel filter-bank energies and wav2vec features as input for the Conv1D SAD neural network. First, Figure 7.4 shows the spectrogram with the SAD ground truth and the speech score for a fragment of file `fsc_p3_dev_001`, that reported a DCF value of 0.02% for log Mel filter-bank energies and 0.01% for wav2vec features. As expected, a high speech score is associated with a strong evidence of speech presence in the audio spectrogram. Generally, it can be observed that both solutions can capture accurately the speech and non-speech fragments. It is also interesting to mention that the system using wav2vec features as input consistently provides a higher score than the setup using log Mel filter-bank energies for speech fragments, and lower scores for the non-speech fragments. Concerning the behaviour of the averaging filter, it can be seen that some of the

high frequency occurrences in the speech score are eliminated, resulting in a smoother signal being used in the thresholding process.

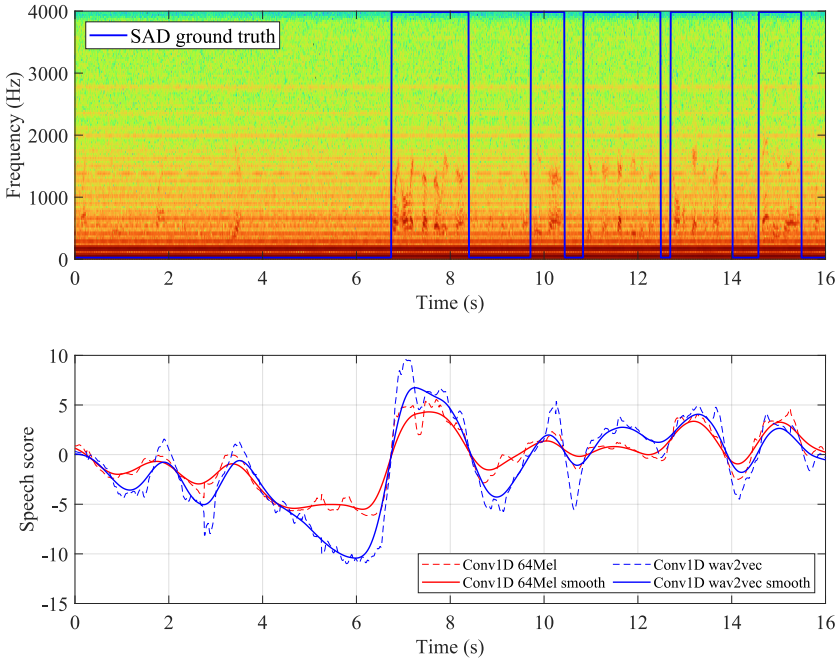


Figure 7.5: Qualitative visualisation of SAD scores in a 16 seconds audio fragment extracted from file "`fsc_p3_dev_020`". From top to bottom: spectrogram with SAD ground truth overlapped, and speech score for different SAD systems proposed.

A similar plot is shown in Figure 7.5, comparing the evolution of speech scores for both considered input features in the case of the file `fsc_p3_dev_020`. Unlike the previous example, this case illustrates more complex acoustic conditions with a reported DCF value of 6.04% for log Mel filter-bank energies and 5.50% for wav2vec features. By observing the spectrogram it can be seen that a stronger background noise is present, being mainly present in lower frequencies. Furthermore, some babble noise is also present in the signal. This may be the reason why the speech score for the first five seconds in the fragment is relatively higher compared to non-speech fragments from Figure 7.4 despite being labelled as silence. For fragments labelled as speech, it can also be observed that ,in general terms, the score tends to be lower compared to the previously shown visualisation. Concerning both set of features, same trend observed in the previous example is repeated, with wav2vec features providing higher scores than the setup using log Mel filter-bank energies for speech fragments, and lower scores for the non-speech fragments in most cases.

7.4.2 Results for Fearless steps challenge phase IV

As done in the previous section with phase III results, Figure 7.6 shows a TSNE projection on a 2D plane of the validation subset for the four set of features considered seeking

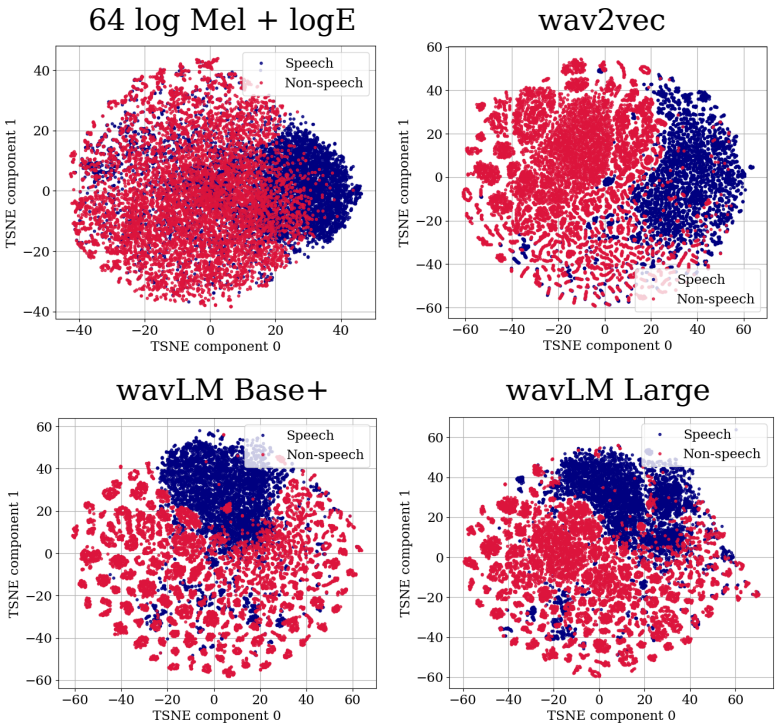


Figure 7.6: TSNE 2D projection for the validation subset for the Mel based baseline features and the three alternatives obtained through wav2vec and wavLM representation learning.

to perform an exploratory analysis of the obtained representations. In this edition results, we evaluate the wav2vec approach and the two alternatives for the wavLM model compared to log Mel energies. It can be observed that, in general terms, when compared to the log Mel filter-bank energies, any of the self-supervised approaches results in a representation where the difference between speech and non-speech samples is more distinguishable. Comparing the three self-supervised features, TSNE representation of the wav2vec approach shows a bigger unified cluster for non-speech class. In the case of wavLM models, TSNE representations behave differently, with most speech samples together in a single cluster and most non-speech sampled separated in several smaller clusters. This fact suggest that wavLM models could provide a more robust representation of speech samples, instead of focusing on the non-speech elements of audio signals.

Table 7.2 summarises results obtained in the 4th edition of the challenge in terms of DCF metric for both development and evaluation partitions. Provided that no significant difference was observed when using convolutional fusion architectures in the case of self-supervised features, in this submissions we only evaluate the CRNN architectures that use 1D and 2D convolution. In this edition of the challenge the same baseline approach as the one presented in previous editions was provided by the organisation, with an statistical solution [155] that yielded a DCF value of 12.50% and 15.61% respectively for the development and evaluation partitions. Concerning the results observed on the

Table 7.2: SAD results in terms of DCF metric on the development and evaluation partitions for the CRNN trained using log Mel filter-bank energies and the self-supervised features from wav2vec system and wavLM systems.

System	Dev DCF(%)			
	Mel	**wav2vec**	**wavLM-Base+**	**wavLM-Large**
CRNN 2D	1.54	1.12	1.49	5.78
CRNN 1D	1.67	1.04	1.26	1.01

System	Eval DCF(%)			
	Mel	**wav2vec**	**wavLM-Base+**	**wavLM-Large**
CRNN 2D	9.66	4.99	4.07	8.72
CRNN 1D	9.35	4.68	4.12	4.92

development partition, it can be seen that our systems based on Mel features already achieve competitive results, largely outperforming the organisation baseline. Best results on the development partition are obtained with our wav2vec approach, showing a relative improvement around 35% compared to Mel features, in the case of the CRNN 1D neural network. Most wavLM based systems also show an improvement compared to perceptual features. The only exception is the wavLM-Large model that underperforms our baseline for the CRNN 2D setup. This may due to the fact that, as features become more refined, they are more likely to show a lower correlation among their different dimensions. This fact could make it harder for 2D convolutions in the CRNN architecture to extract relevant information among the feature axis.

When observing results from the evaluation partition, a strong degradation in performance becomes apparent when comparing to development partition results. This degradation is more significant in our baseline system using Mel features, increasing DCF metric values from around 1.5% to 9.5%. As indicated by the organisation, data from new unseen Apollo missions was included in the evaluation partition, so this degradation may be motivated by the mismatched acoustic conditions between development and evaluation datasets. In general terms, self-supervised features allow to reduce the gap between development and evaluation results in this specific scenario. Best results are obtained using the wavLM Base+ model, with a relative improvement around 56% when compared to Mel features, and 13% when compared to wav2vec features. Furthermore, the use of this specific model can reduce the inference time and number of operations per second of the SAD neural network due to its bigger stride of 20 ms and its reduced feature dimension (1024 from wav2vec versus 768 wavLM Base+).

7.5 Conclusions

The experiments described in this chapter introduced ViVoLab submissions to two consecutive editions of the SAD task of the Fearless steps challenge, namely phase III from 2021 and phase IV from 2022. For both editions, the focus was set on testing the systems generalisation capabilities, with an evaluation dataset that included new channel data, and data from unseen Apollo missions respectively. Inspired by recent advances in self-supervised representation learning, and seeking to obtain new and more robust audio representations, our submissions explored different approaches to the self-supervised representation learning paradigm. For that we considered two different systems: in the edition of 2021, we followed an approach inspired on the wav2vec model that is based on a contrastive loss that learns a feature representation combining an strided CNN encoder with a RNN in order to provide a context embedding. Furthermore, for the 2022 edition we also evaluated recent advances in self-supervised learning by using wavLM models in the context of the SAD task. In both cases, obtained features are then used to train different variants of a CRNN architecture.

In both set of submissions considered for this chapter, experimental findings suggest that recent advances in self-supervised representation learning allow to build systems that are significantly less sensitive to domain mismatch when compared to a set of traditional perceptual features such as log Mel-filterbank energies. In the 2021 edition, our best submission achieved a DCF metric of 0.65% and 2.98% respectively in the development and evaluation sets, ranking third among the 6 participant teams in the SAD task. Concerning results from 2022 edition, compared to the wav2vec approach, best results using wavLM models report a 13% relative improvement on the evaluation partition in a mismatched domain scenario. Additionally, results obtained using wavLM models are also interesting due to its lower computational cost, showing a bigger stride of 20ms and a reduced dimensional space when compared to the wav2vec approach.

Part III

Multiclass audio segmentation

"Every time I fire a linguist, the performance of the speech recogniser goes up."

Frederick Jelinek

8

Recurrent neural networks for multiclass audio segmentation

8.1 Introduction and motivation

The previous part of this thesis dissertation has focused mainly on the SAD task seeking to detect speech segments in audio signals. In this part, a new kind of audio segmentation task designed from a multiclass perspective is introduced. While SAD deals only with an specific acoustic class and classifies the remaining information as not relevant, the audio segmentation task described in this chapter seeks to simultaneously

segment audio signals as being speech, music, noise or a combination of these. This task may be specially relevant from the perspective of audio information retrieval, combining in a single application several binary audio segmentation systems such as speech, music and noise detection. Furthermore, the detection of overlapping fragments where speech is combined with music, or speech is combined with noise, opens new possibilities to provide meaningful labels in the indexation of audio content.

Albayzín international evaluation campaigns have proposed a wide range of challenges from speech transcription to spoken term detection. The mentioned multiclass audio segmentation task was first introduced in these campaigns in 2010, within the context of broadcast news data [11]. Furthermore, more recent Albayzín challenge corpora was released in 2012, namely the CARTV dataset [131], with around 20 hours of audio from Aragón radio archive. In this chapter, we evaluate recent advances in deep learning audio segmentation systems in the tasks originally proposed in these challenges, with the goal of improving the results obtained in the original evaluation.

Work presented in this chapter describes an approach to the multiclass audio segmentation task based on the use of recurrent neural networks, namely the well-known LSTM networks. RNNs are combined with an HMM resegmentation module as a smoothing technique in order to correct possible errors due to high frequency transitions in the neural network output. A new block is introduced in the neural architecture seeking to remove redundant temporal information, reducing the number of operations per second without increasing the number of parameters in the model. Furthermore, we also show how these models can benefit from mixup augmentation, a data agnostic augmentation technique that does not add external data sources to the training dataset.

8.2 RNNs for multiclass audio segmentation

Several studies have proven the performance of LSTM networks in a binary segmentation task such as SAD [45] [46]. A preliminary study aimed to replicate these results in a multiclass segmentation environment, evaluating the results of BLSTM networks on the Albayzín 2010 audio segmentation evaluation with competitive results [202]. This chapter aims to explore the behaviour of recurrent neural architectures for this task in a wider sense, including improvements both in performance and in computational complexity to baseline RNN models. Our proposal to enhance the neural architecture is the incorporation of a new block with different variations that we refer to as "Combination & Pooling" block, that is described in the following subsections. These blocks seek to remove redundant temporal information through the use of temporal pooling.

Furthermore, as several works have shown the improvement in performance obtained when using data augmentation techniques in audio classification [203] [204], we aim to incorporate this kind of techniques in multiclass audio segmentation. Due to the data restriction imposed by the Albayzín 2010 evaluation, no further data can be used in training in order to obtain results that are comparable with other systems. In this context, we eval-

uate the use of mixup augmentation [205], which combines linearly pairs of examples to generate new virtual examples.

In conclusion, our proposal for this set of experiments consists of an RNN based segmentation by classification system. The presented approach combines the modelling capabilities of BLSTM networks with a resegmentation module to get smoothed segmentation hypothesis. In the following lines we describe the two main blocks the system comprises: an RNN based classifier and the final resegmentation module.

8.2.1 Recurrent neural network classifiers

The central idea of our proposed segmentation system is the use of RNNs as the classification algorithm in a segmentation by classification approach. We propose two different variations of this RNN classifier: the first one, that will be considered as our baseline system, is described in Figure 8.1. As shown, the neural architecture is mainly composed by one or more stacked BLSTM layers with 256 neurons each. The outputs of the last BLSTM layer are then independently classified by a linear perceptron sharing its values (weights and biases) among all time steps.

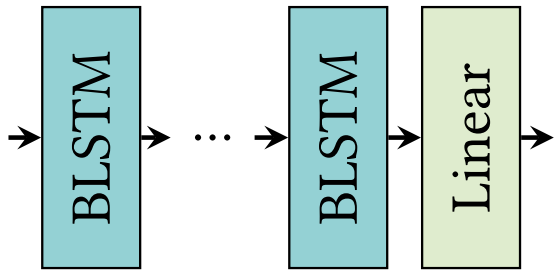


Figure 8.1: BLSTM based neural architecture description for the baseline RNN classifier.

The details of the proposed "Combination & Pooling" block are presented in Figure 8.2. The main idea behind it is to lessen the redundant temporal information through

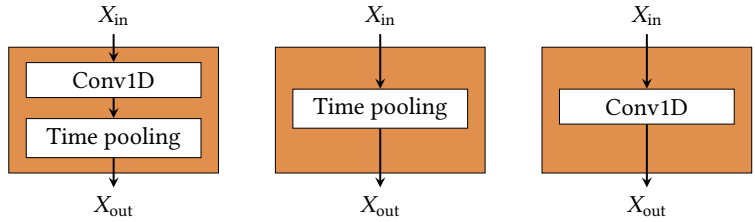


Figure 8.2: Description of the proposed "Combination and Pooling" block and its three variations in this study.

a time pooling mechanism while, at the same time, a more appropriate representation is learned through a 1D convolutional layer. Furthermore, we propose three different variations of this block. The first one combines both the temporal pooling and the 1D

convolutional layer, while the other two variations only use time pooling or a 1D convolutional layer respectively, to evaluate its separate influence on the system too.

In Figure 8.3 we present the second approach to the RNN classifier, incorporating our proposed "Combination & Pooling" block. The linear layer is configured in the same conditions as in the baseline system.

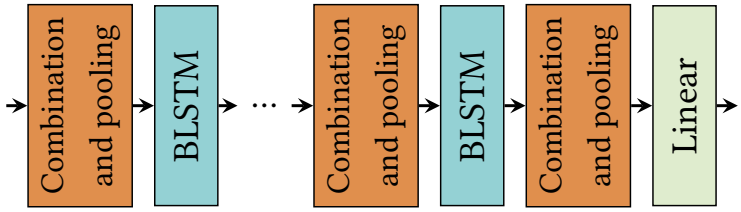


Figure 8.3: Alternative BLSTM based neural architectures including the proposed combination and pooling block.

All systems have been trained and evaluated using finite length sequences (3 seconds, 300 frames), limiting the delay of dependencies that the network may take into account. These sequences have a length of 3 seconds with an advance of 2.5 seconds, thus 0.5 seconds are overlapped. In order to generate the final prediction, the first half of this overlapped part is taken from the previous window, and the second half is taken from the next window. This way the labels corresponding to the boundaries of each fragment are discarded as they may not be reliable. However, the neural network emits a segmentation label for each frame processed at the input for the first system, and one segmentation label for each N frames processed when using the pooling setup, with N being the temporal pooling factor applied.

Neural networks are trained using Adam optimiser [105]. Cross entropy criterion is chosen as loss function, as usually done in multiclass classification tasks. All neural architectures in this work have been evaluated using the PyTorch toolkit [154]. In addition to the emitted RNN segmentation labels, we are also considering the final linear layer scores for each class in order to perform the resegmentation step. In our results we evaluate two different points in our system: the neural network output by using the RNN emitted labels and the final labels produced by the resegmentation module.

8.2.2 Resegmentation module

RNN outputs may contain high frequency transitions which are unlikely to occur in highly temporal correlated signals such as human speech or music. Aiming to avoid spurious changes in the segmentation hypothesis, we incorporate a resegmentation module in our system. Our implementation is based on an ergodic hidden Markov model where each class is modelled trough a state in the Markov chain. Every state is represented by a multivariate Gaussian distribution with full covariance matrix. No a priori information is required for this block to be fully functional because statistical distributions are estimated using the labels hypothesised by the RNN for each file in the database.

Similarly, neural network output may provide a noisy estimation of class boundaries. Seeking to avoid high frequency transitions, neural network scores are downsampled by a factor L using an L order averaging filter. This filter is implemented as a zero-phase FIR filter [206] to avoid delays in the output signal. Moreover, each state in the Markov chain consists of a left-to-right topology of N_{ts} tied states that share the same statistical distribution. These two strategies allow us to impose a certain amount of inertia in the output, forcing a minimum segment length before a class change occurs. This length can be computed as follows:

$$T_{\min} = T_s L N_{ts} \tag{8.1}$$

where T_s is the neural network output sampling period, L is the downsampling factor, and N_{ts} is the number of tied states per state in the Markov chain.

8.3 Experimental setup

The experimental setup for our experiments is based on the proposed originally in the Albayzín 2010 audio segmentation evaluation. A complete description of the task, results obtained by the participants and a description of the different approaches used can be found in [11]. In the following lines we describe the database used in the evaluation and the metrics taken into account in our experiments. We also present the CARTV database introduced in the Albayzín 2012 evaluation [131], and that is used in the final part of our experimentation to check the generalisation capabilities of our proposal.

8.3.1 Data description

Most experiments described in this chapter use the dataset released for the Albayzín 2010 audio segmentation task. As described in section 4.1.2, this dataset features around 87 hours of manually annotated audio sampled at 16 kHz. Two thirds of the database are reserved for training, making a total of 58 hours in 16 different sessions. The remaining third, 29 hours in 8 sessions, is used for test purposes. Furthermore, 15% of training subset is reserved for training validation, which translates to a total of 49 hours of audio for training and 9 hours for validation. The Albayzín 2010 database defines five different acoustic classes that need to be separated: clean speech (sp), music (mu), speech over music (sm), speech over noise (sn) and others (ot). The class others is not evaluated in the final scoring.

Additionally, data released in the Albayzín 2012 audio segmentation evaluation is used in the final part of our experimentation to check the generalisation capabilities of our proposal. This dataset is a collection of emissions from Aragón radio archive. Three separated subsets were released: two developments sets of 2 hours each (dev1 and dev2), and a test set consisting of 18 hours. The main difference comes from the fact that no prior classes are defined and a multiple layer labelling is proposed, allowing 3 possible overlapped classes, speech, music and noise, to be present at any time in the audio document. Even tough, this format is different to the setup presented for the Albayzín 2010

evaluation, an equivalent version can be obtained if a set of non-overlapped labels are generated considering the different combinations of speech, music and noise. A more detailed explanation of this process is given in Section 8.4.6.

8.3.2 Feature extraction

Concerning feature extraction, the neural network is fed with a set of features that combines perceptual features with some musical theory motivated features. First, log Mel filter-bank energies and the log energy of each frame are extracted. Then, Mel features are combined with chroma features [85], extracted using the openSMILE toolkit [207]. Audio is initially resampled at 16 kHz and converted to a single channel input. All features are computed every 10 ms using a 25 ms Hamming window. In the experiments that consider feature derivatives, First and second order derivatives of the features are computed using 2 FIR filters of order 8 to take into account the dynamic information in the audio signal. Finally, feature mean and variance normalisation are applied at recording level.

8.3.3 Evaluation metrics

The main metric used to evaluate our results is the segmentation error rate (SER), that can be seen as the ratio of the total length of the incorrectly classified audio to the total length of the audio in the reference. Additionally, the original metric proposed in the Albayzín 2010 evaluation is considered in our experiments in order to favour the comparison with previous publications. This metric represents the average class error over all the classes. Let C be the set of the five acoustic classes defined in the evaluation, $C = \{\text{mu}, \text{sp}, \text{sm}, \text{sn}, \text{ot}\}$. This way the error metric can be computed according to the following equation:

$$\text{Avg error} = \frac{1}{|C|} \sum_{i \in C} \frac{\text{dur}(\text{miss}_i) + \text{dur}(\text{fa}_i)}{\text{dur}(\text{ref}_i)} \tag{8.2}$$

where $\text{dur}(\text{miss}_i)$ is the total duration of all miss errors for the ith acoustic class, $\text{dur}(\text{fa}_i)$ is the total duration of all false alarm errors for the ith acoustic class, and $\text{dur}(\text{ref}_i)$ is the total duration of the ith acoustic class according to the reference. Using this metric an incorrectly classified segment computes as a miss error for an acoustic class and a false alarm error for another. Due to the fact that class distribution is clearly unbalanced, errors from different acoustic classes are weighted differently according to the total duration of the class in the database. This metric was originally proposed in the evaluation because, by computing the average of the error over all the acoustic classes, participants were encouraged not to focus only on the best represented classes in the database. In both metrics, SER and average class error, a collar of ± 1 second around each reference boundary is not scored in order to avoid uncertainty about when an acoustic class ends or begins, and to consider inconsistent human annotations.

Furthermore, in the final discussion, other metrics traditionally shown for classification tasks such as the overall accuracy, and the precision, recall and F_1 score per class are reported.

8.4 Results

8.4.1 Feature analysis

Two different kind of errors can be differentiated in a segmentation system: a classification error due to an incorrectly labelled frame, and a segmentation error due to a temporal mismatch between the hypothesis and reference class boundaries. In a first set of experiments only classification errors are considered by providing ground truth segments to the system. In this case, classification labels are obtained by choosing the class maximising the neural network score averaged for the whole ground truth segment. Then, the classification error is computed simply dividing the number of oracle segments incorrectly classified by the total number of oracle segments. In the following sets of experiments both segmentation and classification errors are shown as no boundaries are given to the system.

In order to validate experimentally our proposal, different frontend configurations were assessed. Our setup started with 64 bands log Mel filter-bank energies. Then, frequency resolution was gradually increased, evaluating an RNN classifier with 80 and 96 bands log Mel filter-bank energies as input. As explained before, chroma features were

Table 8.1: Classification error with oracle boundaries for the 1BLSTM RNN classifier on the test partition for different frontend configurations (Δ,ΔΔ: 1st and 2nd order derivatives).

Input features	**Classification Error (%)**	**# Parameters**
64 bands	16.22	200K
80 bands	16.20	217K
96 bands	**16.05**	**233K**
64 bands + chroma	**13.40**	**213K**
80 bands + chroma	13.80	229K
96 bands + chroma	14.51	246K
64 bands + chroma + Δ,ΔΔ	13.35	368K
80 bands + chroma + Δ,ΔΔ	**13.34**	**418K**
96 bands + chroma + Δ,ΔΔ	13.37	467K

also incorporated aiming to discriminate correctly the music classes. Eventually, first and second order derivatives were computed to include information about the audio signal dynamics. All these cases use a simple setup consisting of an RNN classifier based on a single BLSTM layer. Table 8.1 shows the classification error obtained with the RNN classifier using oracle boundaries for different frontend configurations. It can be seen that

increasing frequency resolution leads to a consistent improvement in the classification accuracy while the number of parameters of the model is also increased. However, when incorporating chroma features the improvement obtained is significantly better than the one obtained with a bigger frequency resolution. If we compare the best log Mel filter-bank setup (96 bands) with the best one with chroma (80 bands + chroma) it can be seen that, with a similar number of parameters, the error drops significantly. Finally, first and second order derivatives are computed, achieving the best result in our experiment at the cost of increasing the number of parameters in the model.

Additionally, Table 8.2 presents the segmentation error rate and the error per class obtained with the RNN classifier. These results now consider both classification and segmentation error. In this case, the best configuration for log Mel filter-bank energies is

Table 8.2: SER, error per class and average class error for the 1BLSTM RNN classifier on the test partition for different frontend configurations (Δ,ΔΔ: 1st and 2nd order derivatives).

Input features	**SER**	**Class Error(%)**				**Avg**
		mu	**sp**	**sm**	**sn**	
64 bands	18.18	18.54	32.43	32.48	35.76	29.80
80 bands	**17.70**	**18.19**	**31.33**	**31.41**	**34.91**	**28.96**
96 bands	17.93	20.68	30.84	32.09	34.25	29.46
64 bands + chroma	**16.97**	**18.83**	**30.88**	**29.92**	**32.76**	**28.10**
80 bands + chroma	17.89	19.77	32.23	29.55	33.92	28.87
96 bands + chroma	17.65	19.75	30.68	31.62	33.66	28.93
64 bands + chroma + Δ,ΔΔ	16.61	17.46	29.93	29.26	32.60	27.31
80 bands + chroma + Δ,ΔΔ	**16.25**	**16.82**	**30.00**	**26.75**	**32.07**	**26.41**
96 bands + chroma + Δ,ΔΔ	16.46	17.38	29.92	27.98	32.70	27.00

achieved using 80 bands. Increasing the frequency resolution seems not to be so relevant when dealing with the segmentation task compared to the classification one. We can notice that, by incorporating chroma features, the error in the class speech over music and speech over noise decreases significantly. For example, when comparing the 64 bands setup to the same one with chroma, a relative improvement of 8.55% and 9.15% can be observed respectively. This is due to the capabilities of chroma features to capture musical information, which helps the system discriminate noise and music in a more accurate way. This behaviour is also consistent with the classification accuracy improvement observed when using chroma features in the ground truth boundary experiments. The best result is obtained with the frontend that combines 80 bands, chroma features and the first and second order derivatives with a SER metric of 16.25%, equivalent to an average class error of 26.41%. Furthermore, it can be observed that including first and second order derivatives shows a greater relative improvement when considering the segmentation errors compared to the case where only classification errors are considered. We can infer then, that dynamic information incorporated by the 1st and 2nd order derivatives may be more relevant to generate class boundaries than to the classification task itself.

So far, only an architecture with a single BLSTM layer has been evaluated. In the following experiment, the goal is to determine the most appropriate number of BLSTM layers for the RNN classifier. Choosing the best feature frontend (80 Mel + chroma + 1st and 2nd derivatives), we evaluate now our system stacking two and three BLSTM layers. Results for this experiment are presented in Table 8.3.

Table 8.3: SER, error per class and average class error for the RNN classifier on the test partition for the best frontend configurations and different number of stacked BLSTM layers.

# Layers	SER	Class Error(%)				Avg
		mu	sp	sm	sn	
1 BLSTM	16.25	16.82	30.00	26.75	32.07	26.41
2 BLSTM	**15.91**	**16.28**	**28.82**	**26.32**	**31.94**	**25.84**
3 BLSTM	16.02	15.71	27.46	30.09	30.74	26.00

Including 2 stacked BLSTM layers shows a slight relative improvement of around 2.20% compared to the case of using a single BLSTM layer. However, no further improvement is observed when including a third layer. That is why we choose the architecture using 2 BLSTM stacked layers as our baseline in futures experiments. An average class error of 25.84% is obtained, equivalent to a SER of 15.91%. These results are the one used to compare against in the following sections where the different neural architectures proposed are evaluated. Furthermore, following sections also evaluate the performance of the full system combining the RNN classifier and the HMM resegmentation module with a new set of experiments.

8.4.2 HMM resegmentation

With the objective of illustrating the influence of the inertia imposed by the resegmentation module on the segmentation system, Figure 8.4 shows the scatter plot of the relative improvement in performance using the HMM resegmentation versus the minimum segment length (T_{min}) for different values of the downsampling factor. It can be seen that the best performing configurations have a minimum segment length between 0.5 and 1.5 seconds, values which are in the order of magnitude of the 2 seconds collar applied in the evaluation. A fast decrease in performance is observed when T_{min} is increased for values above 3 seconds. With such configuration our system is not able to capture some of the fast transitions happening in the audio, thus a considerable amount of errors are likely to happen, and the performance is decreased. However, no configuration showed a decrease in performance when compared to the case of not using the HMM resegmentation.

In Table 8.4, we show the results on the test partition for the full segmentation system that combines the RNN classifier and the HMM resegmentation for the best feature configuration and different values of downsampling factor, L, and minimum segment length, T_{min}. Compared to the best results in Table 8.2, it can clearly be seen that HMM resegmentation reduces significantly the system error by forcing a minimum segment length for class labels. This error reduction is equivalent to a 21.68% relative improvement in

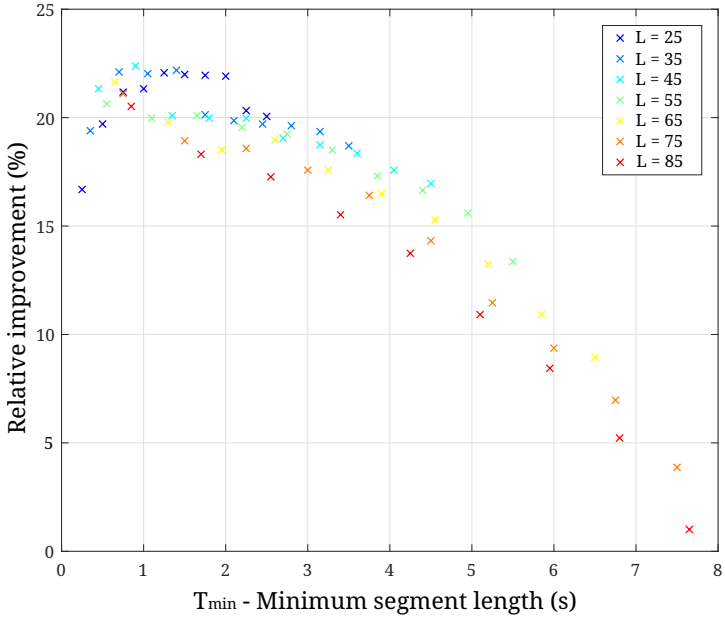


Figure 8.4: Relative improvement over the RNN classifier using the HMM resegmentation module for the best feature configuration versus minimum segment length forced by the system.

terms of SER for the best configuration. Again, it can be observed that, as long as the T_{min} value stays in the range between 0.5 and 1.5 seconds, the performance of the system is not highly affected by the variations in the downsampling factor. SER metric in the four parameters configuration presented in Table 8.4 varies from 12.46% to 12.57%, an absolute difference of only 0.11% between the best and the worst case. This way, reducing the high frequency transitions through imposing a certain amount of inertia in the neural network output, our segmentation system achieves a SER of 12.46%. This value serves also for comparison in the following lines, where new neural architectures are evaluated.

Table 8.4: SER, error per class and average class error for the RNN classifier combined with the resegmentation module over test partition for the best feature configuration and different values of the downsampling factor, L, and minimum segment length, T_{min}.

L , T_{min}	SER	Class Error(%)				Avg
		mu	sp	sm	sn	
25, 1.25s	12.49	14.55	21.99	19.08	24.88	20.13
35, 1.4s	12.48	14.31	22.26	18.70	25.10	20.10
45, 0.9s	**12.46**	**14.19**	**22.14**	**18.82**	**25.04**	**20.05**
55, 0.55s	12.57	16.12	22.00	18.94	24.95	20.50

8.4.3 Combination and pooling experiments

Our initial experiments using the HMM resegmentation module proved that reducing the temporal resolution of the output is beneficial for the segmentation system. Our goal introducing the "Combination and pooling" block is that this downsampling could be implemented inside the neural network itself. Temporal pooling layers are configurable via a pooling factor parameter, N, that controls the length of the output sequences compared to the input length. Pooling layers separate an input sequence into N different subsequences with the same length and no overlapping. Then, the output is computed applying a given pooling mechanism for each of these subsequences. In all cases, a single element is returned for every N frames in the input. On the other hand, the 1D convolutional layer is configured to have the same number of input and output channels in all cases, a kernel size of 1 and no padding.

In Table 8.5, we present the results obtained when using the "Combination and Pooling" block in all its variations before the first BLSTM layer. For this experiment we consider an average pooling mechanism with a pooling factor of N=10. Experimental results

Table 8.5: Average class error and relative improvement over the baseline system for the RNN classifier with the combination and pooling block before the first BLSTM layer on the test partition.

Model configuration	Avg Class Error(%)	Rel. improvement(%)
$\text{ConvPoolBLSTM}_{1,2}$	29.30	-11.80
$\text{ConvBLSTM}_{1,2}$	**26.26**	**-1.60**
$\text{PoolBLSTM}_{1,2}$	28.31	-8.72

show that using temporal pooling before the first BLSTM layer strongly degrades the performance of the segmentation system. The degradation is even stronger when first combining the input features using a 1D convolutional layer with a relative degradation of 11.80% compared to the baseline RNN classifier. This may come motivated by an early reduction of the input dimensionality, before the neural network has been able to process any kind of information. By observing these results, we can discard this type of configurations in future experiments.

In the following lines we present the results for the other two remaining configurations implemented: the one using the "Combination and Pooling" block between the first and second BLSTM layers and the one with the block right after the last BLSTM layer. Figure 8.5 shows the relative improvement compared to the RNN baseline classifier for the setup using the combination and pooling block between both BLSTM layers and the setup using the combination and pooling block after the last BLSTM layer. In these architectures we have experimented with three different values for the pooling factor: 5,10 and 25, and an average pooling mechanism. We also present in green straight lines the architectures using only a 1D conlutional layer, with an independent behaviour of the pooling factor. Concerning the convolutional only architectures (green lines), it can be observed that recombining internal BLSTM representations does not show a significant improvement in performance, with the $\text{BLSTM}_1\text{ConvBLSTM}_2$ system really close to the

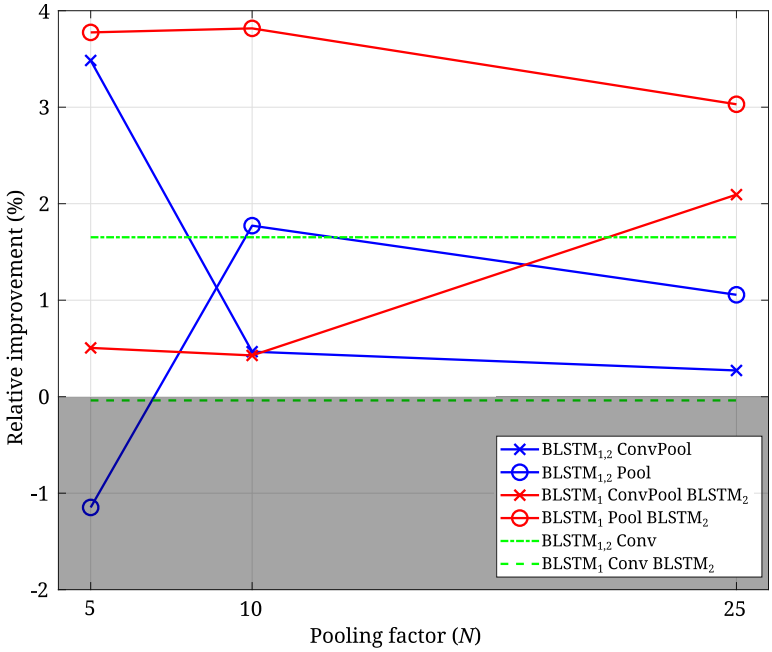


Figure 8.5: Relative improvement over the baseline RNN classifier for the setup using the combination and pool block between both BLSTM layers and the setup using the combination and pooling block after the last BLSTM layer.

baseline classifier and the $BLSTM_{1,2}Conv$ showing a relative improvement of around 1.8%. The best results are obtained consistently among three evaluated pooling factors for the $BLSTM_1PoolBLSTM_2$ configuration (red circles), where a temporal pooling layer is used in between the first and the second BLSTM layers. The best case achieves a relative improvement of 3.8% compared to the baseline system. Furthermore, this new approach does not add more parameters to the model and it decreases computational complexity because the second BLSTM layer is working at a smaller sampling rate.

Being proved that the best performing setup is the one using only a pooling layer in between the first and second BLSTM layer, in the following experiment a deeper analysis of this neural architecture is performed. Two different pooling mechanisms are considered now: average pooling and max pooling. Furthermore, a wider variation range of the pooling factor is tested, ranging from 10 to 100. Figure 8.6 presents the results for all the evaluated configurations in terms of the relative improvement obtained when compared to the baseline RNN classifier output using the best feature configuration (80 bands + chroma + derivatives). Two differentiated behaviours can be observed: the average pooling configurations (blue line) show a general improvement between 3% and 4% without a strong dependence on the pooling factor. However, max pooling (red line) degrades its performance significantly when increasing the pooling factor, even showing worse results than the baseline for pooling factors greater than 25. Bearing this results in mind, only the average pooling configurations are taken into account in the following experiments.

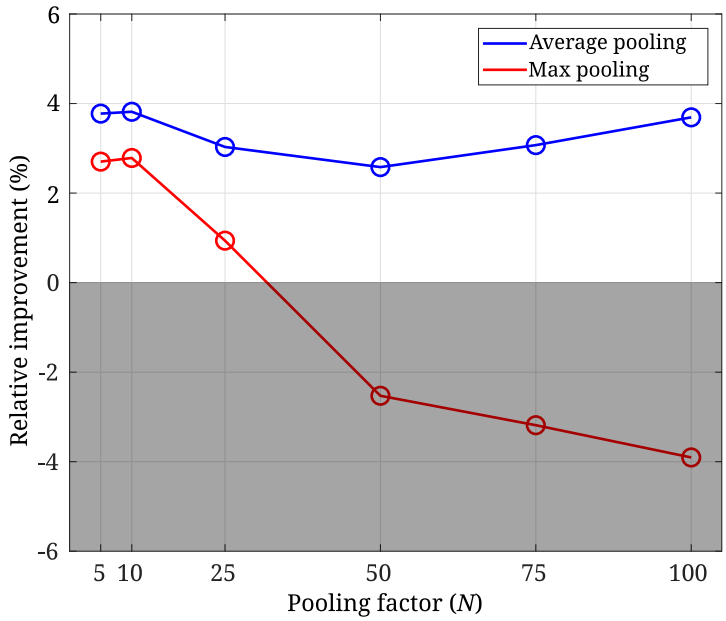


Figure 8.6: Relative improvement over the baseline RNN classifier for different pooling factors and pooling techniques using the $BLSTM_1PoolBLSTM_2$ architecture.

Table 8.6 describes the detailed results for the average pooling setup experiments in terms of SER, error per class and average class error. The best result is obtained for the

Table 8.6: SER, error per class and average class error for the $BLSTM_1PoolBLSTM_2$ RNN classifier on the test partition for different pooling factors (N) and average pooling.

N	SER	Class Error(%)				Avg
		mu	sp	sm	sn	
5	15.49	15.57	28.71	24.63	30.71	24.90
10	**15.47**	**15.55**	**29.16**	**24.34**	**30.51**	**24.89**
25	15.51	16.85	27.17	26.23	30.06	25.08
50	15.53	16.40	28.60	24.92	30.85	25.19
75	15.54	16.87	27.97	25.55	29.88	25.07
100	15.49	18.22	26.77	24.91	29.80	24.92
No pool	15.91	16.28	28.82	26.32	31.94	25.84

pooling factor 10, immediately followed by the configurations of 5 and 100. These results show a relative improvement of 3.82%, 3.77% and 3.69% respectively, without increasing the number of parameters in the neural network and reducing the computational load of our system because the second BLSTM layer is working at a smaller sampling rate.

Finally, results of the $BLSTM_1PoolBLSTM_2$ system combined with the HMM resegmentation module are shown in Table 8.7. Compared to the RNN baseline system using

the HMM module, no significant improvement is observed, with the SER decreasing from 20.05% to 19.90% in the N=10 setup. A performance degradation is even observed for bigger pooling factor setup. This could be motivated by the fact that the pooling layer has already performed part of the smoothing that the HMM enforced in the RNN baseline architecture, so the combination of both pooling layers and HMM module could not lead to a significant improvement. However, this architecture is interesting because this way computational load of the HMM module can be decreased, working at a sampling rate ten times smaller.

Table 8.7: SER, error per class and average class error for the $BLSTM_1PoolBLSTM_2$ RNN classifier combined with the HHM resegmentation on the test partition for different pooling factors (N) and average pooling.

N	**SER**	**Class Error(%)**				**Avg**
		mu	**sp**	**sm**	**sn**	
5	12.48	14.31	22.16	18.69	25.12	20.07
10	**12.41**	**12.87**	**22.58**	**19.16**	**24.99**	**19.90**
100	13.47	22.85	24.32	19.31	26.20	23.17
No pool	12.46	14.19	22.14	18.52	25.04	20.05

8.4.4 Mixup data augmentation

Mixup is a data-agnostic data augmentation routine [205] that generates new virtual training examples. These virtual examples are generated according to the following equations:

$$\begin{cases} \tilde{\mathbf{X}} = \lambda \mathbf{X_i} + (1-\lambda)\mathbf{X_j}\,, \\ \tilde{\mathbf{Y}} = \lambda \mathbf{Y_i} + (1-\lambda)\mathbf{Y_j}\,, \end{cases} \tag{8.3}$$

where $(\mathbf{X_i}, \mathbf{X_j})$ are two feature vectors randomly drawn from the training dataset and $(\mathbf{Y_i}, \mathbf{Y_j})$ are their corresponding one hot encoding labels, and $\lambda \in [0, 1]$. In the practical implementation $\lambda \sim Beta(\alpha, \alpha)$, with α being the mixup hyperparameter that controls the strength of the interpolation for the pairs of examples. Furthermore, this technique is simple to implement and it does not require a high computational overhead. The use of mixup augmentation leads to no addition of more external datasets, so the proposed system is still under the conditions imposed by the Albayzín 2010 evaluation. Mixup augmentation has been shown to improve model generalisation capabilities in different domains, including some audio classification tasks [208]. In our set of experiments, mixup augmentation is applied directly in the feature space.

Table 8.8 shows the results obtained training the $BLSTM_1PoolBLSTM_2$ architecture using mixup data augmentation for different α values compared to the same system trained without mixup augmentation. It can be seen that the result is not highly dependent on the α hyperparameter. Best result is obtained for α=0.2, however all the evaluated configurations show similar results. In general terms, mixup augmentation is able to achieve a relative improvement of 5% compared to a system not trained using

Table 8.8: SER, error per class and average class error on the test partition for the $BLSTM_1PoolBLSTM_2$ RNN classifier (Avg pooling, $N = 10$) trained using mixup augmentation with hyperparameter α.

Mixup	SER	Class Error(%)				Avg
		mu	sp	sm	sn	
$\alpha = 0.1$	14.84	15.21	27.99	23.05	29.34	23.90
$\alpha = 0.2$	**14.80**	**14.64**	**28.20**	**22.01**	**29.01**	**23.56**
$\alpha = 0.3$	14.81	16.03	26.32	23.89	28.22	23.62
No mixup	15.47	15.55	29.16	24.34	30.51	24.89

mixup. The most significant improvement observed when using mixup augmentation is in the speech over music class, with an absolute improvement of 2.23%. This class is the one defined as a pure combination of other two classes in the dataset. This fact shows that generating new virtual examples as a linear combination in the feature domain can be beneficial for our segmentation system.

As our final experiment, Table 8.9 presents the results obtained with the full segmentation system that combines the $BLSTM_1PoolBLSTM_2$ RNN classifier trained with mixup data augmentation and the HMM resegmentation module. With this setup, we achieve

Table 8.9: SER, error per class and average class error on the test partition for the $BLSTM_1PoolBLSTM_2$ RNN classifier (Avg pooling, $N = 10$) trained with mixup augmentation with hyperparameter α combined with the HHM resegmentation.

Mixup	SER	Class Error(%)				Avg
		mu	sp	sm	sn	
$\alpha = 0.1$	12.88	14.39	23.87	18.68	25.70	20.66
$\alpha = 0.2$	**11.80**	**12.46**	**22.86**	**17.34**	**24.35**	**19.25**
$\alpha = 0.3$	12.14	14.80	21.71	17.37	23.54	19.36
No mixup	12.41	12.87	22.58	19.16	24.99	19.90

the best performing segmentation system in this work, which is equivalent to a SER of 11.80% and an average class error of 19.25%.

8.4.5 Discussion

Once our different system proposals have been experimentally evaluated, in this section we aim to compare our results with the ones obtained previously in the literature and perform an analysis on the segmentation system errors. Figure 8.7 shows the results obtained in the Albayzín 2010 test partition by different systems already presented in the literature. The winner team of the original Albayzín 2010 evaluation proposed a segmentation by classification approach based on a hierarchical GMM/HMM (dark blue) including MFCCs, chroma and spectral entropy as input features [209]. The best result so far in this database was obtained with a solution based on factor analysis combined

with a Gaussian backend (orange) and MFCCs with 1st and 2nd order derivatives as input features [38]. Our three previously explained final results combining the RNN classifier and the HMM resegmentation are also presented: the RNN baseline (light blue), the $BLSTM_1PoolBLSTM_2$ RNN approach (light green) and the $BLSTM_1PoolBLSTM_2$ RNN trained using mixup augmentation (lighter blue).

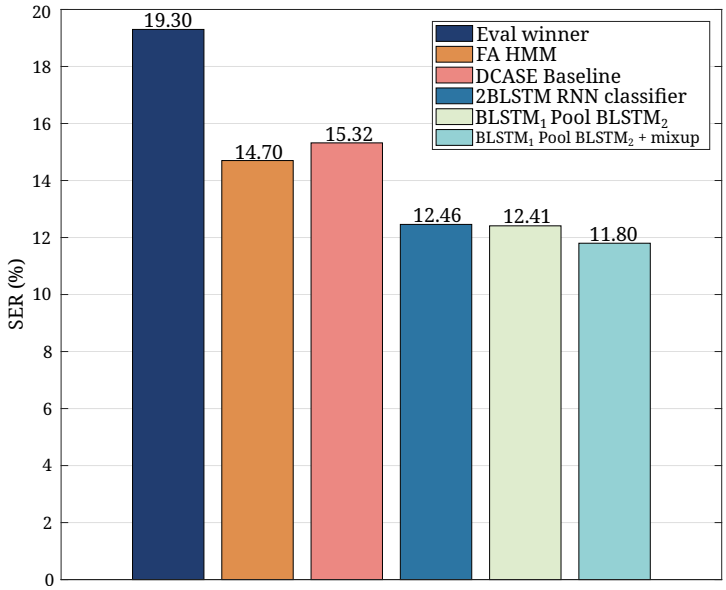


Figure 8.7: Results obtained on the Albayzín 2010 test partition for different systems proposed in the literature compared to our proposed RNN approaches in terms of SER.

Additionally, in order to compare our results with a DNN based system, we trained and evaluated a different system using the neural architecture proposed as baseline in the DCASE challenge for sound environment detection [210], a task similar to the one presented in this chapter. This approach is based on 3 2D CNN layers with 64 channels each followed by a single GRU cell with 64 hidden units. The input features are 64 dimensional log Mel filter-bank energies. It can be seen that our three systems outperform previous results in this database, with our RNN combined with the pooling setup and trained with mixup augmentation achieving a relative improvement of 19.72% in terms of SER compared to the FA HMM approach. Furthermore, if the comparison is made with the DCASE baseline neural architecture, a DNN based system, a 22.97% relative improvement is obtained with our best system.

Results presented in Figure 8.7 are complemented with the ones shown in Table 8.10, that introduces the average class error and the error per class obtained for the same systems presented before. A general improvement over all the classes can be observed comparing to the previous approaches to this task. It is specially significant the error difference obtained in the classes that contain music (absolute decrease of 6.34% for music and 6.26% for speech over music comparing our best system to the FA HMM system). On the other hand, the class speech obtains really similar results. This difference in perform-

Table 8.10: Average class error and error per class obtained on the Albayzín 2010 test partition for different systems proposed in the literature compared to our proposed RNN approaches.

System	Class Error(%)				Avg
	mu	sp	sm	sn	
Eval winner [209]	19.20	39.50	25.00	37.20	30.30
FA HMM [38]	18.80	23.70	23.60	29.10	23.80
DCASE Baseline	19.03	25.58	23.59	29.52	25.18
RNN baseline	14.19	22.14	18.82	25.04	20.05
RNN+Pool	12.87	22.58	19.16	24.99	19.90
RNN+Pool+mixup	**12.46**	**22.86**	**17.34**	**24.35**	**19.25**

ance may be motivated by the introduction of chroma features, helping the adequately representation of music, and the linear combination of classes in training done using mixup augmentation. If the comparison is made with another DNN based approach like the proposed DCASE baseline, again we can observe a general improvement in performance over all the classes evaluated.

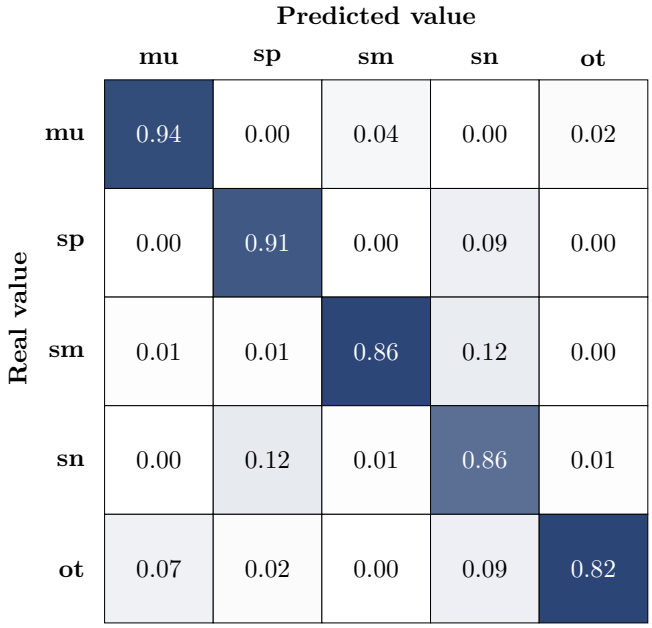


Figure 8.8: Confusion matrix for the best parameter configuration evaluated in this chapter: $\text{BLSTM}_1\text{PoolBLSTM}_2$ RNN ($N = 10$) with mixup (α=0.2) combined with the HMM resegmentation.

As a different performance measure, Figure 8.8 shows the confusion matrix for the best system presented in this work. It can be seen that one of the highest error terms is obtained for the frames predicted as speech over noise that are labelled as speech over music, happening in the 12% of the actual speech over music frames. Something similar

happens with 12% of the speech over noise frames incorrectly classified as speech. The class music obtains the best classification results despite being significantly underrepresented in the database (only 5% of total). As it was observed when comparing with the other systems in the literature, this fact may come motivated by the use of chroma features, capturing adequately the musical structures and helping discriminate correctly music. The worst classification results are given for the others class, not taken into account for scoring. This class, like music, is also heavily underrepresented in the database (3% of total) but, in this case, it comprises any other signal outside the definition of the other 4 classes what results in a unspecific definition that makes the classification harder.

Finally, we report the results of our best performing system in some of the traditional classifications metrics shown in the literature. Table 8.11 shows the overall accuracy and the precision, recall and F1 score, both per class and averaged, at frame level for the Albayzín 2010 test data evaluated using our $\text{BLSTM}_1\text{PoolBLSTM}_2$ proposal trained using mixup augmentation. Our system achieves an overall accuracy of 85%, with a balanced average result in precision and recall.

Table 8.11: Overall accuracy, precision, recall and F_1 score per class and on average for the Albayzín 2010 test data evaluated using the best performing system presented in this chapter.

Class	Precision	Recall	F_1
Music	0.88	0.89	0.88
Speech and music	0.93	0.84	0.89
Speech and noise	0.85	0.84	0.84
Speech	0.83	0.88	0.85
Average	0.87	0.86	0.87
		Accuracy	**0.85**

8.4.6 Evaluation on Albayzín 2012 dataset

In previous sections, we have analysed the results achieved with the proposed segmentation system on the Albayzín 2010 test data and we have proven the performance of our method compared to previous results in the literature. In order to evaluate the generalisation capabilities of our proposal, in this section we aim to evaluate it on a different dataset, namely CARTV dataset, proposed in the Albayzín 2012 evaluation. This dataset differs from the data presented in the 2010 version: the overlap of three different classes is allowed (speech, music and noise). In order to match our multiclass classification framework, this format needs to be converted to non-overlapping classes, obtaining 8 different classes: speech, music, noise, speech and music, speech and noise, speech and music and noise, music and noise, and silence. Due to the similarity with the class defined as others in Albayzín 2010 evaluation, the decision was made to combine music and noise, noise and silence (they represent only the 3% of the total time in the database) in a single class that we also name as others. Therefore, we can see this problem in a similar way to the

task in Albayzín 2010 dataset, but including a new class, speech and music and noise, that was not present in the 2010 version of the evaluation.

The evaluation of our system trained using Albayzín 2010 data on Albayzín 2012 test data would imply the consistent loss of the speech and music and noise class, that is not present in the data seen by the neural network. The change of domain from television to radio data could also affect the results obtained. Furthermore, the low amount of data available from CARTV dataset in development subsets (dev1 and dev2 contain only 4 hours of audio) suggests that training a new neural network from scratch may not be the most suitable solution.

Considering all these statements, we opted for a solution that adapts our best performing model trained on the Albayzín 2010 data to the radio domain using the 4 hours of development data available from the CARTV dataset. This adaptation process is described in the following lines:

- The pretrained model on Albayzín 2010 data is taken as the training starting point of the neural network. The final classification layer is removed and then a new one with 6 output neurons is randomly initialised.
- The whole neural network (BLSTM layers and final classification layer) is fine tuned with CARTV dev1 and dev2 data using the same strategies presented in the previous sections (temporal pooling and mixup augmentation). Learning rate used in the BLSTM layers is ten times smaller than the learning rate used for the final classification layer.

HMM resegmentation is used in the same way as described previously in this chapter. Aiming to compare with a different DNN based architecture, a similar approach is done in the DCASE baseline architecture, removing the last linear layer and randomly initialising a new one with 6 neurons, to then fine tune the neural network with development data from CARTV dataset. Table 8.12 presents the results on the Albayzín 2012 test data for

Table 8.12: SER on the Albayzín 2012 test partition for different systems proposed in the literature compared to our proposed RNN approach.

System	SER
RNN proposal (pool + mixup)	**24.93**
DCASE baseline	31.21
GMM + Viterbi decoding [211]	26.34
HMM-GMM [212]	26.53

different systems compared to our proposed RNN based approach combined with the HMM resegmentation module in terms of SER. In addition to our proposal and the DCASE baseline architecture, we show the results of the two best performing systems in the original Albayzín 2012 evaluation. The first system presented [211] is based on the use of GMM models with 1024 components to describe each of the possible combinations

of acoustic classes. Then a Viterbi decoding is performed to obtain the segmentation labels. Input features are MFCCs and first and second order derivatives. The second system presented [212] applied an HMM-GMM speech recognition approach in which the vocabulary set is defined by the possible acoustic classes. Input is based on MFCC features, considering first and second order derivatives too. It can be observed that if we compare the result obtained with a different DNN approach such as the DCASE baseline architecture, our systems achieves a relative improvement of 20.12% in terms of SER. This improvement is in the same order of magnitude as the improvement observed in the 2010 evaluation data, reflecting a consistent behaviour for our proposed neural architecture. If the focus is set on the results achieved in the original Albayzín 2012 evaluation, a 5.35% relative improvement can be observed compared to the evaluation winner. This improvement is significantly smaller than the improvement achieved in the Albayzín 2010 evaluation. This fact may come motivated by the small amount of in-domain data released for the 2012 evaluation. Our DNN approach would be able to profit from a bigger amount of data, whereas more traditional approaches such as GMM-HMM can achieve competitive results with less data.

8.5 Conclusions

Work presented in this chapter has explored several architectures of RNN based classifiers for the multiclass audio segmentation task. Namely, this set of experiments focuses on the audio segmentation tasks proposed for the Albayzín 2010 and 2012 evaluation, where the goal was to separate simultaneously speech, music, noise and the possible combinations of these three classes. Our proposal, that is based on a segmentation by classification approach, combines the BLSTM modelling capabilities with an HMM backend to smooth the results. Different frontends have been evaluated, proving how useful chroma features can be when representing music, and the importance of feature derivatives in capturing class boundaries. Furthermore, the combination of BLSTM and HMM was proved to be appropriate, reducing significantly the system error by forcing a minimum segment length for the segmentation labels.

This chapter proposes the introduction of a "Combination & Pooling" block in the neural architecture in several configurations. Through a detailed set of experiments, we showed that time pooling architectures can be used between two BLSTM layers to get a subsampled output, removing temporal redundant information and achieving a relative improvement of around 5% in the neural network output. Furthermore, through the introduction of mixup data augmentation, a data-agnostic data augmentation technique, another 5% relative improvement on the neural network can be observed. This improvement was specially significant for classes that can be described as the combination of two classes. This technique allows to remain under the Albayzín 2010 evaluation conditions as no additional datasets were included in the augmentation process.

Competitive results have been obtained with our RNN based approach, resulting in a relative improvement of 19.72% and 5.35% respectively compared to the best result in the literature so far for the Albayzín 2010 and 2012 evaluations. The improvement is

more noticeable for the Albayzín 2010 evaluation, Results are also compared with a DNN based system proposed for the DCASE sound environment detection task, achieving a consistent relative improvement of around 20% in the two datasets evaluated.

"The key to artificial intelligence has always been the representation."

Jeff Hawkins

9

Wav2vec representation learning for multiclass audio segmentation

9.1 Introduction and motivation

Inspired by recent advances in self-supervised representation learning, several works have considered its use among different audio and speech processing tasks, with audio segmentation being one of them. Experimental results from Chapter 7 showed the potential of self-supervised learning in audio segmentation, with significant improvements in performance under mismatched test conditions for the SAD task. In general terms, most works applying self-supervised learning to audio segmentation are limited to speech related tasks [72] [73], with few systems using them in more general audio segmentation tasks.

Deeming this issue as main motivation, the goal of this chapter is to explore representation learning techniques in the context of multiclass audio segmentation. An analysis of different ways to benefit from these abstract representation in more general segmentation problems is performed, focusing on remarking its possible limitations when representing different acoustic classes simultaneously.

Multiclass audio segmentation experiments described in Chapter 8 already demonstrated the capabilities of deep neural networks in the Albayzín 2010 dataset, seeking to separate speech, music, noise or a combination of them. The work presented in this chapter builds on the knowledge acquired from these previous experiences, aiming to introduce the recent advances in self-supervised representation learning in the multiclass audio segmentation tasks. While all previous experiments relied on the use of a combination of different sets of features, namely perceptual features such as log Mel filter-bank energies, or musical features such as chroma, the approach followed in this chapter aims to replace these features with representations obtained through self-supervised learning. Namely, an approach inspired by wav2vec models is evaluated as feature extractor under different training conditions and using different models as classifiers.

9.2 Self-supervised representation learning

9.2.1 wav2vec approach

The same approach presented in Section 7.2.1 for self-supervised representation learning is adopter in this chapter. As explained in Chapter 7, this solution, strongly inspired by wav2vec models presented in [110], combines a CNN encoder mapping the input signal to a latent space and a bidirectional GRU recurrent neural network providing a context embedding. That embedding is extracted in inference time to be used as an audio representation. The main difference with the system presented in Chapter 7 comes from the audio input. In this case, the CNN encoder handles 16 KHz audio signals, so the total downsampling factor of the network is 160 to enforce a feature sequence with 10 ms stride.

9.2.2 Data augmentation

Firstly proposed self-supervised learning models were meant to be applied in task such as ASR, where background noise or music needs to be compensated to focus on the uttered speech. Several works tried to compensate that variability in traditional ASR systems by applying data augmentation techniques [213] [214]. In a similar way, self-supervised models use data augmentation techniques that combine clean signals with noise or music. The model is forced then to predict the unmodified reference signal from the augmented context embeddings. This way, the intra-class variability for each phoneme may be compensated even if it is presented under different acoustic conditions (combined with music, noise, etc, ...). While this compensation may be interesting from an ASR perspective, in a multiclass audio segmentation task such as the one evaluated in this chapter it

would significantly decrease the inter-class separation, bringing together samples with clean speech and speech combined with other acoustic classes.

Under the intuition that augmentation performed in the training process of self-supervised models may compensate variability for classes containing music and noise, we evaluate a version of wav2vec without any kind of data augmentation and a version applying standard augmentation techniques. In the case where data augmentation is applied, augmented versions of the context embedding C_t serve as input to predict an unmodified reference Z_t.

9.2.3 Training conditions

We distinguish three different wav2vec models depending on how they have been trained. Particularly, we differentiate if data augmentation has been used or not, and consider two training conditions: a first one trained with a smaller amount of data, and a second one using a larger amount of data in training. With these considerations three different models were evaluated:

- **wav2vec base**: this model was trained using mainly audio data in Spanish. Data from Albayzin 2018 and 2020 evaluations [125] and from the Spanish partition of the Commonvoice 2 dataset was used. This makes a total of around 1200 hours of unlabelled audio for training. Concerning the data augmentation process, noises from MUSAN [133] database are added sampling a uniform distribution in the range (3,15) dB to obtain the signal to noise ratio (SNR). Furthermore, a variety of room impulse responses (RIR) are simulated using the gpuRIR toolkit [200].
- **wav2vec base no aug**: this model was trained on the exact same training data as the wav2vec base model with the only difference that no data augmentation of any kind was applied in training time.
- **wav2vec extended no aug**: a larger amount of data was added to the datasets already described in the base model. Several well known English corpora were included in the training data: LibriSpeech, LibriLight, Tedlium and VoxCeleb 1&2. This makes a total of around 60000 hours of speech audio. Additionally, a dataset of music scrapped from different sources, and presenting around 1700 hours of audio, was also considered, seeking to improve the modelling capabilities of the system in classes containing music. As done in the previous model, no data augmentation was applied in training time.

9.3 Experimental setup

9.3.1 Data description

Data used for the set of experiments described in this chapter is the same as the one described in Section 8.3.1 for the Albayzín 2010 evaluation. The 87 hours of audio available

are splitted in the same way for training and testing. As explained previously, the database separates five different acoustic classes: clean speech (sp), music (mu), speech over music (sm), speech over noise (sn) and others (ot). The class others is not evaluated in the final test. Unlike previous work presented in Chapter 8, this work does not follow Albayzín 2010 evaluation conditions because several external datasets apart from the one provided for the original challenge were included in the self-supervised representation learning stage.

9.3.2 Neural network classifiers

Following on the working line for the Albayzín 2010 audio segmentation task described in Chapter 8, neural network architectures used in this chapter are based on the use of BLSTM layers. Specifically, three different variants with increasing complexity that have been proven to show relevant results are evaluated. In the following lines a brief description is provided for each one of them:

- **1BLSTM**: a single BLSTM layer is used to process the input. Final classification scores are obtained through a linear layer. One segmentation label is emitted for every frame at the input, this is one each 10 milliseconds.
- **2BLSTM pool**: a BLSTM layer is followed by an average pooling mechanism on the temporal dimension as a way to generate a smoother output score. The output of the average pooling is then fed to another BLSTM layer working at one tenth of the original frame rate. As in the previous architecture, final classification scores are obtained with a linear layer. Due to the average pooling, this model produces a segmentation label every 100 milliseconds.
- **2BLSTM pool + mixup**: this alternative incorporates mixup augmentation [205] on top of the 2BLSTM pool architecture. Mixup is a data agnostic augmentation technique that generates new virtual examples through linear combinations of examples from training data. Weights for this combination are usually sampled from a beta distribution $X \sim \text{Beta}(\alpha, \beta)$ with parameters $\alpha = \beta$.

All wav2vec models are frozen during the classifier training stage and, therefore, are used only as a feature extractor. All classifier models have been trained using Adam optimiser, with a learning rate that decays exponentially from 10^{-3} to 10^{-4} during the 20 epochs that data is presented. Cross entropy loss is used as training objective as usually done in classification tasks.

9.3.3 Evaluation metric

In order to evaluate our results we follow the same metric as the one proposed in the original Albazín 2010 evaluation, the average class error over the considered acoustic classes. More details of this metric are provided in Section 8.3.3. Additionally, results

provided for discussion in the final section of this chapter also report SER as metric for comparison. As described in the original evaluation protocol, a collar of 1 second is not scored around each reference boundary.

9.4 Results

As a comparison to results obtained using the presented wav2vec representations, we also evaluate the neural network classifiers using a set of traditional features that combines log Mel filter-bank energies and chroma features. Namely, we concatenate 80 dimension log Mel filter-bank energies and the log energy of the frame with 12 chroma coefficients. Furthermore, first and second derivatives are computed and concatenated to the input. As it was proven by experimental results from Chapter 8, this frontend can lead to competitive results in the multiclass audio segmentation task.

In a similar way as done in Chapter 7, an exploratory analysis of the wav2vec representations capabilities is performed in a first approximation. For that, a TSNE [201] dimensionality reduction of the best performing wav2vec system (wav2vec extended no aug) is computed in order to obtain a 2D visualisation. These features are also compared to the traditional frontend based on log Mel energies and chroma features. The mentioned representation extracted for the validation partition of the Albayzín 2010 dataset is presented in Figure 9.1. It can be observed that the use of a wav2vec representation

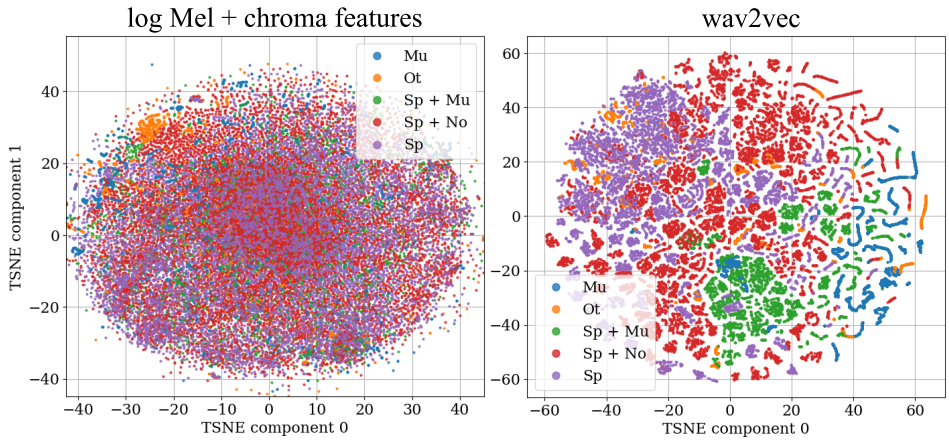


Figure 9.1: TSNE 2D representation of the Albayzín 2010 validation subset for the traditional set of features using log Mel and chroma (left), and the best performing wav2vec system (right).

provides a 2D TSNE plane that is much more easily separable. The TSNE plane obtained for the log Mel and chroma features shows only a limited number of clusters that can be identified, e.g., for the class others in orange. On the other hand, the wav2vec representation results in a plane where separation between classes is much more distinguishable, and that shows no significant overlap between classes such as speech and music, clean

speech and speech and noise. In the case of the wav2vec system, the music class seems to obtain the most disperse representation, with several small sparse subclusters.

Having used a non-linear projection in order to study the class separability provided by the wav2vec representations, we aim now to evaluate these kind of systems in the multiclass audio segmentation task. In the following lines we describe the different experiments carried out with different wav2vec systems and different neural network classifiers. Table 9.1 presents the results obtained using the 1BLSTM classifier on the Albayzín 2010 test partition with different frontend configurations, comparing traditional features and wav2vec representations, in terms of per class error and average error. It can be

Table 9.1: Error per class and average class error for the 1BLSTM classifier on the Albayzín 2010 test partition for different frontend configurations (Δ,ΔΔ: 1st and 2nd order derivatives).

Features	**Class err.(%)**				**Avg err.(%)**
	mu	**sp**	**sm**	**sn**	
80 Mel + chroma + Δ,ΔΔ	**16.28**	28.82	26.32	31.94	25.84
wav2vec base	21.46	28.09	26.72	30.35	26.65
wav2vec base no aug	20.37	28.21	25.87	31.04	26.37
wav2vec extended noaug	19.65	**27.32**	**22.10**	**28.80**	**24.47**

observed that the wav2vec base and wav2vec base no aug systems underperform when compared to the baseline system that uses log Mel energies and chroma features. This drop in performance comes mainly motivated by a large increase in the music class classification error. Furthermore, class error in the rest of classes remains similar. Due to both facts mentioned, the average error is increased. In the case of the wav2vec base no aug model, which used no data augmentation in the pretraining stage of the wav2vec system, the error in the music and speech with music classes decreases compared to the model using data augmentation. Even though these differences are significant, they do not affect in a major way the average error observed, that remains similar to one from the version using data augmentation. Best results are observed when incorporating more data in the pretraining phase of the wav2vec system. The wav2vec extended noaug version yields a relative improvement of 5.2%, 16.03% and 6.7% respectively in the classes speech, speech with music and speech with noise compared to the baseline frontend configuration. The music class error rate still remains high compared to the performance of the baseline system, limiting the overall performance of the wav2vec representations to a 5.3% relative improvement in the average error metric.

Tables 9.2 and 9.3 describe respectively results obtained using the 2BLSTM pool and 2BLSTM pool + mixup classifiers on the Albayzín 2010 test partition comparing traditional features to the best performing wav2vec representation, in terms of per class error and average error. In general terms, trends observed in previous results hold for the new neural architectures evaluated in results reported in Tables 9.2 and 9.3. Focusing first on results described in Table 9.2, that uses the 2BLSTM pool classifier, it can be appreciated that again all the classes but music show an improvement when using a wav2vec

Table 9.2: Error per class and average class error for the 2BLSTM pool classifier on the Albayzín 2010 test partition comparing traditional features and the best performing wav2vec representation (Δ,ΔΔ: 1st and 2nd order derivatives).

Features	Class err.(%)				Avg err.(%)
	mu	sp	sm	sn	
80 Mel + chroma + Δ,ΔΔ	**15.55**	29.16	24.34	30.51	24.89
wav2vec extended noaug	19.29	**24.75**	**21.96**	**26.69**	**23.24**

representation as input for the neural network. The improvement between the 1BLSTM classifier and the 2BLSTM pool classifier is consistent when using both the log Mel energies and chroma features and the wav2vec representation.

Finally, experiments depicted in Table 9.3 evaluate the mixup augmentation strategy under with the 2BLSTM pool + mixup classifier. Same conditions as described in the set of experiments from Chapter 8 are used: mixup is applied on the feature space, and using 3 different α values 0.1, 0.2 and 0.3. The best performance is obtained using a setup

Table 9.3: Error per class and average class error for the 2BLSTM pool + mixup classifier on the Albayzín 2010 test partition comparing traditional features and the best performing wav2vec representation (Δ,ΔΔ: 1st and 2nd order derivatives).

Features	α	Class err.(%)				Avg err.(%)
		mu	sp	sm	sn	
80 Mel + chroma + Δ,ΔΔ	0.1	15.21	27.99	23.05	29.34	23.90
wav2vec extended noaug		19.27	24.87	20.40	26.62	22.79
80 Mel + chroma + Δ,ΔΔ	0.2	**14.64**	28.20	22.01	29.01	23.56
wav2vec extended noaug		17.41	**24.58**	**20.01**	**26.23**	**22.06**
80 Mel + chroma + Δ,ΔΔ	0.3	16.03	26.36	23.89	28.82	23.62
wav2vec extended noaug		18.66	25.70	20.78	26.65	22.95

with $\alpha = 0.2$ for both the traditional set of features and the wav2vec representation. Furthermore, all setups showed an improvement when using mixup augmentation and the wav2vec representation. The best result in this chapter using a wav2vec system is obtained with the 2BLSTM pool architecture, mixup augmentation and $\alpha = 0.2$, achieving a relative improvement close to 6.8% when compared to the same setup using log Mel and chroma features. Even though wav2vec models show better performance than traditional features in classes containing speech, experimental results suggest that one limitation of the wav2vec representations evaluated come from its capacity to model music fragments, underperforming traditional features in all the experiments shown. This fact strongly limits the boost in average performance observed, that could be improved by decreasing music class error.

9.5 Discussion

Once all the experiments presented have been described, this sections aims to put them in context within previous solutions and other approaches in the literature. In addition to the results presented, we also compare the best wav2vec representation obtained with one of the most recent approaches to self-supervised learning, wavLM features [118]. Considering the 2BLSTM pool + mixup classifier, a new audio segmentation system was trained using the available pretrained wavLM Base+[1] features as input, and using the same experimental setup described in Section 9.3. Furthermore, experiments from Chapter 8 demonstrated the capabilities of HMM smoothing to improve segmentation results. We compare the application of these techniques using the new features introduced. HMM smoothing is used under the same conditions as explained in Section 8.2.2. This final comparison is presented in Table 9.4, showing SER, per class error and average class error for systems with and without HMM smoothing, and using traditional features, the best performing wav2vec system and the mentioned wavLM representation. When comparing

Table 9.4: SER, error per class and average class error for the 2BLSTM pool + mixup classifier ($\alpha = 0.2$) with or without HMM smoothing on the Albayzín 2010 test partition comparing traditional features with the best performing wav2vec representation and a wavLM representation (Δ,ΔΔ: 1st and 2nd order derivatives).

Features	SER	Class err.(%)				Avg err.(%)
		mu	sp	sm	sn	
80 Mel + chr + Δ,ΔΔ	14.80	**14.64**	28.20	22.01	29.01	23.56
wav2vec ext. noaug	**13.22**	17.41	**24.58**	**20.01**	**26.23**	**22.06**
wavLM Base+	15.61	20.52	27.79	27.64	30.51	26.62
80 Mel + chr + Δ,ΔΔ/HMM	11.80	**12.46**	22.86	17.34	24.35	**19.25**
wav2vec ext. noaug/HMM	**11.51**	19.62	**20.91**	**17.23**	**22.81**	20.14
wavLM Base+/HMM	14.03	22.82	24.62	25.35	27.62	25.10

previous results with wavLM features, it can be seen that error metrics rise significantly compared to the baseline features. In fact, error for all classes but clean speech is higher. This may come motivated by how wavLM models are trained: using several noise conditions and speech overlap at the input to improve the model robustness in challenging acoustic environments. While this is interesting from the perspective of applications that are focused on the information contained in speech, it seems that it compensates the variability for the different classes present in the specific audio segmentation task considered in this chapter.

Concerning the results obtained applying HMM smoothing, the same general behaviour as the one observed in Chapter 8 can be appreciated. An overall improvement in performance is seen when using the HMM smoothing in terms of average class error and SER. While for the baseline features error was reduced in all classes after the HMM smoothing, wav2vec and wavLM features showed a degradation in performance for the

[1] https://github.com/microsoft/unilm/tree/master/wavlm

music class after applying HMM smoothing. Obtained neural networks scores in these cases may not be representative enough of musical content, what could result in a wrong estimation of HMM parameters affecting its overall performance.

9.6 Conclusions

This chapter presents a study on the application of self-supervised representation learning to the multiclass audio segmentation task. Namely, different wav2vec models have been evaluated with the goal of jointly discriminate audio information belonging to speech, music and noise classes in the Albayzín 2010 dataset. Following previous studies that showed competitive performance using a combination of log Mel filter-bank energies and chroma features, we apply the same neural network classifier models using the new wav2vec features seeking to obtain a comparable experimental setup. The task is evaluated using the average error among all acoustic classes as done in the original evaluation.

Self-supervised learning was initially meant to be applied in task such as ASR, where background noise or music needs to be compensated to focus on the uttered speech. A common trend in these models is to use data augmentation techniques that combine clean signals with noise or music. The model is forced then to predict the unmodified reference signal from the augmented context embeddings. Three different data conditions are evaluated in this chapter, considering cases with and without data augmentation. Results suggest that data augmentation techniques are not beneficial for this specific multiclass segmentation task. Under these conditions, feature obtained underperform when compared to the baseline set of features considered.

In general terms, experimental results demonstrate that the use of wav2vec representations can lead to a significant improvement in the performance of audio segmentation systems for classes containing speech. For instance, the relative improvement observed using wav2vec representations for clean speech and speech over noise classes is around 12.84% and 9.58% respectively. However, a degradation in performance is observed in the segmentation of isolated music when compared to traditional features. Any configuration evaluated using wav2vec representations was able to obtain better results than baseline features in the music class. In fact, the best results for the music class still show a relative degradation close to 18%. This limits the overall improvement observed to a relative improvement close to 6.8% on the Albayzín 2010 segmentation task in terms of average class error.

Part IV

AUC optimisation for audio segmentation

"Tell me how you measure me,
and I will tell you how I will behave."

Eliyahu M. Goldratt

10

Binary AUC optimisation for audio segmentation

10.1 Introduction and motivation

During the description of the experiments performed in previous chapters, several approaches have been applied in order to develop audio segmentation systems under different scenarios. Yet, all these experiments have one common component, the loss function. A unified approach based on the cross entropy loss is used to optimise the neural networks in all previous systems. A significant part of the success in training a DNN system comes from the loss function chosen since, in binary tasks, this loss can

help discriminate in an easier way positive and negative examples. Recently, a number of research works have investigated the development of new loss functions in order to find the most appropriate one for training deep learning solutions. Some significant approaches can be cited here such as triplet loss [215], or prototypical loss [216]. One of the most relevant research lines in this domain is the metric learning paradigm. Metric learning loss functions seek to bring together the training and evaluation process. In general terms, metric learning aims to reduce the distance between similar samples, while, at the same time, it also aims to increase the distance between dissimilar samples [217].

Decision threshold for most detection systems is normally determined previously to the deployment of the model, with different applications having different requirements to the operating point. Therefore, it would be interesting to optimise the performance of detection systems at a wide range of decision thresholds. An overall metric describing the detection performance for all possible thresholds is the AUC metric. Under this context, several works have already proposed to train a classifier with a loss based on the AUC metric. Furthermore, as a generalisation of the AUC, partial AUC only measures the area under the ROC curve that is restricted between two false positive rates. Using this generalisation as a loss function may be interesting in applications where the difference in cost of false positive compared with false negatives is large.

In closing, this chapter introduces the recently proposed AUC optimisation techniques for neural networks in the music detection task. Recent studies have proved that these new optimisation techniques outperform traditional training objectives such as cross entropy in other detection tasks. This is done with the main goal of overcoming the possible issues arising when using a limited amount of training data in deep learning systems. One of the main disadvantages of deep learning systems is that their performance is strongly dependent on the data they were trained on. Particularly, labelled data for music detection is limited when compared to other tasks such as SAD. With the introduction of these new optimisation techniques we aim to increase the performance of music detection systems in this context of limited training data.

10.2 AUC and pAUC optimisation framework

10.2.1 Problem formulation

Suppose a dataset $\Pi = \{\mathbf{X}, \mathbf{Y}\}$ where $\mathbf{X} = \{x_1, \dots, x_N\}$ is the set of acoustic features with N different examples, and $\mathbf{Y} = \{y_1, \dots, y_N\}$ the music labels defining each of the elements in $\mathbf{X}$ as music or non-music examples (1 or 0 respectively). The classifier model can be expressed then as a function $f_\theta : \mathbb{R}^D \rightarrow \mathbb{R}$ depending on a set of parameters θ and mapping the input space of dimension D to a real number representing the music score. The parameters θ are estimated seeking to minimise (or maximise) a metric given by a loss function $\mathcal{L}(\, f_\theta(x_i), y_i \,)$ that measures the difference (or similarity) between the classifier output and the labels.

10.2.2 Area under the ROC curve optimisation

The receiver operating characteristic (ROC) curve is a well known method to represent the performance of a detection system. This curve plots false positive rates on the x axis versus true positive rates on the y axis for all the possible detection thresholds. Furthermore, the area under the ROC curve (AUC) is a measure of the performance of a binary classifier system as the discrimination threshold is varied. This metric is calculated as the surface area between the FPR versus TPR curve and the x axis, as all possible classification thresholds are considered. The AUC represents the probability that a randomly selected positive sample will have a higher predicted probability of being positive than a randomly selected negative sample. Depending on the desired behaviour, the detection threshold may be chosen differently. That is why it would be desirable to optimise the system for all the possible decision thresholds, taking into account the trade-off between false positives and false negatives.

Several papers have already proposed to directly optimise the AUC for different applications with promising results [218] [219]. Focusing on those using neural networks, an AUC optimisation framework is adopted in [220] for the text-dependent speaker verification task. In [147], a deep learning based speech activity detection system is trained with an AUC optimisation criterion.

To compute the AUC metric two new subsets need to be defined: $\mathbf{S}^+ = \{ f_\theta(x_i) \; \forall x_i \in \mathbf{X} \,|\, y_i = 1\}$, which is the neural network scores for the positives examples in $\mathbf{X}$, and $\mathbf{S}^- = \{ f_\theta(x_i) \; \forall x_i \in \mathbf{X} \,|\, y_i = 0\}$ that represents the neural network scores for the negatives examples in $\mathbf{X}$. Cardinalities of those sets are N^+ and N^- respectively. Then, the AUC loss can be defined as

$$\mathbb{L}_{\text{AUC}} = \frac{1}{N^+N^-} \sum_{i=1}^{N^+} \sum_{j=1}^{N^-} \mathbb{1}\left(s_i^+ > s_j^-\right), \tag{10.1}$$

where $\mathbb{1}(\cdot)$ is equal to '1' whenever $s_i^+ > s_j^-$ and '0' otherwise. This expression can be rewritten using the unit step function as

$$\mathcal{L}_{\text{AUC}} = \frac{1}{N^+N^-} \sum_{i=1}^{N^+} \sum_{j=1}^{N^-} u\left(s_i^+ - s_j^-\right). \tag{10.2}$$

In order to enable the backpropagation of the gradients, an approximation must be done to obtain a differentiable function. In our implementation, we adopt the expression proposed in [220], that modifies the step function for a sigmoid function according to

$$\mathcal{L}_{\text{aAUC}} = \frac{1}{N^+N^-} \sum_{i=1}^{N^+} \sum_{j=1}^{N^-} \sigma\left(\delta\left(s_i^+ - s_j^-\right)\right), \tag{10.3}$$

where $\sigma(\cdot)$ is the sigmoid function and δ is an hyperparameter that controls the slope of the sigmoid.

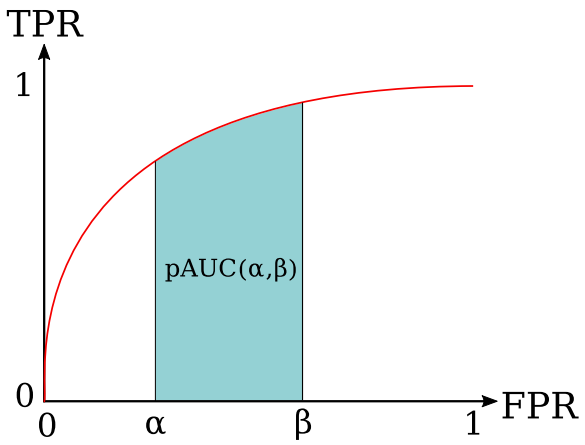


Figure 10.1: Schematic representation of ROC curve and partial AUC given α and β.

10.2.3 Partial AUC optimisation

The usefulness of AUC can be limited in some cases due to its threshold invariance. If there are wide disparities in the cost of false positives compared with the cost of false negatives, it may be critical to minimise one type of error. Furthermore, optimising the whole ROC curve can be costly and, in some specific applications, needless as the system is going to operate only in a certain region of the ROC curve. The partial AUC (pAUC) overcomes these issues evaluating the performance for a region of interest of FPR values. It is formally defined as the area under the ROC curve between two FPR values α and β. This metric is presented schematically in Figure 10.1, with the blue area representing the pAUC.

pAUC optimisation was firstly proposed in [221] for the speaker verification task outperforming the current state-of-the-art on the NIST 2016 SRE data. The hard negative mining approach presented in [220] could also be interpreted as a partial AUC solution that chooses the hardest examples for training. A more recent study [222] has compared both AUC and pAUC training objectives to obtain speaker embeddings for text-independent speaker verification. This work shows that pAUC training achieves better results than AUC training in most cases.

Given the already defined $\mathbf{S}^-$ set, a new set $\mathbf{S}^-_{\alpha\beta}$ must be obtained constraining $\mathbf{S}^-$ to the range where the false positive rate lies in the interval $[\alpha, \beta]$. This is done through the following three main steps:

1. The interval $[\alpha, \beta]$ must be replaced for its integer equivalent, this is $[\frac{n^-_\alpha}{N^-}, \frac{n^-_\beta}{N^-}]$. with n^-_α and n^-_β computed according to

$$n^-_\alpha = \lceil \alpha N^- \rceil + 1\,, \quad n^-_\beta = \lfloor \beta N^- \rfloor\,. \tag{10.4, 10.5}$$

2. Sort $\mathbf{S}^-$ in descending order

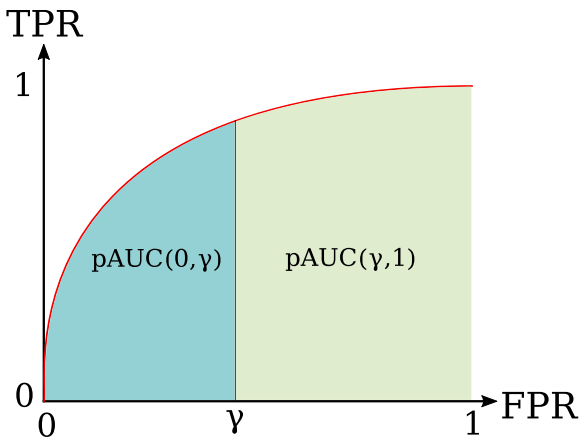


Figure 10.2: Schematic representation of ROC curve and our proposed computation of AUC as the sum of two partial AUC defined by the parameter γ.

3. $\mathrm{S}^-_{\alpha\beta}$ is selected as the set of samples ranked from the top n^-_αth to the n^-_βth position of the sorted S^- set. This results in a set of length $N^-_{\alpha\beta} = n^-_\beta - n^-_\alpha + 1$

Now, the partial AUC can be computed in a similar way to the expression proposed in Equation (10.3), but substituting the set of negative examples S^- for the new set of constrained negative examples $\mathrm{S}^-_{\alpha\beta}$. This partial AUC is expressed as

$$\mathcal{L}_{\text{apAUC}}(\alpha,\beta) = \frac{1}{N^+ N^-_{\alpha\beta}} \sum_{i=1}^{N^+} \sum_{j=1}^{N^-_{\alpha\beta}} \sigma\Big(\delta\big(s^+_i - s^-_{\alpha\beta_j}\big)\Big)\,. \tag{10.6}$$

10.2.4 AUC optimisation as sum of two partial AUCs

When computing the pAUC loss, a fraction of the training examples is discarded. This is done after sorting the S^- set, where only a subset of the sorted set is used in training. If we suppose $\alpha = 0$ as done in [222], a fraction $1-\beta$ of the examples are consistently dropped in the training process. This fact could be seen as a way of speeding up training because it is reducing the number of operations per iteration. However, as we are dealing with a limited training data scenario, we believe that it would be interesting to incorporate those discarded examples in training somehow.

Our idea to incorporate those examples in training is presented schematically in Figure 10.2. The proposed training objective decomposes the entire AUC as the sum of two partial AUC metrics, assuming the first one is using $\alpha = 0$ and the second one is using $\beta = 1$. The parameter γ is introduced as the FPR point that separates the area in two subareas. Then, our new training objective can be computed as

$$\mathcal{L}_{\text{aAUCsum}}(\gamma,\lambda) = \mathcal{L}_{\text{apAUC}}(0,\gamma) + \lambda\mathcal{L}_{\text{apAUC}}(\gamma,1)\,. \tag{10.7}$$

A scalar hyperparameter λ is used for balancing both parts, seeking to give more or less importance to the second pAUC term in training.

10.3 Experimental setup

We set a fixed experimental setup for all our experiments as our main goal is not to evaluate this specific neural architecture or set of features compared to other proposals, but to evaluate whether AUC and pAUC optimisation can improve the performance of our music detection system over traditional training objectives. Considering the limited amount of training data that is used, the neural network model is chosen accordingly with a simple setup consisting of two stacked RNN layers.

10.3.1 Data description

In order to perform our experiments we consider two different datasets, one for training and another for testing. Both of them correspond to the broadcast domain, coming from different television emissions. We use the OpenBMAT dataset [134] to train our music detection system. As explained in Section 4.1.3, this dataset contains 27 hours of television broadcast audio from different countries. For this set of experiments only the binary labels for music and non-music detection are considered. Furthermore, a 10% of the data is reserved for training validation, so the total amount of data used for training is around 24 hours. No prior data picking is performed in any case; minibatches are sampled randomly for the training dataset.

As test data we use the dataset originally presented in the paper "Automatic Music Detection in Television Productions" [135]. It consists of around 9 hours of television recordings from the Austrian national broadcasting corporation that were manually labelled as music or non-music. In both train and test datasets considered, audios were first downsampled to 16 kHz and mixed down to a single channel.

10.3.2 Neural network classifier

In this set of experiments, the neural architecture is based on an RNN structure. The model proposed for the AUC optimisation is described in Figure 10.3. As it can be observed, 2 bidirectional GRU layers [97] with 128 neurons each are stacked. These layers are then followed by a linear layer that performs the final classification. Adam optimiser is used with a learning rate that decays exponentially from 10^{-3} to 10^{-4} during the 20 epochs that data is presented to the neural network. The hyperparameter δ, introduced in subsection 10.2.2 and used in the AUC based optimisation experiments, has a fixed value of 10 seeking to obtain a shape close to the unit step function, in a similar way as it is done in [220]. Training and evaluation is done using limited length sequences (3 seconds, 300 frames) in order to limit the delay of dependencies to take into account by the RNN. However, an output label is emitted for every frame processed at the input.

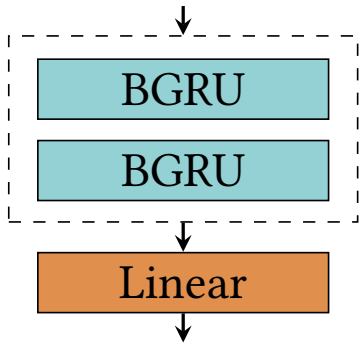


Figure 10.3: Schematic representation of the neural architecture proposed for the AUC optimisation experiments.

10.3.3 Feature extraction

Concerning feature extraction, we combine a traditional set of perceptual features with some musical theory motivated features. First, 128 log Mel filter-bank energies are extracted between 64 Hz and 8kHz. Additionally, they are combined with chroma features [85], extracted using the openSMILE toolkit [207]. All features are computed every 10 ms using a 25 ms Hamming window. Before being concatenated, log Mel energies and chroma features are first normalised to be in the range between 0 and 1.

10.3.4 Evaluation metrics

The music detection task can be interpreted as a binary classification task, so traditional metrics for binary tasks are applicable. In this set of experiments, we evaluate the results of our system using the AUC metric and the EER. We also present the complete ROC curve for the best systems presented in the work. Additionally, we report in our results recall and F_1 measure for a system using a threshold so that precision = 0.90.

10.4 Results

As the starting point of our experimentation, our aim was to obtain a baseline system so that our further results could be compared. With the experimental framework described in Section 10.3, we trained a neural network using the well-known cross entropy loss. This system serves as a point of comparison for our AUC and pAUC optimisation experiments. In Table 10.1, we compare this baseline with a system trained using the aAUC training criterion. It can be observed that, by shifting from traditional loss functions, such as cross entropy, to the aAUC criterion, a significant 4.30% relative improvement can be observed in terms of AUC and a 19.11% relative improvement in terms of EER.

Table 10.1: AUC, EER, recall and F_1 measure (both computed for a system with precision fixed at 0.90) on test data for the music detection system trained using the aAUC training criterion compared to a cross entropy based training.

Training criterion	**AUC (%)**	**EER (%)**	**Recall**	$\mathbf{F_1}$
Cross entropy	87.02	21.40	0.45	0.60
aAUC	**90.76**	**17.31**	**0.65**	**0.75**

Once it has been proved that AUC optimisation can improve music detection performance in a limited data scenario with this first experiment, in the next set of experiments we explore a more general training criterion, this is the apAUC explained previously. It can be easily observed that the aAUC criterion is a specific case of the apAUC criterion that uses parameters $\alpha = 0$ and $\beta = 1$. Results using the pAUC training criterion are presented in Table 10.2.

Table 10.2: AUC, EER, recall and F_1 measure (both computed for a system with precision fixed at 0.90) on test data for the music detection system trained using the apAUC training objective and different values of parameters α and β.

apAUC optimisation		**AUC (%)**	**EER (%)**	**Recall**	$\mathbf{F_1}$
$\alpha = 0$	$\beta = 0.25$	87.21	20.85	0.54	0.68
$\alpha = 0$	$\beta = 0.5$	90.34	17.53	0.64	0.75
$\boldsymbol{\alpha = 0}$	$\boldsymbol{\beta = 0.75}$	**91.88**	**16.03**	**0.71**	**0.79**
$\alpha = 0.25$	$\beta = 0.5$	86.95	21.18	0.40	0.55
$\alpha = 0.25$	$\beta = 0.75$	88.79	19.56	0.55	0.64
$\alpha = 0.25$	$\beta = 1$	86.83	21.69	0.42	0.57

It can be observed that two different parameter setups were assessed. The first set of parameters uses $\alpha = 0$, in a similar way as done in [222]. This configuration is equivalent to discard the fraction $1 - \beta$ of non-target examples with the lower scores, therefore the ones that are easier to classify. The best system performance is achieved for $\beta = 0.75$, with evaluation metrics even better than the ones obtained for the aAUC training objective. This results in a relative improvement compared to the baseline system of 5.60% in terms of AUC and 25.10% in terms of EER. The setup using $\alpha = 0.25$ is presented for comparison purposes. In this case the non-target examples with the higher scores (harder to classify) are discarded for training. It can be clearly observed that this setup underperforms the one with $\alpha = 0$, with AUC and EER values which are close to the baseline system.

Our final experiment explored the proposed aAUCsum training criterion that separates the AUC optimisation in two partial AUCs. The results obtained are presented in Table 10.3. Gamma value was chosen to be 0.75 because it obtained the best performance in the pAUC optimisation experiments. This parameter setup slightly outperforms the apAUC training objective, however, experimental results suggest that incorporating the discarded examples in training does not lead to a consistent performance increment.

Table 10.3: AUC, EER, recall and F_1 measure (both computed for a system with precision fixed at 0.90) on test data for the music detection system trained using the aAUCsum training objective, $\gamma = 0.75$ and different values of λ.

aAUCsum optimisation		AUC (%)	EER (%)	Recall	F_1
	$\lambda = 1$	91.78	16.78	0.70	0.79
	$\lambda = 0.1$	91.31	16.59	0.68	0.77
$\gamma = 0.75$	$\lambda = 0.01$	91.45	16.20	0.69	0.78
	$\boldsymbol{\lambda = 0.001}$	**92.02**	**15.78**	**0.72**	**0.80**
	$\lambda = 0.0001$	91.31	16.78	0.70	0.79

This effect may be motivated by the fact that the hardest examples could be sufficient for the neural network to learn an effective classification mapping. It is also remarkable that, our proposed aAUCsum criterion achieves state-of-the-art performance, with an AUC and EER better than the aAUC training criterion in all cases. This results matches with previous studies that suggest that pAUC optimisation techniques outperform AUC techniques.

Finally, Figure 10.4 presents the ROC curves for the different systems trained in this work using the best parameter configuration obtained. Again, it can be observed that

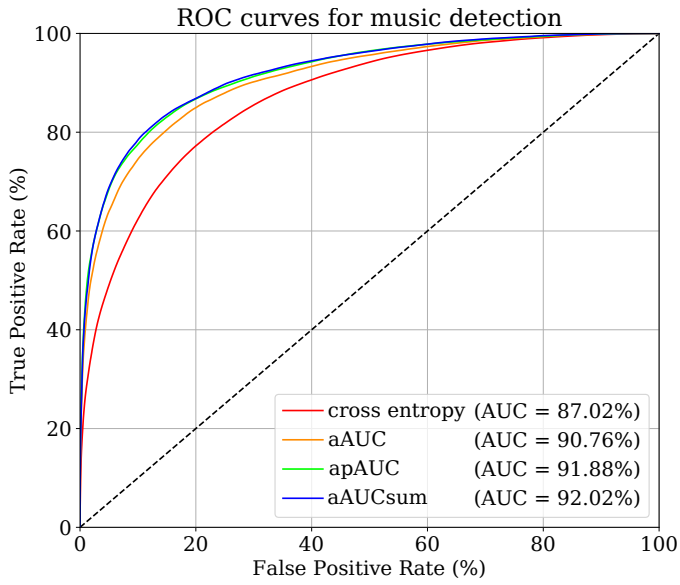


Figure 10.4: ROC curve on the test data for the different training objectives presented in the work using the best parameter configuration obtained (pAUC: $\alpha = 0$ and $\beta = 0.75$, AUCsum: $\gamma = 0.75$ and $\lambda = 0.001$).

the baseline system trained using cross entropy criterion is significantly below in performance compared to the other systems presented in this work. Furthermore, a similar ROC curve is obtained for the apAUC and aAUCsum training objectives, both with a performance better than the one obtained for the aAUC training.

10.5 Conclusions

In this chapter, we have introduced the AUC and pAUC optimisation techniques into the music detection task, aiming to overcome the issues derived from data limitations. We present several approximations to threshold-independent metrics, that can be directly optimised in a deep learning framework by applying an approximation based on the sigmoid function. Considering the AUC loss as the starting point of the experimentation, we also evaluated pAUC based loss functions that considers only a region of interest of FPR values. Furthermore, considering that when computing the pAUC loss a fraction of the training examples is discarded, we propose a novel training objective based on the decomposition of AUC as the sum of two pAUC. The experimental setup proposed is made of a classifier based on recurrent neural networks, that is trained with a limited training dataset consisting only of around 20 hours of audio.

Experimental results suggest that AUC and partial AUC training both outperform traditional training objectives such as cross entropy. All results obtained using any of the AUC based loss functions provide a better performance than cross entropy training under the experimental setup described. By using the described aAUC loss function, a significant improvement in performance can already be observed, resulting in a relative improvement around 4% in terms of AUC and 19% in terms of EER. When considering the apAUC optmisation, systems using $\alpha = 0$ outperform the aAUC approach. This suggests that the fraction $1 - \beta$ of non-target examples with the lowest scores is not so relevant in the learning process. Furthermore, the best result is obtained using our proposed aAUCsum loss function, with a relative improvement close to 5.75% in terms of AUC, and 26.26% in terms of EER. By observing this result, it can be inferred that reincorporating the previously mentioned fraction $1 - \beta$ of non-target examples in the training process with an appropriate weight is able to provide a boost in performance. However, this increase is not consistent for different weights. In general, our results match with the previously published studies that state that pAUC optimisation techniques report better performance than AUC based optimisation.

"Particular facts are never scientific;
only generalisation can establish science."

Claude Bernard

11

Generalising AUC optimisation to multiclass classification

11.1 Introduction and motivation

Results from Chapter 10 have shown the capabilities of new AUC based optimisation techniques for audio segmentation tasks, focusing on the music detection task. In addition, a number of works have already proposed to directly optimise the AUC for different applications with promising results. Nevertheless, up until now, most presented works that aim to optimise the AUC metric using a deep learning framework are limited to binary tasks, as this is the usual situation when applying the AUC metric. As already shown in Part III of this dissertation, multiclass tasks pose new challenges for audio segmentation systems, increasing the possible confusion between classes and, in general terms, showing a lower performance when compared to binary segmentation

systems. Yet, multiclass descriptions allow to provide a more detailed segmentation of the audio content, with acoustic classes providing more specific information. Furthermore, the problem of data scarcity is even more relevant for multiclass audio segmentation systems. Labelling audios with detailed taxonomies involving multiples options is significantly more complex than using a binary labelling schema. That is the reason why the amount of labelled audio for some multiclass audio segmentation tasks is still limited.

Considering these ideas, it would be interesting to be able to benefit from the improvement in performance observed when using AUC optimisation techniques in audio segmentation tasks with more than two classes. As a natural extension of the working line presented in Chapter 10, the major contribution of this chapter is the introduction of a generalisation for the AUC optimisation techniques presented in the previous chapter, so that they can be applied to an arbitrary number of classes. This is done considering as starting point some of the the multiclass variations of the AUC metric proposed in the literature. To demonstrate experimentally its validity, the proposed training objectives are evaluated on the relative music loudness estimation task, whose goal is to separate an audio signal into three classes: foreground music, background music and no music. This task was proposed initially in the MIREX [223] 2018 and 2019 technological evaluations, and it is supported by the release of the OpenBMAT dataset [134] for training and evaluation, which we use in this set of experiments. Using a limited training dataset of around 20 hours of audio, this chapter seeks to overcome data limitations for audio segmentation systems in multiclass setups.

11.2 Beyond binary AUC optimisation

11.2.1 From binary to multiclass AUC

As described in Section 4.2.2, due to its intrinsic nature, ROC analysis [136] and the AUC metric are commonly used in binary tasks, where the goal is to separate two different sets: targets and non-targets. However, different studies have proposed an extension of the AUC metric in order to be used as a performance measure in tasks with more than two classes [224] [225] [226] [227]. Generally, two main approaches are known:

- **one-versus-one (OVO) approach**: originally proposed in [225], this approach extends the AUC metric to the multiclass domain by averaging pairwise comparisons of classes. Supposing a problem with c classes ($c > 2$), we define the set C_i as the set of elements belonging to class i, and $\text{AUC}(C_i, C_j)$ as the binary AUC metric obtained using the elements belonging to class i as targets and the elements belonging to class j as non-targets. Therefore, the overall multiclass AUC can be defined according to the following equation:

$$\text{AUC}_{\text{OVO}} = \frac{2}{c\,(c-1)} \sum_{i=0}^{c-2} \sum_{j=i+1}^{c-1} \widehat{\text{AUC}}(C_i, C_j)\,. \tag{11.1}$$

In general, for more than two classes, $\text{AUC}(C_i, C_j) \neq \text{AUC}(C_j, C_i)$. This problem is tackled in [225] by using the modified version $\widehat{\text{AUC}}(C_i, C_j)$ as the measure of separability between classes i and j. This is done according to the following equation:

$$\widehat{\text{AUC}}(C_i, C_j) = \frac{1}{2}[\text{AUC}(C_i, C_j) + \text{AUC}(C_j, C_i)]\,. \tag{11.2}$$

- **one-versus-rest (OVR) approach**: This approach was firstly proposed in [227], and is based on the idea that a classification system with c classes ($c > 2$) can also be interpreted as a system running c binary classifiers in parallel. With this idea in mind, the multiclass AUC metric can be defined as the mean of the c AUC binary metrics obtained, as expressed in the following equation:

$$\text{AUC}_{\text{OVR}} = \frac{1}{c}\sum_{i=0}^{c-1}\text{AUC}(C_i, C_i^C)\,, \tag{11.3}$$

where C_i^C is the complement set of C_i and $\text{AUC}(C_i, C_i^C)$ is the AUC metric obtained setting the elements belonging to class i as targets, and the elements belonging to the other classes as non-targets.

Experiments described in Chapter 10 have shown that binary AUC optimisation techniques can lead to significant improvements in performance for the music detection task when compared to traditional training criterion in a limited training data scenario. In addition, other binary audio related tasks have also demonstrated the capabilities of the AUC and partial AUC optimisation techniques in a deep learning framework: speaker verification systems have been trained by optimising the AUC [220] and partial AUC [228] [222] metrics. An equivalent approach to AUC optimisation is used in [229] considering tuples of positives and negatives examples for speaker verification. An evaluation of the most popular loss functions based on the metric learning paradigm is presented in [230] for the speaker recognition task.

Building upon the description of the multiclass AUC metrics shown, in this chapter we aim to extend the AUC optimisation techniques to the multiclass classification framework in order to provide a generalisation that can be applied to an arbitrary number of classes. To the best of our knowledge, this is one of the first approaches to multiclass AUC optimisation in deep learning applications. Posterior studies have also proposed methods to optimise multiclass AUC metrics. Work presented in [231] introduced an empirical surrogate risk minimisation framework to approximately optimise the AUC_{OVO} metric. Authors in [232] perform multiclass AUC optimisation for keyword spotting applications by applying a modified hinge loss to obtain a differentiable expression.

11.2.2 Multiclass AUC Optimisation

As the starting point of the formulation for the multiclass AUC loss functions, we need to refer to the already described binary AUC optimisation definition. Here, the main elements of it are summarised in the following lines as a reminder , but more details can be

found in Section 10.2.2. Considering the formal definition of AUC shown previously in Equation (10.2), the sigmoid approximation described in [220] was applied to overcome the non-differentiability issues derived from the unit step function, allowing the back-propagation of gradients. This leads to the expression of the aAUC loss (where '*a*' stands for approximated) described in the following equation:

$$\mathcal{L}_{\text{aAUC}} = \frac{1}{N^+N^-} \sum_{i=1}^{N^+} \sum_{j=1}^{N^-} \sigma\Big(\delta\big(s_i^+ - s_j^-\big)\Big), \tag{11.4}$$

where $\sigma(\cdot)$ is the sigmoid function and δ is a hyperparameter that controls the slope of the sigmoid. As it has already been explained in the previous subsection, the multiclass variations of the AUC metric are built upon the computation of several binary AUC metrics, so Equation (11.4) can be used as the main element to build our proposed multiclass AUC training objectives. Just by modifying which classes are used as targets and non-targets we can derive both multiclass loss functions. In the case of the OVO approach, through the sigmoid approximation and by combining Equation (11.1) and Equation (11.2), it is straightforward to obtain the expression for the aAUC_{OVO} loss:

$$\mathcal{L}_{\text{aAUC}_{\text{OVO}}} = \frac{2}{c\,(c-1)} \sum_{i=0}^{c-2} \sum_{j=i+1}^{c-1} \frac{1}{2}\left[\mathcal{L}_{\text{aAUC}}(C_i, C_j) + \mathcal{L}_{\text{aAUC}}(C_j, C_i)\right], \tag{11.5}$$

where $\mathcal{L}_{\text{aAUC}}(C_i, C_j)$ represents the AUC sigmoid approximation loss computed using the neural network scores belonging to class i as targets and the neural network scores belonging to class j as non-targets.

Similarly, for the the OVR approach the expression for the aAUC_{OVR} loss can be obtained directly applying the sigmoid approximation to Equation (11.3).

$$\mathcal{L}_{\text{aAUC}_{\text{OVR}}} = \frac{1}{c} \sum_{i=0}^{c-1} \mathcal{L}_{\text{aAUC}}(C_i, C_i^C), \tag{11.6}$$

where $\mathcal{L}_{\text{aAUC}}(C_i, C_i^C)$ represents the AUC sigmoid approximation computed using the neural network scores belonging to class i as targets and the neural network scores belonging to the remaining $c-1$ classes as non-targets.

Considering the computational complexity of the proposed multiclass training objectives, in the case of the $\mathcal{L}_{\text{aAUC}_{\text{OVO}}}$, it implies computing $c\,(c-1)$ different aAUCs, resulting in a quadratic growth with the number of classes. On the other hand, in the case of the $\mathcal{L}_{\text{aAUC}_{\text{OVR}}}$, c different aAUCs have to be computed to obtain the final loss, observing a linear growth with the number of classes.

11.3 Experimental setup

With this set of experiments being an extension of the analysis described in Chapter 10, most of the experimental setup is similar to the one described in the previous chapter. The main difference comes from the data. Due to its specific labelling format that allows a multiclass music segmentation, only the OpenBMAT dataset is used for train and test in the evaluation of the proposed multiclass AUC loss functions.

11.3.1 Data description

In order to demonstrate the capabilities of the proposed multiclass AUC optimisation techniques, we apply them in the context of a 3-class audio segmentation task that aims to separate foreground, background and no music fragments. As already mentioned in Section 4.1.3, this task is proposed in the OpenBMAT dataset [134], being the first of its kind to provide annotations separating foreground and background music. Audio data comes from broadcast domain, with emissions from different countries manually labelled.

Dataset authors predefined 10 splits, that we use to separate data for training, validation and test purposes: splits zero to seven are used for training, while split eight is reserved for training validation. Split number nine constitutes the testing set. This translates in a total of 22 hours of audio for training, and around 3 different hours for validation and test respectively. All the audio files have been downsampled to 16 kHz and mixed down to a single channel.

11.3.2 Neural network classifier

The same neural architecture as described in Section 10.3.2 is implemented for the evaluation of multiclass AUC training objectives. The only difference comes from the final linear layer, that uses a 3 neurons output in this set of experiments. Concerning training details, Adam optimiser is applied with a learning rate that decays exponentially from 10^{-3} to 10^{-4} during the 20 epochs that data is presented to the neural network with a minibatch size of 64. Model selection is done by choosing the best performing model in terms of frame classification accuracy using the validation subset. The hyperparameter δ used to control the slope of the sigmoid has a value of 10 seeking to obtain a shape close to the unit step function, as in [220]. After neural network training, the 3-dimensional scores for the validation subset are extracted and then used to adjust a logistic regression model via a stochastic gradient descent algorithm. This model is used to obtain the final classification labels.

11.3.3 Feature extraction

The same approach to feature extraction described in Section 10.3.3 is followed in this new set of experiments, combining 128 log Mel filter-bank energies and chroma features [85]. Again, all features are computed every 10 ms using a 25 ms Hamming window. Before being concatenated, log Mel energies and chroma features are first normalised to be in the range between 0 and 1.

11.3.4 Evaluation metrics

In order to evaluate the performance of the audio segmentation system, we distinguish two kinds of metrics, those showing information for all the possible thresholds considering the neural network scores, and those showing information for the final decisions

made by the classification system. As an aggregate measure of the discrimination of all the classes, we report multiclass AUC in the two variations presented previously: AUC_{OVO} and AUC_{OVR}. We also show the average area under the precision versus recall curve. In order to illustrate each of the class performance, we include the precision versus recall curve per class with the corresponding F_1 isocurves. Furthermore, we report final classification performance with overall accuracy, and per class precision, recall and F_1 score.

All metrics are computed at frame level and without collar on the full test data described previously. Seeking to validate the statistical robustness of our proposal, all our experiments are run 10 different times reporting mean and standard deviation or both of them for the metrics described.

11.4 Results

Table 11.1: AUC_{OVO}, AUC_{OVR} and average area under the precision versus recall curve on test data for the audio segmentation systems trained using the proposed multiclass AUC training objectives compared to two variants of cross entropy based training (Mean ± standard deviation over 10 different experiments).

Training objective	**AUC_{OVO}(%)**	**AUC_{OVR}(%)**	**Avg. AUC (%) prec vs recall**
Softmax CE	81.69±0.84	79.42±0.69	66.05±0.97
Angular softmax	80.95±0.71	79.23±0.65	65.89±0.91
$aAUC_{OVO}$	83.67±0.54	81.28±0.61	69.55±0.80
$aAUC_{OVR}$	82.46±0.81	80.33±0.71	68.51±0.90

As the starting point of our experimentation, we aim to obtain a baseline system so that further results could be compared. Using the experimental framework described previously, a neural network was trained using a cross entropy loss function. Note that, in this work, cross entropy is implemented as a multiclass loss, also known as softmax cross entropy in the literature. Furthermore, results are also compared with a well known variant of the softmax cross entropy, the angular softmax loss [233]. Both systems serve as a comparison point for our proposed multiclass AUC training objectives. Table 11.1 compares the AUC_{OVO}, AUC_{OVR} and average area under the precision versus recall curve on test data for our proposed multiclass AUC loss functions and the two cross entropy training variants. It must be noted that both $aAUC_{OVO}$ and $aAUC_{OVR}$ provide an improvement compared to softmax and angular softmax training in all the metrics presented. This is most noticeable in the OVO approach that yields an approximate 2.5% average relative improvement in both AUC multiclass metrics and an approximate 5% average relative improvement in area under the precision versus recall curve when compared to softmax CE training.

Figure 11.1 summarises the neural network scores performance per class for different thresholds comparing our proposed AUC multiclass loss functions to the two cross en-

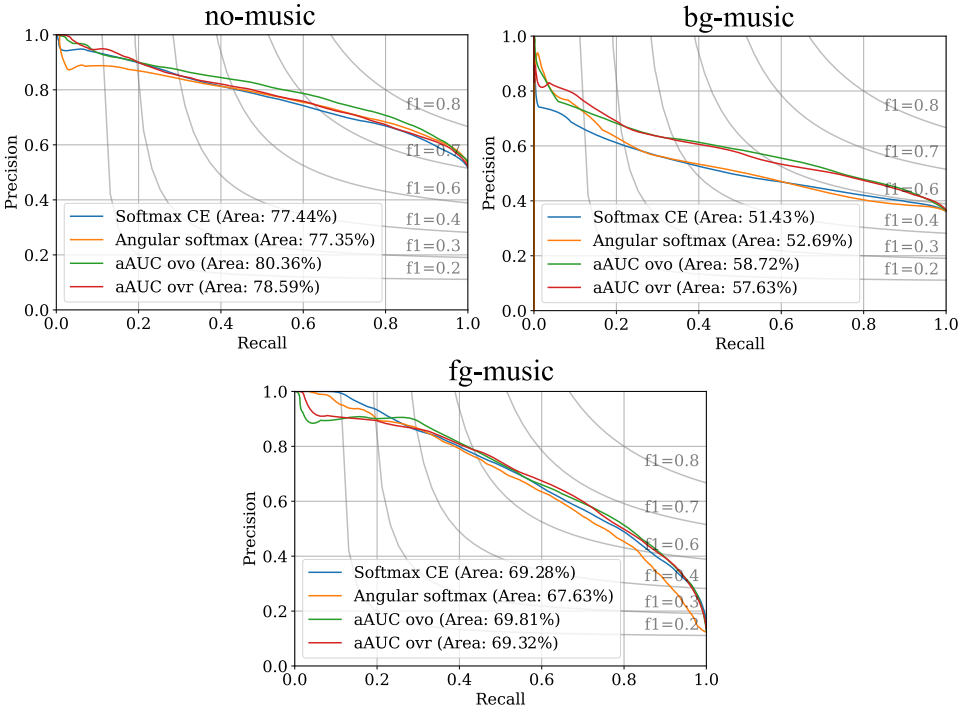


Figure 11.1: Precision versus recall curves, F_1 isocurves, and area under the precision versus recall curve per class on the test data for the proposed multiclass AUC training objectives compared to two variants of cross entropy based training (Average curve obtained over 10 different experiments).

tropy variations presented, showing precision versus recall curves and the corresponding F_1 isocurves. In general, we can see that both of our proposals achieve an improvement in the classification of no music and background music classes, without a significant drop in performance in the foreground music class. The improvement in the background music class is consistent between the OVO and OVR approach (around 14% average relative improvement compared to cross entropy training in area under the precision versus recall curve). However, the OVO approach is able to obtain better results on the no music class (around 4% average relative improvement compared to cross entropy training in area under the precision versus recall curve), achieving a better overall result and matching the metrics shown in Table 11.1. It can be observed again that, when comparing softmax CE and angular softmax losses, the last one yields similar results to softmax CE training with a minor overall drop in performance.

Finally, Table 11.2 shows the results obtained by the final classification labels in terms of overall accuracy and per class precision, recall and F_1 for our proposed multiclass AUC training objectives compared to both cross entropy variants. In this case, the AUC optimisation techniques derive in an average relative improvement of 14.03% and 8.77% respectively for the OVO and OVR approach, compared to softmax CE in terms of overall accuracy. Again, the OVO approach seems to outperform the OVR approach. This suggests that, in general terms, the combination of pairs of classes performed in the OVO

Table 11.2: Final classification overall accuracy, precision, recall and F_1 per class on the test data for the audio segmentation systems trained using the proposed multiclass AUC function losses compared to two variants of cross entropy based training (Mean ± standard deviation over 10 different experiments).

Training objective	**Overall accuracy**
Softmax CE	0.57±0.03
Angular softmax	0.56±0.03
$aAUC_{OVO}$	0.65±0.02
$aAUC_{OVR}$	0.62±0.02

Training objective	**no-music**		
	Precision	**Recall**	**F_1**
Softmax CE	0.76±0.04	0.46±0.05	0.57±0.04
Angular softmax	0.71±0.04	0.48±0.03	0.57±0.03
$aAUC_{OVO}$	0.81±0.03	0.70±0.03	0.75±0.03
$aAUC_{OVR}$	0.80±0.03	0.51±0.03	0.63±0.02

Training objective	**bg-music**		
	Precision	**Recall**	**F_1**
Softmax CE	0.49±0.02	0.69±0.06	0.57±0.03
Angular softmax	0.51±0.03	0.63±0.05	0.56±0.04
$aAUC_{OVO}$	0.50±0.04	0.72±0.05	0.59±0.03
$aAUC_{OVR}$	0.49±0.02	0.70±0.04	0.58±0.02

Training objective	**fg-music**		
	Precision	**Recall**	**F_1**
Softmax CE	0.66±0.06	0.63±0.04	0.63±0.03
Angular softmax	0.60±0.04	0.64±0.03	0.62±0.04
$aAUC_{OVO}$	0.54±0.03	0.67±0.04	0.60±0.03
$aAUC_{OVR}$	0.58±0.02	0.71±0.05	0.61±0.03

solution is more powerful than the binary transformation performed in the OVR solution. It can be observed that this boost in performance comes mainly from the increment in the F_1 metric for the no music class that rises from an average value of 0.57 in the softmax CE training to 0.75 in the $aAUC_{OVO}$ training. Angular softmax shows a similar trend as the one observed in the previous results, with a performance slightly below softmax CE training and underperforming our proposed multiclass AUC loss functions.

11.5 Conclusions

In this chapter, a generalisation of the AUC optimisation framework that can be applied to an arbitrary number of classes has been introduced and validated experimentally. De-

rived from the presented multiclass extensions of the AUC metric, we introduce two training objectives based on a one-versus-one and a one-versus-rest approach respectively. The first one extends the AUC metric to the multiclass domain by averaging pairwise comparisons of classes, while the second one defines a multiclass AUC metric as the mean of c binary AUC metrics. For both cases, extending the optimisation framework to the multiclass domain implies computing a number of AUC metrics using different sets of scores appropriately. In order to obtain a differentiable expression for these AUC computations and following the same approach described in Chapter 10, a sigmoid approximation is applied. The formulation of the multiclass AUC loss functions is straightforward then, being the average of different binary AUC sigmoid approximations using the different sets of target and non-target scores as described in the multiclass AUC metrics.

This proposal is evaluated in a 3 class audio segmentation task that aims to separate foreground music, background music and no music using an experimental setup based on recurrent neural networks with a limited training dataset consisting only of around 20 hours of audio. Despite the validity of the theoretical framework presented, further research would be needed to explore the possible effects of using a larger number of classes on the performance of the proposed training objectives.

Experimental results suggest that the multiclass AUC optimisation outperforms traditional training objectives such as softmax cross entropy or angular softmax, under the circumstances of a limited training dataset being used. In particular, for this task, we report an average relative improvement close to 14% in terms of overall accuracy using our proposed aAUC_{OVO} training criterion. Comparing the two different approaches considered, even though both of them outperform cross entropy criterion, the OVO approach outperforms the OVR approach in all of our experiments. This fact suggests that the combination of pairs of classes is a more robust training criterion than the use of one-versus-rest binarisation solutions.

Part V

Conclusions

"Genius begins great works; labour alone finishes them."

Joseph Joubert

12

Conclusions & future lines

12.1 Conclusions

The application of deep learning solutions in order to automatically extract information from audio signals has become a common practice among the audio processing community. Current technology capabilities support its use in industry and other environments outside laboratory conditions. Even tough deep learning research has observed huge advances in recent years, several questions still need further research. Within the different chapters of this thesis dissertation, a study of several audio segmentation tasks has been presented, exploring its current limitations for different scenarios. This study has mainly focused on two important topics: data availability and generalisation. For the first one, part of the work presented in this thesis has investigated ways to improve the performance of audio segmentation systems even when the training datasets are limited in size. Concerning generalisation, some experiments performed aimed to train robust audio segmentation models that can work in different domain conditions.

Research efforts presented in this thesis dissertation have been centred around three main areas: speech activity detection in challenging environments, multiclass audio segmentation, and AUC optimisation for audio segmentation. In the following lines a summary of the main conclusions obtained for each of them is provided.

12.1.1 Speech activity detection in challenging environments

The first part of this thesis has been devoted to the SAD task, considering the challenging domain proposed in the Fearless steps challenge with audio from Apollo space missions. Motivated by the release of new data in the second phase of the challenge that allowed the development of supervised deep learning solutions, initial experiments performed in Chapter 5 explored the use CRNN models in the context of the SAD task. In addition to several CRNN systems using 1D and 2D filters in the convolutional stage, a novel architecture is proposed that combines information from both kind of filters in an intermediate feature space. The results obtained using the described CRNN models largely outperform the baseline provided by the challenge organisation. Performance observed for these models is in line with previous results in the literature where 2D convolutional stages achieved better results than 1D filters. Moreover, experimental results hint that combining the information provided by 1D and 2D filters is beneficial for the SAD system, showing the best results in the development and evaluation sets. In terms of the evaluation metric, competitive results in the challenge were obtained, with the best submission ranking seventh among the 28 submissions to the challenge SAD task, and fourth among the 7 participant teams.

With the introduction of Apollo space missions audio data by the Fearless steps challenge, the need for methods to transfer knowledge from other domains to new environments became apparent. The research work described in Chapter 6 evaluates the capabilities of a SAD system trained on common datasets to perform on a new domain such as the one presented in the Fearless steps challenge. Different methods to adapt a baseline model to this new domain are tested. Furthermore, this adaptation is done in a fully unsupervised way, increasing its practical application in several scenarios. Experimental results suggest that methods minimising statistical distribution shift between source and target domains are the most promising. In particular, the Deep CORAL method provides a significant improvement in the original challenge evaluation metric. In addition, the cascaded application of Deep CORAL and pseudo-labelling reports the best result in this analysis, suggesting that both techniques may be complementary. The first one provides an overall improvement in the DET curve, while the second one improves the AUC value by modifying the DET curve in areas with high FPR and FNR values. In general terms, the improvement in performance observed can substantially reduce the gap between a system trained using in domain data and a system adapted in an completely unsupervised way.

Novel advances in self-supervised representation learning showed improvements in a number of speech processing tasks under a variety of domains. Motivated by this fact, the work detailed in Chapter 7 incorporates these advances to be applied in audio seg-

mentation systems, namely in the SAD task. These methods are used with the goal of solving possible mismatches in test data from new datasets released for the Fearless steps challenge phase III and IV from years 2021 and 2022 respectively. Two different systems were considered in our submissions: in the 2021 edition, an approach inspired on the wav2vec model is followed. This solution is based on a contrastive loss that learns a feature representation combining a strided CNN encoder with a RNN in order to provide a context embedding. Furthermore, for the 2022 edition recent advances in self-supervised learning were also evaluated by using wavLM models in the context of the SAD task. In both cases, the representations obtained are then used as input to train different variants of a CRNN architecture. For both editions of the challenge considered, empirical findings indicate that recent advances in self-supervised feature learning allow to develop audio segmentation systems that are significantly less sensitive to domain mismatch when compared to systems using traditional perceptual features such as log Mel filter-bank energies.

12.1.2 Multiclass audio segmentation

In the second part of this dissertation, the focus has been set on a more general audio segmentation task seeking to separate speech, music, noise and a combination of them. This task may be specially relevant from the perspective of audio information retrieval, combining in a single system two binary audio segmentation systems such as speech and music detection, and going beyond that description modelling the possible combinations of both classes. Chapter 8 introduced an approach to the multiclass audio segmentation task proposed by the Albayzín 2010 evaluation that is based on the use of recurrent neural networks. These RNNs are combined with an HMM resegmentation module as a smoothing technique in order to correct possible errors due to spurious transitions in the neural network scores. A new block is proposed for the neural architecture seeking to remove redundant temporal information. By applying a temporal pooling layer, the number of operations per second is reduced without increasing the number of parameters of the model. Also, mixup data agnostic augmentation technique is applied to improve the model generalisation capabilities without adding external data sources. The set of experiments presented in this chapter showed that time pooling architectures can be used between two BLSTM layers to remove redundant temporal information, improving the overall performance of the segmentation system. The improvement observed when using mixup augmentation was specially significant for classes that can be modelled as the combination of two classes. In general terms, competitive results were obtained with the RNN based approach, outperforming the best results in the literature so far and a similar approach based on DNN proposed for the DCASE sound environment detection task.

Results from Chapter 7 showed the potential of self-supervised representation learning in a binary audio segmentation task such as SAD. Aiming to obtain a wider perspective of these techniques in audio segmentation applications, the work described in Chapter 9 explored representation learning in the context of multiclass audio segmentation tasks. For that, an analysis of different ways to benefit from these abstract representations is performed, with an special focus on remarking its possible limitations when representing

different acoustic classes simultaneously. While experiments on Albayzín 2010 dataset from Chapter 8 relied on a combination of different sets of features, the approach followed in this chapter aims to replace these features with representations obtained through self-supervised learning. Namely, an approach inspired by wav2vec models is evaluated as feature extractor under different training conditions and using different models as classifiers. A general trend can be observed in results for wav2vec representations leading to a significant improvement in the performance of audio segmentation system for classes containing speech. Yet, the opposite behaviour occurs for the segmentation of isolated music. Any configuration evaluated using wav2vec representation was able to improve the results for the baseline features in the music class. This fact limits the overall improvement observed when using wav2vec features in terms of average class error.

12.1.3 AUC optimisation for audio segmentation

The third and last part of this manuscript has investigated the application of new training objectives in audio segmentation with the main goal of mitigating the issues derived from a limited training dataset. The AUC metric serves as an overall description of the detection performance for all possible thresholds. In the context of metric learning techniques, a number of works have already proposed to train a classifier using a loss function based on the AUC metric. Additionally, partial AUC only measures the area under the ROC curve that is restricted between two false positive rates. Using this generalisation as a loss function may be interesting in applications with large differences in cost between false positives and false negatives. The first approximation to AUC optimisation is introduced in Chapter 10, that presents an optimisation framework based on deep neural networks applied to the music detection task. The experimental setup proposed consists of a classifier based on recurrent neural networks, that is trained with a limited training dataset consisting only of around 20 hours of audio. With the introduction of these new optimisation techniques, the aim was to increase the performance of music detection in this context of limited training data. Findings extracted from the set of experiments described in this chapter suggest that AUC and partial AUC optimisation techniques both outperform traditional training objectives such as cross entropy. Particularly, a performance better than the one obtained using a cross entropy loss function is observed for any of the AUC based loss functions evaluated, with the best performance obtained by the system using the proposed aAUCsum loss function. These results correlate with different published studies dealing with well-known speech processing tasks affirming that pAUC optimisation techniques report better performance than AUC based optimisation techniques.

Most of the works presented so far that aim to optimise the AUC metric using a deep learning framework are limited to binary tasks, as this is the usual situation when applying the AUC metric. Part of this dissertation has described in Chapters 8 and 9 how multiclass tasks pose new challenges for audio segmentation systems. Furthermore, the problem of data scarcity is even more relevant for multiclass audio segmentation systems, using detailed taxonomies that are more complex than using a binary labelling schema. Considering these ideas, it would be interesting to be able to benefit from the improve-

ment in performance observed when using AUC optimisation techniques in audio segmentation tasks with more than two classes. As a natural extension of the work from Chapter 10, Chapter 11 goes beyond binary AUC optimisation techniques to propose a generalisation of these techniques so that they can be applied to an arbitrary number of classes. For that, multiclass variations of the AUC metric proposed in the literature are used as the starting point for the formulation. Namely, two training objectives were proposed: one that is based on a one-versus-one approach, and one that is based on a one-versus-rest approach. Experimental results suggest that, under the circumstances of a limited training dataset being used, multiclass AUC optimisation outperforms traditional training objectives such as softmax cross entropy or angular softmax. Concerning the two approaches tested, even though both of them outperform cross entropy criterion, in all the experiments conducted the OVO approach provides consistently better results than the OVR approach. This hints that the combination of pairs of classes may be a more robust training criterion than using one-versus-rest binarisation solutions.

12.2 Scientific contributions

The research work carried out along this thesis has produced multiple contributions to peer-review journals and conference proceedings, involving the different research lines presented. These contributions are listed in the following lines:

12.2.1 Journal articles

- **P. Gimeno**, I. Viñals, A. Ortega, A. Miguel, and E. Lleida, "Multiclass audio segmentation based on recurrent neural networks for broadcast domain data," *EURASIP Journal on Audio, Speech, and Music Processing*, 2020, vol. 2020, no. 1, pp. 1–19.
- **P. Gimeno**, V. Mingote, A. Ortega, A. Miguel, and E. Lleida, "Generalising AUC optimisation to multiclass classification for audio segmentation with limited training data," *IEEE Signal Processing Letters*, 2021, vol. 28, pp. 1135–1139.
- V. Mingote, I. Viñals, **P. Gimeno**, A. Miguel, A. Ortega, and E. Lleida, "Multimodal diarization systems by training enrollment models as identity representations," *Applied Sciences*, 2022, vol. 12, no. 3, pp. 1141.
- **P. Gimeno**, D. Ribas, A. Ortega, A. Miguel, and E. Lleida, "Unsupervised adaptation of deep speech activity detection models to unseen domains," *Applied Sciences*, 2022, vol. 12, no. 4, pp. 1832.

12.2.2 Conference proceedings

- I. Viñals, **P. Gimeno**, A. Ortega, A. Miguel, and E. Lleida, "Estimation of the number of speakers with variational bayesian PLDA in the DIHARD diarization challenge.," in *Proc. ISCA Interspeech*, 2018, pp. 2803–2807.

- **P. Gimeno**, I. Viñals, A. Ortega, A. Miguel, and E. Lleida, "A recurrent neural network approach to audio segmentation for broadcast domain data," in *Proc. IberSPEECH*, 2018, pp. 87–91.
- **P. Gimeno**, A. Ortega, A. Miguel, and E. Lleida, "Partial AUC optimisation using recurrent neural networks for music detection with limited training data," in *Proc. ISCA Interspeech*, 2020, pp. 3067–3071.
- **P. Gimeno**, D. Ribas, A. Ortega, A. Miguel, and E. Lleida, "Convolutional recurrent neural networks for speech activity detection in naturalistic audio from Apollo missions," in *Proc. IberSPEECH*, 2021, pp. 26–30.
- V. Mingote, I. Viñals, **P. Gimeno**, A. Miguel, A. Ortega, and E. Lleida, "ViVoLAB multimodal diarization system for RTVE 2020 challenge," in *Proc. IberSPEECH*, 2021, pp. 76–80.
- I. Viñals, **P. Gimeno**, A. Ortega, A. Miguel, and E. Lleida, "Diarization and identity attribution compatibility in the Albayzin 2020 challenge," in *Proc. IberSPEECH*, 2021, pp. 94–98.
- **P. Gimeno**, A. Ortega, A. Miguel, and E. Lleida, "Unsupervised representation learning for speech activity detection in the Fearless Steps challenge 2021," in *Proc. ISCA Interspeech*, 2021, pp. 4359–4363.
- **P. Gimeno**, A. Ortega, A. Miguel, and E. Lleida, "A study on the use of wav2vec representations for multiclass audio segmentation," in *Proc. IberSPEECH*, 2022, pp. 56–60.

12.3 Future lines

This thesis dissertation has presented different approaches to improve the performance of deep learning based audio segmentation systems under a variety of scenarios. Despite the fact that presented solutions showed gains in both binary and multiclass segmentation tasks, there still exist a number of limitations that may be further investigated in future works. In the following lines we describe some possible future research lines that could be studied in order to solve some of the challenges described in this thesis dissertation.

- Recent research has shown the great performance of self attention architectures in different audio processing tasks. While work developed in this dissertation has focused mainly on other structures such as RNN and CRNN models, transformer models are also being introduced in different audio segmentation tasks with promising results [67]. Data conditions explored in this dissertation deal with limited training datasets in some cases. A number of studies have demonstrated that transformer architectures need much more data than previous approaches such as CNN models for training [234]. Under these circumstances, it would be interesting to apply a reverse approach to current transformer experimentation, where the model

should be scaled down instead of scaled up [235]. In this context, future work may explore the application of mentioned transformer architectures for audio segmentation tasks, with the goal of evaluating relatively small datasets.

- Domain adaptation techniques evaluated in Chapter 6 were able to reduce the mismatch in performance observed when evaluating a SAD model on a different domain from the one that it was trained on. In addition, this was done in a completely unsupervised way, without the need of labels in the new domain. Domain adaptation methods that minimise the statistical distribution shift between source and target domains, such as Deep CORAL, showed one of the most promising results. Some recent work has introduced the use of higher-order statistics for unsupervised domain adaptation [236] [237], generalising the idea presented in Deep CORAL as an arbitrary order moment matching technique. Some future work lines may point in this direction, applying this idea to the SAD task.

- The combination of RNN classifiers and an HMM resegmentation module achieved state-of-the-art results in the Albayzín 2010 multiclass segmentation task presented in Chapter 8. Through the introduction of temporal pooling techniques in the RNN classifier, results were improved without increasing the complexity of the model and reducing the number of operations per seconds. However, temporal pooling techniques in the RNN classifier are still underperforming when compared to our proposed HMM resegmentation module. Results presented in this dissertation provide an interesting insight into the introduction of pure DNN smoothing in audio segmentation tasks. Yet further research is needed on sequence compression techniques to fill the gap between HMM and DNN smoothing. While techniques evaluated in this thesis such as average pooling or max pooling are fixed length subsampling techniques, some works have presented new approaches that rely on variable length subsampling to reduce the temporal information of a sequence [238] [239]. Future work could evaluate these recent advances in variable length subsampling in audio segmentation tasks.

- The limitations of current self-supervised representation learning models when used in multiclass segmentation environments was shown in experiments described in Chapter 9. Evaluated wav2vec representations demonstrated significant improvements in performance for classes containing speech. However, these wav2vec representations underperform traditional features in the segmentation of isolated music. This problem is partially mitigated by eliminating standard augmentation techniques in the training process, but even in the best case a poor performance is observed for the music class. Further research should investigate the generation of robust audio representations, capable of dealing with speech and music simultaneously. This topic is already an active research line with several researchers working towards that direction. For instance, work developed in the context of the HEAR challenge [240] seeks to develop a general purpose audio representation that provides a strong basis for learning in a wide variety of tasks and scenarios. Some recent models such as BYOL [241] have developed new learning methods to

obtain general audio representations that can be applied to different downstream tasks.

- The generalisation of AUC optimisation techniques for multiclass classification presented in Chapter 11 can be theoretically applied to an arbitrary number of classes. Experimental validation of this proposal was performed in the context of an audio segmentation task that needs to separate three acoustic classes. In order to compute these new loss functions a number of binary AUC metric need to be obtained first, increasing computational complexity linearly with the number of classes in the case of the OVR approach, and quadratically in the case of the OVO approach. Further research should evaluate the mentioned training objectives in other multiclass audio segmentation tasks, focusing on investigating how to efficiently scale them to tasks that involve a larger number of classes.

References

[1] YouTube LLC, "YouTube copyright transparency report H1 2022," https://storage.googleapis.com/transparencyreport/report-downloads/pdf-report-22_2022-1-1_2022-6-30_en_v1.pdf, September 2022. [pg. 3]

[2] Spotify AB, "Spotify shareholder deck Q3 2022," https://s29.q4cdn.com/175625835/files/doc_financials/2022/q3/Q3-2022-Shareholder-Deck-FINAL-LOCKED.pdf, October 2022. [pg. 3]

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. [pg. 4]

[4] J. J. Burred and A. Lerch, "Hierarchical automatic audio signal classification," *Journal of the Audio Engineering Society*, vol. 52, no. 7/8, pp. 724–739, 2004. [pg. 5]

[5] M. Casey, "General sound classification and similarity in MPEG-7," *Organised Sound*, vol. 6, no. 2, pp. 153–164, 2001. [pg. 5]

[6] E. Alexandre, R. Gil-Pita, L. Cuadra, L. Alvarez, and M. Rosa-Zurera, "Speech/music/noise classification in hearing aids using a two-layer classification system with MSE linear discriminants," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2008, pp. 1–5. [pg. 5]

[7] P. Del Moral, S. Nowaczyk, and S. Pashami, "Why is multiclass classification hard?" *IEEE Access*, vol. 10, pp. 80 448–80 462, 2022. [pg. 12]

[8] J. Ramirez, J. M. Gorriz, and J. C. Segura, "Voice activity detection. fundamentals and speech recognition system robustness," in *Robust Speech Recognition and Understanding*, M. Grimm and K. Kroschel, Eds. IntechOpen, 2007, ch. 1. [pg. 12]

[9] P. Lopez-Otero, L. Docio-Fernandez, and C. Garcia-Mateo, "Introducing a framework for the evaluation of music detection tools," in *Proc. International Conference on Language Resources and Evaluation (LREC)*, 2014, pp. 568–572. [pg. 12]

[10] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012. [pg. 12]

[11] T. Butko and C. Nadeu, "Audio segmentation of broadcast news in the Albayzín-2010 evaluation: overview, results, and discussion," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2011, no. 1, pp. 1–10, 2011. [pg. 13], [pg. 41], [pg. 42], [pg. 71], [pg. 100], [pg. 103]

[12] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021. [pg. 13]

[13] B. Gygi and V. Shafiro, "Environmental sound research as it stands today," in *Proc. ASA Meetings on Acoustics*, vol. 1, 2007, p. 050002. [pg. 13]

[14] J. Abeßer, "A review of deep learning based methods for acoustic scene classification," *Applied Sciences*, vol. 10, no. 6, p. 2020, 2020. [pg. 13]

[15] S. Duan, J. Zhang, P. Roe, and M. Towsey, "A survey of tagging techniques for music, speech and environmental sound," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 637–661, 2014. [pg. 13]

[16] T. Theodorou, I. Mporas, and N. Fakotakis, "An overview of automatic audio segmentation," *International Journal of Information Technology and Computer Science (IJITCS)*, vol. 6, no. 11, p. 1, 2014. [pg. 13]

[17] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics*, pp. 461–464, 1978. [pg. 14]

[18] S. Chen and P. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *Proc. DARPA broadcast news transcription and understanding workshop*, vol. 8, 1998, pp. 127–132. [pg. 15]

[19] C.-H. Wu, Y.-H. Chiu, C.-J. Shia, and C.-Y. Lin, "Automatic segmentation and identification of mixed-language speech using delta-BIC and LSA-based GMMs," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 266–276, 2005. [pg. 15]

[20] M. Kotti, E. Benetos, and C. Kotropoulos, "Computationally efficient and robust BIC-based speaker segmentation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 5, pp. 920–933, 2008. [pg. 15]

[21] J. Neyman and E. S. Pearson, "On the problem of the most efficient tests of statistical hypotheses," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 231, no. 694-706, pp. 289–337, 1933. [pg. 15]

[22] D. Wang, R. Vogt, M. Mason, and S. Sridharan, "Automatic audio segmentation using the generalized likelihood ratio," in *Proc, IEEE International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2008, pp. 1–5. [pg. 15]

[23] A. Dessein and A. Cont, "An information-geometric approach to real-time audio segmentation," *IEEE Signal Processing Letters*, vol. 20, no. 4, pp. 331–334, 2013. [pg. 15]

[24] J. T. Foote and M. L. Cooper, "Media segmentation using self-similarity decomposition," in *Storage and Retrieval for Media Databases 2003*, vol. 5021. SPIE, 2003, pp. 167–175. [pg. 15]

[25] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Proc, IEEE International Conference on Multimedia and Expo (ICME)*, vol. 1, 2000, pp. 452–455. [pg. 15]

[26] H. Meinedo and J. Neto, "A stream-based audio segmentation, classification and clustering pre-processing system for broadcast news using ANN models," in *Proc. ISCA Interspeech*, 2005, pp. 237–240. [pg. 15], [pg. 18]

[27] R. Yin, H. Bredin, and C. Barras, "Speaker change detection in broadcast TV using bidirectional long short-term memory networks," in *Proc. ISCA Interspeech*, 2017, pp. 3827–3831. [pg. 15]

[28] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 1996–2000. [pg. 15]

[29] M. Kos, M. Grasic, D. Vlaj, and Z. Kacic, "On-line speech/music segmentation for broadcast news domain," in *Proc. International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2009, pp. 1–4. [pg. 16]

[30] A. Misra, "Speech/nonspeech segmentation in web videos," in *Proc. ISCA Interspeech*, 2012, pp. 1977–1980. [pg. 16]

[31] T. Zhang and C.-C. Kuo, "Hierarchical classification of audio data for archiving and retrieving," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1999, pp. 3001–3004. [pg. 16]

[32] J. Ajmera, I. McCowan, and H. Bourlard, "Speech/music segmentation using entropy and dynamism features in a HMM classification framework," *Speech communication*, vol. 40, no. 3, pp. 351–363, 2003. [pg. 16]

[33] E. Didiot, I. Illina, D. Fohr, and O. Mella, "A wavelet-based parameterization for speech/music discrimination," *Computer Speech & Language*, vol. 24, no. 2, pp. 341–357, 2010. [pg. 16]

[34] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, vol. 14, no. 28-29, p. 2, 2005. [pg. 16]

[35] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000. [pg. 17]

[36] N. Dehak, P. Kenny, R. Dehak, O. Glembek, P. Dumouchel, L. Burget, V. Hubeika, and F. Castaldo, "Support vector machines and joint factor analysis for speaker verification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 4237–4240. [pg. 17]

[37] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007. [pg. 17]

[38] D. Castán, A. Ortega, A. Miguel, and E. Lleida, "Audio segmentation-by-classification approach based on factor analysis in broadcast news domain," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2014, no. 1, pp. 1–13, 2014. [pg. 17], [pg. 114], [pg. 115]

[39] D. Castan, A. Ortega, J. Villalba, A. Miguel, and E. Lleida, "Segmentation-by-classification system based on factor analysis," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 783–787. [pg. 17]

[40] G. Richard, M. Ramona, and S. Essid, "Combined supervised and unsupervised approaches for automatic segmentation of radiophonic audio streams," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007, pp. 461–464. [pg. 17]

[41] Y. Patsis and W. Verhelst, "A speech/music/silence/garbage/classifier for searching and indexing broadcast news material," in *Proc. International Workshop on Database and Expert Systems Applications (DEXA)*, 2008, pp. 585–589. [pg. 17]

[42] M. Liu, C. Wan, and L. Wang, "Content-based audio classification and retrieval using a fuzzy logic system: towards multimedia search engines," *Soft Computing*, vol. 6, no. 5, pp. 357–364, 2002. [pg. 17]

[43] S. Ganapathy, P. Rajan, and H. Hermansky, "Multi-layer perceptron based speech activity detection for speaker verification," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2011, pp. 321–324. [pg. 18]

[44] J.-s. Keum and H.-s. Lee, "Speech/music discrimination based on spectral peak analysis and multi-layer perceptron," in *Proc. International Conference on Hybrid Information Technology (ICHIT)*, vol. 2, 2006, pp. 56–61. [pg. 18]

[45] F. Eyben, F. Weninger, S. Squartini, and B. Schuller, "Real-life voice activity detection with LSTM recurrent neural networks and an application to Hollywood movies," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 483–487. [pg. 18], [pg. 100]

[46] J. Kim, J. Kim, S. Lee, J. Park, and M. Hahn, "Vowel based voice activity detection with LSTM recurrent neural network," in *Proc. International Conference on Signal Processing Systems (ICSPS)*, 2016, p. 134–137. [pg. 18], [pg. 100]

[47] Y.-H. Wang, C.-T. Chung, and H.-Y. Lee, "Gate activation signal analysis for gated recurrent neural networks and its correlation with phoneme boundaries," in *Proc. ISCA Interspeech*, 2017, pp. 3822–3826. [pg. 18]

[48] Z. Chen, X. Zhang, J. Deng, J. Li, Y. Jiang, and W. Li, "A practical singing voice detection system based on GRU-RNN," in *Proc. Conference on Sound and Music Technology (CSMT)*, 2019, pp. 15–25. [pg. 18]

[49] S. Venkatesh, D. Moffat, and E. R. Miranda, "You only hear once: A YOLO-like algorithm for audio segmentation and sound event detection," *Applied Sciences*, vol. 12, no. 7, 2022. [pg. 18]

[50] B.-Y. Jang, W.-H. Heo, J.-H. Kim, and O.-W. Kwon, "Music detection from broadcast contents using convolutional neural networks with a Mel-scale kernel," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2019, no. 1, pp. 1–12, 2019. [pg. 18]

[51] D. Doukhan and J. Carrive, "Investigating the use of semi-supervised convolutional neural network models for speech/music classification and segmentation," in *Proc. International Conferences on Advances in Multimedia (MMEDIA)*, 2017. [pg. 18]

[52] S. Thomas, S. Ganapathy, G. Saon, and H. Soltau, "Analyzing convolutional neural networks for speech activity detection in mismatched acoustic conditions," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 2519–2523. [pg. 18]

[53] T. Sainath, R. J. Weiss, K. Wilson, A. W. Senior, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *Proc. ISCA Interspeech*, 2015, pp. 1–5. [pg. 18]

[54] P. Ghahremani, V. Manohar, D. Povey, and S. Khudanpur, "Acoustic modelling from the signal domain using CNNs," in *Proc. ISCA Interspeech*, 2016, pp. 3434–3438. [pg. 18]

[55] S. Venkatesh, D. Moffat, and E. R. Miranda, "Investigating the effects of training set synthesis for audio segmentation of radio broadcast," *Electronics*, vol. 10, no. 7, 2021. [pg. 18]

[56] D. de Benito-Gorron, A. Lozano-Diez, D. T. Toledano, and J. Gonzalez-Rodriguez, "Exploring convolutional, recurrent, and hybrid deep neural networks for speech and music detection in a large audio dataset," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2019, no. 1, pp. 1–18, 2019. [pg. 18]

[57] A. Vafeiadis, E. Fanioudakis, I. Potamitis, K. Votis, D. Giakoumis, D. Tzovaras, L. Chen, and R. Hamzaoui, "Two-dimensional convolutional recurrent neural networks for speech activity detection," in *Proc. ISCA Interspeech*, 2019, pp. 2045–2049. [pg. 18], [pg. 54], [pg. 58]

[58] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *Proc. European Conference on Computer Vision (ECCV)*, 2016, pp. 47–54. [pg. 19]

[59] Q. Lemaire and A. Holzapfel, "Temporal convolutional networks for speech and music detection in radio broadcast," in *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, 2019, p. 229–236. [pg. 19]

[60] B. Meléndez-Catalán, E. Molina, and E. Gómez, "Relative music loudness estimation using temporal convolutional networks and a CNN feature extraction front-end," in *Proc. International Conference on Digital Audio Effects (DAFx)*, vol. 5, 2020, pp. 273–280. [pg. 19]

[61] K. Guirguis, C. Schorn, A. Guntoro, S. Abdulatif, and B. Yang, "SELD-TCN: Sound event localization & detection via temporal convolutional networks," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2021, pp. 16–20. [pg. 19]

[62] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. International Conference on Learning Representations (ICLR)*, 2015. [pg. 19], [pg. 31]

[63] J. Kim and M. Hahn, "Voice activity detection using an adaptive context attention model," *IEEE Signal Processing Letters*, vol. 25, no. 8, pp. 1181–1185, 2018. [pg. 19]

[64] T. Fernando, H. Ghaemmaghami, S. Denman, S. Sridharan, N. Hussain, and C. Fookes, "Heart sound segmentation using bidirectional LSTMs with attention," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 6, pp. 1601–1609, 2020. [pg. 19]

[65] Y.-H. Shen, K.-X. He, and W.-Q. Zhang, "Learning how to listen: A temporal-frequential attention model for sound event detection," in *Proc. ISCA Interspeech*, 2019, pp. 2563–2567. [pg. 19]

[66] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017. [pg. 19], [pg. 31]

[67] Y. R. Jo, Y. K. Moon, W. I. Cho, and G. S. Jo, "Self-attentive VAD: Context-aware detection of voice from noise," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 6808–6812. [pg. 19], [pg. 160]

[68] Y. Zhao and B. Champagne, "An efficient transformer-based model for voice activity detection," in *Proc. IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2022, pp. 1–6. [pg. 19]

[69] Y. C. Liu, E. Han, C. Lee, and A. Stolcke, "End-to-End Neural Diarization: From Transformer to Conformer," in *Proc. ISCA Interspeech*, 2021, pp. 3081–3085. [pg. 19]

[70] S. Liu, A. Mallol-Ragolta, E. Parada-Cabaleiro, K. Qian, X. Jing, A. Kathan, B. Hu, and B. W. Schuller, "Audio self-supervised learning: A survey," *Patterns*, vol. 3, no. 12, p. 100616, 2022. [pg. 19]

[71] A. Mohamed, H.-y. Lee, L. Borgholt, J. D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S.-W. Li, K. Livescu, L. Maaløe, T. N. Sainath, and S. Watanabe, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1179–1210, 2022. [pg. 19]

[72] M. Lebourdais, M. Tahon, A. Laurent, and S. Meignier, "Overlapped speech and gender detection with wavLM pre-trained features," in *Proc. ISCA Interspeech*, 2022, pp. 5010–5014. [pg. 19], [pg. 121]

[73] S. Alisamir, F. Ringeval, and F. Portet, "Cross-domain voice activity detection with self-supervised representations," *arXiv preprint arXiv:2209.11061*, 2022. [pg. 19], [pg. 121]

[74] P. Singh and S. Ganapathy, "Self-supervised representation learning with path integral clustering for speaker diarization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1639–1649, 2021. [pg. 19]

[75] J. Zhou, J. Wang, J. Zhang, W. Sun, J. Zhang, S. Birchfield, D. Guo, L. Kong, M. Wang, and Y. Zhong, "Audio-visual segmentation," in *Proc. European Conference on Computer Vision (ECCV)*, 2022. [pg. 20]

[76] V. Mingote, I. Viñals, P. Gimeno, A. Miguel, A. Ortega, and E. Lleida, "Multimodal diarization systems by training enrollment models as identity representations," *Applied Sciences*, vol. 12, no. 3, 2022. [pg. 20]

[77] E. Z. Xu, Z. Song, S. Tsutsui, C. Feng, M. Ye, and M. Z. Shou, "AVA-AVD: Audio-visual speaker diarization in the wild," in *Proc. ACM International Conference on Multimedia (ICM)*, 2022, p. 3838–3847. [pg. 20]

[78] S. Khurana, A. Laurent, and J. Glass, "SAMU-XLSR: Semantically-aligned multimodal utterance-level cross-lingual speech representation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1493–1504, 2022. [pg. 20]

[79] Z. Chen, Y. Zhang, A. Rosenberg, B. Ramabhadran, P. J. Moreno, A. Bapna, and H. Zen, "MAESTRO: Matched speech text representations through modality matching," in *Proc. ISCA Interspeech*, 2022, pp. 4093–4097. [pg. 20]

[80] J. Ao, R. Wang, L. Zhou, C. Wang, S. Ren, Y. Wu, S. Liu, T. Ko, Q. Li, Y. Zhang, Z. Wei, Y. Qian, J. Li, and F. Wei, "SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing," in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022, pp. 5723–5738. [pg. 20]

[81] A. Guzhov, F. Raue, J. Hees, and A. Dengel, "Audioclip: Extending clip to image, text and audio," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 976–980. [pg. 20]

[82] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 28, no. 4, pp. 357–366, 1980. [pg. 22]

[83] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990. [pg. 22]

[84] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937. [pg. 23]

[85] T. Fujishima, "Real-time chord recognition of musical sound: a system using common lisp music," in *Proc. International Computer Music Conference (ICMC)*, 1999. [pg. 24], [pg. 104], [pg. 139], [pg. 147]

[86] M. Bartsch and G. Wakefield, "To catch a chorus: using chroma-based representations for audio thumbnailing," in *Proc. IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2001, pp. 15–18. [pg. 24]

[87] R. N. Shepard, "Circularity in judgments of relative pitch," *The Journal of the Acoustical Society of America*, vol. 36, no. 12, pp. 2346–2353, 1964. [pg. 24]

[88] T. Cho and J. P. Bello, "On the relative importance of individual components of chord recognition systems," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 2, pp. 477–492, 2013. [pg. 25]

[89] S. Ewert, M. Muller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 1869–1872. [pg. 25]

[90] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943. [pg. 25]

[91] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017. [pg. 25]

[92] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012. [pg. 25]

[93] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990. [pg. 27]

[94] C. Olah, "Understanding LSTM networks," accessed: 2023-01-03. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ [pg. 28]

[95] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994. [pg. 28]

[96] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [pg. 28], [pg. 34]

[97] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000. [pg. 29], [pg. 86], [pg. 138]

[98] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [pg. 29]

[99] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016. [pg. 30]

[100] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. International Conference on Machine Learning (ICML)*, 2010, pp. 807–814. [pg. 30], [pg. 55], [pg. 72]

[101] N. Chen, S. Watanabe, J. Villalba, P. Żelasko, and N. Dehak, "Non-autoregressive transformer for speech recognition," *IEEE Signal Processing Letters*, vol. 28, pp. 121–125, 2020. [pg. 31]

[102] J. D. M.-W. C. Kenton and L. K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT*, 2019, pp. 4171–4186. [pg. 31], [pg. 87]

[103] S. Alfasly, C. K. Chui, Q. Jiang, J. Lu, and C. Xu, "An effective video transformer with synchronized spatiotemporal and spatial self-attention for action recognition," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022. [pg. 31]

[104] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986. [pg. 33]

[105] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. International Conference on Learning Representations (ICLR)*, 2015. [pg. 34], [pg. 56], [pg. 102]

[106] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013. [pg. 34]

[107] L. R. Rabiner and J. G. Wilpon, "Considerations in applying clustering techniques to speaker-independent word recognition," *The Journal of the Acoustical Society of America*, vol. 66, no. 3, pp. 663–673, 1979. [pg. 34]

[108] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. International Conference on Learning Representations (ICLR)*, 2014. [pg. 34]

[109] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. ISCA Interspeech*, 2019, pp. 146–150. [pg. 34]

[110] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018. [pg. 34], [pg. 35], [pg. 86], [pg. 87], [pg. 122]

[111] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 539–546. [pg. 34]

[112] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607. [pg. 34], [pg. 87]

[113] T. H. Trinh, M.-T. Luong, and Q. V. Le, "Selfie: Self-supervised pretraining for image embedding," *arXiv preprint arXiv:1906.02940*, 2019. [pg. 35]

[114] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019. [pg. 35], [pg. 87]

[115] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 12 449–12 460, 2020. [pg. 35]

[116] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 29, pp. 3451–3460, 2021. [pg. 35]

[117] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu, "w2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2021, pp. 244–250. [pg. 35]

[118] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022. [pg. 35], [pg. 87], [pg. 88], [pg. 128]

[119] S. wen Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K. tik Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H. yi Lee, "SUPERB: Speech processing universal performance benchmark," in *Proc. ISCA Interspeech*, 2021, pp. 1194–1198. [pg. 35]

[120] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology," *Bulletin of the American Mathematical Society*, vol. 73, no. 3, pp. 360–363, 1967. [pg. 35]

[121] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970. [pg. 35]

[122] J. H. Hansen, A. Joglekar, M. C. Shekhar, V. Kothapally, C. Yu, L. Kaushik, and A. Sangwan, "The 2019 inaugural Fearless steps challenge: A giant leap for naturalistic audio," in *Proc. ISCA Interspeech*, 2019, pp. 1851–1855. [pg. 40], [pg. 81]

[123] J. H. Hansen, A. Sangwan, A. Joglekar, A. E. Bulut, L. Kaushik, and C. Yu, "Fearless steps: Apollo-11 corpus advancements for speech technologies from earth to the moon," in *Proc. ISCA Interspeech*, 2018, pp. 2758–2762. [pg. 40]

[124] A. Joglekar, J. H. Hansen, M. C. Shekar, and A. Sangwan, "FEARLESS STEPS challenge (FS-2): Supervised learning with massive naturalistic apollo data," *Proc. ISCA Interspeech*, pp. 2617–2621, 2020. [pg. 40], [pg. 54], [pg. 73]

[125] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, and A. De Prada, "Albayzín 2018 evaluation: the IberSPEECH-RTVE challenge on speech technologies for spanish broadcast media," *Applied sciences*, vol. 9, no. 24, p. 5412, 2019. [pg. 41], [pg. 71], [pg. 123]

[126] P. Bell, M. J. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Wester *et al.*, "The MGB challenge: Evaluating multi-genre broadcast media recognition," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 687–693. [pg. 41], [pg. 71]

[127] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos *et al.*, "The AMI meeting corpus," in *Proc. International Conference on Methods and Techniques in Behavioral Research*, vol. 88, 2005, p. 100. [pg. 41], [pg. 71]

[128] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke *et al.*, "The ICSI meeting corpus," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2003. [pg. 41], [pg. 71]

[129] A. Ortega, A. Miguel, E. Lleida, V. Bazán, C. Pérez, M. Gómez, and A. de Prada, "Albayzín evaluation: IberSPEECH-RTVE 2020 speaker diarization and identity assignment," *Albayzin 2020 evaluation plan*, 2020. [pg. 41], [pg. 71]

[130] A. Canavan, D. Graff, and G. Zipperlen, "Callhome american english speech," *Linguistic Data Consortium*, 1997. [pg. 41], [pg. 71]

[131] A. Ortega, D. Castan, A. Miguel, and E. Lleida, "The Albayzín 2012 audio segmentation evaluation," in *Proc. IberSPEECH*, 2012, pp. 283–289. [pg. 42], [pg. 100], [pg. 103]

[132] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeature speech/music discriminator," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, 1997, pp. 1331–1334. [pg. 43]

[133] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015. [pg. 43], [pg. 87], [pg. 123]

[134] B. Meléndez-Catalán, E. Molina, and E. Gómez, "Open broadcast media audio from TV: A dataset of TV broadcast audio with relative music loudness annotations," *Transactions of the International Society for Music Information Retrieval*, vol. 2, no. 1, 2019. [pg. 43], [pg. 138], [pg. 144], [pg. 147]

[135] K. Seyerlehner, T. Pohle, M. Schedl, and G. Widmer, "Automatic music detection in television productions," in *Proc. International Conference on Digital Audio Effects (DAFx)*, 2007. [pg. 44], [pg. 138]

[136] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006. [pg. 48], [pg. 144]

[137] N. Brümmer and J. Du Preez, "Application-independent evaluation of speaker detection," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 230–275, 2006. [pg. 49]

[138] NIST, "The 2009 (RT-09) rich transcription meeting recognition evaluation plan," Melbourne 28-29 May 2009. [pg. 49]

[139] S. V. Gerven and F. Xie, "A comparative study of speech detection methods," in *Proc. European Conference on Speech Communication and Technology*, 1997. [pg. 54]

[140] J.-C. Junqua, B. Mak, and B. Reaves, "A robust algorithm for word boundary detection in the presence of noise," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 3, pp. 406–412, 1994. [pg. 54]

[141] J.-H. Chang, N. S. Kim, and S. K. Mitra, "Voice activity detection based on multiple statistical models," *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 1965–1976, 2006. [pg. 54]

[142] T. Ng, B. Zhang, L. Nguyen, S. Matsoukas, X. Zhou, N. Mesgarani, K. Veselỳ, and P. Matějka, "Developing a speech activity detection system for the DARPA RATS program," in *Proc. ISCA Interspeech*, 2012, pp. 1969–1972. [pg. 54]

[143] K.-H. Woo, T.-Y. Yang, K.-J. Park, and C. Lee, "Robust voice activity detection algorithm for estimating noise spectrum," *Electronics Letters*, vol. 36, no. 2, pp. 180–181, 2000. [pg. 54]

[144] J. Ramırez, J. C. Segura, C. Benıtez, A. De La Torre, and A. Rubio, "Efficient voice activity detection algorithms using long-term speech information," *Speech Communication*, vol. 42, no. 3-4, pp. 271–287, 2004. [pg. 54]

[145] B. Liu, Z. Wang, S. Guo, H. Yu, Y. Gong, J. Yang, and L. Shi, "An energy-efficient voice activity detector using deep neural networks and approximate computing," *Microelectronics Journal*, vol. 87, pp. 12–21, 2019. [pg. 54]

[146] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "Deep neural networks for multi-room voice activity detection: Advancements and comparative evaluation," in *Proc. IEEE International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 3391–3398. [pg. 54]

[147] Z.-C. Fan, Z. Bai, X.-L. Zhang, S. Rahardja, and J. Chen, "AUC optimization for deep learning based voice activity detection," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6760–6764. [pg. 54], [pg. 135]

[148] F. Byers and O. Sadjadi, *2017 Pilot Open Speech Analytic Technologies Evaluation (2017 NIST Pilot OpenSAT): Post Evaluation Summary*. US Department of Commerce, National Institute of Standards and Technology, 2019. [pg. 54]

[149] K. Walker and S. Strassel, "The RATS radio traffic collection system," in *Proc. ISCA Speaker and Language Recognition Workshop (Odyssey)*, 2012, pp. 291–297. [pg. 54]

[150] I. Viñals, P. Gimeno, A. Ortega, A. Miguel, and E. Lleida, "Estimation of the number of speakers with variational bayesian PLDA in the DIHARD diarization challenge," in *Proc. ISCA Interspeech*, 2018, pp. 2803–2807. [pg. 55]

[151] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. International Conference on Machine Learning (ICML)*, 2015, pp. 448–456. [pg. 55], [pg. 72]

[152] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1492–1500. [pg. 56]

[153] H. Zeinali, L. Burget, and H. Cernocky, "Acoustic scene classification using fusion of attentive convolutional neural networks for DCASE2019 challenge," in *Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2019. [pg. 56]

[154] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2019, vol. 32, pp. 8024–8035. [pg. 56], [pg. 102]

[155] A. Ziaei, L. Kaushik, A. Sangwan, J. H. Hansen, and D. W. Oard, "Speech activity detection for NASA apollo space missions: Challenges and solutions," in *Proc. ISCA Interspeech*, 2014, pp. 1544–1548. [pg. 58], [pg. 60], [pg. 91], [pg. 94]

[156] J. Heitkaemper, J. Schmalenstroeer, and R. Haeb-Umbach, "Statistical and neural network based speech activity detection in non-stationary acoustic environments," in *Proc. ISCA Interspeech*, 2020, pp. 2597–2601. [pg. 60]

[157] Q. Lin, T. Li, and M. Li, "The DKU speech activity detection and speaker identification systems for Fearless steps challenge phase-02," in *Proc. ISCA Interspeech*, 2020, pp. 2607–2611. [pg. 60]

[158] A. I. Mezza, E. A. Habets, M. Müller, and A. Sarti, "Unsupervised domain adaptation for acoustic scene classification using band-wise statistics matching," in *Proc. European Signal Processing Conference (EUSIPCO)*, 2021, pp. 11–15. [pg. 64]

[159] C. Anoop, A. Prathosh, and A. Ramakrishnan, "Unsupervised domain adaptation schemes for building ASR in low-resource languages," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2021. [pg. 64]

[160] X.-L. Zhang, "Unsupervised domain adaptation for deep neural network based voice activity detection," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6864–6868. [pg. 64]

[161] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2507–2516. [pg. 64]

[162] C. Chu and R. Wang, "A survey of domain adaptation for neural machine translation," in *Proc. International Conference on Computational Linguistics (COLING)*, 2018, pp. 1304–1319. [pg. 64]

[163] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020. [pg. 64]

[164] S. Sun, B. Zhang, L. Xie, and Y. Zhang, "An unsupervised deep domain adaptation approach for robust speech recognition," *Neurocomputing*, vol. 257, pp. 79–87, 2017. [pg. 64]

[165] Q. Wang, W. Rao, S. Sun, L. Xie, E. S. Chng, and H. Li, "Unsupervised domain adaptation via domain adversarial training for speaker recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4889–4893. [pg. 64]

[166] S. Mavaddaty, S. M. Ahadi, and S. Seyedin, "A novel speech enhancement method by learnable sparse and low-rank decomposition and domain adaptation," *Speech Communication*, vol. 76, pp. 42–60, 2016. [pg. 64]

[167] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018. [pg. 65]

[168] B. Tan, Y. Zhang, S. Pan, and Q. Yang, "Distant domain transfer learning," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 31, 2017. [pg. 65]

[169] G. Csurka, *A Comprehensive Survey on Domain Adaptation for Visual Applications*. Springer International Publishing, 2017, pp. 1–35. [pg. 65]

[170] J. Hu, J. Lu, and Y.-P. Tan, "Deep transfer metric learning," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 325–333. [pg. 66]

[171] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, 2016, pp. 136–144. [pg. 66]

[172] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016. [pg. 66], [pg. 69]

[173] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," *arXiv preprint arXiv:1603.04779*, 2016. [pg. 66]

[174] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, pp. 469–477, 2016. [pg. 66]

[175] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016. [pg. 66]

[176] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised representation learning: Transfer learning with deep autoencoders," in *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. [pg. 66]

[177] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2223–2232. [pg. 66]

[178] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, vol. 3, no. 2, 2013, p. 896. [pg. 66], [pg. 67]

[179] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Label propagation for deep semi-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5070–5079. [pg. 66]

[180] W. Shi, Y. Gong, C. Ding, Z. M. Tao, and N. Zheng, "Transductive semi-supervised deep learning using min-max features," in *Proc. European Conference on Computer Vision (ECCV)*, 2018, pp. 299–315. [pg. 66]

[181] G.-H. Wang and J. Wu, "Repetitive reprediction deep decipher for semi-supervised learning," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6170–6177. [pg. 66]

[182] M. N. Rizve, K. Duarte, Y. S. Rawat, and M. Shah, "In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning," in *Proc. International Conference on Learning Representations (ICLR)*, 2021. [pg. 67]

[183] M. Zhong, J. LeBien, M. Campos-Cerqueira, R. Dodhia, J. L. Ferres, J. P. Velev, and T. M. Aide, "Multispecies bioacoustic classification using transfer learning of deep convolutional neural networks with pseudo-labeling," *Applied Acoustics*, vol. 166, p. 107375, 2020. [pg. 67]

[184] Y. Takashima, Y. Fujita, S. Horiguchi, S. Watanabe, L. P. G. Perera, and K. Nagamatsu, "Semi-supervised training with pseudo-labeling for end-to-end neural diarization," in *Proc. ISCA Interspeech*, 2021, pp. 3096–3100. [pg. 67]

[185] Q. Xu, T. Likhomanenko, J. Kahn, A. Hannun, G. Synnaeve, and R. Collobert, "Iterative pseudo-labeling for speech recognition," in *Proc. ISCA Interspeech*, 2020, pp. 1006–1010. [pg. 67]

[186] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015. [pg. 67], [pg. 68], [pg. 78]

[187] P. Shen, X. Lu, S. Li, and H. Kawai, "Feature representation of short utterances based on knowledge distillation for spoken language identification." in *Proc. ISCA Interspeech*, 2018, pp. 1813–1817. [pg. 67]

[188] J. Li, R. Zhao, Z. Chen, C. Liu, X. Xiao, G. Ye, and Y. Gong, "Developing far-field speaker system via teacher-student learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5699–5703. [pg. 67]

[189] A. Korattikara Balan, V. Rathod, K. P. Murphy, and M. Welling, "Bayesian dark knowledge," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, pp. 3438–3446, 2015. [pg. 68]

[190] T. Asami, R. Masumura, Y. Yamaguchi, H. Masataki, and Y. Aono, "Domain adaptation of DNN acoustic models using knowledge distillation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 5185–5189. [pg. 68]

[191] J. Li, M. L. Seltzer, X. Wang, R. Zhao, and Y. Gong, "Large-scale domain adaptation via teacher-student learning," in *Proc. ISCA Interspeech*, 2017, pp. 2386–2390. [pg. 68]

[192] J. Luckenbaugh, S. Abplanalp, R. Gonzalez, D. Fulford, D. Gard, and C. Busso, "Voice activity detection with teacher-student domain emulation," *Proc. ISCA Interspeech*, pp. 4374–4378, 2021. [pg. 69]

[193] H. Dinkel, S. Wang, X. Xu, M. Wu, and K. Yu, "Voice activity detection in the wild: A data-driven approach using teacher-student training," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 29, pp. 1542–1555, 2021. [pg. 69]

[194] B. Sun and K. Saenko, "Deep CORAL: Correlation alignment for deep domain adaptation," in *Proc. European Conference on Computer Vision (ECCV)*, 2016, pp. 443–450. [pg. 69]

[195] P. Morerio and V. Murino, "Correlation alignment by riemannian metric for domain adaptation," *arXiv preprint arXiv:1705.08180*, 2017. [pg. 70]

[196] Z. Huang and L. Van Gool, "A riemannian network for SPD matrix learning," in *Proc. AAAI Conference on Artificial Intelligence*, 2017. [pg. 70]

[197] C. C. MacDuffee, *The theory of matrices*. Springer Science & Business Media, 2012, vol. 5. [pg. 70]

[198] P. Gimeno, D. Ribas, A. Ortega, A. Miguel, and E. Lleida, "Convolutional recurrent neural networks for speech activity detection in naturalistic audio from apollo missions," in *Proc. IberSPEECH*, 2021, pp. 26–30. [pg. 72], [pg. 73]

[199] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6429–6433. [pg. 87]

[200] D. Diaz-Guerra, A. Miguel, and J. R. Beltran, "gpuRIR: A python library for room impulse response simulation with GPU acceleration," *Multimedia Tools and Applications*, vol. 80, no. 4, pp. 5653–5671, 2021. [pg. 87], [pg. 123]

[201] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of machine learning research*, vol. 9, no. 11, 2008. [pg. 90], [pg. 125]

[202] P. Gimeno, I. Viñals, A. Ortega, A. Miguel, and E. Lleida, "A recurrent neural network approach to audio segmentation for broadcast domain data," in *Proc. IberSPEECH*, 2018, pp. 87–91. [pg. 100]

[203] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. ISCA Interspeech*, 2015, pp. 3586–3589. [pg. 100]

[204] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *Proc. International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 121–126. [pg. 100]

[205] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. International Conference on Learning Representations (ICLR)*, 2018. [pg. 101], [pg. 112], [pg. 124]

[206] F. Gustafsson, "Determining the initial states in forward-backward filtering," *IEEE Transactions on Signal Processing*, vol. 44, no. 4, pp. 988–992, 1996. [pg. 103]

[207] F. Eyben, M. Wöllmer, and B. Schuller, "OpenSMILE: the munich versatile and fast open-source audio feature extractor," in *Proc. ACM international conference on Multimedia*, 2010, pp. 1459–1462. [pg. 104], [pg. 139]

[208] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," in *Proc. Pacific Rim Conference on Multimedia (PCM)*, 2018, pp. 14–23. [pg. 112]

[209] A. Gallardo Antolín and R. San Segundo Hernández, "UPM-UC3M system for music and speech segmentation," in *Proc. IberSPEECH*, 2010. [pg. 113], [pg. 115]

[210] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2018, pp. 19–23. [pg. 114]

[211] D. Tavarez, E. Navas, D. Erro, and I. Saratxaga, "Audio segmentation system by Aholab for Albayzín 2012 evaluation campaign," in *Proc. IberSPEECH*, 2012, pp. 577–584. [pg. 117]

[212] S. Cerdà, J. Albert, A. Giménez Pastor, J. Andrés Ferrer, J. Civera Saiz, and A. Juan Císcar, "Albayzín evaluation: the PRHLT-UPV audio segmentation system," in *Proc. IberSPEECH*, 2012, pp. 596–600. [pg. 117], [pg. 118]

[213] M. J. Gales, A. Ragni, H. AlDamarki, and C. Gautier, "Support vector machines for noise robust ASR," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009, pp. 205–210. [pg. 122]

[214] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014. [pg. 122]

[215] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823. [pg. 134]

[216] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017. [pg. 134]

[217] M. Kaya and H. Ş. Bilge, "Deep metric learning: A survey," *Symmetry*, vol. 11, no. 9, p. 1066, 2019. [pg. 134]

[218] L. P. Garcia-Perera, J. A. Nolazco-Flores, B. Raj, and R. Stern, "Optimization of the DET curve in speaker verification," in *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2012, pp. 318–323. [pg. 135]

[219] K.-A. Toh, J. Kim, and S. Lee, "Maximizing area under ROC curve for biometric scores fusion," *Pattern Recognition*, vol. 41, no. 11, pp. 3373–3392, 2008. [pg. 135]

[220] V. Mingote, A. Miguel, A. Ortega, and E. Lleida, "Optimization of the area under the ROC curve using neural network supervectors for text-dependent speaker verification," *Computer Speech & Language*, p. 101078, 2020. [pg. 135], [pg. 136], [pg. 138], [pg. 145], [pg. 146], [pg. 147]

[221] Z. Bai, X.-L. Zhang, and J. Chen, "Partial AUC metric learning based speaker verification back-end," in *Proc. ISCA Speaker and Language Recognition Workshop (Odyssey)*, 2020, pp. 380–384. [pg. 136]

[222] ——, "Partial AUC optimization based deep speaker embeddings with class-center learning for text-independent speaker verification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6819–6823. [pg. 136], [pg. 137], [pg. 140], [pg. 145]

[223] J. S. Downie, A. F. Ehmann, M. Bay, and M. C. Jones, "The music information retrieval evaluation exchange: Some observations and insights," in *Advances in music information retrieval*. Springer, 2010, pp. 93–115. [pg. 144]

[224] M. S. Wandishin and S. J. Mullen, "Multiclass ROC analysis," *Weather and Forecasting*, vol. 24, no. 2, pp. 530–547, 2009. [pg. 144]

[225] D. J. Hand and R. J. Till, "A simple generalisation of the area under the ROC curve for multiple class classification problems," *Machine learning*, vol. 45, no. 2, pp. 171–186, 2001. [pg. 144], [pg. 145]

[226] R. Kleiman and D. Page, "AUCμ: A performance metric for multi-class machine learning models," in *Proc. International Conference on Machine Learning*, 2019, pp. 3439–3447. [pg. 144]

[227] F. Provost and P. Domingos, "Well-trained PETs: Improving probability estimation trees," *Raport instytutowy IS-00-04, Stern School of Business, New York University*, 2000. [pg. 144], [pg. 145]

[228] Z. Bai, X.-L. Zhang, and J. Chen, "Speaker verification by partial AUC optimization with mahalanobis distance metric learning," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 28, pp. 1533–1548, 2020. [pg. 145]

[229] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4879–4883. [pg. 145]

[230] J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han, "In defence of metric learning for speaker recognition," in *Proc. ISCA Interspeech*, 2020, pp. 2977–2981. [pg. 145]

[231] Z. Yang, Q. Xu, S. Bao, X. Cao, and Q. Huang, "Learning with multiclass AUC: Theory and algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7747–7763, 2022. [pg. 145]

[232] M. Xu, S. Li, C. Liang, and X.-L. Zhang, "Multi-class AUC optimization for robust small-footprint keyword spotting with limited training data," in *Proc. ISCA Interspeech*, 2022, pp. 3278–3282. [pg. 145]

[233] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 212–220. [pg. 148]

[234] Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. D. Nadai, "Efficient training of visual transformers with small datasets," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 23 818–23 830. [pg. 160]

[235] R. Shao and X.-J. Bi, "Transformers meet small datasets," *IEEE Access*, vol. 10, pp. 118 454–118 464, 2022. [pg. 161]

[236] C. Chen, Z. Fu, Z. Chen, S. Jin, Z. Cheng, X. Jin, and X.-S. Hua, "HoMM: Higher-order moment matching for unsupervised domain adaptation," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 3422–3429. [pg. 161]

[237] Z. Cheng, C. Chen, Z. Chen, K. Fang, and X. Jin, "Robust and high-order correlation alignment for unsupervised domain adaptation," *Neural Computing and Applications*, vol. 33, no. 12, pp. 6891–6903, 2021. [pg. 161]

[238] L. Dong and B. Xu, "CIF: Continuous integrate-and-fire for end-to-end speech recognition," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6079–6083. [pg. 161]

[239] Y. Meng, H.-J. Chen, J. Shi, S. Watanabe, P. Garcia, H.-y. Lee, and H. Tang, "On compressing sequences for self-supervised speech models," in *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2022. [pg. 161]

[240] J. Turian, J. Shier, H. R. Khan, B. Raj, B. W. Schuller, C. J. Steinmetz, C. Malloy, G. Tzanetakis, G. Velarde, K. McNally *et al.*, "HEAR: Holistic evaluation of audio representations," in *NeurIPS Competitions and Demonstrations Track*, 2021, pp. 125–145. [pg. 161]

[241] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "BYOL for audio: Exploring pre-trained general-purpose audio representations," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 31, pp. 137–151, 2022. [pg. 161]