UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Universitat Politècnica de Catalunya

Grau en Enginyeria de Sistemes Audiovisuals

Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

DETECCIÓ I CLASSIFICACIÓ DE SONS SUPERPOSATS

Treball de Fi de Grau

Autor

Marc León Gimeno

Director

Ignasi Esquerra Llucià

22 de Juny de 2021

AGRAÏMENTS

Vull expressar el meu profund agraïment al tutor de seguiment d'aquest projecte, l'Ignasi Esquerra Llucià, per la seva amabilitat i contínua predisposició a resoldre els diversos dubtes i dificultats que han anat sorgint al llarg del desenvolupament d'aquest projecte.

RESUM

Actualment, els models basats en l'aprenentatge automàtic estan prenent gran importància en l'àmbit del processament de dades. Els sistemes desenvolupats a partir d'aquesta tecnologia permeten processar una gran quantitat d'informació i extreure'n les característiques més rellevants, oferint així, una gran varietat d'aplicacions, cadascuna donant solució a tasques concretes.

Aquest treball es centra en l'àmbit de la detecció i la classificació d'esdeveniments acústics, concretament en la identificació de sons reals enregistrats a la via pública.

Al llarg d'aquest projecte es realitzarà el desenvolupament d'un sistema basat en l'aprenentatge automàtic, utilitzant tècniques d'aprenentatge profund. Així doncs, el sistema ha de ser capaç, per una banda, de detectar i classificar els esdeveniments acústics presents en diversos senyals d'àudio i, per altra banda, de generar les etiquetes temporals per cada un d'ells on s'indiqui l'instant de temps d'inici i final de cada un dels esdeveniments detectats.

Per poder dur a terme aquest projecte, s'utilitzarà la base de dades facilitada en el repte 'DCASE Challenge' de l'any 2017. Concretament, s'utilitzarà la base de dades facilitada per la tasca 3, '*Sound event detection in real life audio'*.

En aquest sentit, es realitzarà una anàlisis de la base de dades i del sistema de referència proposats per l'organització del repte per comprendre les tecnologies de processament utilitzades i poder desenvolupar el sistema propi.

L'objectiu no és participar en el repte escollit ni proposar un sistema amb millors resultats que el sistema de referència. Amb el desenvolupament d'aquest projecte es pretén entendre el funcionament dels sistemes de detecció i classificació d'àudio i les tècniques utilitzades pel processament dels senyals acústics.

Finalment, en base als resultats obtinguts i les observacions realitzades, es presentaran noves línies de desenvolupament que permetin la continuació d'aquest projecte.

En trets generals, d'aquest projecte es pot extreure que les solucions a una tasca d'aprenentatge automàtic no són úniques, de tal manera que, els criteris seguits per escollir la configuració per desenvolupar el sistema poden influir considerablement en els resultats obtinguts.

ABSTRACT

Nowadays, systems based on machine learning are taking a great importance into the data processing field. The variety of systems developed by this kind of technology allow us to process a wide amount of information and extract the most relevant features. In this way, there is a variety of applications developed to solve concrete tasks.

This project is focused in the detection and the classification of sound events, specifically in detecting real sounds recorded in several public roads.

Throughout this project, the development of a system based on machine learning will be carried out using deep learning techniques. The system has to be able to, in one hand, detect and classify the sound events present in various audio signals and, on the other hand, to be able to generate the temporary labels for each one of them where it indicates the starting and finishing time of each of the detected events.

To carry out this project, the database given from the 2017 DCASE Challenge will be used, specifically, the database from the task 3, *'Sound event detection in real life audio'.*

In this way, an analysis of the database and the baseline system provided by the challenge's organization will be done to understand the processing technologies used and being able to develop the main system.

The main goal is not to participate in the selected challenge nor propose a system with better results than the baseline system. The development of this project is aimed to understand the functioning of the sound detection and classification systems and the techniques used to process audio signals.

Finally, based on the obtained results and the observations made, new lines of development will be presented that will allow the continuation of this project.

In general, this project shows that the solutions to a machine learning task are not unique, so that the criteria followed to choose the configuration to develop the system can significantly influence the results obtained.

ÍNDEX

ÍNDEX DE TAULES

ÍNDEX DE FIGURES

ACRÒNIMS

DCASE Detection and Classification of Acoustic Scenes and Events (Detecció i classificació d'escenes i esdeveniments acústics).

TP True positives (Veritables positius).

TN True negatives (Veritables negatius).

FP False positives (Falsos positius).

FN False negatives (Falsos negatius).

MLP Multilayer Perceptron (Perceptró Multicapa).

Adam Adaptative Moment Estimation (Estimació del moment adaptatiu).

MFCC Mel-Frequency Cepstral Coefficients (Coeficients cepstrals de freqüència Mel).

1 INTRODUCCIÓ

1.1 Motivació

Actualment, els sistemes basats en l'aprenentatge automàtic estan prenent una gran importància en la vida quotidiana. Per exemple, aquests tipus de sistemes es poden trobar en els assistents de veu, en la conducció autònoma de vehicles o bé, en sistemes de reconeixement facial, entre d'altres.

Per altra banda, al llarg del grau en enginyeria de sistemes audiovisuals, s'han anat adquirint conceptes i tècniques sobre el processament de senyals acústics, gràcies als quals, s'ha pogut aprofundir sobre aquest camp de coneixement.

És per això que s'ha decidit realitzar un projecte que unifiqui ambdós camps de coneixement, de tal manera que pugui ser utilitzat com a introducció en el camp del processament de senyals acústics utilitzant tècniques d'aprenentatge automàtic.

1.2 Abast

Per dur a terme aquest projecte es realitzarà, en primer lloc, una recerca documental de les diverses tècniques actualment utilitzades en el processament dels senyals d'àudio, així com les tècniques i sistemes basats en l'aprenentatge automàtic per tal d'escollir aquelles més adients per desenvolupar el sistema.

En segon lloc, es realitzarà una anàlisi del sistema de referència i la base de dades proposats per l'organització del repte DCASE Challenge de l'any 2017 per tal d'entendre el funcionament dels sistemes de detecció i classificació d'esdeveniments acústics.

En tercer lloc, es desenvoluparà el sistema de detecció i classificació d'esdeveniments acústics propi, utilitzant les tècniques vistes durant la recerca documental.

Finalment, es desenvoluparà la memòria on es recullin els conceptes més rellevants i el procediment seguit per desenvolupar el sistema propi.

1.3 Requeriments

Per aquest projecte, es requereix un elevat cost computacional per tal d'executar el sistema de detecció i classificació, sobretot en el procés d'entrenament. En aquest sentit, la principal limitació es troba en l'equip utilitzat per desenvolupar l'algorisme. Les especificacions computacionals de l'ordinador utilitzat són, en general, poc eficients per executar una tasca d'aquestes dimensions. Això comporta que el temps requerit d'execució sigui elevat respecte altres dispositius més potents.

1.4 Objectius

Amb el desenvolupament d'aquest projecte es pretén assolir els següents principals objectius:

- Adquirir una base de coneixement sobre els sistemes i tècniques basades en l'aprenentatge automàtic.

- Conèixer i estructurar els passos a realitzar per donar solució a un problema relacionat amb l'aprenentatge automàtic.

- Entendre el funcionament dels sistemes de detecció i classificació d'esdeveniments acústics.

- Analitzar la configuració de les bases de dades utilitzades per entrenar i avaluar aquest tipus de sistemes.

- Dissenyar, desenvolupar i avaluar un sistema propi que sigui capaç de detectar esdeveniments acústics en un senyal d'àudio i generar les etiquetes temporals per cadascun dels esdeveniments detectats.

1.5 Estructura i Planificació

El projecte ha estat dividit en cinc fases diferents per tal d'assolir les diverses fites proposades. Les fases del projecte són les següents:

- **Fase 1**: En aquesta fase es pretén, per una banda, concretar l'abast del projecte i, per altra banda, planificar les setmanes per tal d'organitzar les tasques a realitzar i establir el temps de dedicació per cada una d'elles.

- **Fase 2**: Aquesta fase fa referència a la recerca documental. Es realitzarà una recerca dels diversos conceptes i tècniques necessàries per tal de desenvolupar el projecte.

- **Fase 3**: Al llarg d'aquesta fase es realitzarà una anàlisis del sistema de referència proposat per l'organització del repte i la base de dades facilitada per desenvolupar el sistema propi.

- **Fase 4**: Al llarg d'aquesta fase, es desenvoluparà el sistema propi. Es realitzarà el preprocessament de les dades, el desenvolupament del model acústic i la seva avaluació, així com analitzar els resultats obtinguts.

- **Fase 5**: En aquesta fase es desenvoluparà la memòria del projecte on es detallarà el procediment seguit per desenvolupar el sistema propi, l'anàlisi del sistema de referència i de la base de dades i els conceptes teòrics rellevants utilitzats al llarg del projecte.

Aquestes fases no han estat seqüencials, és a dir, una fase no ha començat quan l'anterior ha finalitzat, sinó que totes les fases s'han pogut desenvolupar de forma global.

En la següent imatge es mostra la planificació de les fases amb les que es divideix el projecte:

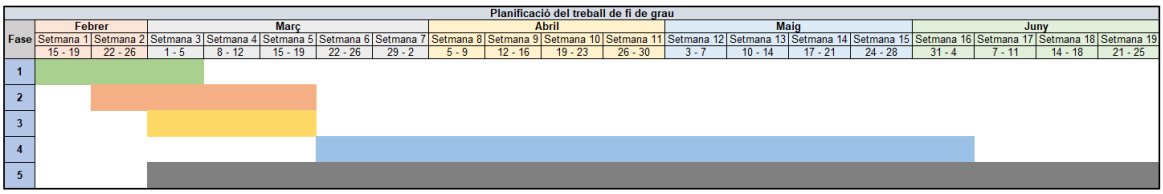

Figura 1. Planificació

Per altra banda, en l'Annex A: Planificació del projecte, es mostra una planificació detallada de les tasques realitzades per desenvolupar el projecte.

2 SISTEMES D'APRENENTATGE AUTOMÀTIC

2.1 Aprenentatge automàtic

L'aprenentatge automàtic[1], també conegut com aprenentatge de les màquines, és un subcamp de les ciències de la computació i una branca de la intel·ligència artificial que té per objectiu desenvolupar tècniques que permetin a les màquines aprendre sense la necessitat de ser programades de forma explícita.

D'aquesta manera, els algorismes desenvolupats amb aquestes tècniques són capaços d'adaptar-se i modelar-se davant de diversos tipus de dades, així com aprendre patrons d'una complexitat elevada.

2.1.1 Aprenentatge profund

El concepte d'aprenentatge profund [2] es pot entendre com el conjunt d'algorismes capaços de modelar sistemes basats en l'aprenentatge automàtic fent ús de les xarxes neuronals.

El sistema processa les dades d'entrada per cada una de les capes que forma la xarxa neuronal, de tal manera que, una vegada processades les dades en una capa determinada, s'envien a la següent capa fins arribar a la sortida del sistema.

2.1.2 Models d'aprenentatge

A partir de l'aprenentatge automàtic es generen models capaços de resoldre problemes o tasques determinades. Entre els models generats es distingeixen els següents:

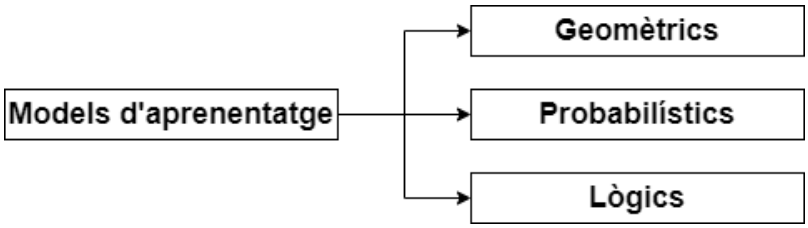

Figura 2. Models d'aprenentatge automàtic

a) **Models geomètrics**

Aquests tipus de models es construeixen en l'espai que formen les característiques d'entrada del sistema, de tal manera que poden tenir una o més dimensions. El model genera una divisió lineal, anomenada límit de decisió, per determinar la classe a la que pertanyen les dades d'entrada. Quan és possible generar una d'aquestes divisions, es diu que les dades son linealment separables.

b) **Models probabilístics**

Aquests models s'utilitzen per determinar la distribució de probabilitats que relaciona les dades d'entrada del sistema amb les de sortida, és a dir, amb les prediccions generades pel model.

[1] Aquest apartat, així com les classificacions presentades al llarg del mateix, s'ha realitzat seguint la referència [1].

c) **Models lògics**

Aquests models transformen i expressen les probabilitats en regles organitzades en arbres de decisió.

2.1.3 Tipus d'algorismes

Els algorismes d'aprenentatge automàtic es poden classificar segons el mètode d'entrenament utilitzat per tal d'adaptar els paràmetres interns del model.

Aquests, es poden categoritzar de la següent manera:

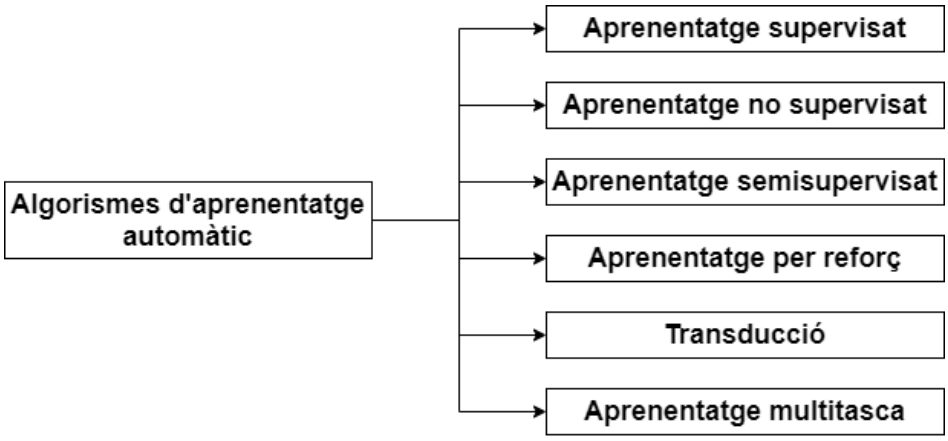

Figura 3. Tipus d'algorismes

a) **Aprenentatge supervisat**

En aquest tipus d'algorismes es proporcionen les entrades i les sortides del sistema, és a dir, coneix amb antelació la sortida que ha de generar. L'algorisme s'adapta per tal de detectar els patrons que relacionen les dades d'entrada amb les sortides. D'aquesta manera, una vegada ha après els paràmetres, és capaç de realitzar prediccions a partir de noves dades d'entrada no vistes prèviament i sense la necessitat de proporcionar-li la sortida amb antelació.

b) **Aprenentatge no supervisat**

En aquest tipus d'algorismes només es proporcionen les dades d'entrada al sistema. El sistema no coneix la sortida desitjada amb antelació, sinó que s'ajusta en base les observacions realitzades de forma autònoma i és capaç d'explorar l'estructura de dades per extreure tota aquella informació rellevant.

c) **Aprenentatge semisupervisat**

Aquests tipus d'algorismes realitzen una combinació entre els algorismes d'aprenentatge supervisat i els algorismes d'aprenentatge no supervisat, de tal manera que es tenen en compte les dades etiquetades, és a dir, aquelles dades de les quals es coneix la sortida i les dades de les quals no se'n coneix la seva sortida.

d) **Aprenentatge per reforç**

Aquests algorismes s'utilitzen per desenvolupar sistemes, anomenats 'agents', capaços d'aprendre a escollir les accions a realitzar dins un entorn, ja sigui real o no. Es basen en recompenses i penalitzacions, en funció del resultat obtingut, d'aquesta manera, basant-se en la prova i l'error, si el sistema escull l'opció desitjada, es recompensa, mentre que si s'equivoca, rep una penalització. Això permet que el sistema s'adapti fins al punt de trobar la solució al problema plantejat tot observant l'entorn.

e) **Transducció**

Aquests algorismes són semblants als algorismes d'aprenentatge supervisat, amb la diferència que aquests no constitueixen de forma explícita una funció. Intenten predir les categories o classes basant-se en les dades d'entrada, les seves respectives categories i les noves dades entrades al sistema. D'aquesta manera, el sistema pot variar en funció de les noves dades mostrades.

f) **Aprenentatge multitasca**

Aquests algorismes utilitzen coneixements apresos prèviament per tal de resoldre problemes o tasques semblants a les ja realitzades. En altres paraules, utilitza criteris anteriors per tal de resoldre tasques futures.

2.1.4 Tècniques de classificació

En tasques de classificació, existeix una diversitat de tècniques utilitzades per poder dur a terme la seva resolució. A continuació es mostren les tècniques principalment utilitzades:

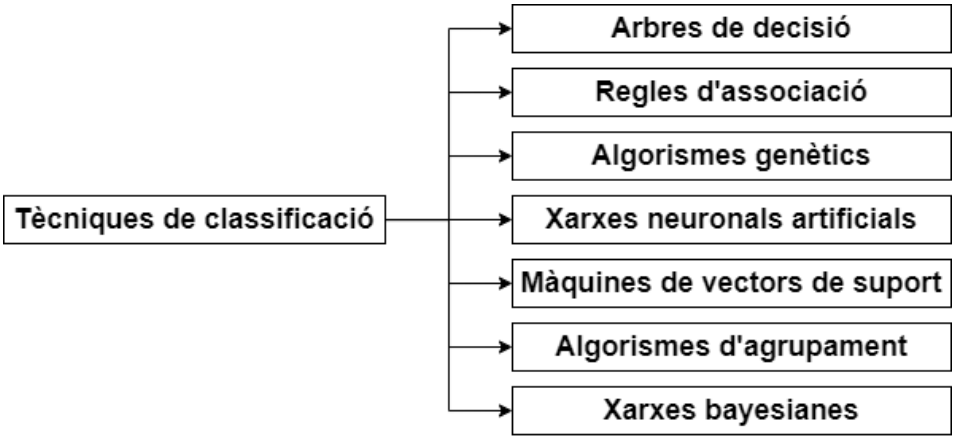

Figura 4. Tècniques de classificació

a) **Arbres de decisió**

Aquesta tècnica utilitza arbres de decisió com a model predictiu per tal de generar les sortides del sistema. Aquests es basen en generar diagrames de construccions lògiques per representar i categoritzar les condicions que ocorren de forma successiva per tal de realitzar la classificació.

b) **Regles d'associació**

Aquesta tècnica es basa en determinar les relacions més rellevants entre les dades existents dins un determinat conjunt.

c) **Algorismes genètics**

Els algorismes genètics es basen en processos de cerca heurística que simulen el comportament natural, de tal manera que, s'acaben seleccionant els models més adaptats, és a dir, els que generen millors resultats i es descarten els que pitjors resultats són capaços de generar.

d) **Xarxes neuronals artificials**

Aquesta tècnica s'inspira en les neurones del sistema nerviós dels animals. Es generen sistemes formats per unitats neuronals artificials enllaçades entre si per tal de produir un estímul de sortida.

Les unitats neuronals contenen paràmetres interns capaços de modelar-se de forma automàtica segons les dades processades. D'aquesta manera, aprenen en base els impulsos generats.

e) **Màquines de vectors de suport**

Les màquines de vectors de suport formen un conjunt de mètodes d'aprenentatge supervisat utilitzats en tasques de classificació i de regressió. Utilitzen un conjunt de dades d'entrenament per tal de generar models que siguin capaços de predir la classe a la que pertany una nova dada, la qual pot ser que no hagi estat vista anteriorment.

f) **Algorismes d'agrupament**

Aquesta tècnica es basa en la classificació d'observacions en subgrups, per tal que les observacions realitzades en cada un dels subgrups tinguin una semblança entre si segons uns determinats criteris. D'aquesta manera, es pot determinar la pertinença d'una dada en un subgrup determinat en funció de les semblances que tingui amb aquest.

g) **Xarxes bayesianes**

Es tracta de models probabilístics que representen una sèrie de variables aleatòries i les seves independències condicionals a través d'un graf dirigit. Aquesta tècnica és molt útil en l'estimació de probabilitats davant noves dades a analitzar.

2.1.5 Altres conceptes

a) **Hiperparàmetres**

Els hiperparàmetres són aquells paràmetres que defineixen el comportament del model i que són controlats i ajustats pels programadors del sistema. Aquests paràmetres s'ajusten per tal de trobar el sistema òptim que doni la millor solució possible al problema o tasca plantejats.

b) **Dropout**

Procediment utilitzat per evitar el sobreajustament del sistema, evitant així que el sistema estableixi dependència entre les unitats neuronals que constitueixen el model. Consisteix en desactivar neurones de forma aleatòria en cada una de les iteracions durant el procés d'entrenament, de tal manera que, l'efecte que té cada neurona sobre el resultat varia en cada una de les iteracions.

c) **Propagació endavant**

Per tal de processar les dades, les xarxes neuronals utilitzen el concepte de propagació endavant (de l'anglès, *forward propagation*) per tal de generar la sortida del sistema. D'aquesta manera, les dades d'entrada avancen per les diverses capes que formen el model fins arribar a la última capa, la capa de sortida. S'anomena propagació endavant ja que, no es generen cicles durant el processament de les dades a través de la xarxa neuronal.

d) **Retropropagació**

La retropropagació (de l'anglès, *backpropagation*) es tracta d'un mètode utilitzat durant el procés d'entrenament per calcular el gradient necessari a aplicar sobre els pesos o coeficients dels nodes de la xarxa, és a dir, sobre les unitats neuronals que formen les capes del sistema. D'aquesta manera, per cada iteració d'entrenament, els paràmetres dels nodes s'ajusten en base el gradient obtingut amb aquest mètode.

e) **Regressió**

El mètode de la regressió és un model matemàtic que es basa en determinar la relació entre una variable dependent y respecte altres variables, anomenades explicatives o independents, x.

En un model de regressió, les variables dependents fan referència a les prediccions generades pel sistema mentre que les variables independents corresponen amb les entrades al mateix sistema.

2.1.6 El Perceptró

Frank Rosenblatt va proposar un algorisme capaç d'aprendre els coeficients òptims de forma automàtica, el Perceptró [3], de tal manera que aquests coeficients es multiplicarien amb les característiques d'entrada per tal de decidir si una neurona artificial s'activa o no.

Per exemple, en el cas d'un problema de classificació binària, es podria predir si una mostra d'entrada pertany a una classe determinada o a una altra.

En la següent imatge es pot veure l'estructura del Perceptró proposat per Frank Rosenblatt:

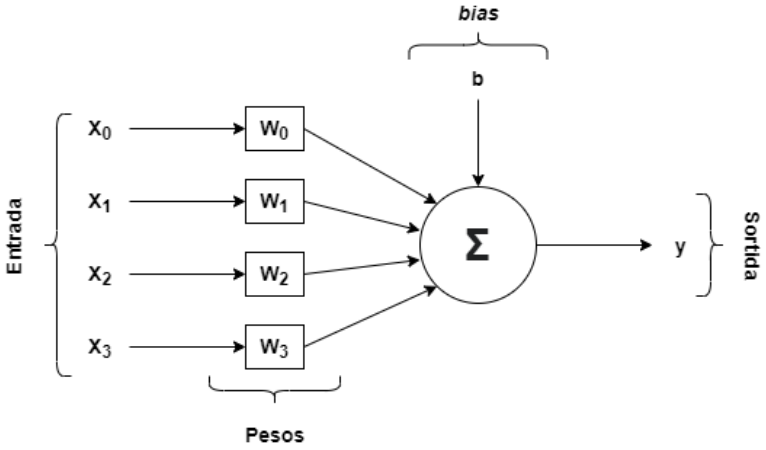

Figura 5. Estructura del Perceptró

La figura mostra els pesos i el paràmetre *bias* que han d'adaptar-se de forma automàtica per tal de relacionar l'entrada del sistema amb la sortida a predir. Això ho fa realitzant el producte escalar entre la entrada i els coeficients i sumant el paràmetre *bias*.

2.1.6.1 El Perceptró Multicapa

El Perceptró Multicapa o MLP (de l'anglès: *Multilayer Perceptron)* [4] sorgeix a partir de la figura del Perceptró i la principal limitació que aquest té, no poder resoldre problemes no lineals.

El MLP està format per diversos nodes, basats en el Perceptró, connectats local o totalment entre si i formant una estructura amb diverses capes.

L'estructura que forma el MLP es divideix en tres tipus de capes:

- **Capa d'entrada**: En aquesta capa s'introdueixen les dades al MLP, és a dir, connecta els valors d'entrada amb la primera capa profunda. En aquesta capa no es realitza cap tipus de processament.

- **Capa profunda**: Les entrades d'aquestes capes provenen de les capes anteriors, de tal manera que, una vegada realitzat el processament de la dada, s'envia a les neurones de la següent capa. Hi poden haver diverses capes profundes en una mateixa estructura.

- **Capa de sortida**: Aquesta capa constitueix l'última capa del MLP, de tal manera que és on es determinen els valors de sortida generats pel model, és a dir, les prediccions.

En la següent figura es mostra una possible estructura d'un MLP:

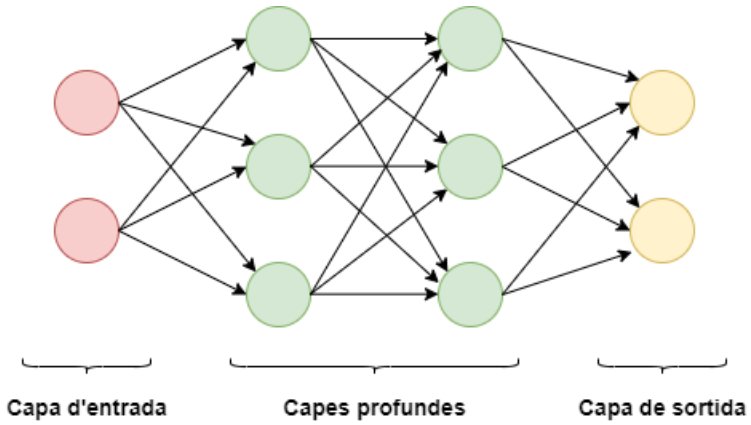

Figura 6. Estructura del MLP

El sistema utilitza el mètode de propagació endavant per tal de generar la sortida final del sistema i, durant el procés d'entrenament, per tal d'actualitzar els paràmetres interns de cada una de les neurones i poder arribar a la solució òptima, utilitza el mètode de retropropagació.

2.1.6.2 Funció d'activació

La funció d'activació [5] s'entén com la funció utilitzada per transmetre les sortides calculades d'una neurona a la següent. Generalment, s'utilitzen funcions no lineals per tal que el sistema pugui ser capaç de modelar els paràmetres interns quan les dades d'entrada són d'una complexitat elevada.

Per altra banda, s'acostumen a utilitzar funcions amb derivades senzilles per tal que el cost computacional sigui el més petit possible.

La funció d'activació s'aplica a la sortida generada per una unitat neuronal, tal i com es pot veure en la següent figura:

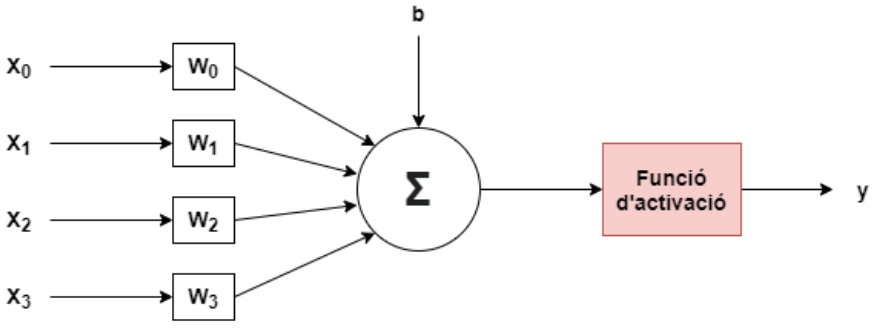

Figura 7. Funció d'activació

Algunes de les funcions d'activació més utilitzades són les següents:

a) **Funció Sigmoide**

Aquesta funció escala els valors de la sortida calculada per una unitat neuronal entre el rang de valors comprès entre 0 i 1.

És utilitzada en problemes de classificació, sobretot en problemes en què l'entrada del sistema pot pertànyer a diverses classes a la vegada.

L'expressió de la funció Sigmoide és la següent:

$$f(x) = \frac{1}{1 + e^{-x}}$$

on x es correspon amb la sortida generada per la neurona artificial.

En la següent figura es pot observar la representació gràfica de la funció Sigmoide:

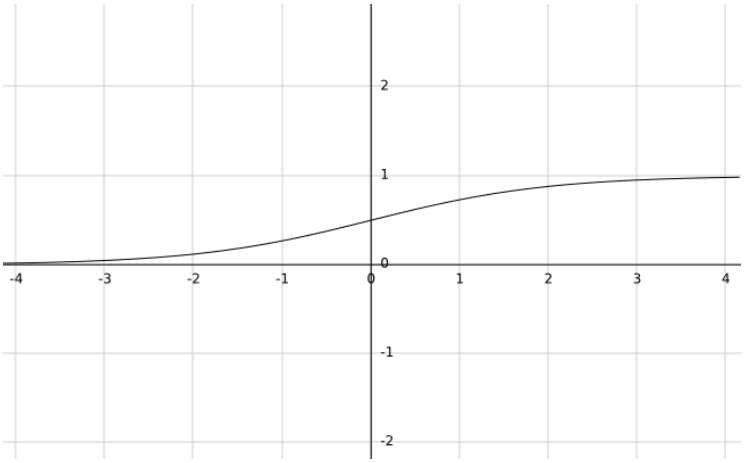

Figura 8. Funció Sigmoide

b) **Funció ReLu**

Aquesta funció modifica els valors calculats per una unitat neuronal quan aquest valor és negatiu. Totes aquelles sortides negatives es converteixen al valor 0 i la resta de valors es mantenen intactes.

L'expressió d'aquesta funció es mostra a continuació:

$$f(x) = \max(0, x)$$

on x correspon amb la sortida calculada per la neurona artificial.

La gràfica d'aquesta funció és la següent:

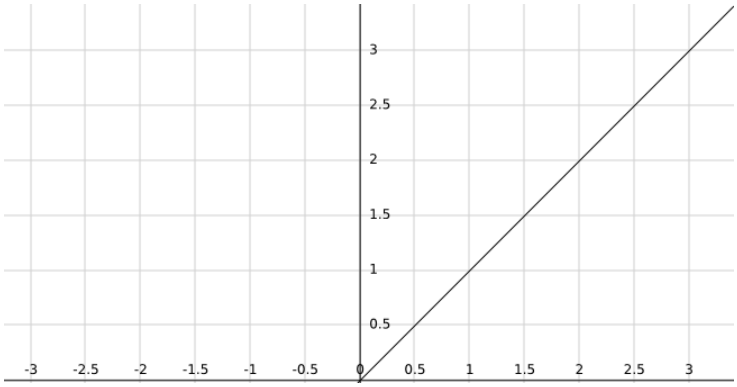

Figura 9. Funció ReLu

c) **Funció Softmax**

Aquesta funció calcula la distribució de probabilitats per cada una de les sortides, de tal manera que la suma dels valors calculats entre totes les sortides ha de ser 1.

S'utilitza en problemes de classificació binària o en problemes d'etiquetatge únic, és a dir, quan l'entrada només pot pertànyer a una sola categoria de totes les possibles. Acostuma a ser utilitzada en l'última capa del model per tal de realitzar la classificació.

L'expressió de la funció es mostra a continuació:

$$f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \; on \; i = 1, \dots, K$$

on x correspon amb al sortida generada per la neurona, i la classe sobre la que s'ha de calcular la seva probabilitat i K correspon amb el nombre total de classes possibles.

d) **Funció TanH**

Aquesta funció modifica la sortida calculada per la unitat neuronal i l'escala entre el rang de valors de -1 i 1.

L'expressió de la funció es mostra a continuació:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

on x correspon amb la sortida generada per la unitat neuronal.

La gràfica de la funció es mostra en la següent figura:

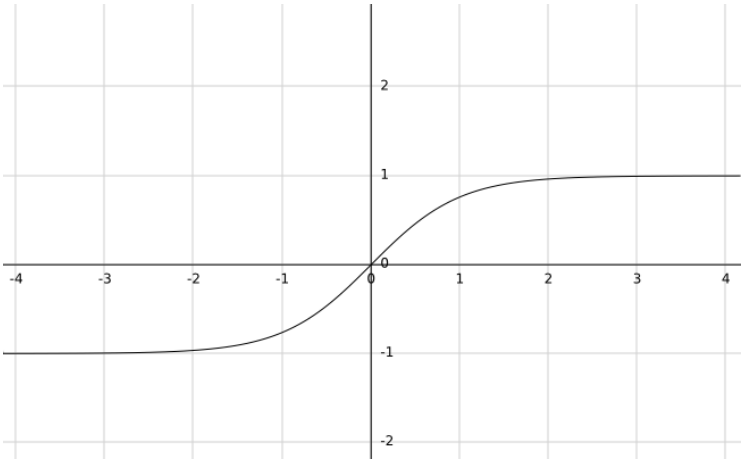

Figura 10. Funció TanH

2.1.6.3 Optimització

Per tal d'obtenir els paràmetres òptims que donin solució a un determinat problema d'aprenentatge automàtic és necessari fer ús de l'optimització [6].

Aquest concepte es basa en trobar els coeficients que minimitzin la funció de cost utilitzada durant l'entrenament, de tal manera que per cada iteració en el procés d'entrenament, el model actualitza els paràmetres en base l'error calculat. Una vegada s'ha aconseguit minimitzar la funció de cost, es diu que el sistema ha trobat la solució que produeix el mínim error possible.

Una vegada ha finalitzat el procés d'entrenament, el model aconsegueix que les prediccions generades siguin el més precises possible.

El mètode d'optimització més comú en xarxes neuronals és el gradient descendent. Aquest és un mètode iteratiu per trobar el mínim d'una funció, que en aquest cas és la funció de cost o pèrdua. Rep aquest nom ja que es prenen els increments proporcionals al negatiu del gradient.

Aquest mètode es basa en seleccionar un punt d'inici i una direcció aleatoris. Una vegada generada la predicció es computa l'error generat. En base aquest error s'obté el gradient negatiu de la funció en aquest punt per tal d'obtenir un nou punt a avaluar. D'aquesta manera, a mesura que avança l'entrenament, el sistema s'acosta al mínim de la funció de cost, produint el mínim error possible i augmentant la precisió del sistema.

Quan el sistema ja no és capaç de minimitzar l'error, es diu que el sistema ha trobat la solució òptima en base els hiperparàmetres utilitzats.

En la següent figura es mostra l'evolució en el procés d'optimització de la funció de cost, de tal manera que els punts avancen en direcció al mínim de la funció, que és el cercle central:

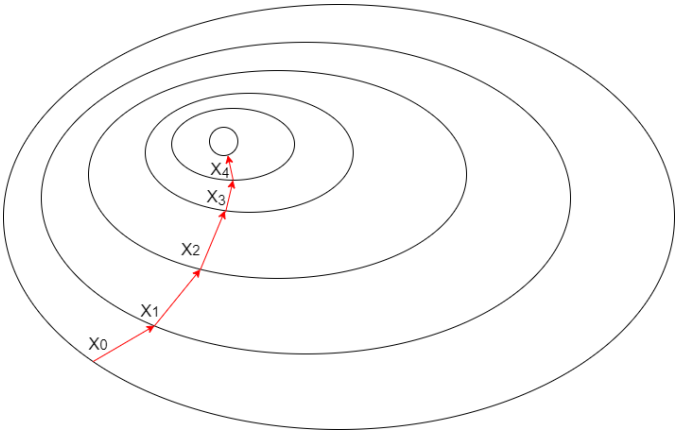

Figura 11. Optimització de la funció de cost

2.2 Anàlisi computacional d'escenes i esdeveniments acústics

L'anàlisi computacional d'escenes i esdeveniments acústics [2] s'encarrega d'extreure, segons el tipus d'aplicació o de tasca a realitzar, la informació present en els senyals acústics utilitzant mètodes computacionals.

2.2.1 Tècniques de processament

Per tal que el model sigui capaç de processar un senyal d'àudio, és necessari un tractament previ per tal d'extreure el contingut més rellevant i eliminar tot aquell contingut que pugui dificultar la resolució de la tasca. Aquest procés s'anomena extracció de característiques.

Per processar un senyal d'àudio cal tenir en compte dos factors. En primer lloc, la informació més significant es troba en el domini freqüencial, de tal manera que cal transformar el senyal del domini temporal al domini freqüencial per poder obtenir aquesta informació. En segon lloc, els senyals d'àudio acostumen a ser no estacionaris, és a dir, varien ràpidament al llarg del temps.

A partir d'aquests dos factors, la tècnica més utilitzada per processar els senyals d'àudio és l'anàlisi per blocs. Aquesta tècnica consisteix en dividir el senyal en petits segments temporals, normalment entre 20ms i 60ms, de tal manera que es pot captar el senyal en un estat quasi estacionari.

Per tal de desplaçar la finestra sobre el senyal s'aplica un salt o desplaçament entre trames que acostuma a ser el 50% de la mida de la finestra.

Per cada un d'aquests segments s'aplica, el que s'anomena, un enfinestrament amb la finalitat d'evitar canvis sobtats als límits de cada un dels segments. Una vegada s'ha aplicat l'enfinestrament es transforma el senyal del domini temporal al domini freqüencial per tal d'extreure tota la informació rellevant del senyal. El mètode més utilitzat per generar l'espectre del senyal és la transformada discreta de Fourier o DFT (de l'anglès, *discrete Fourier transform)*.

[2] Per realitzar aquest apartat s'ha seguit la referència [7].

Una vegada realitzat el procés de transformació del domini es procedeix a generar els vectors de característiques que formen el conjunt de dades que el model ha de processar.

Una de les característiques acústiques més utilitzades per representar el contingut de l'espectre de l'àudio del senyal són els coeficients cepstrals de freqüència Mel o MFCC (de l'anglès, *Mel-frequency cepstral coefficients)* [8]. Aquests proporcionen una representació compacta i fluida de l'espectre analitzat però no fan referència als canvis temporals de l'espectre amb el pas del temps.

2.2.2 Sistemes d'anàlisis computacional d'àudio

Els sistemes encarregats de realitzar l'anàlisi computacional d'àudio es poden categoritzar en tres grups principals, segons el tipus de tasca a realitzar:

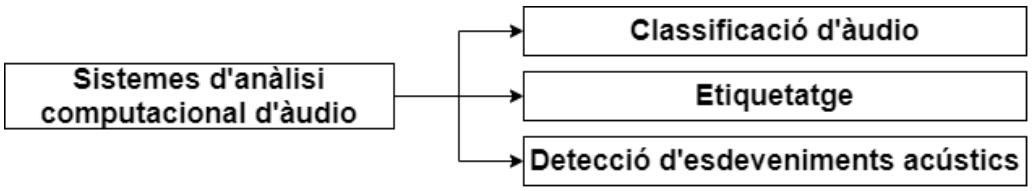

Figura 12. Sistemes d'anàlisi computacional d'àudio

a) **Classificació d'àudio**

Els sistemes de classificació es basen, tal i com indica el seu nom, en la classificació de senyals d'àudio segons la seva categoria o el seu origen.

Per altra banda, un model realitza classificació d'àudio quan el senyal analitzat només pot pertànyer a una sola categoria.

En general, els sistemes de classificació no ofereixen informació temporal sobre les classes detectades en els senyals d'entrada. Simplement, són utilitzats per categoritzar i classificar els senyals entre una varietat de classes.

Els tipus de sistemes de classificació d'àudio es poden veure en la següent figura:

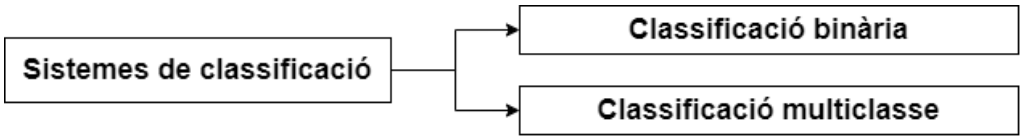

Figura 13. Tipus de classificació d'àudio

- **Classificació binària**: Classifica les entrades del sistema en una de dues categories possibles.

- **Classificació multiclasse**: Es basen en el mateix principi que els classificadors binaris, amb la diferència que el nombre de classes possibles és superior a dues.

El diagrama dels sistemes de classificació es pot visualitzar en la següent figura:

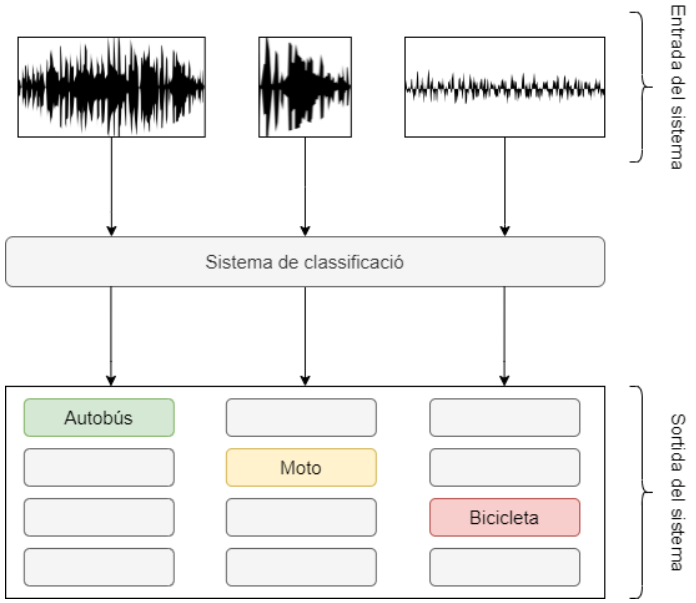

Figura 14. Sistema de classificació (Font: Adaptació de la referència [7])

b) **Etiquetatge**

En el camp de la classificació , es pot donar el cas que un senyal acústic pugui pertànyer a diverses categories de forma simultània. És per això que sorgeixen els models d'etiquetatge.

Al igual que els sistemes de classificació, la principal limitació és que només aporten informació categòrica, de tal manera que no són capaços d'aportar informació temporal sobre la classe reconeguda.

Els algorismes d'etiquetatge es classifiquen en les categories mostrades en la següent figura:

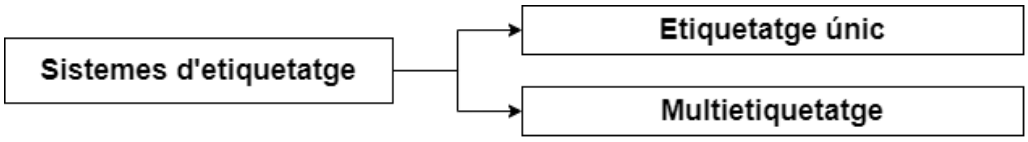

Figura 15. Tipus d'etiquetatge

- **Etiquetatge únic**: Els classificadors d'etiqueta única es basen classificar els senyals d'entrada tenint en compte que només poden pertànyer a una sola classe.

- **Multietiquetatge**: Els classificadors multietiqueta es basen en detectar les classes existents en un senyal d'àudio, és a dir, en un mateix senyal hi pot haver més d'una classe a detectar.

El diagrama dels sistemes d'etiquetatge es mostra en la següent figura:

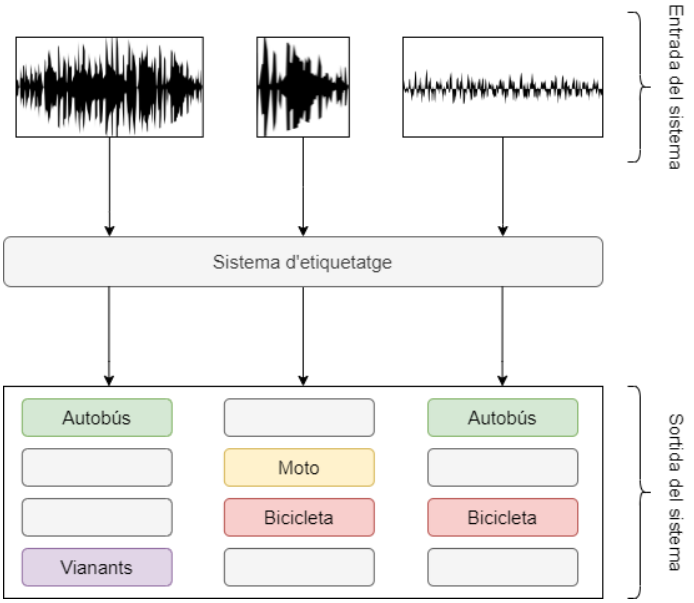

Figura 16. Sistema d'etiquetatge (Font: Adaptació de la referència [7])

c) **Detecció d'esdeveniments acústics**

Un altre tipus de sistema és el de la detecció d'esdeveniments acústics presents en un senyal d'àudio. A diferència dels sistemes de classificació convencionals, els sistemes de detecció aporten informació temporal per cada una de les classes detectades en el senyal d'entrada. A banda d'indicar la classe a la qual pertany l'esdeveniment detectat en un segment determinat del senyal, també s'indica el temps d'inici i final del mateix.

Aquests tipus de sistemes segueixen l'estructura mostrada en la següent figura:

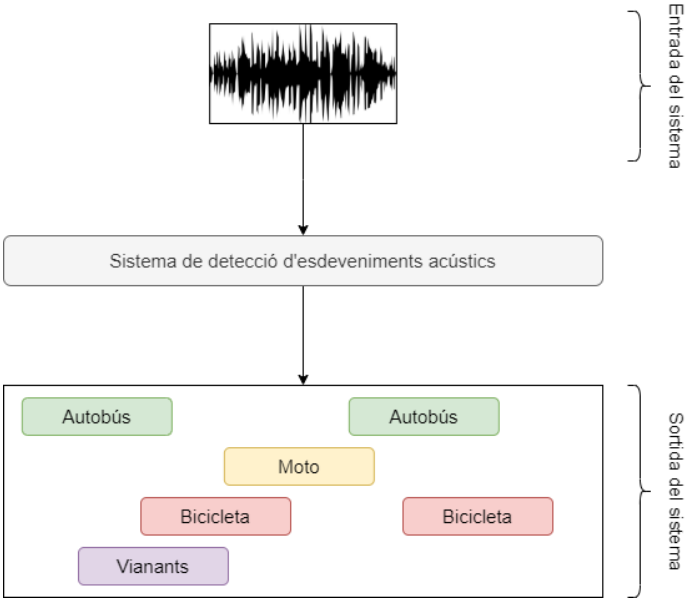

Figura 17. Sistema de detecció d'esdeveniments acústics (Font: Adaptació de la referència [7])

3 DCASE CHALLENGE

Detection and classification of acoustic sound scenes and events Challenge, conegut com DCASE Challenge [9] és un repte anual sobre la detecció i la classificació de senyals acústics en el qual diversos participants presenten una solució a algunes o totes les tasques que formen el repte. L'objectiu principal d'aquest repte és augmentar el camp de coneixement del processament d'àudio i compartir els resultats entre investigadors i participants.

Les tasques es plantegen de tal manera que la naturalesa de totes elles estigui relacionada directament amb la detecció i la classificació d'àudio però, a la vegada, les tasques proposades siguin diferents entre si, ja que cadascuna planteja una problemàtica diferent.

En general, els reptes proposats es divideixen en quatre o cinc tasques diferents i, per cada una de les tasques, s'ofereix als participants una base de dades amb la que poden desenvolupar i avaluar els seus sistemes. A més, com a contingut extra, també es proposa un sistema desenvolupat pels organitzadors del repte que pot ser utilitzat com a referència per comparar els resultats obtinguts. Aquest sistema és coneix com *Baseline System*.

Finalment, per tal d'avaluar els diferents sistemes i presentar els resultats finals, des de l'organització s'ofereix una nova base de dades d'avaluació. Aquesta base de dades està formada només pels senyals d'àudio que els sistemes han de processar. D'aquesta manera, els participants executen els sistemes amb aquesta base de dades sense conèixer les solucions de forma prèvia. L'organització del repte s'encarrega d'analitzar, avaluar i presentar els resultats obtinguts per cada un dels sistemes presentats.

3.1 DCASE Challenge 2017

El repte plantejat l'any 2017 es divideix en les següents tasques:

Tasca	Nom de la tasca	Objectiu
1	Classificació d'escenes acústiques	Classificar senyals d'àudio en classes que identifiquin l'entorn on ha estat enregistrat
2	Detecció d'esdeveniments acústics rars	Detectar esdeveniments acústics rars en mescles d'àudio creades de forma artificial
3	Detecció d'esdeveniments acústics en la vida real	Detectar i classificar, per categories, esdeveniments acústics en senyals d'àudio enregistrats en entorns reals
4	Detecció d'esdeveniments acústics a gran escala dèbilment supervisats per automòbils intel·ligents	Avaluar sistemes per detectar a gran escala esdeveniments acústics utilitzant dades d'entrenament etiquetades de forma dèbil

Taula 1. Tasques del repte DCASE 2017 [10]

Aquest projecte es centre en l'anàlisi i el desenvolupament de la tasca 3, detecció d'esdeveniments acústics en la vida real.

3.1.1 Tasca 3 del repte DCASE Challenge 2017

Un esdeveniment acústic pot ser definit com tot aquell so produït per una font sonora física, per exemple, un cotxe amb el motor engegat o persones conversant. Seguint aquesta línia, la tasca 3 del repte DCASE Challenge de l'any 2017 té com a objectiu el desenvolupament i l'avaluació dels sistemes de detecció i la classificació d'esdeveniments acústics presents en senyals d'àudio.

La particularitat d'aquesta tasca es troba en què els sons no es troben aïllats entre si, és a dir, els esdeveniments poden succeir de forma simultània en el temps, de tal manera que diverses fonts poden generar sons a la vegada.

Per altra banda, el sistema no només ha de detectar i classificar els sons, sinó que ha de generar les etiquetes temporals on s'ha d'indicar el temps d'inici i de finalització per cadascun d'ells.

En la següent imatge, es mostra la representació gràfica de l'objectiu d'aquesta tasca:

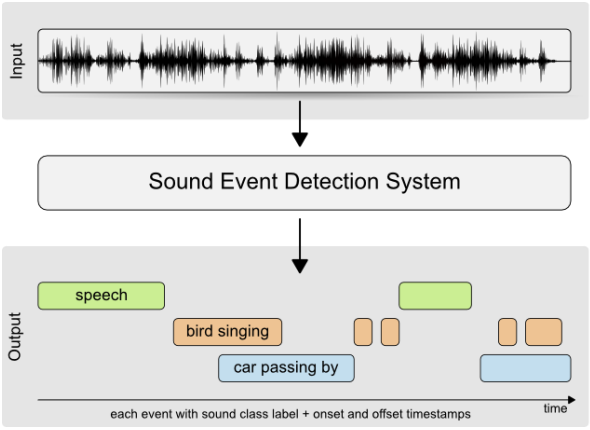

Figura 18. Detecció i classificació d'esdeveniments acústics [10]

Les classes d'esdeveniments[3] acústics que es poden trobar per aquesta tasca són les següents:

Esdeveniment acústic
Brakes squeaking
Car
Children
Large vehicle
People speaking
People walking

Taula 2. Classes d'esdeveniments [10]

[3] S'ha mantingut l'idioma original en el nom de les classes, ja que els arxius de la base de dades es troben en l'idioma original.

3.2 Base de dades

3.2.1 Estructura

La base de dades està formada per un directori que conté tota la informació necessària per poder desenvolupar i avaluar el sistema.

Concretament, conté un seguit d'arxius d'àudio i els arxius d'etiquetes. Aquestes dues dades constitueixen les dades d'entrada i de sortida del sistema, respectivament, ja que el model ha de processar els arxius d'àudio i generar les etiquetes temporals.

L'estructura general que segueix la base de dades es pot visualitzar en la següent figura:

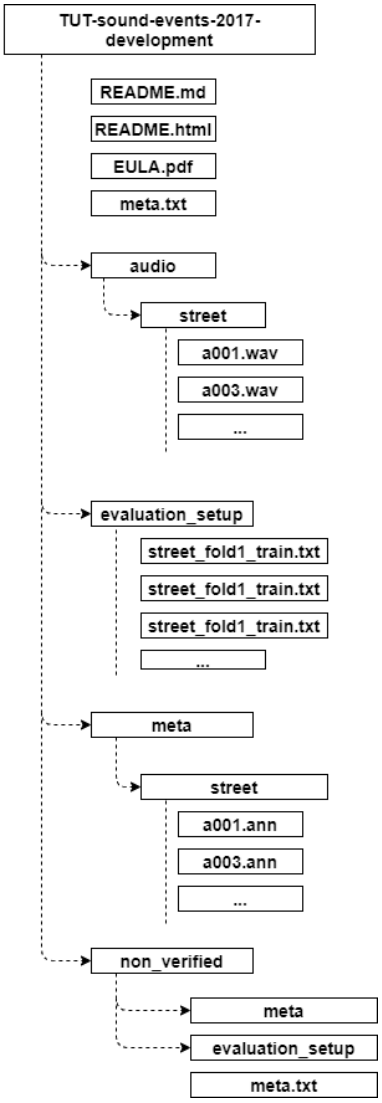

Figura 19. Estructura de la base de dades (Font: Adaptació de la referència [10])

De la base de dades es poden destacar dos directoris principals:

- **Audio***:* Aquest directori conté la escena de referència, *street*. Aquesta escena fa referència a on s'han enregistrat els senyals d'àudio. Tal i com indica el nom de l'escena, aquests àudios han estat enregistrats al carrer. Dins el subdirectori *street* s'hi poden trobar els diferents arxius d'àudio que formen les mostres per desenvolupar el sistema.

- **Evaluation setup**: Aquest directori conté els diferents arxius d'etiquetes necessaris per desenvolupar i avaluar el sistema.

3.2.2 Arxius d'àudio

Els senyals d'àudio que conté la base de dades van ser enregistrats per la Universitat de Tecnologia de Tampere, a Finlàndia, entre el juny de l'any 2015 i el gener de l'any 2016.

Concretament, hi ha un total de 24 arxius d'àudio en format WAV i en estèreo, és a dir, tenen dos canals. La resolució de les mostres és 24 bits i tenen una freqüència de mostratge de 44100Hz.

Els arxius es van enregistrar utilitzant un micròfon electret Soundman OKM II Klassik/studio A3 i una gravadora Roland Edirol R-09.

En la següent taula es proporciona un resum sobre les característiques dels arxius d'àudio:

Característica	Valor
Total d'arxius d'àudio	24
Format del fitxer	WAV
Número de canals	2
Bits per mostra (bits)	24
Freqüència de mostratge (Hz)	44100
Duració total entre tots els arxius (s)	5528
Duració mitja per fitxer (s)	230.3

Taula 3. Característiques dels senyals d'àudio

3.2.3 Arxius d'etiquetes

Els arxius d'etiquetes consisteixen en arxius de text que contenen, per cada un dels arxius d'àudio, un llistat dels esdeveniments presents i els temps d'inici i final de cadascun d'ells, en segons.

En concret, per cada esdeveniment s'indica el nom de l'arxiu al que es fa referència, l'escena de referència, el temps d'inici i final i el nom de la classe a la que pertany l'esdeveniment comprès en el temps indicat.

En la següent figura es pot veure el format que segueix una possible etiqueta:

```
audio/street/b094.wav    street    12.152502    13.195551    people speaking
```

Figura 20. Format dels arxius d'etiquetes

Per tal de generar les etiquetes, una vegada enregistrats els senyals d'àudio, membres del grup de recerca van procedir a anotar tots els esdeveniments que poguessin estar presents en els senyals, de tal manera que es va obtenir una llista dels possibles esdeveniments etiquetats.

Degut al nivell de subjectivitat en el procés d'etiquetatge, es van escollir tres persones que van escoltar els senyals acústics i van decidir si l'esdeveniment etiquetat era correcte o no.

Aquest procés de verificació va ser útil per unificar els criteris de decisió i generar les etiquetes finals adjuntades a la base de dades. D'aquesta manera, totes les etiquetes que van passar aquest procés es consideren vàlides.

Amb aquest procés es va eliminar un 10% dels esdeveniments de les anotacions originals, és a dir, de les no verificades.

Dins el directori de la base de dades *non_verified* es poden trobar les anotacions originals, és a dir, les etiquetes abans de passar el procés de validació.

A la següent taula es pot visualitzar el nombre total d'esdeveniments per classe, tan per les etiquetes validades com per les no validades, entre tots els arxius d'àudio de la base de dades:

Esdeveniment	Total d'esdeveniments validats	Total d'esdeveniments no validats
Brakes squeaking	52	59
Car	304	304
Children	44	58
Large vehicle	61	61
People speaking	89	117
People walking	109	130
Total	**659**	**729**

Taula 4. Total d'esdeveniments [10]

En la següent figura es mostra la distribució de les diferents classes dins la base de dades:

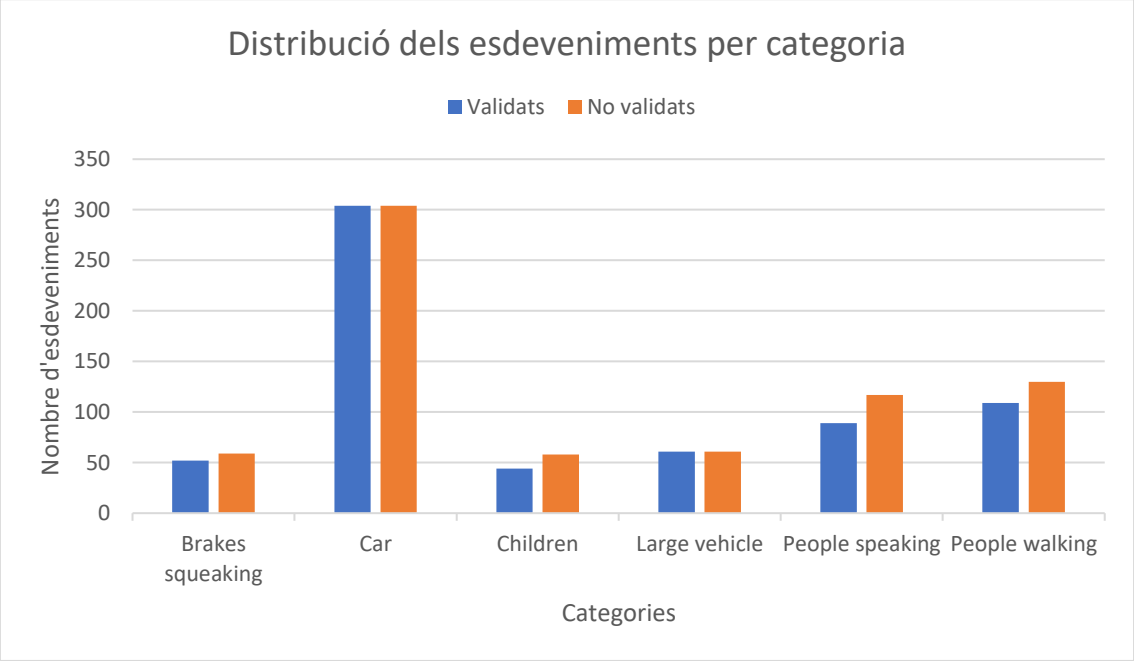

Figura 21. Distribució dels esdeveniments per classe

Es pot veure que la distribució entre totes les classes no és homogènia degut a que hi ha un nombre molt elevat d'esdeveniments de cotxes, *Car*, mentre que la resta de classes sí que mantenen una certa equitat entre el nombre total d'esdeveniments. Això és un factor a tenir en compte durant el desenvolupament del model, ja que pot influir en el seu comportament.

Per altra banda, els arxius d'etiquetes estan preparats per poder entrenar el sistema utilitzant el mètode de validació creuada. Concretament, utilitza el mètode *k-folds*, que consisteix en dividir les dades en *k* divisions.

Cadascuna d'aquestes divisions conté totes les dades repartides de forma diferent, de tal manera que algunes dades s'utilitzen per entrenar el sistema mentre que la resta de dades s'utilitzen per avaluar-lo.

Aquest mètode s'utilitza per analitzar el comportament del model sobre totes les dades d'entrenament i comprovar que el sistema és capaç de resoldre la tasca tan pels valors vistos com per les dades no vistes. En altres paraules, garantir que el sistema generalitzi correctament.

Per aquesta tasca, la base de dades es divideix en 4 subdivisions, de tal manera que es realitzen un total de 4 entrenaments del model.

3.3 Sistema de referència

Des de l'organització, es proporciona un sistema de referència [11, 12], el qual pot ser utilitzat pels participants del repte com a punt de partida per desenvolupar els sistemes propis.

En general, el sistema de referència proposa una solució per cada una de les tasques proposades però aquesta secció del document es centra en l'anàlisi del sistema de referència proposat per la tasca 3 del repte DCASE Challenge de l'any 2017.

3.3.1 Preprocessament de dades

El sistema de referència utilitza l'espectrograma de Mel per construir el vector de característiques dels senyals d'àudio. Concretament, fa ús d'un total de 40 bandes d'energia, de tal manera que, per cada un dels segments d'àudio, s'obté un total de 40 valors que s'utilitzaran com a característiques d'entrada del sistema.

Per altra banda, els senyals d'àudio es divideixen en segments de 40ms, utilitzant un desplaçament entre segments del 50%, és a dir, de 20ms. Per cadascun d'aquests segments es calcula el seu espectrograma de Mel.

Finalment, com entrada del sistema, es construeix un vector de característiques format per 5 segments consecutius, de tal manera que l'entrada del sistema és un vector de 200 característiques per cada un dels segments amb que s'ha dividit el senyal.

En la següent taula es mostra un resum dels paràmetres utilitzats pel sistema de referència per generar els vectors de característiques:

Paràmetre	Valor
Número de bandes de Mel	40
Mida de la finestra (ms)	40
Mida del desplaçament entre segments (ms)	20
Número de segments consecutius	5
Mida del vector de característiques	200

Taula 5. Paràmetres pel preprocessament de dades

3.3.2 Estructura del model

El model utilitzat pel sistema de referència es basa en el MLP. Cadascun dels nodes que formen la xarxa neuronal es basa en la figura del Perceptró.

La primera capa del sistema es correspon amb la capa d'entrada, de tal manera que connecta els valors dels vectors de característiques amb la primera capa profunda del sistema. Concretament, la mida de la capa d'entrada és de 200 unitats, ja que cada vector de característiques està format per un total de 200 valors.

La segona i tercera capa del sistema es corresponen amb les capes profundes. Cada capa està formada per un total de 50 nodes o neurones artificials. Aquestes capes s'encarreguen de modelar el sistema per tal d'establir la correlació entre l'entrada i la sortida del model.

La funció d'activació aplicada en aquestes dues capes és la ReLu i, entre capes, s'aplica un Dropout del 20% per evitar el sobreentrenament del sistema.

La capa de sortida té un total de 6 unitats, ja que, en total, hi ha 6 classes possibles per detectar i utilitza la funció Sigmoide com a funció d'activació.

En la següent taula es mostra un resum de l'estructura del sistema:

Capa	Unitats	Funció d'activació
Entrada	200	-
Capa profunda 1	50	ReLu
Capa profunda 2	50	ReLu
Sortida	6	Sigmoide

Taula 6. Configuració del model utilitzat pel sistema de referència

3.3.3 Entrenament

El sistema de referència realitza l'entrenament sobre tots els senyals d'àudio que conté la base de dades. A més, realitza un total de 4 entrenaments, ja que, tal i com s'ha comentat en el punt *3.2.3*, s'utilitza la validació creuada per avaluar el comportament del sistema.

El nombre d'iteracions que realitza per cada entrenament és de 200 iteracions, de tal manera que realitza el procés d'optimització un total de 200 vegades. A més, utilitza el mètode d'aturada anticipada per tal de reduir el sobreentrenament del model. Concretament, aquest procés consisteix en finalitzar l'entrenament de forma autònoma quan el model ja no millora els resultats obtinguts en base uns límits establerts per hiperparàmetre. En aquest cas, si el model no millora en un interval de 10 iteracions d'entrenament, aquest finalitza. Aquest criteri es té en compte a partir de la iteració d'entrenament número 100.

Per altra banda, el rang d'aprenentatge [13] és de 0.001. Aquest valor controla com de ràpid aprèn el sistema. A més, utilitza una mida de lot de 256 mostres d'entrada. És a dir, cada 256 vectors de característiques processats s'executa l'actualització dels pesos de la xarxa en comptes de fer-ho d'un en un.

Pel procés d'optimització, el model utilitza l'optimitzador Adam per tal d'actualitzar els paràmetres interns i minimitzar la funció de cost, la qual és la funció d'entropia creuada.

3.3.4 Avaluació

Per tal d'avaluar el sistema, utilitza el valor *F* i la taxa d'error *ER*, dels quals se'n parlarà en profunditat en el punt *4.7*. L'avaluació es realitza una vegada finalitzat l'entrenament per totes les subdivisions realitzades de la base de dades.

El procés d'avaluació es realitza amb segments d'un segon sense solapament entre trames. Els resultats s'obtenen a partir de les prediccions generades pel classificador, considerant un esdeveniment com actiu quan en qualsevol de les trames utilitzades per generar un segon, es troba en estat actiu.

El *valor F* s'obté a través de la següent expressió:

$$F = \frac{2 \cdot \sum_{k=1}^{K} TP(k)}{2 \cdot \sum_{k=1}^{K} TP(k) + \sum_{k=1}^{K} FP(k) + \sum_{k=1}^{K} FN(k)}$$

on *TP* correspon amb els veritables positius (de l'anglès, *True positives), TN* correspon amb els veritables negatius (de l'anglès, *True negatives), FP* correspon amb els falsos

positius (de l'anglès, *False positives)* i *FN* correspon amb els falsos negatius (de l'anglès, *False negatives). K* fa referència al total de segments d'1 segon de duració i *k* correspon amb el segment avaluat.

La *taxa d'error ER* s'obté amb la següent expressió:

$$ER = \frac{\sum_{k=1}^{K} S(k) + \sum_{k=1}^{K} D(k) + \sum_{k=1}^{K} I(k)}{\sum_{k=1}^{K} N(k)}$$

on k és un segment d'1 segon de duració i K el nombre total de segments d'1 segon.

$N(k)$ és el nombre total d'esdeveniments actius reals, $S(k)$ és el nombre de vegades que un esdeveniment és detectat però no és l'etiqueta correcta, $I(k)$ fa referència al nombre d'esdeveniments predits pel model que no són correctes i $D(k)$ al nombre d'esdeveniments en la referència que no són correctes.

Aquests valors s'obtenen de la següent manera:

$$S(k) = \min(FN(k), FP(k))$$

$$D(k) = \max(0, FN(k) - FP(k))$$

$$I(k) = \max(0, FP(k) - FN(k))$$

En la següent taula es mostren els resultats obtinguts pel sistema de referència:

Mètrica	Valor
Valor F (%)	56.7
ER	0.69

Taula 7. Avaluació del sistema de referència

Tenint en compte que un model es considera òptim quan té un valor *F* del 100% i una taxa d'error *ER* de 0, aquest model presenta uns bons resultats, tenint en compte la simplicitat de l'estructura utilitzada pel sistema de detecció.

4 DESENVOLUPAMENT DEL SISTEMA

Prèviament, per tal d'obtenir una visió global dels tipus de sistemes desenvolupats per la tasca 3 del repte de l'any 2017, a banda d'analitzar el sistema de referència, també s'ha realitzat una anàlisi dels diferents sistemes proposats pels participants del repte [14, 15, 16, 17, 18, 19]. Concretament, aquesta anàlisi s'ha realitzat posant èmfasi en els procediments seguits en el procés d'extracció de característiques dels senyals i el tipus de tècniques d'aprenentatge automàtic utilitzades per desenvolupar el model de detecció.

Pel que fa les tècniques utilitzades per extreure les característiques dels senyals d'àudio, la tendència general ha estat utilitzar els MFCC o l'espectrograma de Mel. Ambdues tècniques són les més utilitzades actualment per caracteritzar la informació del senyal.

Per altra banda, també s'ha pogut veure que els tipus de xarxes neuronals utilitzades han sigut, en general, les xarxes neuronals convolucionals o CNN (de l'anglès, *Convolutional Neural Network*) i les xarxes neuronals recurrents o RNN (de l'anglès, *Recurrent Neural Network*).

D'aquesta manera, s'ha vist que no només existeix una solució única per desenvolupar un sistema de detecció d'àudio i que els resultats poden variar considerablement en funció dels paràmetres o les tècniques utilitzades per configurar el sistema.

Així doncs, en base les tècniques presentades i l'anàlisi realitzat del sistema de referència, així com l'anàlisi dels diversos sistemes presentats per part dels participants, en aquest apartat es procedirà a detallar les tècniques utilitzades per desenvolupar el sistema propi, així com l'avaluació del funcionament.

Finalment, es presentarà una anàlisi dels resultats obtinguts i les modificacions afegides fins a desenvolupar el sistema final.

Abans de procedir a l'explicació del desenvolupament realitzat, comentar que es pot trobar el codi del sistema en el següent enllaç: https://github.com/marcleon7/TFG2021-Sound_Event_Detection.git.

4.1 Programari utilitzat

Abans d'iniciar el desenvolupament del sistema propi, s'ha fet una recerca del programari a utilitzar i de les diferents llibreries Python necessàries per poder desenvolupar l'algorisme del sistema.

Pel que fa al programari, s'ha utilitzat Anaconda [20] per configurar l'entorn de treball amb el que s'ha desenvolupat el sistema. Aquest programari de codi obert és una distribució dels llenguatges de programació Python, entre d'altres, pel processament de dades de gran escala, analítica predictiva i computació científica. Inclou un gestor de paquets propi per tal de descarregar i configurar l'entorn de treball de forma àgil.

Pel que fa al desenvolupament del codi, s'ha escollit el llenguatge Python [21], ja que aquest llenguatge està prenent gran importància en el camp de l'aprenentatge automàtic degut a la seva versatilitat i simplicitat respecte altres llenguatges de programació.

Pel que fa a les llibreries Python utilitzades, les més destacables són les següents:

- **Soundfile** [22]: Llibreria utilitzada per importar els senyals d'àudio de la base de dades.

- **Numpy** [23]: Llibreria utilitzada per generar i emmagatzemar els vectors de característiques dels arxius d'àudio i realitzar càlculs entre vectors.

- **Matplotlib** [24]: Llibreria utilitzada per generar les gràfiques presentades en el document.

- **Librosa** [25]: Llibreria utilitzada per extreure i calcular els MFCC dels senyals d'àudio.

- **Tensorflow** [26]: Llibreria utilitzada per desenvolupar l'estructura de la xarxa neuronal, entrenar-la i realitzar les prediccions.

En la següent taula es mostra el recull dels principals recursos utilitzats, així com les respectives versions:

Recurs	Versió
Anaconda	2021.05
Python	3.8.8
Soundfile	0.10.3.post1
Numpy	1.19.2
Matplotlib	3.3.4
Librosa	0.8.0
Tensorflow	2.4.1

Taula 8. Recursos principals

4.2 Estructura del programa

El sistema s'ha estructurat en diversos arxius Python[4]. Cada arxiu conté les funcions necessàries per tal d'executar els diferents processos que formen el sistema de detecció i classificació d'esdeveniments acústics.

L'estructura de fitxers que s'ha utilitzat és la següent:

a) **main.py**

Aquest arxiu conté el flux general del sistema. Executa els diversos processos per tal de realitzar l'execució del sistema en base els paràmetres configurats en l'arxiu *params.py*.

b) **dataset.py**

Aquest arxiu conté les funcions necessàries per tal de tractar els arxius d'àudio i d'etiquetes de la base de dades. També conté el procés per preprocessar les dades, tan dels arxius d'àudio com dels arxius d'etiquetes.

c) **model.py**

Aquest arxiu conté les funcions necessàries per tal de generar i entrenar el model acústic. També conté els processos per realitzar l'avaluació i poder generar prediccions d'etiquetes.

d) **params.py**

Aquest arxiu conté un diccionari on es poden modificar els paràmetres per tal de configurar el sistema.

[4] Veure Annex B: Codi font dels processos més rellevants.

e) **utils.py**

Aquest arxiu conté funcions útils i necessàries per desenvolupar el sistema però que no tenen una relació directa amb els altres processos.

Per altra banda, el sistema conté dos arxius de text útils per la seva execució:

a) **requirements.txt**

Aquest arxiu conté un llistat de les diferents llibreries necessàries, així com les respectives versions, per poder executar el sistema.

b) **LOG.txt**

Aquest arxiu de text és generat pel sistema una vegada finalitzada la seva execució. Conté un resum de l'execució realitzada, indicant el temps d'inici i finalització i les puntuacions obtingudes pel model.

4.3 Preprocessament

En el camp de l'aprenentatge automàtic, un dels processos més importants és el preprocessament de dades.

Aquest procés es realitza de forma prèvia a l'entrenament del model per tal de facilitar el procés d'aprenentatge.

L'objectiu d'aquest procés és, per una banda, seleccionar la informació rellevant de les dades i descartar tota aquella informació irrellevant que pugui dificultar el procés d'entrenament. Per altra banda, es tradueix la informació per tal que el sistema sigui capaç de comprendre les dades i poder processar-les correctament.

Pel que fa el sistema propi, el preprocessament s'ha realitzat sobre els arxius d'àudio i sobre els arxius d'etiquetes.

4.3.1 Normalització

Els senyals d'àudio han estat enregistrats en format estèreo, per tant, la primera consideració que s'ha tingut en compte ha estat la conversió d'estèreo a mono, és a dir, reduir el nombre de canals d'àudio de dos canals a un. D'aquesta manera, el sistema treballa amb un vector d'entrada d'una dimensió.

En la següent figura es pot visualitzar un dels senyals d'àudio de la base de dades sense aquest tractament previ, és a dir, mostrant els dos canals, dret i esquerre, sense normalitzar:

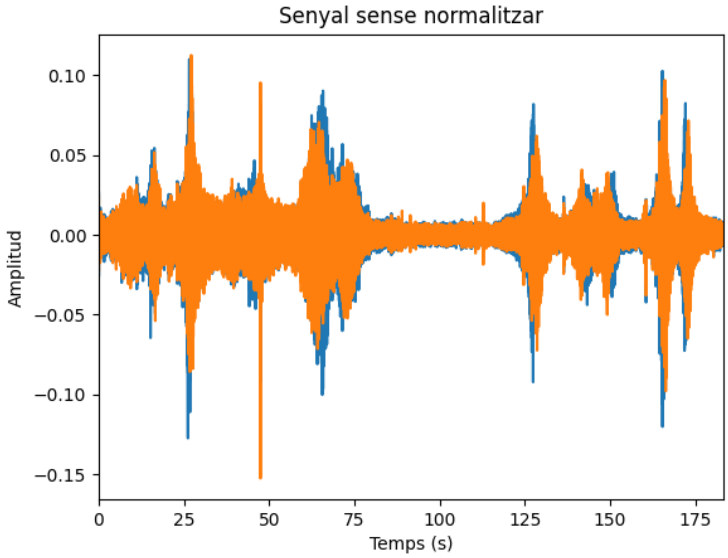

Figura 22. Senyal sense normalitzar

Per poder fer la conversió del senyal a mono, s'ha realitzat la mitjana aritmètica entre els dos canals, tal i com es pot veure en la següent expressió:

$$x_{mono} = \frac{x_d + x_e}{2}$$

on x_d es correspon amb les mostres del canal dret, x_e es correspon amb les mostres del canal esquerre i x_{mono} fa referència a les mostres del senyal obtingut després de la conversió d'estèreo a mono.

Per evitar variacions de volum sobtades i mantenir les mostres en un mateix rang de valors, s'ha decidit normalitzar les mostres del senyal en el rang de -1 a 1.

Aquesta normalització s'ha realitzat dividint les mostres dels senyals pel seu valor màxim d'amplitud, en valor absolut. Això es pot veure en la següent expressió:

$$x_{norm} = \frac{x}{max(abs(x))}$$

on x es correspon amb el vector de mostres del senyal després de la conversió d'estèreo a mono.

En la següent figura es pot veure el senyal d'àudio una vegada realitzada la conversió a mono i normalitzats els valors de l'amplitud:

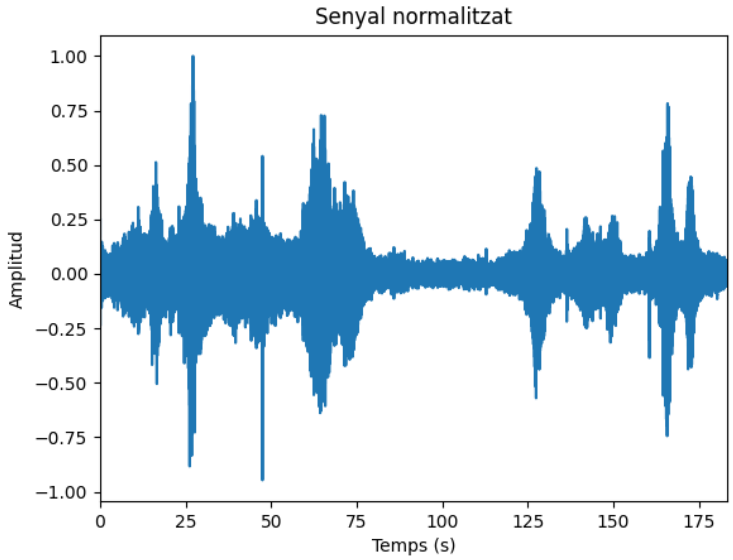

Figura 23. Senyal normalitzat

4.3.2 Extracció de característiques

Una vegada normalitzats els senyals d'àudio, es procedeix a realitzar l'extracció de característiques dels senyals. En aquest cas, s'han utilitzat els MFCC[5], ja que són molt utilitzats en el camp del processament d'àudio.

El procediment seguit per obtenir els valors dels MFCC s'ha basat en segmentar el senyal utilitzant una mida de finestra de 40ms i un desplaçament entre finestres del 50% de la mida de la finestra, és a dir, de 20ms.

Per tal de suavitzar cada un dels segments del senyal, s'ha utilitzat una finestra de tipus Hann. Aquest tipus de finestra té un bon comportament en el processament de senyals d'àudio, fent que sigui de les més utilitzades en aquest camp.

La funció de la finestra utilitzada és la següent:

$$f(n) = \frac{1}{2}\left(1 - \cos\frac{2\pi n}{N-1}\right)$$

on n és una mostra del senyal i N és el la mida de la finestra, en mostres.

Per cada un dels segments obtinguts s'ha calculat la transformada de Fourier amb finestra o STFT (de l'anglès, *Short-time Fourier transform)* i s'ha obtingut la potència de l'espectre generat.

[5] Per obtenir aquests coeficients s'ha seguit el procediment de la referència [27].

La següent figura mostra l'espectrograma obtingut per un dels senyals:

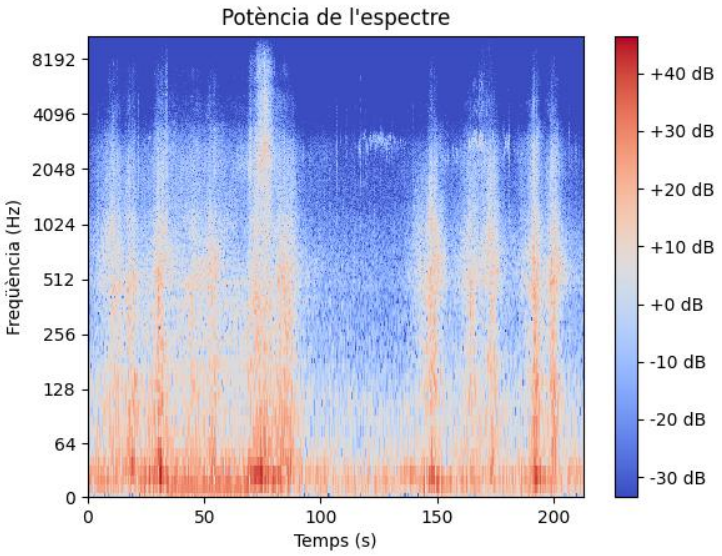

Figura 24. Espectrograma

A continuació, s'ha generat el banc de filtres Mel corresponent. En aquest cas, s'han generat un total de 40 bandes Mel.

Amb aquesta base generada, s'ha realitzat el producte amb la potència de l'espectre del senyal per tal d'obtenir l'espectrograma de Mel i poder calcular els MFCC.

El resultat obtingut ha estat utilitzat per calcular els MFCC del senyal. Concretament, s'han obtingut els primers 20 coeficients MFCC.

En la següent figura es pot visualitzar la representació dels MFCC obtinguts:

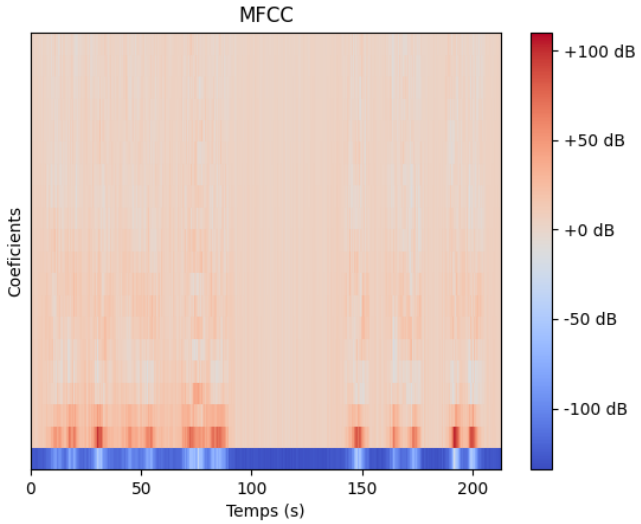

Figura 25. Coeficients MFCC

Per altra banda, també s'ha utilitzat la primera derivada, $\Delta MFCC$ i la segona derivada, $\Delta\Delta MFCC$, dels MFCC.

Els coeficients de la primera i la segona derivada dels MFCC defineixen la velocitat i l'acceleració de canvi entre els segments amb què es divideix el senyal, respectivament.

D'aquesta manera, per cada un dels segments es construeix un vector de característiques concatenant els coeficients MFCC, els coeficients $\Delta MFCC$ i els coeficients $\Delta\Delta MFCC$:

$$v_t = \{MFCC, \Delta MFCC, \Delta\Delta MFCC\}$$

En la següent figura es mostra la representació dels coeficients corresponents de la primera i la segona derivada dels MFCC:

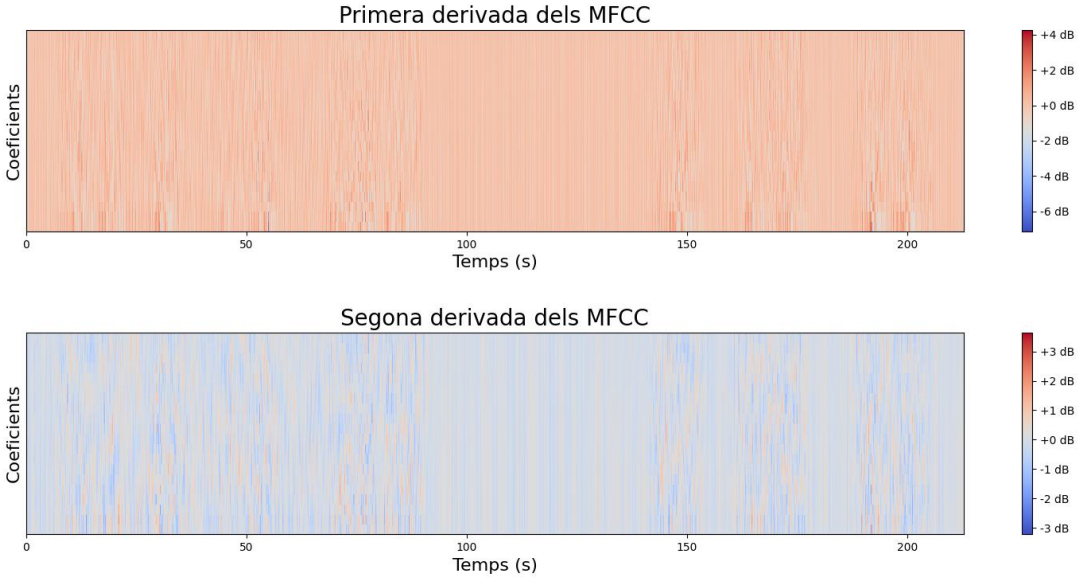

Figura 26. Primera i segona derivada dels MFCC

D'aquesta manera, per cada segment del senyal es genera un vector de 60 valors, repartits en 20 valors pels MFCC, 20 valors per la primera derivada i 20 valors més per la segona derivada.

Per altra banda, per tal de proporcionar més informació al sistema i donar a conèixer l'evolució del senyal, s'ha afegit el segment anterior i posterior al segment processat, de tal manera que l'entrada del sistema està formada per un total de tres segments.

$$v_{entrada} = \{v_{t-1}, v_t, v_{t+1}\}$$

Finalment, el vector de característiques obtingut per cada segment del senyal és el que forma l'entrada del model, és a dir, els valors que aquest ha de processar. D'aquesta manera, el vector d'entrada del model és un vector d'1 dimensió format per un total de 180 característiques, repartides en els 60 valors del vector anterior, els 60 valors del vector actual i els 60 valors del vector posterior.

Per altra banda, els valors d'entrada s'han normalitzat en el rang de -1 a 1, utilitzant el següent mètode de normalització:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

on x es correspon amb el vector de característiques, x_{min} al valor mínim del vector i x_{max} al valor màxim.

En la taula següent es mostra un recull dels paràmetres utilitzats al llarg del procés d'extracció de característiques dels senyals d'àudio:

Paràmetre	Valor
Mida de la finestra (ms)	40
Mida del desplaçament (ms)	20
Número de bandes Mel	40
MFCC	20
ΔMFCC	20
ΔΔMFCC	20
Segments afegits	2
Mida del vector de característiques	180

Taula 9. Paràmetres utilitzats pel preprocessament

4.3.3 Preprocessament dels arxius d'etiquetes

A banda de preprocessar els senyals d'àudio, també s'ha realitzat un preprocessament de les etiquetes temporals, per convertir la informació existent en les etiquetes en dades que el sistema sigui capaç de comprendre i processar.

En primer lloc, el mètode de codificació utilitzat per generar els vectors de sortida del sistema ha estat el mètode *One-Hot Encoding*. Aquest mètode es basa en generar una seqüència de 0 i 1 on cada posició dins el vector es correspon amb una classe diferent. D'aquesta manera, per tal de recuperar la classe a la que correspon la sortida, cal recuperar l'índex de la posició que es trobi en estat 1.

La sortida del sistema està formada per un vector de 6 posicions, ja que en total hi ha 6 classes possibles a detectar. D'aquesta manera, per poder identificar les classes actives dins el vector de sortida, cada una d'elles ha estat codificada amb un número enter del 0 al 5. Aquests indicadors fan referència a la posició dins el vector de sortida.

En la següent taula es mostra la classe i el seu respectiu indicador:

Etiqueta	Identificador
Brakes squeaking	0
Car	1
Children	2
Large vehicle	3
People speaking	4
People walking	5

Taula 10. Codificació de les classes

D'aquesta manera, per cada un dels segments amb que s'ha dividit el senyal, es genera un vector de 6 posicions on els possibles valors en cada una de les posicions pot ser 0 o 1. Quan s'indica el valor 0, significa que la classe no està activa en el segment processat, mentre que, si existeix una posició dins el vector amb valor 1, indica que la classe està activa en el segment.

Per exemple, una possible sortida del sistema podria ser la següent:

$$y_t = [0\ 1\ 0\ 0\ 0\ 1]$$

En aquest cas, es pot veure que hi ha dues classes actives que es corresponen amb els índex 1 i 5. Si es compara amb la taula 10, en aquest segment hi ha actives les classes *Car* i *People Walking*, és a dir, cotxes i persones caminant.

Per generar els vectors de sortida del sistema, s'ha fet ús dels arxius d'etiquetes on s'indica el temps d'inici i finalització i la classe d'esdeveniment actiu en aquest interval de temps.

D'aquesta manera, seleccionant el temps d'inici i finalització, es poden calcular els segments on es troba actiu l'esdeveniment.

El segment d'inici d'un esdeveniment es pot obtenir aplicant la següent expressió:

$$segment_{inici} = floor\left(\frac{t_{inici} * fm}{hop}\right)$$

on t_{inici} correspon amb el temps de inici en segons, fm amb la freqüència de mostreig i hop amb les mostres de desplaçament entre segments.

Pel temps de finalització, s'ha aplicat la següent expressió:

$$segment_{finalització} = ceil(\frac{t_{finalització} * fm}{hop})$$

on $t_{finalització}$ correspon amb el temps finalització en segons, fm amb la freqüència de mostreig i hop amb les mostres de desplaçament entre segments.

Finalment, s'ha creat una matriu amb un total de vectors igual al nombre de segments de senyal generats, de tal manera que per cada vector d'entrada del sistema, se li assigna el respectiu vector de sortida.

4.4 Model

El model proposat es basa en el MLP, el qual s'encarrega d'aprendre els paràmetres adequats per relacionar l'entrada del sistema i la sortida que ha de ser capaç de generar.

Una vegada entrenat el sistema, aquest ha de ser capaç de detectar i classificar els esdeveniments acústics presents en les mostres d'entrada sense la necessitat de conèixer la sortida del sistema.

En concret, s'ha utilitzat una estructura formada per una capa d'entrada, tres capes profundes i una capa de sortida que s'encarrega d'indicar les classes actives i inactives en el vector processat.

Concretament, l'entrada del sistema està formada per un total de 180 unitats, ja que, tal i com s'ha comentat en el punt *4.3.2*, l'entrada està formada pel vector de característiques del segment a processar i els segments anterior i posterior a aquest.

Les tres capes profundes del sistema estan formades per un total de 50 unitats neuronals o nodes, a les quals se'ls aplica la funció ReLu com a funció d'activació.

La capa de sortida està formada per un total de 6 neurones on cada una de les unitats fa referència a una classe. La funció d'activació que s'ha aplicat en aquesta capa ha estat la funció Sigmoide.

Per determinar l'ús de la funció Sigmoide, s'ha tingut en compte el tipus de valors que es generen a la sortida. Aquests poden ser 0 o 1, tal i com s'ha comentat en el punt *4.3.3*. Per tant, per una banda, la funció Sigmoide escala els valors en aquest rang de 0 i 1. Per altra banda, hi poden haver diferents esdeveniments succeint al mateix temps, de tal manera que, en el vector de sortida hi poden haver més d'un tipus d'esdeveniment a detectar i classificar. La funció Sigmoide tracta els valors de sortida de forma independent, de tal

manera que permet detectar més d'una classe a la vegada. Per tant, és la funció més adient per aquesta tasca.

Si en comptes d'utilitzar la funció Sigmoide s'utilitzés la funció Softmax, els resultats no serien correctes ja que, recordant el funcionament de la funció, aquesta retorna la distribució de probabilitats d'entre totes les classes possibles. Per tant, com a màxim, només es podria detectar una classe per cada segment, ja que la suma de totes les probabilitats ha de ser 1. Per tant, la funció Softmax no tindria un bon comportament en la resolució d'aquesta tasca ja que no és capaç de detectar correctament quan dos esdeveniments succeeixen al mateix instant de temps.

En la següent taula es mostra un resum de l'estructura utilitzada per generar el model de detecció i classificació:

Capa	Unitats	Funció d'activació
Entrada	180	-
Capa profunda 1	50	ReLu
Capa profunda 2	50	ReLu
Capa profunda 3	50	ReLu
Sortida	6	Sigmoide

Taula 11. Estructura del model

Per altra banda, després de cada una de les capes internes, s'ha utilitzat un *Dropout* amb un valor del 20% per reduir el sobreentrenament del model.

4.5 Entrenament del sistema

Per entrenar el sistema s'ha utilitzat el mètode de validació creuada. Tal i com està preparada la base de dades, s'ha realitzat un entrenament per cada una de les subdivisions amb la que es divideix la base de dades. En total, s'han realitzat 4 entrenaments.

Per tal que el sistema sigui capaç de trobar els paràmetres òptims, s'ha configurat l'entrenament tenint en compte els següents hiperparàmetres:

a) **Rang d'aprenentatge**

El rang d'aprenentatge (en anglès, *learning rate*), és un hiperparàmetre utilitzat en el camp de l'aprenentatge automatitzat per tal d'optimitzar els models.

Aquest paràmetre s'encarrega de determinar la mida del pas que es fa en cada iteració durant l'entrenament, és a dir, controla l'increment que es realitza en l'actualització dels paràmetres de la unitat neuronal per tal de minimitzar la funció de cost utilitzada pel sistema.

En aquest sentit, un rang d'aprenentatge massa baix pot suposar que el sistema necessiti una gran quantitat de temps per optimitzar model, cosa que pot fer que augmenti el cost computacional. Per altra banda, un rang d'aprenentatge massa alt pot suposar que el sistema no sigui capaç de trobar el mínim de la funció de cost i, per tant, no sigui capaç de trobar els paràmetres òptims per donar solució al problema plantejat [28].

Pel sistema desenvolupat, el rang d'aprenentatge que millors resultats ha donat ha sigut de 0.001.

b) **Nombre d'iteracions**

El nombre d'iteracions (en anglès, e*pochs)*, fa referència a les vegades que l'algorisme processa les dades d'entrada i actualitza els paràmetres.

Un nombre d'iteracions baix pot fer que el sistema no sigui capaç de minimitzar la funció de cost i, per tant, no poder optimitzar els paràmetres per tal d'obtenir bons resultats. En canvi, un nombre d'iteracions elevat pot desajustar el sistema, de tal manera que, a mesura que avancen les iteracions, el sistema trobi la solució en una de les iteracions però, degut a que el sistema continua calculant l'optimització dels paràmetres, s'allunyi de la solució òptima [29].

El nombre d'iteracions que s'ha utilitzat per entrenar el sistema ha estat de 50 iteracions.

c) **Mida del lot**

La mida del lot (en anglès, *batch size*), és un hiperparàmetre que s'utilitza per definir la quantitat de dades a processar abans d'actualitzar els paràmetres.

Quan no s'utilitza una mida de lot, el procés actualitza els paràmetres interns per cada una de les entrades del sistema, cosa que pot provocar un augment del cost computacional i del temps d'execució [29].

Per l'entrenament s'ha utilitzat una mida de lot de 256 vectors, de tal manera que, el model processa 256 vectors de característiques abans de procedir al càlcul de l'error i l'actualització dels paràmetres interns. S'ha utilitzat aquesta mida de lot tenint en compte el cost computacional requerit per processar les dades. D'aquesta manera, s'ha minimitzat el temps d'execució, ja que el nombre de segments a processar és elevat.

d) **Optimització**

L'optimitzador escollit ha estat l'optimitzador Adam. Es tracta d'un mètode de descens per gradient estocàstic que es basa en l'estimació adaptativa de moments.

Tal i com s'ha explicat en el punt *2.1.6.3*, el mètode del gradient descendent utilitza un únic rang d'aprenentatge durant tot l'entrenament. L'optimitzador Adam és capaç de modificar i adaptar el rang d'aprenentatge segons els resultats obtinguts al llarg de l'entrenament. En aquest sentit, adapta el rang d'aprenentatge segons la distribució i l'actualització dels paràmetres, de tal manera que quan els paràmetres es troben molt dispersos, el rang d'aprenentatge augmenta [30].

e) **Funció de cost**

La funció de cost és un hiperparàmetre que s'utilitza per determinar l'error entre els valors estimats i els valors reals. Amb l'ús d'aquesta funció es pretén optimitzar els paràmetres de la xarxa neuronal per tal que generin l'error mínim possible en base altres hiperparàmetres establerts.

La funció de cost que s'ha utilitzat ha estat la funció d'entropia creuada binària [30].

Aquesta funció és utilitzada en l'àmbit de la classificació multietiqueta i s'acostuma a utilitzar quan la capa de sortida del model té com a funció d'activació la funció Sigmoide.

L'expressió d'aquesta funció és la següent:

$$L = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(p_i) + (1 - y_i)\log(1 - p_i)$$

on y_i és el valor real a generar, p_i és la sortida generada pel sistema i n és el nombre total de classes.

En la següent taula, es mostra un resum dels diferents hiperparàmetres utilitzats per configurar l'entrenament del sistema:

Hiperparàmetre	Valor
Rang d'aprenentatge	0.001
Nombre d'iteracions	50
Mida de lot	256
Optimitzador	Adam
Funció de cost	Funció d'entropia creuada binària

Taula 12. Hiperparàmetres d'entrenament

4.6 Postprocessament

El postprocessament de dades ha consistit en recuperar les prediccions que el sistema ha generat per cadascun dels segments d'àudio i construir els arxius d'etiquetes indicant la classe d'esdeveniment actiu en l'interval de temps.

Per realitzar el postprocessament, s'han tingut en compte dos factors a l'hora de generar les etiquetes per tal de compensar els errors de predicció del model.

En primer lloc, s'ha tingut en compte l'interval de temps entre dos esdeveniments de la mateixa categoria. Concretament, s'ha decidit que si dos esdeveniments de la mateixa classe es troben separats per un interval de temps menor a 0.5 segons, es consideren un mateix esdeveniment. De tal manera que, a l'hora de construir l'etiqueta de l'esdeveniment, aquests es consideren un mateix esdeveniment.

Aquest criteri s'ha decidit basant-se en la principal limitació que tenen els sistemes de detecció i classificació d'esdeveniments acústics. No són capaços de diferenciar dos esdeveniments de la mateixa classe superposats en el temps, de tal manera que el sistema de detecció els identificarà com a un únic esdeveniment.

Analitzant els arxius d'etiquetes de la base de dades, s'ha vist que hi ha esdeveniments de la mateixa classe separats un instant de temps menor a 1 segon. Tot i ser un interval de temps petit, ha de ser considerat suficient com perquè el sistema els hagi de detectar com a dos esdeveniments diferents. El nombre d'esdeveniments separats per un interval de temps menor a 0.5 segons és mínim, per tant, s'ha decidit utilitzar aquest temps per determinar la distància entre dos esdeveniments diferents de la mateixa categoria.

En segon lloc, una vegada realitzat el pas anterior, per tal de generar l'etiqueta final per cada un dels esdeveniments, s'ha tingut en compte la seva durada. Concretament, el criteri seguit per generar l'etiqueta final ha estat que, si l'esdeveniment té una durada inferior a 1.5 segons, es determina que és un esdeveniment mal generat pel sistema i que, per tant, s'ha d'eliminar de la llista d'etiquetes.

Aquests criteris s'han decidit a través d'una anàlisi de les etiquetes existents en la base de dades. Pràcticament, les anotacions que figuren a la base de dades tenen una durada elevada, sent les de menor durada, pròximes a 1.5 segons. Per tant, s'ha vist que és un

bon límit per diferenciar els esdeveniments correctament detectats dels que no s'han detectat correctament.

Per tal de determinar el nom de la etiqueta, cal recordar que, el mètode utilitzat per codificar les classes ha estat el *One-hot encoding*, del qual se n'ha parlat en el punt *4.3.3*. Per tant, per cada segment, s'han recuperat les posicions en estat actiu del vector i s'han comparat amb el valor de l'indicador de la classe corresponent. D'aquesta manera, s'ha pogut obtenir el nom de cada classe per generar les etiquetes.

La següent figura mostra algunes de les prediccions generades pel sistema durant un entrenament sense aplicar el postprocessament. No es mostren totes les prediccions generades ja que, sense aquest processament, la llista d'etiquetes generada és extensa. Tal i com es pot veure, els esdeveniments tenen una durada molt curta, de tal manera que, sense tractar les etiquetes, cada un dels esdeveniments detectats es tracta com un esdeveniment nou quan realment poden pertànyer al mateix.

```
audio/street/b099.wav   0.0     0.08    car
audio/street/b099.wav   0.12    0.24    car
audio/street/b099.wav   0.36    0.42    car
audio/street/b099.wav   0.44    0.5     car
audio/street/b099.wav   0.54    0.92    car
audio/street/b099.wav   0.96    1.12    car
audio/street/b099.wav   1.14    1.26    car
audio/street/b099.wav   1.3     1.36    car
audio/street/b099.wav   1.4     1.44    car
audio/street/b099.wav   1.46    1.54    car
audio/street/b099.wav   1.56    1.6     car
audio/street/b099.wav   1.76    1.82    car
audio/street/b099.wav   1.84    1.9     car
audio/street/b099.wav   1.92    2.04    car
```

Figura 27. Prediccions sense postprocessar per l'arxiu b099.wav

En la següent figura es mostra les prediccions generades durant un entrenament amb el mateix senyal que l'utilitzat per generar la llista anterior però aplicant el postprocessament sobre les etiquetes. En aquest cas, sí que es mostra tota la llista generada. Es pot veure, els intervals són majors respecte els intervals anteriors ja que el sistema detecta, segons els paràmetres establerts, quins esdeveniments formen part del mateix i quins no.

```
audio/street/b099.wav   0.02    12.5    car
audio/street/b099.wav   18.5    20.42   car
audio/street/b099.wav   38.78   40.96   car
audio/street/b099.wav   42.76   45.66   car
audio/street/b099.wav   88.52   97.6    car
audio/street/b099.wav   101.34  106.8   car
audio/street/b099.wav   107.86  120.8   car
audio/street/b099.wav   121.44  137.24  car
audio/street/b099.wav   137.94  147.66  car
audio/street/b099.wav   156.58  175.9   car
audio/street/b099.wav   176.92  183.24  car
audio/street/b099.wav   192.86  197.24  car
audio/street/b099.wav   205.7   216.06  car
audio/street/b099.wav   221.88  234.68  car
audio/street/b099.wav   235.38  240.58  car
```

Figura 28. Prediccions postprocessades per l'arxiu b099.wav

4.7 Avaluació del sistema

Per realitzar l'avaluació del sistema existeixen diferents mètriques [31, 32] utilitzades per obtenir una visió acurada del comportament del model. D'aquesta manera, es pot obtenir una visó global d'aquelles dades que el sistema és o no capaç de processar correctament.

Abans d'entrar en detall del procediment seguit per avaluar el sistema, cal dir que el mètode utilitzat per l'avaluació no és el mateix que el que utilitza el sistema de referència. Tal i com s'ha comentat, el sistema utilitza segments d'1 segon per obtenir les diferents mètriques.

Per avaluar el model desenvolupat, degut a la limitació de temps, s'ha decidit realitzar l'avaluació utilitzant els segments predits pel sistema per cada un dels segments de senyal, en comptes de realitzar l'avaluació amb segments de 1 segon.

A més, l'avaluació s'ha fet una vegada finalitzat l'entrenament per les 4 subdivisions de la base de dades, obtenint el funcionament general del sistema fent la mitja entre els resultats obtinguts pels 4 entrenaments. És a dir, s'han obtingut les puntuacions per cada un dels entrenaments realitzats i s'ha calculat la mitja per cada una de les mètriques calculades.

Per altra banda, en problemes de classificació, és molt habitual utilitzar la matriu de confusió. En aquest cas, la matriu de confusió s'ha construït amb els veritables positius i negatius i els falsos positius i negatius per tal de veure les prediccions correctes i incorrectes.

Els veritables positius són aquells esdeveniments predits pel sistema com actius correctament.

Els veritables negatius són considerats els esdeveniments classificats com inactius correctament.

Els falsos positius són aquells esdeveniments classificats com actius quan realment haurien d'haver estat predits com inactius.

Els falsos negatius són aquells esdeveniments que el sistema prediu com inactius quan en realitat haurien d'haver estat predits com actius.

A continuació es mostra la matriu de confusió del model. Els diferents valors que es presenten s'han obtingut fent la mitja entre els valors resultants dels 4 entrenaments.

		Valors Predits	
		Inactiu (0)	Actiu (1)
Valors Reals	Inactiu (0)	318837	27240
	Actiu (1)	44228	24400

Taula 13. Matriu de confusió

Per altra banda, per obtenir una visió més acurada del comportament real del sistema, s'han utilitzat les següents mètriques per la seva avaluació:

a) **Exactitud**

El valor d'exactitud (en anglès, *Accuracy*), indica l'encert del sistema. Per tal d'obtenir el valor, es tenen en compte les classificacions realitzades correctament respecte el total d'esdeveniments classificats.

Aquest valor s'ha obtingut utilitzant la següent expressió:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

En aquest cas, el valor indica els esdeveniments que han estat classificats com actius i inactius correctament.

Aquest valor és dels més utilitzats per avaluar un sistema d'aprenentatge automàtic però no és dels més fiables, de tal manera que, un model que classifica amb molta exactitud una classe concreta i la resta de classes no, podria indicar un equivocat correcte funcionament del sistema.

b) **Exhaustivitat**

El valor d'exhaustivitat (en anglès, *Recall*), és una mètrica que indica la quantitat de classificacions positives, en aquest cas, la quantitat d'esdeveniments actius identificats correctament entre el total d'esdeveniments actius reals. Relaciona els veritables positius amb el total de veritables positius i falsos negatius.

Per obtenir el valor d'aquesta mètrica, s'ha utilitzat la següent expressió:

$$recall = \frac{TP}{TP + FN}$$

c) **Precisió**

La precisió (en anglès: *Precision*) s'utilitza per mesurar la qualitat del model en tasques de classificació. Identifica, d'entre el total d'esdeveniments classificats com actius quins, s'han indicat correctament com actius.

Aquesta mètrica s'obté a partir de la següent expressió:

$$precision = \frac{TP}{TP + FP}$$

d) **Valor F**

El valor F s'utilitza per tal relacionar les mètriques de precisió i exhaustivitat en un únic valor. Això es fa per poder comparar el rendiment del model.

Per tal d'obtenir aquest valor, s'ha utilitzat la següent expressió:

$$F = 2 \cdot \frac{recall \cdot precision}{recall + precision}$$

En models òptims, el valor F ha de ser pròxim a 1, expressant el resultat en una escala de 0 a 1.

e) **Taxa d'error**

La taxa d'error ER s'utilitza per analitzar, tal i com indica el seu nom, l'error produït en funció de les insercions I, supressions D i substitucions S.

Pel càlcul d'aquest valor, s'ha utilitzat la següent expressió:

$$ER = \frac{S + D + I}{N}$$

on N és el nombre d'esdeveniments actius reals.

Per obtenir les substitucions, les supressions i les insercions, s'han utilitzat les següents expressions:

$$S = \min(FN, FP)$$

$$D = \max(0, FN - FP)$$

$$I = \max(0, FP - FN)$$

En models òptims, el valor de la taxa d'error ha de ser pròxim a 0.

Per tal d'obtenir els resultats generals del sistema, s'han utilitzat les dades d'avaluació per obtenir els valors de les diverses mètriques utilitzades.

Una vegada presentades les diferents mètriques utilitzades per avaluar el sistema, en la següent taula es mostra un recull dels diferents resultats obtinguts per cada una de les mètriques:

Mètrica	Valor
Exactitud (%)	82.64
Exhaustivitat (%)	37
Precisió (%)	48.39
Valor F (%)	40.37
Taxa d'error	0.73

Taula 14. Avaluació del model

4.8 Anàlisi dels resultats

El sistema presentat al llarg d'aquest desenvolupament ha estat resultat de diverses modificacions realitzades sobre el sistema de detecció i classificació al llarg del projecte per veure com influeix en els resultats l'ús de diverses configuracions en el procés d'extracció de característiques.

En aquest punt, es presentaran els diferents resultats obtinguts al llarg del projecte fins arribar al sistema que més bons resultats ha obtingut i que, com es pot intuir, el sistema més òptim ha estat el presentat en al llarg del punt *4*.

En primer lloc, com a primera configuració per generar els vectors de característiques, s'ha optat per generar vectors d'entrada formats pels coeficients MFCC, és a dir, per cada un dels segments amb els que s'han dividit els senyals d'àudio s'ha obtingut un vector format pels 20 primers coeficients MFCC.

Una vegada finalitzat l'entrenament amb aquesta configuració, s'ha pogut veure com el sistema presenta un problema de sobreajustament, és a dir, és capaç de classificar correctament les dades d'entrenament però disminueix considerablement l'encert sobre les dades d'avaluació.

En la següent figura es presenta la gràfica d'encert i de pèrdua del primer sistema desenvolupat:

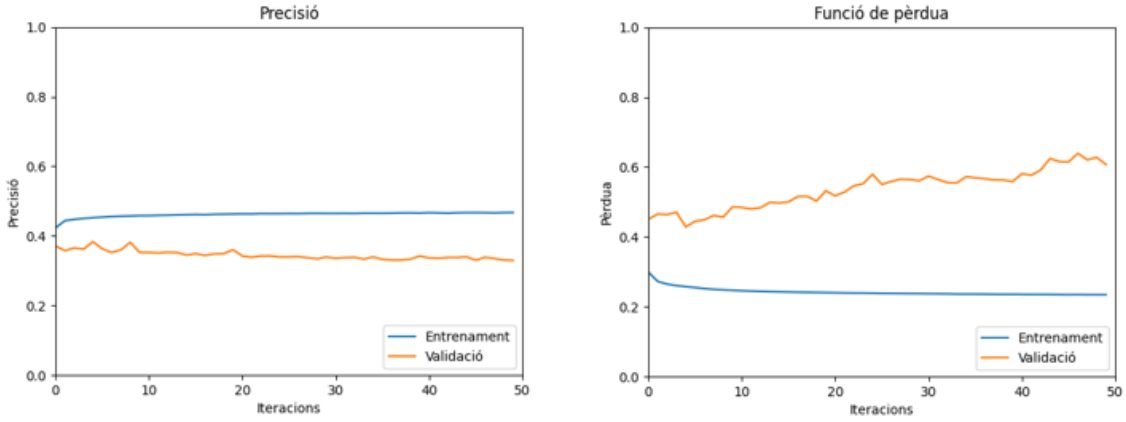

Figura 29. Precisió i pèrdua del primer sistema

En un comportament ideal, a mesura que avança l'entrenament, l'encert hauria d'augmentar pràcticament fins a 1, mentre que la funció de cost, hauria de veure's reduïda fins pràcticament a 0. A més, els resultats, tan per les dades d'entrenament com per les dades d'avaluació, haurien de ser semblants. Normalment, l'encert del sistema acostuma a ser menor sobre les dades d'avaluació que sobre les dades d'entrenament, ja que són les dades no vistes pel sistema, però, la diferència ha de ser mínima.

En aquest cas, es pot veure que les dades d'entrenament mostren una evolució uniforme. L'encert augmenta a mesura que avancen les iteracions d'entrenament mentre que la funció de pèrdua es veu reduïda. Per contra, sobre les dades d'avaluació, el comportament es mostra contrari. Mentre que l'encert disminueix a mesura que avança l'entrenament, la funció de pèrdua augmenta.

A partir d'aquests primers resultats, els esforços s'han centrat en buscar una configuració que permeti, per una banda, augmentar la puntuació del sistema però, més important encara, reduir el problema de sobreentrenament.

Aquestes modificacions s'han basat en augmentar la quantitat d'informació que es mostra al sistema, és a dir, oferir més característiques d'entrada per tal que el sistema pugui tenir un context més ampli del contingut del segment del senyal processat.

La primera de les modificacions s'ha basat en afegir la primera i la segona derivada dels coeficients MFCC, tal i com s'ha explicat en el punt *4.3.2*. D'aquesta manera, s'ha augmentat el vector de característiques, per cada segment generat, de 20 a 60 característiques, ja que s'han afegit 20 coeficients MFCC, 20 coeficients de la primera derivada i 20 coeficients de la segona derivada.

Amb aquesta primera modificació s'han obtingut els resultats següents de l'entrenament:

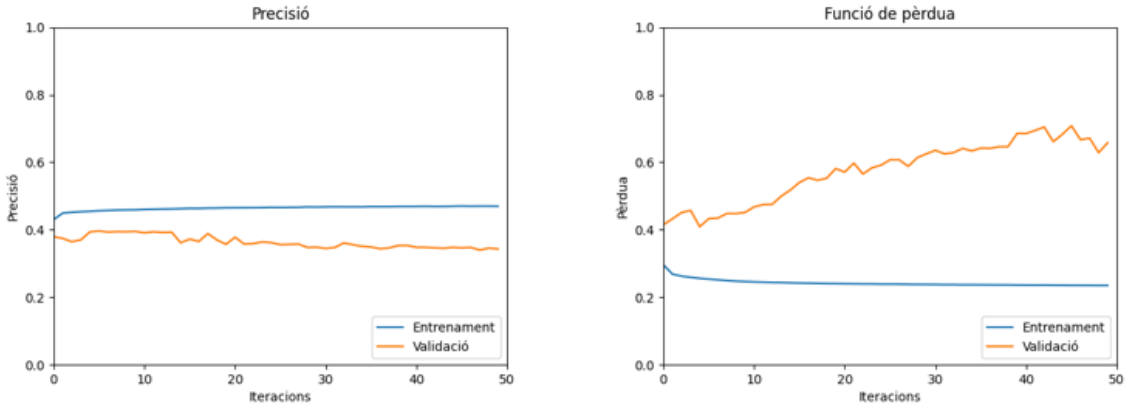

Figura 30. Precisió i pèrdua del segon sistema

Tal i com es pot veure, aquesta solució tampoc ha donat solució al problema de sobreentrenament que presenta el sistema. Si que es pot veure que, en les 10 primeres iteracions d'entrenament, el sobreajustament no és tan elevat però, a mesura que avancen les iteracions, l'encert segueix disminuint i la funció de cost augmenta sobre les dades d'avaluació.

Finalment, l'última modificació realitzada ha estat l'explicada al llarg del desenvolupament, en aquest cas, s'ha afegit a l'anterior modificació el segment anterior i posterior al segment processat, de tal manera que no només es mostra al sistema la velocitat i l'acceleració de canvi entre segments, sinó que s'afegeix informació addicional dels segments anteriors i posteriors.

En aquest cas, els resultats obtinguts de l'entrenament del sistema amb aquesta configuració han estat els següents:

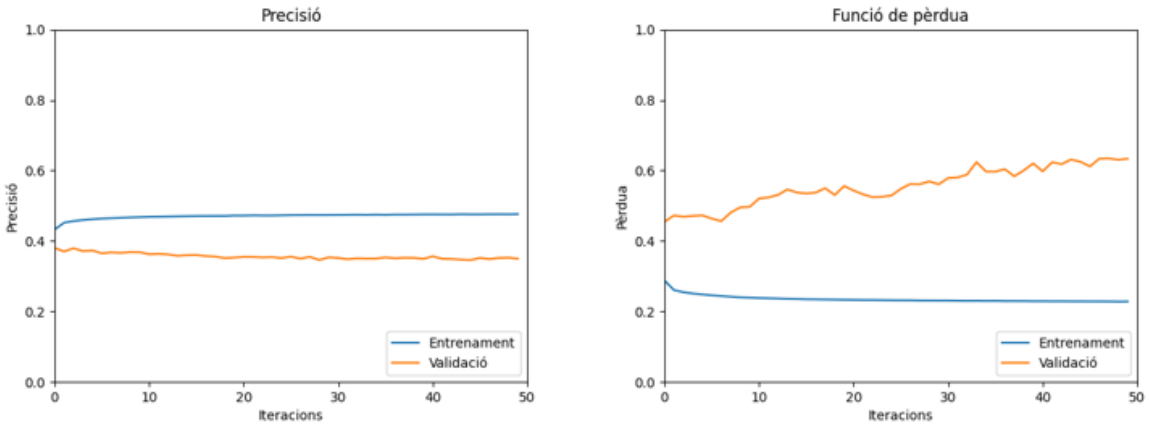

Figura 31. Precisió i pèrdua del tercer sistema

Com es pot veure, aquesta modificació ha permès augmentar lleugerament els resultats d'entrenament però no de forma significativa per afirmar que s'ha solucionat el poblema de sobreajustament del sistema.

Una vegada analitzades les modificacions realitzades per desenvolupar el sistema final, es mostra una taula comparativa dels resultats obtinguts pels tres sistemes:

Mètrica	MFCC	MFCC + ΔMFCC + ΔΔMFCC	MFCC + ΔMFCC + ΔΔMFCC + SEGMENTS
Exactitud (%)	82.27	82.38	82.64
Exhaustivitat (%)	36.14	37.05	37
Precisió (%)	46.76	47.25	48.39
Valor F (%)	39.39	40.26	40.37
Taxa d'error	0.74	0.73	0.73

Taula 15. Comparativa de les modificacions

Per analitzar el funcionament real, cal centrar-se en els valors F, l'exhaustivitat i la precisió, ja que es centren en analitzar els esdeveniments classificats com actius.

En primer lloc, el valor obtingut d'exhaustivitat pel sistema final és del 37%. Aquest valor relaciona els esdeveniments detectats pel sistema com actius respecte el total d'esdeveniments actius reals. En aquest cas, indica que només s'han detectat un 37% d'esdeveniments actius, respecte el total d'esdeveniments actius que hauria d'haver estat capaç de detectar.

En segon lloc, la precisió del model és del 48.39%. Aquest indicador mostra, dels esdeveniments detectats pel sistema com a actius, quins han estat detectats correctament com actius. És a dir, que un 48.39% dels esdeveniments detectats com actius pel sistema són esdeveniments detectats correctament.

En tercer lloc, tal i com s'ha explicat, per relacionar els dos indicadors anteriors, s'utilitza el valor F. En aquest cas, el valor F és del 40.37%. En models òptims, aquest valor hauria de ser pròxim al 100%, per tant, indica que el funcionament del sistema no és bo.

Finalment, com a taxa d'error ER, el valor obtingut és del 0.73. Aquest valor no pot ser expressat en percentatge, ja que, en certs casos, podria ser superior a 1. Per altra banda, en models òptims, el valor de la taxa d'error hauria de ser pròxim a 0. En aquest cas, el

valor obtingut és elevat, en comparació amb el cas d'un model òptim, cosa que fa reafirmar el baix encert del sistema sobre les dades d'avaluació.

A més, tot i aplicant un dropout entre les capes que forma la xarxa neuronal, el sobreentrenament no s'ha pogut reduir.

Per altra banda, per avaluar la precisió del sistema durant l'entrenament, en tasques de classificació multietiqueta, és habitual utilitzar la precisió binària [33] per mesurar l'encert del sistema. Aquesta mètrica analitza valor per valor si coincideix amb el valor corresponent del segment real.

Aquesta mètrica d'encert no mostra el comportament real del sistema ja que, per exemple, en segments en que només hi ha un esdeveniment actiu i la resta no, si la predicció indica que no hi ha esdeveniments actius, la precisió seguirà sent alta, però no mostra la realitat del sistema.

És per això que, per generar les gràfiques mostrades en aquest apartat, s'ha utilitzat la mètrica de precisió. Aquesta mètrica compara vector a vector, de tal manera que, es considera un resultat correcte quan tots els valors coincideixen amb el vector real.

Tot i així, s'ha trobat interessant comparar aquestes dues mètriques per mostrar que una mala interpretació dels resultats pot portar a pensar que el sistema funciona correctament, quan realment no ho fa.

La figura 32 ha estat obtinguda utilitzant la precisió binària com a mètrica per mesurar l'encert del sistema amb els mateixos paràmetres que el sistema final presentat en aquest desenvolupament. Només s'ha canviat el tipus de mètrica per avaluar l'encert del sistema.

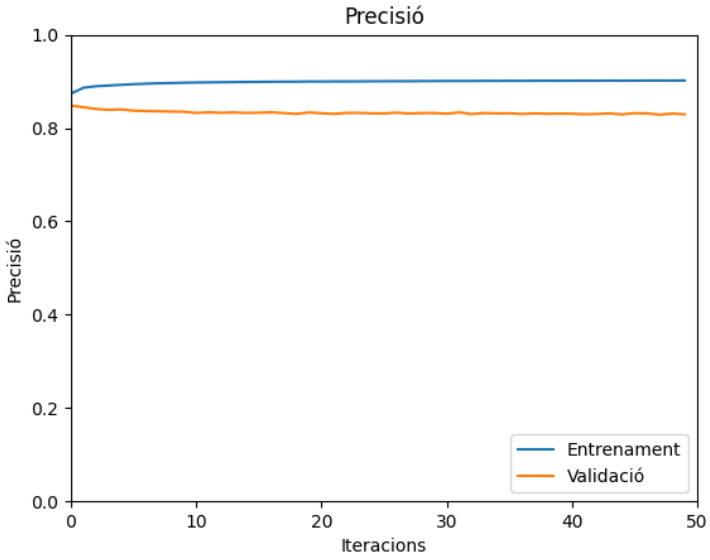

Figura 32. Precisió binària

Analitzant la figura anterior, es pot veure com el sistema sembla que presenti uns molt bons resultats, però, si es compara amb la figura 31, és pot veure que la realitat no és aquesta.

Utilitzant la precisió binària, s'obté un encert del 90.21%, mentre que utilitzant com a mètrica la precisió, s'obté un encert del 47.63% per les dades d'entrenament. Pel que fa les dades d'avaluació, l'encert és del 82.69% en el cas de la precisió binària i del 34.99% en el cas d'utilitzar la precisió com a mètrica d'encert. Per tant, es veu que, segons la mètrica a utilitzar, cal posar èmfasi a la interpretació realitzada.

En relació al comportament general del sistema, analitzant el comportament sobre les dades d'entrenament, es pot veure que arriba a optimitzar els paràmetres amb molt poques iteracions d'entrenament, de tal manera que, tan la precisió com la funció de cost, varien molt poc a mesura que avança l'entrenament. Això és un comportament a tenir en compte, ja que, això mostra que el sistema és capaç de trobar la solució òptima en base els hiperparàmetres establerts amb molt poques iteracions. Per contra, tal i com s'ha vist, el problema es troba sobre les dades d'avaluació, és a dir, les no vistes prèviament pel sistema. Sobre aquestes, el sistema no és capaç de generalitzar correctament.

5 CONCLUSIONS

Al llarg d'aquest projecte s'ha pogut conèixer el funcionament dels sistemes encarregats de la detecció i la classificació de senyals acústics, concretament en tasques de detecció i classificació de sons superposats temporalment. En concret, s'ha fet una anàlisi de la tasca 3 del repte *DCASE Challenge 2017*: *Sound event detection in real life audio*, així com el sistema de referència i s'ha fet ús de la base de dades proporcionada per l'organització del repte per proposar un sistema de detecció i classificació propi.

Per altra banda, l'objectiu d'aquest projecte no només ha estat desenvolupar el sistema, sinó que s'ha intentat elaborar un projecte que pugui ser utilitzat com a introducció al camp de l'aprenentatge automàtic. En aquest sentit, s'ha pogut analitzar el procediment que cal seguir per plantejar una solució a una tasca d'aquestes característiques, completant així cada una de les fases del problema.

Pel que fa a les dificultats trobades al llarg del projecte, destacar que la falta d'experiència prèvia ha sigut determinant per poder trobar una solució al principal problema que presenta el model, el sobreajustament. S'ha hagut de destinar una quantitat notable de temps per analitzar aquest problema, sense trobar una solució concreta. En aquest sentit, cal dir que no hi ha una solució única a la tasca i que, per tant, les configuracions que els usuaris realitzin del sistema poden modificar notablement el seu comportament.

Seguint amb les dificultats presentades, destacar que l'equip utilitzat per desenvolupar un sistema d'aprenentatge automàtic, concretament, durant el procés d'entrenament, té un paper molt important en relació al cost computacional. D'aquesta manera, s'ha vist que, segons el tipus d'equip utilitzat, el temps necessari per processar totes les dades es pot veure afectat, sent molt elevat en equips de menors característiques.

Així doncs, es pot afirmar que els objectius plantejats en el punt *1.4* han estat assolits. Destacar que, tot i el problema de sobreajustament que presenta el model, s'ha pogut desenvolupar un sistema que realitza totes les fases per donar una solució, ja sigui millor o pitjor, a la tasca plantejada.

Per concloure, m'agradaria destacar que aquest projecte ha estat tot un repte. Tot i no tenir una gran experiència prèvia sobre el tema, considero que he estat capaç de desenvolupar un sistema que cobreix totes les fases per donar solució a una tasca en concret. Per altra banda, aquest projecte m'ha motivat a aprofundir i descobrir un gran interès sobre el camp de l'aprenentatge automàtic i el processament de senyals d'àudio.

5.1 Pressupost

A banda de les conclusions generals que es poden extreure d'aquest projecte, també s'ha pogut plantejar un possible pressupost per obtenir una visió general dels costos econòmics que comporta la realització d'aquest projecte.

Aquest projecte no requereix una gran inversió a nivell de material, ja que, el programari utilitzat és d'accés lliure o de codi obert, per tant, tots els recursos utilitzats resten a disposició de qualsevol usuari.

Així doncs, el pressupost general d'aquest projecte es basa en el sou mig d'un enginyer, en aquest cas, de categoria *Junior*, concretament, a Espanya, que és on s'ha realitzat el desenvolupament del projecte. Concretament, els càlculs d'aquests costos, s'han realitzat tenint en compte que s'ha destinat un total de 20 hores a la setmana a la resolució d'aquest projecte, distribuïdes en 19 setmanes.

Per altra banda, també s'han afegit els costos del tutor del projecte. Per obtenir els seus honoraris, s'ha tingut en compte la seva categoria professional, considerat com enginyer *Senior*.

A més, per tal d'obtenir una visió per que fa el cost d'execució dels entrenaments del sistema, s'ha fet una estimació utilitzant el servei *Google Cloud Platform* [34]. Concretament, s'ha realitzat el càlcul utilitzant una configuració bàsica de GPU i tenint en compte les hores destinades a la execució del sistema.

Finalment, tenir en compte que els costos s'han obtingut a partir del preu/hora, per tant, s'ha fet una estimació de les hores totals requerides per desenvolupar el projecte per tal d'obtenir el cost total final.

En la següent taula es mostra el resum del pressupost realitzat per aquest projecte:

Recurs	Dedicació (hores)	Cost hora (€/hora)	Total (€)
Enginyer *Junior*	380	12	4.560
Enginyer *Senior*	38	25	950
Servidor	150	0,76	114
Total	**-**	**-**	**5.624 €**

Taula 16. Pressupost

5.2 Propostes d'estudi

El sistema proposat en aquest projecte s'ha basat en una de les estructures més bàsiques, el Perceptró Multicapa. Tanmateix, existeixen altres tipus d'estructures que poden ser utilitzades per donar solució a la mateixa tasca que la realitzada en aquest projecte.

Algunes de les estructures més utilitzades són les *Xarxes Neuronals Convolucionals* o les *Xarxes Neuronals Recurrents*, de les quals no s'ha parlat en aquest projecte.

Per altra banda, tal i com s'ha comentat, el model desenvolupat presenta sobreajustament, és a dir, no és capaç de generalitzar correctament sobre totes les dades, disminuint l'encert sobre les dades no vistes.

Com a possible alternativa, es proposa utilitzar una de les estructures esmentades anteriorment per tal d'avaluar el comportament utilitzant diferents estructures, així com intentar solucionar el problema de sobreajustament que presenta el sistema d'aquest projecte. D'aquesta manera, es podria determinar quines estructures poden ser més adients per ser utilitzades per resoldre aquesta tasca.

Un dels punts en que l'organització del repte DCASE posa més èmfasi, és en els mètodes d'avaluació del model. Concretament, es proposa un programa, desenvolupat en Python, per avaluar el comportament del sistema.

Degut a la limitació de temps, no s'ha pogut analitzar el sistema d'avaluació que s'utilitza ni entrar en detall sobre el funcionament del programa utilitzat avaluar els models. És per això que es proposa analitzar aquests mètodes d'avaluació i implementar-los per tal de comparar els resultats amb els sistemes presentats per aquesta tasca en concret.

Finalment, l'última alternativa proposada fa referència al procés d'extracció de característiques dels senyals. Per aquest projecte, els vectors de característiques generats es basen en els MFCC però, existeixen altres mètodes, com per exemple, l'espectrograma de Mel. Per tant, es proposa utilitzar algun mètode d'extracció de característiques diferent a l'utilitzat en aquest projecte per tal d'analitzar com influeixen aquests mètodes sobre l'entrenament del model.

BIBLIOGRAFIA

[1] Aprendizaje automático. A: *Wikipedia*. [En línia]. Wikimedia Foundation, 2021. [Consulta: 3 abril 2021]. Disponible a: <https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico>

[2] Aprendizaje profundo. A: *Wikipedia*. [En línia]. Wikimedia Foundation, 2021. [Consulta: 3 abril 2021]. Disponible a: <https://es.wikipedia.org/wiki/Aprendizaje_profundo>

[3] Mirjalili, V.; Raschka, S. Entrenar algoritmos simples de aprendizaje automático para clasificación. A: Mirjalili, V.; Raschka, S. *Python Machine Learning.* [En paper] Marcombo, 2019. ISBN 9788426727206.

[4] Perceptrón multicapa. A: *Wikipedia.* [En línia]. Wikimedia Foundation, 2021. [Consulta 5 abril 2021]. Disponible a: <https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa>

[5] Calvo, D. Función de activación – Redes Neuronales. A: *Diegocalvo*. [En línia]. Desembre, 2018. [Consulta: 20 maig 2021]. Disponible a: <https://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/>

[6] Serokell. Machine Learning Optimization Methods and Techniques. [En línia] A: *BetterProgramming*. [Consulta: 5 juny 2021]. Disponible a: <https://betterprogramming.pub/machine-learning-optimization-methods-and-techniques-56f5a6fc5d0e>

[7] Ellis. D.; Plumbley, M. D.; Virtanen, T. *Computational Analysis of Sound Scenes and Events*. [Electrònic]. Springer, 2018. ISBN 9783319634494.

[8] MFCC. A: *Wikipedia* [En línia]. Wikimedia Foundation, 2020. [Consulta: 15 març 2021]. Disponible a: <https://es.wikipedia.org/wiki/MFCC>

[9] DCASE. *Detection and Classification of Acoustic Scenes and Events*. [En línia] [Consulta: 3 març 2021]. Disponible a: <http://dcase.community/>

[10] DCASE. *DCASE 2017 Challenge.* [En línia]. [Consulta: 3 març 2021]. Disponible a: <http://dcase.community/challenge2017/task-sound-event-detection-in-real-life-audio>.

[11] Diment, A.; Elizalde, B.; Heittola, T.; Mesaros, A.; Raj, B; Vincent, E.; Virtanen, T. *Sound event detection in the DCASE 2017 challenge*. [En línia] IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2019. [Consulta: 3 març 2021]. Disponible a: <doi:10.1109/TASLP.2019.2907016>

[12] Diment, A.; Elizalde, B.; Heittola, T.; Mesaros, A.; Shah, A.; Raj, B.; Vincent, E.; Virtanen, T. *DCASE 2017 challenge setup: tasks, datasets and baseline System*. [En línia]. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017), 85–92. Novembre 2017. [Consulta: 3 març 2021]. Disponible a: <http://dcase.community/documents/workshop2017/proceedings/DCASE2017Workshop_Mesaros_100.pdf>

[13] Learning rate. A: *Wikipedia.* [En línia]. Wikimedia Foundation, 2021. [Consulta: 10 maig 2021]. Disponible a: <https://en.wikipedia.org/wiki/Learning_rate>

[14] Dang, A.; Vu, T. H.; Wang, J. *Deep Learning For DCASE2017 Challenge*. [En línia]. Munic, 2017. [Consulta: 5 març 2021]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Dang_209.pdf>

[15] Liu, Y.; Wang, C.; You, J.; *Sound Event Detection From Real-Life Audio By Training A Long Short-Term Memory Networks With Mono And Stereo Features.* [En línia]. Munic, 2017. [Consulta: 5 març 2017]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Wang_168.pdf>

[16] Zhou, J. *Sound Event Detection In Multichannel Audio LSTM Network*. [En línia]. Munic, 2017. [Consulta: 5 març 2017]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Zhou_151.pdf>

[17] Adavanne, S.; Virtanen, T. *A report on sound event detection with diferent binaural features.* [En línia]. Munic, 2017. [Consulta: 5 març 2021]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Adavanne_130.pdf>

[18] Hou, Y.; Li, S. *Sound Event Detection In Real Life Audio Using Multi-model System.* [En línia]. Munic, 2017. [Consulta: 5 març 2021]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Hou_155.pdf>

[19] Chen, Y.; Duan, Z.; Zhang, Y. *DCASE2017 Sound Event Detection Using Convolutional Neural Networks*. [En línia]. Munic, 2017. [Consulta: 5 març 2021]. Disponible a: <http://dcase.community/documents/challenge2017/technical_reports/DCASE2017_Chen_124.pdf>

[20] Anaconda Inc. *Anaconda*. [En línia]. [Consulta: 2 març 2021]. Disponible a: <https://www.anaconda.com/>

[21] Python. *Python*. [En línia]. [Consulta: 2 març 2021]. Disponible a: <https://www.python.org/>

[22] Python Software Foundation. *Soundfile*. [En línia]. [Consulta: 5 abril 2021]. Disponible a: <https://pypi.org/project/SoundFile/>

[23] NumPy. *NumPy*. [En línia]. [Consulta: 15 març 2021]. Disponible a: <https://numpy.org/>

[24] Matplotlib. *Matplotlib*. [En línia]. [Consulta: 10 abril 2021]. Disponible a: <https://matplotlib.org/>

[25] Librosa. *Librosa*. [En línia]. [Consulta: 10 abril 2021]. Disponible a: <https://librosa.org/doc/latest/index.html>

[26] Tensorflow. *Tensorflow*. [En línia]. [Consulta: 7 abril 2021]. Disponible a: <https://www.tensorflow.org/>

[27] Heittola, T.; Mesaros, A.; Virtanen, T. *TUT database for acoustic scene classification and sound event detection.* [En línia]. In 24th European Signal Processing Conference 2016 (EUSIPCO 2016). Budapest, Hungary, 2016. [Consulta: 3 abril 2021]. Disponible a: <https://homepages.tuni.fi/annamaria.mesaros/pubs/mesaros_eusipco2016-dcase.pdf>

[28] Brownlee, J. Understand the Impact of Learning Rate on Neural Network Performance. A: *machinelearningmastery.* [En línia]. Gener 2019. [Consulta: 4 maig 2021]. Disponible a: <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/>

[29] Brownlee, J. Difference Between a Batch and an Epoch in a Neural Network. A: *machinelearningmastery*. [En línia]. Juliol, 2018. [Consulta: 4 maig 2021]. Disponible a: <https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>

[30] Calvo, D. Función de coste – Redes neuronales. A: *diegocalvo*. [En línia]. Desembre 2018. [Consulta: 5 maig 2021]. Disponible a: <https://www.diegocalvo.es/funcion-de-coste-redes-neuronales/>

[31] Mesaros, A.; Heittola, T.; Virtanen, T. *Metrics for polyphonic sound event detection.* [En línia]. Applied Sciences, 6(6):162, 2016. [Consulta 14 abril 2021]. Disponible a: <http://www.mdpi.com/2076-3417/6/6/162>

[32] Martínez Heras, Jose. Precision, Recall, F1, Accuracy en clasificación. A: *IArtificial.net* [En línia]. Jose Martínez Heras, 2020. [Consulta: 12 maig 2021]. Disponible a: <https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/>

[33] Karakaya, M. How to solve Multi-Label Classification Problems in Deep Learning with Tensorflow & Keras? A: *Medium.* Gener 2016. [En línia]. [Consulta: 10 abril 2021]. Disponible a: <https://medium.com/deep-learning-with-keras/how-to-solve-multi-label-classification-problems-in-deep-learning-with-tensorflow-keras-7fb933243595>

[34] Google Cloud Platform. *Google Inc.* [En línia]. [Consulta: 8 juny 2021]. Disponible a: <https://cloud.google.com/ai-platform/training/pricing#note-2>

ANNEXOS

Annex A: Planificació del projecte

	Tasques	Febrer		Març					Abril				Maig				Juny			
		Setmana 1	Setmana 2	Setmana 3	Setmana 4	Setmana 5	Setmana 6	Setmana 7	Setmana 8	Setmana 9	Setmana 10	Setmana 11	Setmana 12	Setmana 13	Setmana 14	Setmana 15	Setmana 16	Setmana 17	Setmana 18	Setmana 19
Fase 1	Descripció del projecte																			
	Definir els objectius																			
	Definir les fases del projecte																			
	Estructurar el calendari																			
Fase 2	Estudi dels reptes i sistemes presentats																			
	Lectura del llibre "Computational Analysis of Sound Scenes and Events" i articles																			
	Anàlisi de les tecnologies actuals en el processament d'àudio																			
	Estudi de les tècniques actuals basades en aprenentatge profund																			
Fase 3	Anàlisi del sistema de referència																			
	Anàlisi de la base de dades																			
Fase 4	Estructurar el directori del sistema acústic																			
	Preprocessament dels arxius d'àudio																			
	Preprocessament dels arxius de text																			
	Desenvolupar la xarxa nuronal																			
	Entrenament de la xarxa neuronal																			
	Prova de funcionament amb les dades de test																			
	Avaluació del sistema																			
	Primer prototip del sistema acústic																			
	Revisió del sistema acústic																			
	Millora del sistema acústic																			
	Segon prototip del sistema acústic																			
	Correcció de possibles errors del sistema acústic																			
	Millora del sistema acústic																			
	Desenvolupament del sistema acústic final																			
	Avaluació del sistema acústic																			
	Desenvolupament del programa d'execució (aplicació de consola)																			
	Execució completa del sistema																			
	Avaluació del sistema complet																			
Fase 5	Redacció de la introducció a la memòria																			
	Redacció dels conceptes teòrics																			
	Redacció de l'anàlisis de la base de dades																			
	Redacció del l'anàlisi del sistema de referència																			
	Redacció del procediment pràctic per al desenvolupament del programa																			
	Redactar les conclusions del projecte																			
	Redactar les noves línies de treball																			
	Redactar la bibliografia																			
	Revisió de la memòria del projecte																			

Annex B: Codi font dels processos més rellevants

Params.py

```
parameters = {

    'features_folder' : 'dataset/TUT-sound-events-2017-development/pre-
processed/',
    'audio_folder' : 'dataset/TUT-sound-events-2017-
development/audio/street/',
    'setup_folder' : 'dataset/TUT-sound-events-2017-
development/evaluation_setup/',

    'folds' : [1, 2, 3, 4],

    'sample_rate': 44100,
    'nfft' : 2048,
    'window_length' : 0.04,
    'hop_length' : 0.02,
    'mels' : 40,
    'mfcc' : 20,
    'frame_padding' : 1,

    'epochs' : 50,
    'learning_rate' : 0.001,
    'batch_size' : 256,

    'extract_features' : True,
    'train_model' : True,
    'tagging' : True,

    'audio_file_for_tagging' : 'dataset/TUT-sound-events-2017-
development/audio/street/b093.wav',
    'tags_file_name' : 'b093.txt'
}
```

Model.py – Funció per generar el model

```
def build_model(input=None, output=None, lr=0.001, show=False):

    """
        Descripció: Funció per configurar el model.

        Arguments ->
            input: Tamany de la entrada al sistema
            output: Tamany de la sortida del sistema
            lr: Pas d'aprenentatge en cada iteració
            show: Mostrar la configuració del model

        Retorn ->
            model: model generat

    """

    model = tf.keras.Sequential()

    model.add(tf.keras.layers.Input(input))
    model.add(tf.keras.layers.Dense(50, activation='relu'))
    model.add(tf.keras.layers.Dropout(0.2))
    model.add(tf.keras.layers.Dense(50, activation='relu'))
    model.add(tf.keras.layers.Dropout(0.2))
    model.add(tf.keras.layers.Dense(50, activation='relu'))
    model.add(tf.keras.layers.Dropout(0.2))
    model.add(tf.keras.layers.Dense(output, activation='sigmoid'))

    optimizer = tf.keras.optimizers.Adam(learning_rate=lr)

    model.compile(optimizer=optimizer, loss='binary_crossentropy',
metrics=['binary_accuracy'])

    if(show): model.summary()

    return model
```

Dataset.py – Funció per extreure les característiques dels senyals d'àudio

```
def extract_features(audio_folder=None, features_folder=None, fm=44100,
nfft=2048, window_length=2048, hop_size=1024, n_mels=40, n_mfcc=12,
frame_padding=None):

    """
        Descripció: Funció per calcular els coeficients de mel dels
senyals d'àudio de la base de dades

        Arguments ->
            audio_folder: Carpeta que conté els arxius d'àudio
            features_folder: Carpeta on es guardaran les dades de
característiques
            fm: Freqüència de mostratge dels senyals d'àudio
            nftt: Número de punts de la transformada
            window_length: Mida de la finestra
            hop_size: Mida del salt entre trames
            n_mels: Número de filtres de Mel
            n_mfcc: Número de coeficients de Mel a calcular
            frame_padding: Número de trames a afegir
    """

    data_dict = load_setup(folder='train', fold_number=1)
    data_dict.update(load_setup(folder='test', fold_number=1))

    create_folder(path=features_folder)

    audio_files = os.listdir(audio_folder)

    for audiofile in tqdm(audio_files):

        data, sr = sf.read(os.path.join(audio_folder, audiofile))

        data = stereo_to_mono(data=data)

        data = normalitzar(data=data)

        mfcc = extract_mfcc(y=data, fm=fm, n_fft=nfft,
win_length=window_length, hop_length=hop_size, n_mels=n_mels,
n_mfcc=n_mfcc)

        mfcc = split_into_sequences(data=mfcc,
frame_padding=frame_padding)

        labels = np.zeros((mfcc.shape[0], len(class_labels)))

        tmp_data = np.array(data_dict[audiofile])

        frame_start = np.floor(tmp_data[:,0] * fm / hop_size).astype(int)

        frame_end = np.ceil(tmp_data[:,1] * fm / hop_size).astype(int)

        se_class = tmp_data[:,2].astype(int)
```

```
        for i, valor in enumerate(se_class):

            labels[frame_start[i]:frame_end[i], valor] = 1

        file_name = os.path.join(audiofile.split('/')[-1].replace(".wav",
"")) + ".npz"

        path = os.path.join(features_folder, file_name)

        np.savez(path, mfcc, labels)
```

Utils.py – Funció per calcular els MFCC i les seves derivades

```
def extract_mfcc(y=None, fm=44100, n_fft=2048, win_length=2048,
hop_length=512, n_mels=40, n_mfcc=20):

    """
        Descripció: Funció per extreure calcular els coeficients
cepstrals de mel

        Arguments ->
            y: senyal
            fm: freqüència de mostratge
            n_fft: punts de la trasformada
            win_length: mida de la finestra
            hop_length: mida del salt entre trames
            n_mels: número de bandes de mel
            n_mfcc: número de coeficients a retornar
    """

    spectrum = np.abs(librosa.stft(y=y, n_fft=n_fft,
hop_length=hop_length, win_length=win_length))**2

    mel_basis = librosa.filters.mel(sr=fm, n_fft=n_fft, n_mels=n_mels,
fmin=0.0, fmax=22050)

    mel_spectrum = np.dot(mel_basis, spectrum)

    mfcc = librosa.feature.mfcc(S=librosa.amplitude_to_db(mel_spectrum),
n_mfcc=n_mfcc)

    features_vector = mfcc

    deltas = librosa.feature.delta(data=mfcc, order=1)

    features_vector = np.vstack((features_vector, deltas))

    deltas2 = librosa.feature.delta(data=mfcc, order=2)

    features_vector = np.vstack((features_vector, deltas))

    return features_vector.T
```

Model.py – Funció per avaluar el sistema

```
def evaluate_model(model=None, X_test=None, Y_test=None):

    """
        Descripció: Funció per avaluar el sistema

        Arguments ->
            model: model a avaluar
            X_test: dades d'entrada per generar les prediccions
            Y_test: prediccions reals

        Retorna ->
            Retorna les puntuacions obtingudes pel model avaluat

    """

    Y_pred = model.predict(X_test)
    Y_pred = np.round(Y_pred)

    m = keras.metrics.TruePositives()
    m.update_state(Y_test, Y_pred)
    TP = m.result().numpy()

    m = keras.metrics.TrueNegatives()
    m.update_state(Y_test, Y_pred)
    TN = m.result().numpy()

    m = keras.metrics.FalsePositives()
    m.update_state(Y_test, Y_pred)
    FP = m.result().numpy()

    m = keras.metrics.FalseNegatives()
    m.update_state(Y_test, Y_pred)
    FN = m.result().numpy()

    S = np.min([FN, FP])
    D = np.max([0, FN-FP])
    I = np.max([0, FP-FN])
    N = Y_test.sum()

    ER = (S + D + I) / N

    accuracy = (TP + TN) / (TP + FP + TN + FN)

    recall = TP / (TP + FN)

    precision = TP / (TP + FP)

    F1 = 2 * ((precision * recall) / (precision + recall))

    return accuracy, recall, precision, F1, ER, TP, TN, FP, FN
```