ESEIAAT

Projecte Final de Grau

Classificació d´àudio amb Deep Learning

Grau en enginyeria de sistemes audiovisuals

Alumne: Sergi Badia Oliver

Tutor: Ignasi Esquerra Llucià

Maig 2019

Agraïments

Primer de tot m´agradaria reservar aquesta pàgina per donar les gràcies a totes les persones (companys, amics i professors) que han fet possible el fet d´haver arribat a aquest punt. Entregar el projecte de final de carrera significa un punt i a part en la meva vida d´estudiant i farà deixar enrere una etapa que sempre recordaré amb molta estima. Aquests últims anys seran els que em guiaran en la meva vida laboral i en el meu futur i personalment crec que m´han permès formar-me d´una forma immillorable.

Al meu tutor Ignasi Esquerra també m´agradaria donar-li les gràcies especialment per la ajuda que m´ha donat per intentar entendre un camp que no és precisament la meva especialitat però que gràcies a ell i al desenvolupament d´aquest projecte, ara puc entendre una mica millor que al començament.

A la meva família donar-li les gràcies també per la seva paciència ja que porten molt de temps esperant per poder celebrar el final d´aquesta etapa tots junts.

Gràcies a totes les persones involucrades directament o indirectament en el desenvolupament d´aquest projecte.

Objectiu del projecte

L´objectiu principal d´aquest projecte ha sigut aprendre sobre el món del Deep Learning i les xarxes neuronals intentant entendre els conceptes principals més importants en el cas de l´àudio i posar-ho en pràctica realitzant un exemple bàsic d´una xarxa neuronal.

Per a realitzar l´exemple de la xarxa neuronal s´aprofitarà el repte DCASE que es celebra anualment i anima als participants a realitzar una sèrie de tasques de classificació d´àudio amb bases de dades públiques i amb sistemes basats en el Deep Learning. Es simularà una participació al concurs (en concret l´edició del 2017) fent servir les mateixes dades i normatives que tots els participants per veure quins resultats es poden aconseguir.

Aprofitant que els resultats i els mètodes seguits per aconseguir-los són públics també comprovarem quins son els sistemes que es fan servir en la actualitat per a classificar àudio.

Índex

Capítol 1: Introducció a la classificació d´àudio

1. Introducció a la classificació d´àudio

El so és qualsevol fenomen físic que involucri la propagació d´ones. Les vibracions que produeixen els objectes al ser colpejats es transmeten a través d´un medi(aire, aigua, etc...) fins que arriba a la nostra oïda i aquesta transforma les vibracions en impulsos elèctrics que envia al cervell produint una sensació sonora [3].

El so es propaga a través de medis elàstics[1] a diferent velocitat, ja que cada medi té una velocitat de propagació diferent.

Algunes característiques dels sons, que definirem a continuació, son perceptibles amb un cop d'oïda. Aquestes característiques són:

- **To:** Depèn de la freqüència i fa que un so s´escolti agut o greu.
- **Duració:** El temps que dura un so. La duració pot variar en funció de si és subjectiu o objectiu. Subjectiu es l´interval del temps en que nosaltres sentim el so i objectiu es la duració del so mesurada físicament.
- **Intensitat:** Depèn de la amplitud i fa que un so s´escolti més fort o més fluix, és a dir, amb més energia o menys energia.
- **Timbre:** Depèn de la forma de la ona sonora. Permet diferenciar un so de la mateixa freqüència i intensitat produïda per fonts sonores diferents.

A part de les característiques comentades anteriorment, els sons porten molta més informació del que sembla en un principi. El to, la intensitat i el timbre són característiques del so que es poden percebre amb un cop d'oïda però hi han altres que són més difícils de percebre. Molta d´aquesta informació està amagada en el domini de la freqüència, el qual ens obliga a realitzar una transformació del so.

Desenvolupar mètodes que extreguin aquesta informació amagada dintre de les vibracions acústiques té un gran potencial a l´hora de crear aplicacions que per exemple, classifiquin música entre gèneres, busquin contingut multimèdia segons la informació del àudio o en general, reconèixer activitats humanes segons la informació que ens dona l´àudio.

Com hem comentat abans, la classificació del àudio mitjançant la informació que conté pot tenir moltes utilitats avui en dia. Però com es realitza aquest procés de extracció de dades i classificació? Ara farem una introducció als conceptes bàsics sobre aquest procés.

[1] Aquell medi en el que es facilita el pas d´una ona comprimint-la i dilatant-la fins recuperar el seu estat original. Aquest es deforma i es recupera vibrant al pas de l´ona. Els medis poden ser líquid, sòlid i gas.

1.1 Mètodes d´extracció de la informació del àudio

Avui en dia existeixen moltes formes de poder extreure la informació d´un àudio. No hi ha cap mètode millor o pitjor que un altre, sinó que es tracta d´un procés purament analític. Depenent dels tipus d´àudios i del mètode de classificació que es vol fer, un tipus de característiques (també anomenats comunament com a features) poden funcionar millor que no pas unes altres. Al final, es tracta de provar unes característiques, obtenir uns resultats, provar amb unes altres diferents i finalment es comprova si els resultats del sistema han millorat.

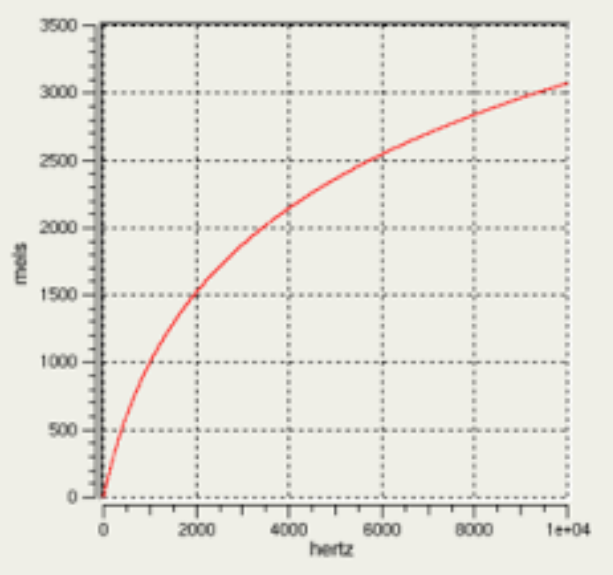
Aquests són alguns dels "features" més usats avui en dia per parametritzar un àudio:

1.1.1 MFCC (Mel Frequency Cepstral Coefficients)

Els MFCC son coeficients per a la representació de la parla i de l´àudio que estan basats en la percepció humana dels sons. Els MFCC representen les característiques locals d´un senyal de veu i donen una idea de quin tipus de senyal és.

Els coeficients cepstrals es calculen mitjançant la Transformada de Fourier (FT) ,de la Transformada de Fourier Discreta (DFT) o de la Transformada Cosinus Discreta (DCT) que és una variant de la DFT. En les MFCC les bandes de freqüència funcionen logarítmicament segons l´escala mel (d´aquí el nom), en la que el punt de referència s'obté equiparant un to de 1000 Hz, 40 dBs per sobre del llindar d´audició de l´oient, amb un to de 1000 mels.

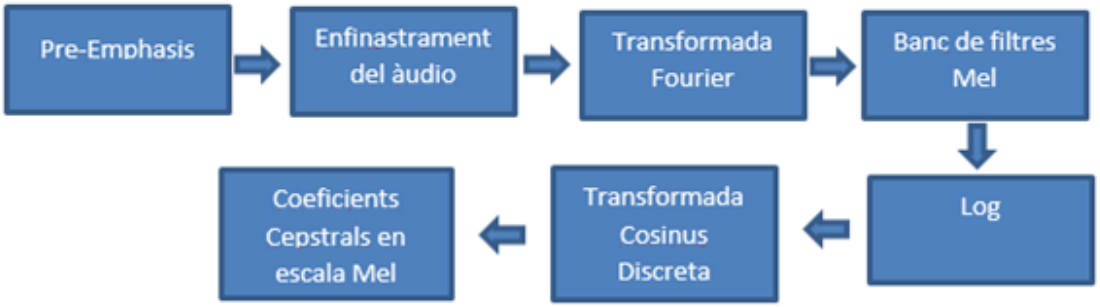
En la següent figura podem veure una taula d'equivalència entre mels-Hz:



Il·lustració 1: Equivalència Mel-Hz

Per sobre dels 500 Hz, els intervals de freqüència separats exponencialment es queden separats de forma lineal en l´escala Mel. Això vol dir que, per exemple, 4 octaves en l´escala freqüencial corresponen només a 2 octaves en l´escala Mel (sempre per sobre de 500 Hz) . Aquest comportament lineal permet representar i modelar millor la nostra resposta auditiva que no pas l´escala freqüencial i aquesta es la raó per la que els MFCC són una de les característiques més usades a l´hora de realitzar classificació d´ àudio.

Aquest es el procés complet per calcular els MFCC:



Il·lustració 2: Procés per al càlcul dels MFCC

1.1.2 Flux espectral

El flux espectral [14] mesura els canvis en l´espectre[2] d´un àudio entre dues trames consecutives. Es calcula com a la diferència al quadrat de la magnitud normalitzada de l´-espectre entre 2 finestres consecutives de l´àudio. Es pot definir amb la següent equació:

$$F_n = \sum_{k=1}^{K} (P_n(k) - P_{n-1}(k))^2$$

on **n** es la trama d´àudio i **Pn(k)** es l´espectre normalitzat d´una trama.

El flux espectral es fa servir per tenir una idea dels canvis en l´energia i la freqüència d´un àudio i es bastant comú fer-ho servir per a classificar àudio.

1.1.3 Descriptors MPEG-7

L´ estàndard MPEG-7 consisteix en crear una representació d´un àudio o una imatge que permeti la descripció dels seus continguts. En el cas de l´àudio ens ofereixen 2 valors diferents que ens permetrà conèixer les característiques i continguts d´un senyal d´àudio. Aquests dos valors són:

[2] L´espectre d´un àudio es una descomposició de les freqüències que conté. Permet veure quines freqüències són més comunes o calcular l´energia espectral.

- **AudioWaveformType (AWF):** Aquest descriptor permet realitzar una descripció de la forma del senyal d´àudio. S'obté un valor mínim i màxim de les mostres d´àudio dintre d´una finestra i es van guardant de manera que es pugui crear una sèrie temporal amb els valors mínim i màxim.
- **AudioPowerType (AP):** Descriu la potència de les mostres del senyal d´àudio. Bàsicament permet saber com evoluciona l´amplitud del senyal en la línia temporal.

Aquests valors es poden fer servir per parametritzar un senyal d´àudio i fer-los servir per entrenar un sistema i realitzar després una classificació entre diferents classes.

1.2 Introducció al Machine Learning

Que una màquina es pugui comportar i tingui la mateixa capacitat de decisió que els humans sempre ha sigut un objectiu en el camp de la intel·ligència artificial. En l´actualitat, encara sembla molt difícil programar alguna aplicació que s´assembli a la ment humana però realment en els últims anys s´han produït avenços bastant importants en aquest camp.

El Machine Learning o aprenentatge automàtic és una disciplina científica en l´àmbit de la intel·ligència artificial que permet crear aplicacions que tenen aprenentatge autònom. Aquestes aplicacions llegeixen dades, aprenen sobre les característiques d´aquestes dades i finalment tenen la capacitat de realitzar una predicció sobre la nova informació que s´introdueix. Aquestes aplicacions fan servir diferents algoritmes per ser el més precís possible a l´ hora de realitzar una feina concreta. La màquina és entrenada utilitzant una gran quantitat de dades d´entrada fent, durant el procés d´entrenament, que aquests algoritmes es vagin perfeccionant.

Es distingeixen 2 tipus de algoritmes segons si les dades venen etiquetades en classes (aprenentatge supervisat) o, en canvi, no venen etiquetades (aprenentatge no supervisat). Per tant, per a resoldre problemes de classificació, com el que farem en aquest projecte, es realitzen mitjançant algoritmes d´aprenentatge supervisat. A part hi ha un altre divisió possible dels algoritmes depenent del tipus de resultats que volem obtenir [11]:

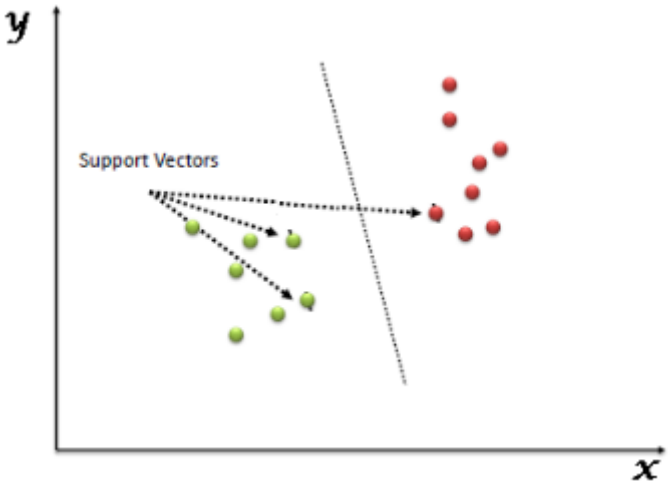
- **Algoritmes de classificació:** Es fan servir quan el resultat final és una classe discreta. Alguns exemples d´aquests algoritmes són: k-nearest neighbors, arbres de decisió i suport vector machines.

- **Algoritmes de regressió:** És útil per predir productes que són continus, és a dir, que no tenen una sortida discreta com és el cas dels algoritmes de classificació. La sortida del algoritme es representa mitjançant un conjunt que pot determinar-se de manera flexible en funció de les entrades del model enlloc de limitar-se a un conjunt de classes. Alguns exemples d´aquests algoritmes són: regressió lineal i Ordinary Least Squares Regression.

Algoritmes de classificació

En aquest apartat explicarem alguns dels algoritmes de classificació més usats en el aprenentatge automàtic:

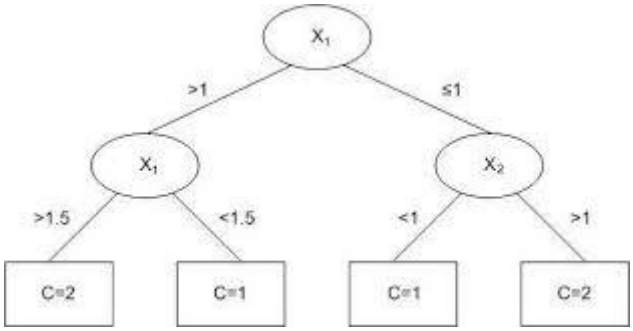
- **Support Vector Machines:** En aquest algoritme es dibuixa cada característica de l´-àudio en un pla n-dimensional on n correspon al nombre de característiques que extraiem de cada dada d´entrada. El valor de cada característica correspondrà a una coordenada dintre del pla n-dimensional. Finalment a l´hora de classificar, es realitza una línia que permet diferenciar de forma clara les diferents classes:



Il·lustració 3: Classificació mitjançant support vector machines de 2 classes

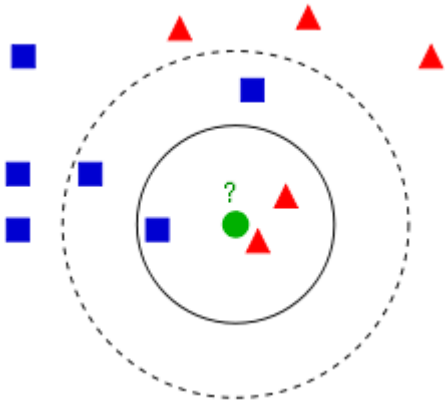
https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

- **Aprenentatge basat en arbres de decisió:** És un mètode molt comú que s´usa en la mineria de dades. L´objectiu és crear un model que permeti predir el valor d´una variable final a partir d´unes variables d´entrada. Un arbre consisteix en un conjunt seqüencial de condicions i accions que relacionen uns determinats factors amb un resultat o classe. Les variables d´entrada en el cas de l´àudio podrien ser els MFCC, log-mel energies, etc...



Il·lustració 4: Exemple de Arbre de decisió amb 2 possibles classes de sortida

- **K-nearest Neighbors:** Aquest algoritme classifica cada nova dada d´entrada en una de les diferents classes segons tingui "k" veïns més a prop d´una classe o d´una altra. Per exemple, amb un àudio, aquest es col·loca en un pla n-dimensional (on n són el nombre de característiques que s´extreuen de cada àudio) i es calcula la distància d´aquest àudio a cada un dels existents, s´ordenen aquestes distàncies de menor a major i finalment es classifica l´àudio en el grup en que major freqüència apareixen amb menors distàncies.



Il·lustració 5: Classificació mitjançant k-nearest neighbor

Com podem veure en l´anterior imatge, la nova dada (cercle) s´ha de classificar entre una de les dues classes (quadrat o triangle). Els veïns més propers que té el cercle de classe indefinida són 2 triangles. Per tant, per k=2 la nova dada es classificarà com a classe triangle. En canvi si k=3, la dada es classificarà com a quadrat ja que té més a prop un conjunt de 3 quadrats que no pas de 3 triangles.

Algoritmes de regressió

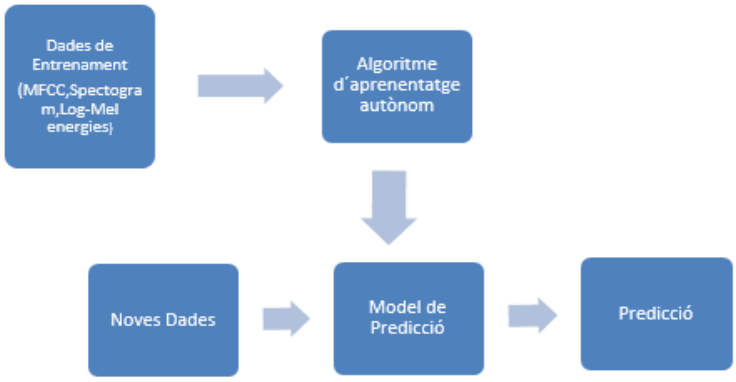
Tot i que aquests algoritmes no permeten classificar, que és l´objectiu del projecte, farem un breu resum d´alguns algoritmes de regressió:

- **Regressió Lineal:** És un algoritme d´aprenentatge supervisat que es fa servir no només en el aprenentatge automàtic sinó també en el camp de l´estadística. Podríem dir que es tracta d´un algoritme que dibuixa una recta que indica la tendència d´un conjunt de dades continues.
- **Ordinary Least Squares Regression**: És un mètode per a realitzar regressió lineal mitjançant l´estadística. La regressió lineal serveix per a estudiar la relació entre diferents variables.

Aquests algoritmes es continuen perfeccionant avui en dia per aconseguir un entrenament més eficient i per tant aconseguir un model més pròxim a la realitat, que permetrà fer prediccions més precises. A part dels comentats anteriorment, existeixen multitud d´-algoritmes per realitzar aprenentatge automàtic.

Seguint amb el Machine Learning, podem identificar dues fases en el procés de l´-aprenentatge autònom:

- **Entrenament del sistema:** Consisteix en la lectura de dades d´entrenament, extracció de característiques i entrenament del sistema mitjançant les dades parametritzades. El algoritme d´entrenament podrà detectar certs patrons en la informació impossibles de detectar pels humans i permetrà crear un model de predicció, que es podrà fer servir per a realitzar prediccions amb noves dades.
- **Fase de predicció:** Havent creat un model de predicció mitjançant l´entrenament del sistema podrem realitzar prediccions i classificacions utilitzant noves dades.



Il·lustració 6: Esquema Machine Learning

1.3 Introducció al Deep Learning

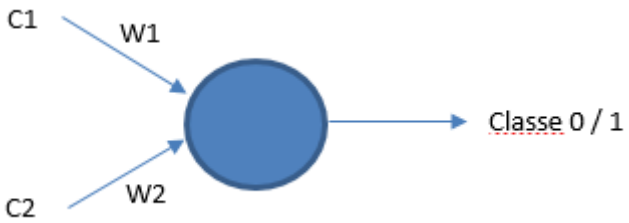
En els últims anys s´ha posat de moda una tècnica d´aprenentatge automàtic coneguda com a Deep Learning o aprenentatge profund [4] [5] [12]. Realment, es pot considerar que l´aprenentatge profund es la suma de realitzar aprenentatge automàtic fent servir moltes més dades. En l´aprenentatge profund entrenem amb unes dades d ´entrada per crear un model de predicció i després es creen un conjunt de instruccions per anar modificant aquest model per tal d´anar minimitzant els errors de predicció.

Un dels mètodes més famosos per a realitzar aprenentatge profund són les xarxes neuronals. Consisteix en simular una xarxa artificial de neurones dins del software que fa l´anàlisi de les dades. Aquestes neurones estan connectades entre si i treballen en conjunt, sense que aquestes neurones tinguin una feina en concret. Amb l´experiència, les neurones van creant i reforçant certes connexions per aprendre alguna cosa que es queda fixat en la xarxa. Es pot dir que la intenció es simular el funcionament real del sistema de neurones que tenim en el nostre cervell humà.

Les xarxes neuronals es basen en una idea senzilla en inici: donats uns paràmetres a l´-entrada hi ha una forma de combinar-los per preveure el resultat final. El gran problema és que no es sap com combinar-los de forma correcta i la xarxa és el model amb el que s´intenta trobar aquesta combinació de paràmetres. La forma de trobar la combinació adequada és entrenant la xarxa neuronal. Un cop entrenada es podrà fer servir per a realitzar prediccions o classificacions, o el que és el mateix, aplicant la combinació de paràmetres correctes.

Per entendre millor com funciona una xarxa neuronal farem un exemple senzill d´un perceptró, que és la unitat fonamental d´una xarxa neuronal, i la farem servir amb un problema de classificació d´àudio, que es del que es tracta aquest projecte.

Suposem que tenim un àudio que s´ha de classificar entre dues classes diferents. La entrada de la xarxa seran, per exemple, 2 coeficients MFCC del àudio (C1 i C2) amb un pes indeterminat (W1 i W2) i la sortida serà la classe 0 o la classe 1:



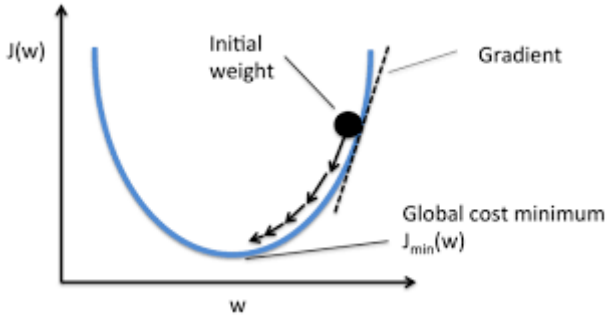
Il·lustració 7: Exemple de perceptró amb 2 entrades i 1 sortida

El que determinarà quina de les dues classes sortirà al final serà el pes de cada coeficient (W1 i W2), és a dir, quin dels 2 coeficients té més importància a l´hora de realitzar la

classificació. Els valors dels coeficients es multipliquen pel seu pes i es sumen entre ells. Aquest resultat serà el que arriba a l´entrada de la neurona i cada neurona (o conjunt de neurones com veurem més endavant) se li assigna una funció[3] que transforma el valor d´entrada a un altre valor diferent. Si aquest valor a la sortida de la neurona es superior a un numero serà d´una classe i sinó serà de l´altra classe.

Al començament de l´entrenament, com no sabem quin és el valor correcte dels pesos s´acostuma a posar 0,5 a cada un per no donar més importància a un coeficient que a l´altre. Per tant, el objectiu al realitzar l´entrenament de la xarxa serà anar variant els pesos de les connexions de les neurones per veure com afecta el resultat final i si la classe que dona a la sortida és la correcta [4] .

Per saber si la sortida és correcta o no es fan servir el que s´anomena funció de cost. La funció de cost indica que tant bo o dolent han sigut els resultats de la xarxa en comparació amb els valors reals. La situació ideal seria que el cost fos 0 però això voldria dir que la xarxa encertarà sempre totes les prediccions, cosa que es gairebé impossible. Més aviat es tracta de minimitzar el valor d´aquesta funció. Per minimitzar la funció de cost hi ha diferents mètodes que es poden fer servir però el més habitual és fer-ho mitjançant gradients.



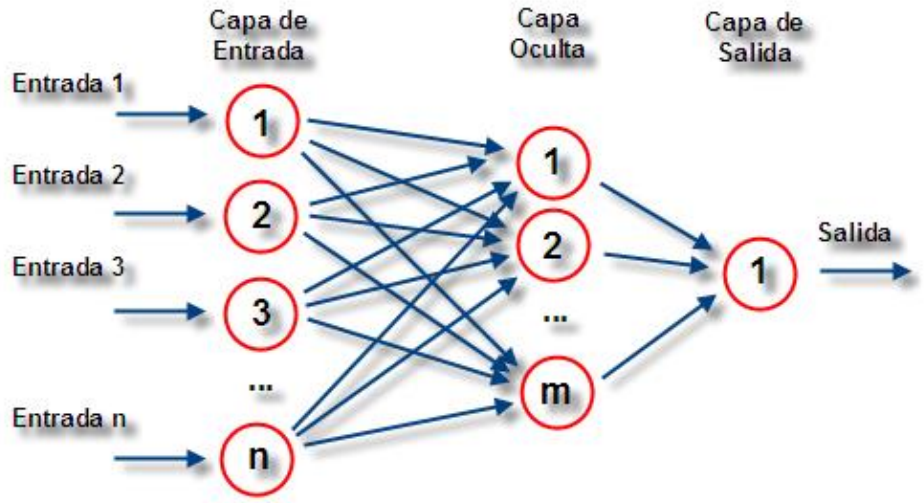
Il·lustració 8:Exemple de gradient descendent

Donada una funció de cost determinada, el gradient es mou en un punt concret respecte a la derivada en aquell punt. D´aquesta manera el gradient anirà avançant fins a trobar el mínim de la funció de cost. En cada pas del gradient o en cada iteració s´anirà variant els pesos en petites variacions definits pel learning-rate[5] fins que s´ aconsegueixi ajustar-los amb uns pesos que permetin minimitzar el valor de la funció de cost. El fet de que s´hagi d´iterar i anar variant els pesos moltes vegades per trobar el mínim d´aquesta funció es una de les raons principals per les quals es necessita un gran poder de computació per entrenar una xarxa.

3 Anomenades funcions d´activació com veurem més endavant.
4 Al realitzar l´entrenament coneixem a quina classe pertany cada àudio.
5 Paràmetre d´una xarxa neuronal que indica si les variacions dels pesos d´una xarxa durant cada iteració són grans o petits. Com més gran és aquest valor més grans són les variacions i més ràpidament s´intenta minimitzar la funció de cost de la xarxa però també es més difícil trobar el mínim exacte. Com més baix el valor, més petites són les variacions i mes lentament s´arribarà al mínim però aquest serà més exacte.

Un cop els pesos tinguin un valor adequat, es pot considerar que tenim un model entrenat i podrem començar a fer prediccions amb àudios que no vinguin etiquetats i classificar-los de forma més eficient.

El perceptró, que hem explicat fa una estona, és el exemple més senzill de xarxa que és pot fer. Però com hem comentat abans, una xarxa neuronal és una interconnexió de neurones i en el cas que hem explicat fa un moment, només tenim una neurona. El concepte que falta per explicar és el de capes. Aquí podem veure un exemple de una xarxa formada per diferents capes:



Il·lustració 9: Estructura bàsica d´una xarxa neuronal multicapa
http://www.esacademic.com/pictures/eswiki/82/RedNeuronalArtificial.png

Les capes son un conjunt de neurones que estan dintre d´un mateix nivell de l´estructura de la xarxa i serveixen per afegir informació extra. Agafen les dades d´entrada, s´exploren i es treuen les característiques que millor ajuden a entendre que està passant. Durant el procés d´aprenentatge cada capa aprèn a trobar i detectar les característiques que millor ajuden a classificar les dades. Agafant el exemple de l´àudio que hem fet servir abans i fent-lo servir amb aquesta nova xarxa, les capes ocultes [6] podrien aprendre a detectar certs valors i patrons en els coeficients MFCC que serveixin per a poder decidir a quina classe pertany l´àudio.

A cada capa oculta de la xarxa neuronal se li assigna una funció no lineal per a determinar quina serà la sortida de les neurones de cada una de les capes de la xarxa. Aquestes funcions s´anomenen Funcions d´activació.

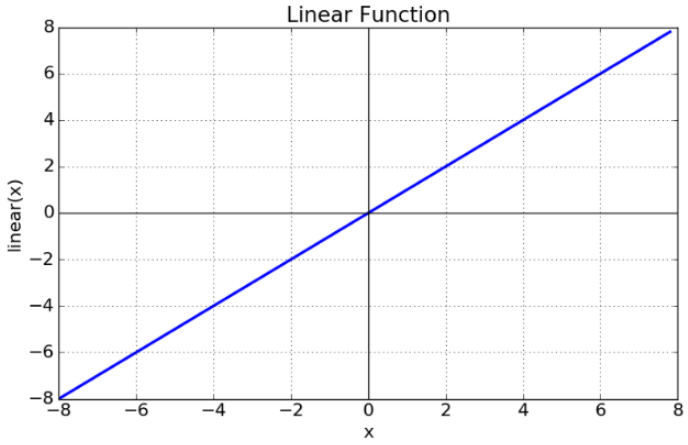
[6] Les capes entre la capa d´entrada i la de sortida s´acostumen a nomenar capes ocultes

1.3.1 Funcions d´activació en el Deep Learning

Les funcions d´activació [13] o també conegudes com a funcions de transferència es fan servir per obtenir el valor a la sortida de la neurona com un "si" o un "no". Bàsicament fan un mapeig del valor d´entrada de la neurona a un valor entre 0 i 1 [7].

Les funcions d´activació es poden dividir en 2 grups:

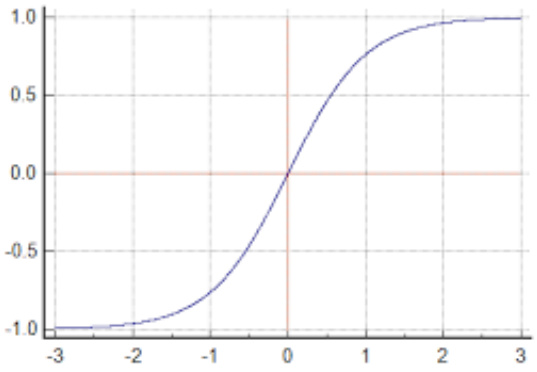
Funcions Lineals:



Il·lustració 10: Exemple Funció Lineal
www.towardsdatascience.com

Com es pot veure correspon a una linea recta. El problema d´aquestes funcions es que no es podria delimitar la sortida de la neurona a un valor entre 0 i 1. No ajuden molt ja que les dades amb les que s´alimenta una xarxa acostumen a ser molt complexes.

Funcions no lineals:



Il·lustració 11: Exemple Funció no lineal
www.towardsdatascience.com

[7] O entre -1 i 1 depenent de la funció

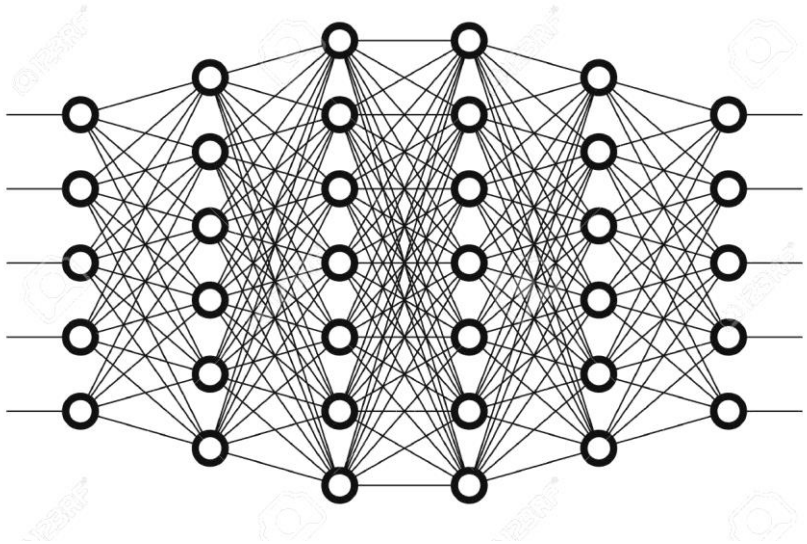
Són les funcions més usades en el aprenentatge profund. Permeten adaptar-se millor a les dades d´entrada i diferenciar clarament les diferents sortides. Alguns exemples d´aquestes funcions són:

- Funció Sigmoid
- Tangent Hiperbòlica
- Funció ReLU

1.4 Estat de l´art en la classificació d´àudio

Les xarxes neuronals explicades anteriorment com el perceptró o la xarxa neuronal multicapa fa un temps que estan entre nosaltres, però que és realment l´últim en classificació de àudio? Ara intentarem anomenar i descriure molt per sobre quines són les tecnologies mes usades en la actualitat:

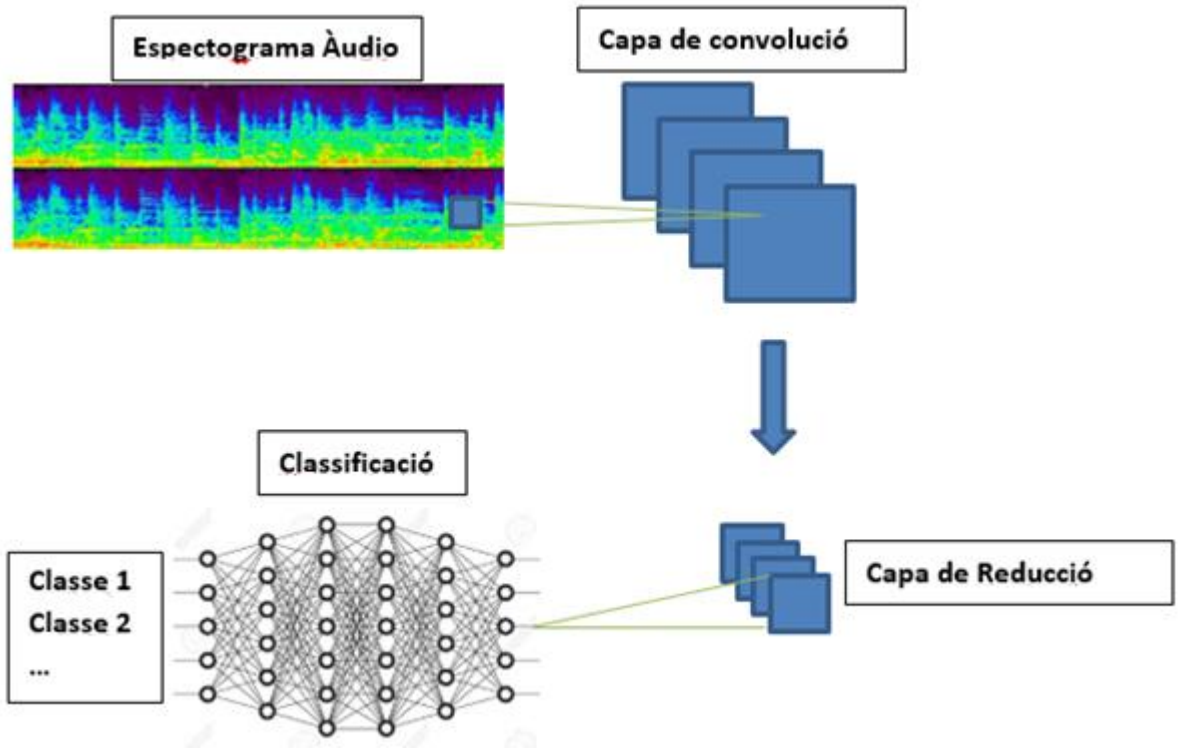
Xarxes neuronals profundes: Agafant com a model la xarxa neuronal bàsica, es poden anar afegint més capes intermitges amb varies neurones per capa fins a arribar al concepte de xarxa neuronal profunda. La idea és que amb més capes i més neurones es poden millorar les prediccions en conjunts de dades que siguin més complexes, per exemple, una base de dades que tingui més de 2 classes diferents.



Il·lustració 12:Xarxa neuronal profunda amb una capa d´entrada, 4 capes intermitges i 1 de sortida. https://es.123rf.com/photo_62610991_red-neuronal-la-red-neuronal-aprendizaje-profundo-concepto-de-tecnolog%C3%ADa-cognitiva-ilustraci%C3%B3n-vectorial.html

Xarxes convolucionals: Les xarxes convolucionals es van començar a fer servir per a classificació d´imatge, ja que la idea de la xarxa convolucional és buscar característiques locals en petites regions d´una imatge que permetin identificar millor el contingut que té. Però llavors en el cas de l'àudio com funciona? La idea es convertir l´àudio d´entrada a una

imatge i això és possible gràcies als espectrogrames. Un espectrograma és una representació freqüencial de l´energia d´un senyal d´àudio al llarg del temps. Per tant, l´-espectrograma ens permetrà buscar característiques no en tot l´àudio sinó en petites regions d´aquest espectrograma (canvis bruscos en la freqüència, canvis de tons, continguts fonètics, etc...). Per posar a prova aquesta idea s´acostuma a crear un mateix grup de neurones per a cada entrada de la capa convolucional (Per exemple uns quants píxels de l´-espectrograma d´un arxiu d´àudio). La idea és que tots els elements que es fiquen a la capa (que es diu capa de convolució) tinguin els mateixos pesos a l´entrada per a reduir el nombre de paràmetres que es fan servir. A mida que anem afegint més capes, la xarxa podrà anar descobrint més característiques de l´espectrograma d´àudio, per exemple, una capa que s´activa amb àudios de freqüència alta i un altre que s´activi amb alguns continguts fonètics. Finalment, després de la capa de convolució s´acostuma a posar una xarxa neuronal més habitual, que tindrà com a entrades les característiques locals que s´han trobat de l´àudio i classificarà entre les diferents classes finals.



Il·lustració 13: Exemple de Xarxa convolucional

Xarxes Convolucionals Recurrents: Una xarxa neuronal recurrent no té una estructura de capes definides com en els exemples anteriors sinó que permeten connexions aleatòries entre les diferents neurones que formen la xarxa. Són una de les xarxes més potents ja que es poden crear cicles, canviar la direcció del flux de dades i retroalimentar-les, i amb això es pot aconseguir que tingui memòria. Hi ha diferents tipus de xarxes recurrents depenent de com es propaguen les dades i del nombre de capes ocultes que tinguin. Alguns exemples són:

- **Xarxes Recurrents simples – SRN**: Arquitectura base sobre la que s´implementen les altres. Incorporen la retroalimentació per crear una línia temporal i que tinguin memòria. Es fan servir molt per a reconeixement de veu.

- **Xarxes LSTM (Long Short Term Memory):** Un dels problemes de les xarxes recurrents simples és que presenten problemes en l´entrenament degut a que al tenir una línia temporal, l´acumulació d´errors provoca dificultats per memoritzar dependències a llarg termini. Aquests problemes es resolen amb les xarxes LSTM que incorporen una sèrie de passos per decidir quina informació es guarda i quina informació és eliminada.

- **Xarxes GRU (Gated Recurrent Unit):** Son un tipus especial de xarxa neuronal recurrent que contenen una unitat de memòria (GRU) que controla la forma en que la informació flueix dintre de la xarxa. Aquest tipus de xarxes han demostrat que funcionen millor amb conjunts de dades més petites. En canvi, per a conjunts de dades més complexes i amplies no són tan útils i seria millor fer servir una xarxa LTSM.

Capítol 2: DCASE Challenge

2. DCASE Challenge

2.1 Què és el repte DCASE?

Es tracta d´un repte anual que permet a diferents grups o organitzacions participar en una competició en detecció i classificació d´escenes i events acústics [1]. DCASE ajuda a desenvolupar mètodes d´extracció d´informació i models d'anàlisis d´aquesta informació fent servir una base de dades (Dataset) d´àudios públics i realitzant una classificació entre els diferents participants del repte tenint en compte la precisió, Recall i F-Score dels diferents sistemes presentats en el concurs.

Per a cada edició del concurs es creen entre 4 o 5 tasques noves i dintre de cada tasca es creen subtasques diferents. Els participants són lliures d´intentar realitzar totes les tasques o de centrar-se només en una per tal de millorar al màxim els resultats.

Per a cada tasca dels reptes, una base de dades d´entrenament (Training) i un sistema baseline són proporcionats als concursants. Finalment, uns dies abans de la data límit d´entrega, s´allibera la base de dades d´avaluació (Evaluation) que es fa servir per avaluar i donar els resultats finals.

2.1.1 Tasques del repte

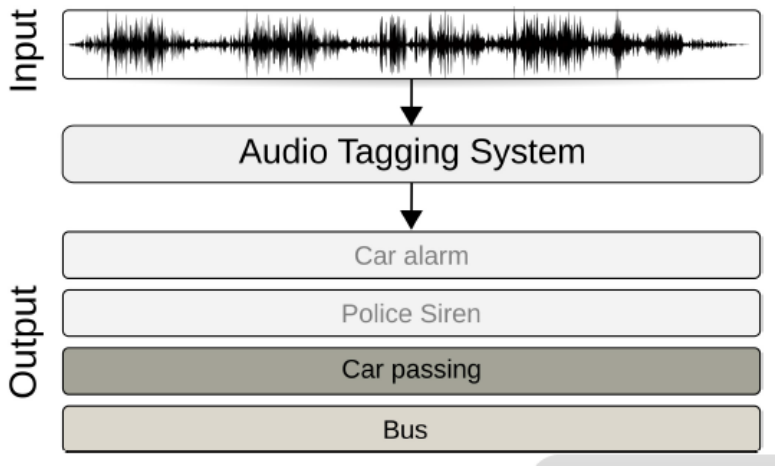
El repte de 2017 consistia de 4 tasques:

- **Tasca 1: Classificació d´escena acústica:** L´objectiu d´aquesta tasca és classificar els àudios en una de les classes que caracteritza el entorn en el qual es va gravar l´àudio (pàrquing, carrer, oficina, ...). La base de dades inclou àudios de 15 contexts diferents i aproximadament una hora d´àudio per cada context. Aquest tipus de repte es realitza en cada edició del concurs i no ha variat molt al llarg dels anys.
- **Tasca 2: Detecció de sons estranys:** Aquesta tasca es centra en la detecció de sons estranys en mescles d´àudio creades artificialment.
- **Tasca 3: Detecció d´esdeveniments sonors en àudio de la vida real:** En aquesta tasca es faran servir àudios gravats en entorns de la vida real. Això permetrà avaluar sistemes de classificació de detecció d´àudio amb múltiples fonts de so, tal i com ens trobem en el nostre dia a dia on els àudios es solapen uns amb els altres. Les anotacions en aquesta tasca es fan manualment i són totalment subjectives.
- **Tasca 4: Detecció a gran escala d´esdeveniments sonors en una ciutat per a cotxes intel·ligents:** La base de dades que es fa servir conté sons que es troben dintre d´una gran ciutat i que podrien ajudar, per exemple, a un cotxe autònom a percebre el que té al voltant i prendre decisions en conseqüència.

En aquest projecte ens centrarem en aquesta última tasca.

2.1.2 Definició de la Tasca 4

L´objectiu de la tasca és entrenar un sistema de classificació automàtica d'àudios d´una base de dades pública. Els àudios seran diferents sorolls comuns en una gran ciutat(diferents vehicles, botzines, etc...). El sistema haurà de ser capaç de classificar entre aquests diferents sorolls amb una precisió mínimament acceptable.



Il·lustració 14: Objectiu de la tasca 4
www.cs.tut.fi/sgn/arg/dcase2017/challenge

Aquesta tasca ha sigut escollida degut a la poca rellevància que es dona a aquest tipus d´àudios. Els resultats ajudaran a la detecció de diferents tipus de sons comuns en les grans ciutats i poden servir per a la implantació de la conducció autònoma de vehicles o altres utilitats en les cada vegada més de moda ciutats intel·ligents.

La tasca 4 està dividida en 2 subtasques:

- **Detecció d´events de so sense marques de temps**: La precisió, el recall i el F-score serà calculat fent servir l´àudio sencer, és a dir, durant el procés d´entrenament i classificació no es pot dividir en segments el àudio.
- **Detecció d´events de so amb marques de temps:** La precisió, el Recall i el F-score serà calculat fent servir petits segments d´àudio on apareixen els events sonors, és a dir, durant el procés de classificació i avaluació es podran dividir els àudios utilitzant les marques de temps com a referència (Les marques de temps indiquen entre quin segon inicial i final apareix dins de l´àudio un determinat esdeveniment sonor).

Nosaltres ens centrarem en fer l´entrenament i la classificació sense marques de temps. Fer servir marques de temps pot complicar i allargar l´entrenament així que ho deixarem per a una possible millora per al futur.

L´organització del repte posa una normativa per a la realització de cada tasca. En el cas concret de la nostra tasca, aquestes són les normes:

- No està permès fer servir àudios externs a la base de dades definida a la tasca 4. Àudios d´alguna de les altres tasques tampoc es poden fer servir per a la realització del sistema.
- No es pot fer servir la base de dades d´entrenament per a avaluar ni la base de dades d´avaluació per a entrenar.
- Està permès la manipulació de les bases de dades d´avaluació i d´entrenament.
- No està permès dividir els àudios fent servir les marques de temps en el moment d´entrenar al sistema.

2.1.3 Avaluació de la Tasca 4

L´avaluació d´aquesta tasca es farà calculant el F-Score del sistema. Aquesta mètrica depèn d´altres com la precisió i el recall així que passarem a explicar en que consisteix cada una d´elles [7]. També explicarem el concepte de matriu de confusió, que serà d´utilitat a l´hora de fer tots aquests càlculs.

2.1.3.1 Matriu de Confusió

La matriu de confusió és una eina per avaluar el rendiment d´un sistema d´aprenentatge autònom. Permet comprovar si el valor predit pel model és el real o es una predicció falsa. Es compten tots els casos de totes les categories i els totals són mostrats en la matriu de confusió. A continuació tenim un exemple d´una matriu de confusió amb 2 categories.

		Predicció	
		Positius	**Negatius**
Real	**Positius**	Verdaders Positius (VP)	Falsos Negatius (FN)
	Negatius	Falsos Positius (FP)	Verdaders Negatius (VN)

Taula 1: Matriu de confusió per a 2 classes

Les files de la matriu de confusió representen els valors reals que ens han donat com a referència i les columnes representen els valors de la predicció. Podem comprovar que en aquesta matriu hi ha 4 opcions possibles:

- **Verdaders positius (VP)** : Nombre de positius que han sigut predits correctament com a positius.
- **Verdaders Negatius (VN):** Nombre de negatius que han sigut predits correctament com a negatius.
- **Falsos Positius (FP):** Nombre de negatius que han estat predits incorrectament com a positius.
- **Falsos Negatius (FN):** Nombre de positius que han estat predits incorrectament com a negatius.

Al veure numèricament els resultats de les prediccions permet fer-se una idea del funcionament del model. Es pot comprovar ràpidament amb un cop d´ull que les prediccions correctes queden en la diagonal de la matriu i els valors restants són prediccions incorrectes.

El exemple que hem posat es de 2 classes però com es calcula si hem de fer la matriu de més de 2 classes? Les coses canvien una mica però també es pot calcular seguint unes fórmules. Suposant que tenim n classes diferents, la estructura de la matriu quedaria així:

		Predicció	
Real	c_{11}	$\cdots$	c_{1n}
	$\vdots$	$\ddots$	
	c_{n1}		c_{nn}

Il·lustració 15: Matriu n-classes

Sabent això, la forma de calcular els diferents elements de la matriu seria així:

Verdaders Positius (VP) → $VP_i = C_{ii}$
Falsos Positius (FP) → $FP_i = \sum_l^L C_{li} - VP_i$
Falsos Negatius (FN) → $\mathrm{FN}_i = \sum_l^L C_{il} - VP_i$
Verdaders Negatius (VN) → $VN_i = \sum_l^L \sum_k^L C_{lk} - VP_i - FP_i - FN_i$

Grup Equacions 1: Equacions per al càlcul de la matriu de n-classes [10]

Gràcies a la matriu de confusió podem calcular els VP,FP,VN i FN mitjançant les fórmules anteriors i aquests ens permetran fer el càlcul de les diferents mètriques que explicarem a continuació. Si analitzem una mica les fórmules veurem que les prediccions correctes continuen estant a la diagonal com en el cas de 2 classes.

2.1.3.2 Exactitud, Precisió, Record i F1-Score

Hi ha diferents mètriques a l´hora d´avaluar el sistema. Aquí comentarem les més comunes i que usarem en aquest projecte.

Exactitud: Bàsicament és el percentatge d´encert en les prediccions. Es pot calcular dividint el total de prediccions correctes entre el total de prediccions tant correctes com incorrectes.

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN}$$

Equació 1: Exactitud

L´exactitud dona una idea general del funcionament del sistema, però en segons quines classificacions pot ser insuficient. És necessari tenir més mètriques.

Precisió: Igual que l´exactitud però fent servir només les prediccions positives. La precisió és una bona forma de determinar el cost que tenen els falsos positius. Per exemple, en un sistema automàtic de detecció de Spam els falsos positius tenen molta més importància (Un correu no s´hauria de classificar com a Spam si no ho és).

$$Precisió = \frac{VP}{VP + FP}$$

Equació 2: Precisió

Per tant, la precisió es una millor forma de classificar donant més importància als falsos positius que no pas la exactitud, que ho té en compte tot.

Record (Recall): Permet donar més importància als falsos negatius enlloc de als falsos positius.

$$Recall = \frac{VP}{VP + FN}$$

Equació 3: Record

F1-Score: Permet fer un balanç entre la precisió i el record.

$$F1 = 2 * \frac{Precisió * Recall}{Precisió + Recall}$$

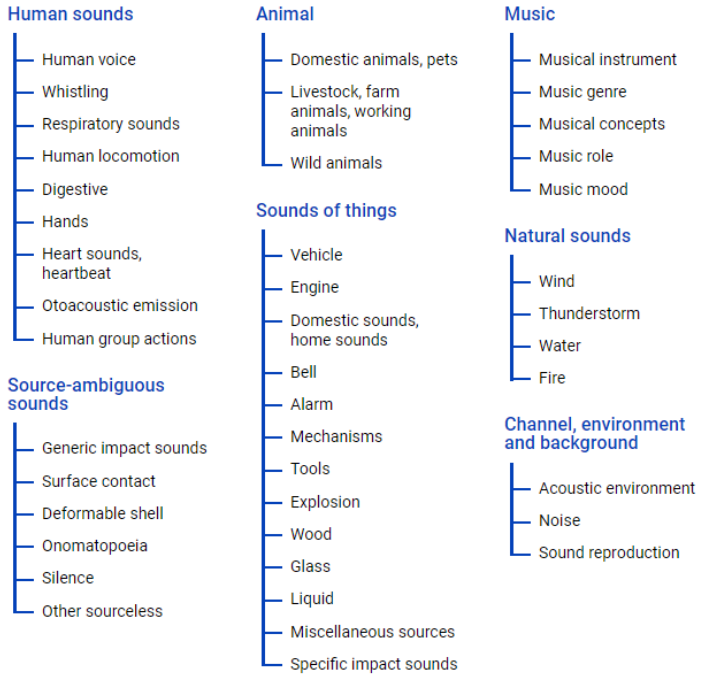
Equació 4: F1-Score

Per a realitzar l´avaluació del sistema en aquest projecte, crearem una matriu de confusió amb les diferents categories i farem servir les mètriques explicades anteriorment. La mètrica que dona una idea general del funcionament del sistema és l´última explicada, el F1-Score.

2.2 Base de dades del repte

2.2.1 Estructura del Audioset de Google

Aquest projecte es realitzarà amb un subconjunt dintre del Audioset públic de Google "***AudioSet: An Ontology And Human-Labeled Dataset For Audio Events***" [6]. L´audioset consisteix en una ontologia en expansió de 632 classes d´esdeveniments de so i una col·lecció de 2 milions de clips etiquetats de so de 10 segons extrets de 2 milions de vídeos de Youtube. L´ontologia està formada com un grup jeràrquic de categories que abarca una amplia gama de sons humans i animals, instruments musicals de diferents gèneres de música i sons ambientals del dia a dia:



Il·lustració 16: Ontología del Audioset de Google

La tasca 4 del repte DCASE en fa servir 17 englobades dins de 2 grans categories:

Sons d´alarma	Sons de vehicles
Botzina tren	Bicicleta
Botzina camió	Skate
Alarma cotxe	Cotxe
Xiulets Marxa enrere	Cotxe en moviment
Sirena Ambulància	Autobús
Sirena Cotxe policia	Camió
Sirena camió de bombers	Moticicleta
Sirena Defensa Civil	Tren
Crits	

Taula 2: Classes usades en la tasca 4 del DCASE 2017

En inici, abans de adaptar la base de dades per a fer-la servir en el projecte, aquesta es la quantitat de àudios que conté:

Sons Alarma:

- Botzina Tren: 441
- Botzina Camió:407
- Alarma cotxe:273
- Xiulets Marxa Enrere:337
- Sirena Ambulància:624
- Sirena Cotxe Policia:2399
- Sirena Camió de bombers:2399
- Sirena Defensa Civil:1506
- Crits:744

Sons Vehicles:

- Bicicleta:2020
- Skate:1617
- Cotxe:25744
- Cotxe en moviment:3745
- Autobús:3745
- Camió:7090
- Motocicleta:3291
- Tren:2301

Cada subconjunt està dividit en 2: la part d´entrenament (development o training) i la part d´avaluació (test o evaluation). Per poder descarregar la base de dades es proporcionen dos arxius, un per descarregar la part d´entrenament i l´altra per la part d´avaluació. Aquests arxius tenen la següent estructura:

```
CID,start_seconds,end_seconds,positive_labels
```

Per exemple:

```
2ceUOv8A3FE,20.000,30.000,"Train horn,Train","/m/0284vy3,/m/07jdr"
```

La primera columna “2ceUOv8A3FE” es el ID del vídeo de Youtube d´on s´ha extret el clip de 10 segons d´àudio. La segona i tercera columnes, t=20 seg a t=30 seg correspon a la part del vídeo on apareix l´àudio que ens interessa. L´última columna /m/0284vy3 (Train Horn) i /m/07jdr (Train) indiquen quines classes i events apareixen en l´àudio.

Hem de comentar que l´àudio no indica els límits on apareix cada classe de so dintre del clip de 10 segons. Per tant es pot considerar que les anotacions són etiquetes dèbils.

El subconjunt d´entrenament que s´entrega per a la realització d´aquesta tasca no està balancejat i per tant, pot comportar alguns problemes que comentarem després a la hora de realitzar l´entrenament. Tot i que el subconjunt està desequilibrat contindrà almenys 30

clips per classe com a mínim. Cal indicar també que un clip d´àudio pot correspondre a més d´una classe a la vegada.

2.3 Sistema Baseline

A l´hora de realitzar la tasca, els organitzadors proporcionen el sistema baseline. El baseline es la implementació bàsica de la tasca, de tal forma que tinguem un punt de partida a l´hora de comparar resultats amb els del nostre sistema propi. Un cop desenvolupat el nostre programa, si els nostres resultats són pitjors que els del baseline significa que no anem per bon camí. Els resultats del sistema baseline són els mínims que es poden obtenir sense que es consideri que són uns resultats totalment aleatoris.

Aquestes són les característiques del sistema que proporciona l´organització:

El sistema baseline ha sigut implementat amb Python i està basat en una arquitectura d´un perceptró multicapa fent servir com a característiques les energies log mel-band. Fent servir aquestes característiques, una xarxa neuronal de 2 capes ocultes amb 50 neurones per capa es entrenada fent 200 iteracions per cada classe. La decisió de la detecció es fa a la capa de sortida de la xarxa i es fa mitjançant neurones amb la funció de sigmoid.

Aquest seria el resum de les característiques del sistema durant la execució:

- Arquitectura de la xarxa neuronal basada en un Perceptró Multicapa
- **Software:** Python 2.7 o 3.6
- **Entrenament:** Adam algorithm for gradient-based optimization.
- **Features:** Log Mel Band
- **Vector de característiques:** 40 valors Mel Band amb 5 trames consecutives = 200 valors. Les trames són de 40 ms amb 50% de solapament entre elles.
- **Learning Rate** = 0.001

Com hem comentat, la tasca s'avaluarà amb el valor del F1-Score. La execució del baseline dona els següents resultats:

Mètrica	Resultat
F1-Score	10,9 %
Precisió	7,8 %
Recall	17,5 %

Taula 3: Resultats Baseline

Aquests seran els resultats que intentarem millorar desenvolupant el nostre sistema propi.

2.4 Sistemes guanyadors en edicions anteriors

En el repte DCASE participen anualment molt grups de recerca d´arreu del món. Pot ser una bona idea repassar una mica els sistemes guanyadors de les últimes edicions del concurs i veure quines són les característiques principals dels sistemes que han presentat. Per sort, al finalitzar el repte, tothom que ha participat està obligat a presentar un document tècnic sobre la tecnologia que han fet servir i nosaltres podrem aprofitar-ho per veure quines tecnologies donen millors resultats.

Any 2017

El guanyador d´aquell any va ser el departament de *Processament de senyal i Àudio de la Universitat de Surrey.* En la Taula 2 es poden veure els resultats que van obtenir i la comparació amb els resultats que et dona el sistema baseline.

Mètrica	Resultat	Baseline
F-Score	55,6 %	10,9 %
Precisió	61,4 %	7,8 %
Recall	50,8 %	17,5 %

Taula 4: Resultats Guanyador de la tasca 4 del DCASE 2017 en comparació amb el Baseline

Aquestes són les característiques principals del sistema que van presentar:

- Arquitectura d´una xarxa neuronal convolucional recurrent o CRNN
- Àudios mono mostrejats a 44.1 Khz
- Log-Mel energies com a features
- Learnable-gate activation function a la capa de sortida

Una explicació amb més detall es pot trobar en l´informe que van presentar al finalitzar el concurs [9].

Any 2018

La tasca 4 del DCASE del any 2018 és molt semblant a la que estem realitzant en el projecte. Es tractava de classificar events acústics sense cap relació entre ells en 10 classes diferents (gossos, gats, aigua en moviment, etc...) . Les característiques principals del sistema que han presentat es poden resumir en:

Els àudios que s´han utilitzats són mono i han estat mostrejats a 22,05 KHz ja que no consideren útils les altes freqüències per a la detecció i no conté molta informació de rellevància. Han fet servir una arquitectura d´una xarxa neuronal convolucional recurrent o CRNN fent servir el espectrograma mel com a característiques dels àudios. L´espectrograma s´ha extret amb una finestra de 2048 mostres (1600 de solapament) que recorre l´àudio. D´aquí s´extreuen 640 trames amb la informació de l´àudio que es farà servir per alimentar la xarxa neuronal. També es fan servir mètodes més avançats com el Context Gating o el Mean-Teacher que serveixen per millorar l´eficàcia del model.

Els resultats aconseguits es poden consultar a la següent taula :

Mètrica	Resultat	Baselyne
F1-Score	34,4 %	14 %
ER [8]	1,09	1,54

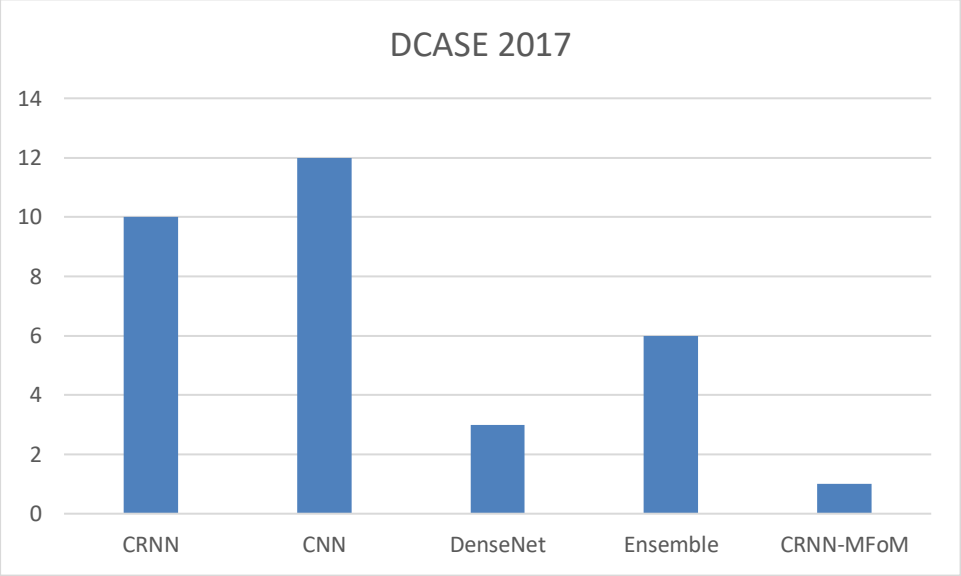
Taula 5: Resultats guanyador de la tasca 4 del DCASE 2018

Un document amb una explicació més detallada del sistema es pot consultar aquí [16].

Tecnologies usades en el Repte

A part de veure els guanyadors, podem donar un repàs de quines tecnologies han fet servir els 32 participants en els reptes d´edicions passades. Això ens permetrà veure realment que és el que es fa servir en la pràctica i cap a on està evolucionant l´aprenentatge profund.

En el any 2017, aquestes van ser les tecnologies usades:



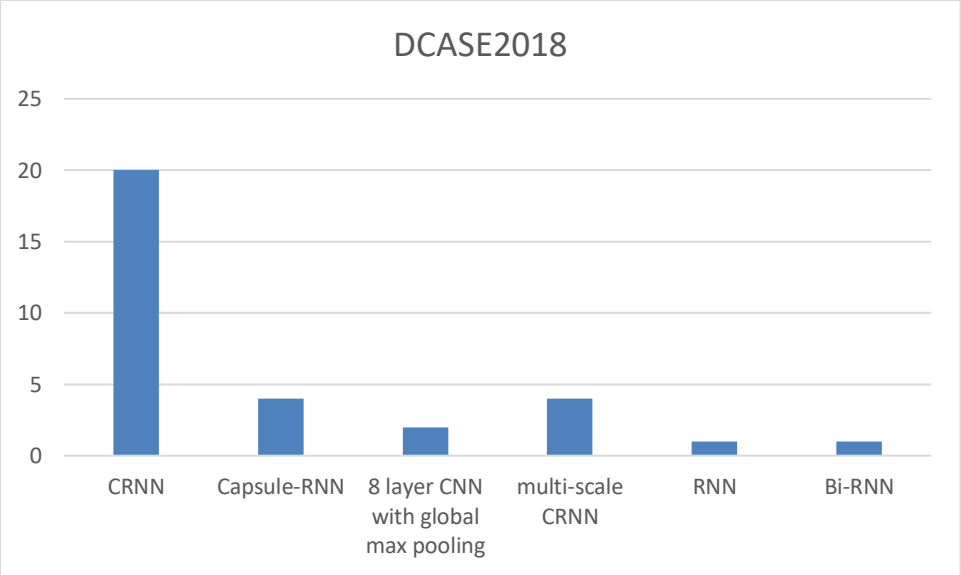
Il·lustració 17: Tecnologies usades en el 2017

Com podem veure, les xarxes convolucionals i les xarxes convolucionals recurrents s'emporten la gran majoria. També apareixen alguns sistemes que són variacions d´aquests, com els DenseNet o xarxes convolucionals denses que tenen les connexions entre neurones més properes que les xarxes tradicionals. Si alguna cosa es pot treure en clar és que el punt

[8] Event Based Error Rate: Els resultats obtinguts es comparen amb el arxiu de referència i es calcula la tassa d´error. Com més baixa és la tassa d´error millor resultats obtinguts.

en comú en els sistemes de tots els participants han estat les xarxes convolucionals i que en casos reals el que dona millors resultats es fer servir aquest tipus de xarxes.

En el any 2018 hi ha una tasca equivalent a la que estem analitzant. En aquest cas, les tecnologies del primers 32 participants són aquestes:



Il·lustració 18: Tecnologies usades en el 2018

El que més sorprèn en aquest cas és que sembla que la tendència es dirigeix a deixar de fer servir les tradicionals xarxes convolucionals i substituir-les per xarxes convolucionals recurrents.

Capítol 3: Sistema Propi

3. Sistema Propi

En aquest capítol explicarem com és el programa que hem creat per a realitzar la classificació i poder treure resultats. El primer objectiu va ser fer servir el sistema baseline proporcionat per l´organització però degut a la seva complexitat i que no es va poder executar degut a la falta d´una llibreria que en el moment de creació d´aquest projecte continua sent privada, es va decidir crear un sistema propi més senzill seguint un exemple d´internet [8].

3.1 Programari Usat

Python

Il·lustració 19: Icona de Python
www.python.org

Python es un llenguatge de programació multiplataforma de codi obert que s´ha convertit en un indispensable avui en dia per a realitzar aplicacions i que té un gran àmbit d´utilització. Python és ideal per treballar amb grans volums de dades perquè afavoreix la seva extracció i processament i aquesta és una de les raons per les qual és el gran favorit de les empreses de Big Data. Compta amb una amplíssima gama de llibreries de recursos que ens permetrà automatitzar moltes de les tasques que haurem de fer. A part, al ser un llenguatge que es usat per molt usuaris arreu del món, per internet podem trobar una comunitat molt activa a l´hora de resoldre dubtes i compartir coneixements. Els seus casi 30 anys des que va sortir al mercat indiquen que es tracta d´un llenguatge molt madur i que la gent continuarà fent servir degut a la seva qualitat i practicitat d´ús.

Tot el codi realitzat per a la creació del sistema ha estat fet amb python 2.7 i diferents llibreries compatibles per a aquest.

Anaconda

Il·lustració 20: Icona Distribució Anaconda
repo.continuum.io

Anaconda es un entorn de treball i una de les moltes distribucions del llenguatge python per a computació científica, ciència de dades, anàlisis estadístics i aprenentatge autònom. Algunes de les característiques d´Anaconda són:

- Es gratuïta tant per ús personal com comercial i per tant, es pot fer servir per al que vulgui cadascú.
- Té versions tant per a Windows, Linux i mac.
- Es poden crear diferents entorns dins anaconda i cadascun d´ells pot tenir diferents versions de paquets. Per exemple, pots tenir un entorn per a fer servir Python 2.7 i un altre diferent per a Python 3.3 .
- Té un administrador de paquets anomenat *conda* que permet instal·lar, crear i actualitzar paquets i llibreries addicionals.

Mitjançant Anaconda, s´ha creat un entorn per al projecte amb python 2.7 i totes les llibreries necessàries per a executar el nostre programa en local. Per obtenir els resultats finals el codi es pujarà als servidors de la UPC sense necessitat de fer servir Anaconda, ja que aquests servidors ja contenen tot el software necessari per a executar el programa.

Tensorflow

Il·lustració 21: Icona Tensorflow
www.towardsdatascience.com

Tensorflow es la plataforma d´aprenentatge profund de Google i una de les més importants del món en el sector de la intel·ligència artificial. Es tracta d´una llibreria de codi obert per a construir i entrenar xarxes neuronals i que permet detectar patrons i correlacions en les dades d´entrada.

Tensorflow es molt escalable ja que permet implementar els càlculs en una o varies CPUs o GPUs en equips d´escriptori, servidors i fins i tot dispositius mòbils.

Hi una llibreria de Tensorflow compatible amb python, que és la que s´ha fet servir per realitzar el projecte.

3.2 Adaptació de la base de dades

Per a la realització del projecte primer hem hagut de fer alguns canvis en la base de dades proporcionada per l´organització per a que funcioni millor amb el nostre codi. En els següents apartats definirem quins han sigut aquests canvis.

3.2.1 Descarregar la base de dades

Hem descarregat la base de dades fent servir els arxius de referència (Ground Truth) comentats anteriorment i els següents programes:

- **Oracle VM virtual box**: Màquina virtual amb sistema operatiu Linux.
- **Youtube-dl:** Permet descarregar un vídeo de Youtube passant-li el link. Si afegim el paràmetre *extract-audio* ens permetrà descarregar només l´àudio del vídeo en format mp3.
- **FFmpeg:** Amb FFmpeg hem tallat l´àudio per la part que ens interessa fent servir el segon inicial i final que ens indica en els arxius per a descarregar les base de dades proporcionats per l´organització del repte.
- **Sox:** El fem servir per convertir els àudios de format mp3 a format wav. En la realització d´aquest projecte farem servir els àudios en format wav.
- **Sed:** Permet extreure strings[9] d´un fitxer de text. L´hem fet servir tant per extreure el id del vídeo de Youtube per poder descarregar-lo com per extreure el segon inicial i final per on s´havia de tallar.

En una màquina virtual de Linux, s´ha creat un script de bash[10] per a descarregar la base de dades fent servir els programes que acabem de comentar.

3.2.2 Eliminació dels àudios multiclasse

En la base de dades del repte hi ha molts àudios que pertanyen a 2 o fins i tot 3 classes a la vegada. Això és degut a que els àudios fan servir una anotació dèbil, és a dir, que en l´àudio de 10 segons no s´indica en quina porció apareix cada classe i que, per exemple, els primers segons poden pertànyer a una classe i els segons finals a una altra totalment diferent. Un dels problemes que pot generar el fet de tenir àudios categoritzats en més d´una classe a la vegada es que es confonguin entre ells. Per tant, per tal d´optimitzar el funcionament del sistema, eliminarem tots els àudios multiclasse de la base de dades d´entrenament per tal de no entrenar amb àudios que puguin portar a confusions en el moment de classificació.

3.2.3 Adaptació dels fitxers de referència

Els fitxers de referència (ground truth) indiquen a quina classe pertany cada àudio. Es fa servir en el moment de l´entrenament per poder indicar a quina classe pertany les característiques que passem de cada àudio i en el moment de la classificació, permet saber si les prediccions que hem fet han estat correctes o no. En el nostre cas, tenim 2 arxius de

[9] Cadena de caràcters

[10] Bash és un intèrpret de comandes que permet executar instruccions contingudes en un script. Funciona en tots els entorn de Unix.

referència: un per la base de dades d´entrenament i l´altra per la d´avaluació. Aquí tenim un exemple de la estructura dels fitxers de referència que farem servir:

```
AR-KmtlXg4Y_70.000_80.000.wav    70.000   80.000   Car alarm
-60XojQWWoc_30.000_40.000.wav    30.000   40.000   Reversing beeps
-6d-zxMvC5E_30.000_40.000.wav    30.000   40.000   Reversing beeps
```

Com podem veure, la primera columna indica l´àudio i l´última a quina classe pertany. Les 2 columnes intermitges indiquen entre quins segons apareixia l´esdeveniment sonor dins de l´àudio, però en aquest cas ja no es d´utilitat ja que hem retallat els àudios anteriorment.

Un cop explicat que és un arxiu de referència, aplicarem uns canvis per poder realitzar el projecte. El primer que farem serà crear 2 arxius de referència nous (un per a l´entrenament i l´altra per a l´avaluació) a partir dels que ens han donat. En aquest arxius unificarem totes les classes en les dues classes principals: sons d´alarma i sons de vehicles. Això ens permetrà començar realitzant experiments fent servir primer les dues classes principals i finalment passar a utilitzar les 17 classes finals. Per tant ara tindrem 4 arxius: 2 que estan unificats amb dues classes i 2 més que estan amb les 17 classes inicials.

Per altra banda, haver d´estar fent servir el nom complet de les classes del àudio pot alentir el procés d´entrenament i classificació i, a més, fer el codi més complex. Per tant, per solucionar aquest problema, assignarem a cada classe un número i treballarem sense fer servir els noms de les classes:

Taula Equivalència 2 Classes

Sons Alarma	0
Sons Vehicles	1

Taula 6: Equivalència Classes-Número (2 classes)

Taula Equivalència 17 Classes

Sons Alarma	Equivalència	Sons Vehicles	Equivalència
Botzina Tren	0	Bicicleta	9
Botzina Camió	1	Skate	10
Alarma Cotxe	2	Cotxe	11
Xiulets Marxa Enrere	3	Cotxe en moviment	12
Sirena Ambulància	4	Autobús	13
Sirena Cotxe Policia	5	Camió	14
Sirena Camió bombers	6	Motocicleta	15
Sirena Defensa Civil	7	Tren	16
Crits	8		

Taula 7: Equivalències Classe-Número (17 classes)

Amb el que hem explicat, el arxius de referència queden de la següent manera, amb la última columna indicant la classe a la que pertany l´àudio:

```
AR-KmtlXg4Y_70.000_80.000.wav    70.000   80.000   2
-60XojQWWoc_30.000_40.000.wav    30.000   40.000   3
-6d-zxMvC5E_30.000_40.000.wav    30.000   40.000   3
```

3.2.4 Àudios dividits per classe

Aquesta és la llista final d`àudios dividits per subconjunt i per classe després d'haver adaptat la base de dades. [11]

Àudios d´entrenament dividits per classe

Sons Alarma	Quantitat	Sons Vehicles	Quantitat
Botzina Tren	94	Bicicleta	1874
Botzina Camió	256	Skate	1509
Alarma Cotxe	149	Cotxe	22644
Xiulets Marxa Enrere	88	Cotxe en moviment	3577
Sirena Ambulància	309	Autobús	3594
Sirena Cotxe Policia	1772	Camió	6881
Sirena Camió bombers	1186	Motocicleta	3162
Sirena Defensa Civil	1368	Tren	2074
Crits	634	**Total**	**45315**
Total	**5856**		

Taula 5: Àudios del subconjunt d´entrenament dividits per classe

[11] El llistat final s´ha realitzat mitjançant el codi que es pot consultar en el **annex 1**.

Àudios d´avaluació dividits per classe

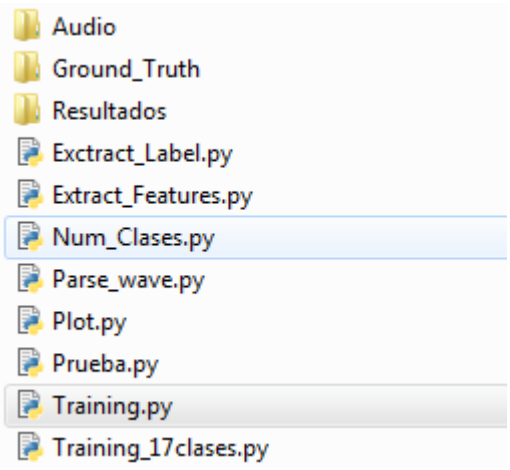
Sons Alarma	Quantitat	Sons Vehicles	Quantitat
Botzina Tren	40	Bicicleta	57
Botzina Camió	47	Skate	59
Alarma Cotxe	47	Cotxe	235
Xiulets Marxa Enrere	24	Cotxe en moviment	60
Sirena Ambulància	38	Autobús	56
Sirena Cotxe Policia	16	Camió	80
Sirena Camió bombers	40	Motocicleta	56
Sirena Defensa Civil	54	Tren	127
Crits	59	**Total**	**730**
Total	**365**		

Taula 6: Àudios del subconjunt d´avaluació dividits per classe

3.3 Definició del sistema

Per a realitzar la nostra xarxa neuronal hem seguit un exemple que es pot consultar aquí [12]. L´hem adaptat i afegit noves funcions per a que funcioni amb la nostra base de dades.

Primerament, aquesta es la estructura de carpetes del programa que hem realitzat:



[12] http://aqibsaeed.github.io/2016-09-03-urban-sound-classification-part-1/

Dintre de la carpeta “Audio” tindrem dues subcarpetes: “Training” i “Evaluation”. La primera tindrà la base de dades ja adaptada de Training i la segona subcarpeta la base de dades amb la que realitzarem l´avaluació. A la carpeta “Ground_Truth” tindrem 4 arxius: 2 arxius per a realitzar l´entrenament i avaluació amb dues classes i 2 arxius per fer-ho amb 17 classes. Finalment a la carpeta resultats apareixeran la precisió, recall, F1-Score i la matriu de confusió després de l´execució del programa.

A part tenim diferents arxius en els que hem realitzat el codi del programa. Els arxius Training.py i Training_17clases.py tenen el codi principal i des d´aquests arxius es criden altres funcions per a extreure les característiques dels àudios i les etiquetes corresponents.

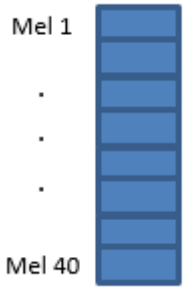
El sistema el podríem dividir en dues parts:

- Extracció de Característiques
- Definició Xarxa Neuronal

3.3.1 Extracció de característiques

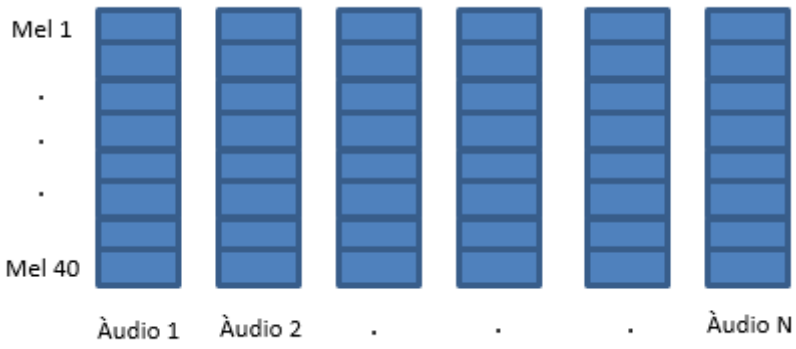
Per extreure les característiques dels àudios farem servir la llibreria **Librosa** de python. Amb aquesta llibreria es poden extreure diferents característiques dels àudios, per exemple els melspectogram, spectral_contrast o tonnetz. En canvi nosaltres farem servir els **mfcc** (Mel-frequency cepstral coeficients) una de les característiques més usades en la classificació d´àudio i basat en les classificacions del repte d´anys anteriors és el que acostuma a donar millors resultats.

El codi recorrerà la carpeta d´ àudios de “Training” i per cada àudio s’extrauran 40 coeficients mel que s´inseriran en un vector de característiques de mida 40:



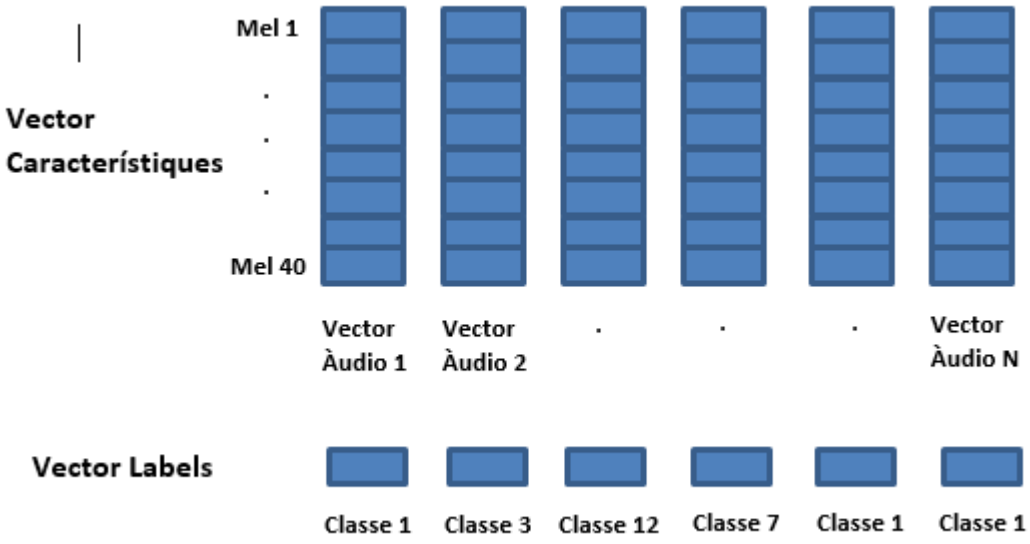
Il·lustració 22: Vector de característiques d´un àudio

Cada vector de característiques s´inserirà en un altre vector, de tal forma que ens quedarà un vector de vectors de mida N, on N es el nombre d´àudios als quals hem extret les característiques:



Il·lustració 23: Vector de vectors de característiques

A la vegada que recorrem la carpeta amb els àudios i anem traient els coeficients mel, també haurem d´extreure la classe a la que pertany l´àudio per poder relacionar les característiques amb la classe a la que corresponen. Les classes les extraurem dels fitxers de referència de la carpeta "Ground_Truth" i les inserirem en un vector de Labels. La forma de relacionar el vector de Labels amb el vector de característiques dels àudios serà de la següent forma:



Il·lustració 24: Relació del vector de característiques amb el vector de labels

Com podem veure, la posició 0 del vector de Labels correspondrà a la posició 0 del vector de característiques. D´aquesta manera podrem saber que els coeficients mel que estan a la posició 0 corresponen a la classe de la posició 0 del vector de Labels.

Finalment, el codi també farà la extracció de les Labels dels àudios de la carpeta "Evaluation" ja que a la hora de comprovar la precisió de la predicció, necessitarem comparar la classe que ha estat predita amb la classe real.

3.3.2 Característiques de la xarxa neuronal

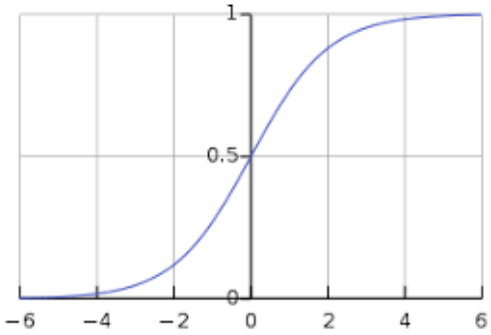
Per a entrenar les nostres dades farem servir una xarxa neuronal multicapa. La creació i l´entrenament s´ha realitzat mitjançant Tensorflow. La xarxa neuronal conté una capa d´entrada, dues capes intermitges i una capa de sortida. Les funcions d´activació que tindrà cada capa de neurones són les següents:

Capa d´entrada: Aquí tindrem les variables d´entrada que seran els diferents vectors amb els coeficients mel dels àudios que hem extret anteriorment.

Capa 1: Tenim una funció de Sigmoid associada a les neurones d´aquesta capa. L´expressió corresponent és la següent:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

L´equació anterior crea una funció d´aquest tipus:



Il·lustració 25: Gràfica de la funció Sigmoid
www.towardsdatascience.com

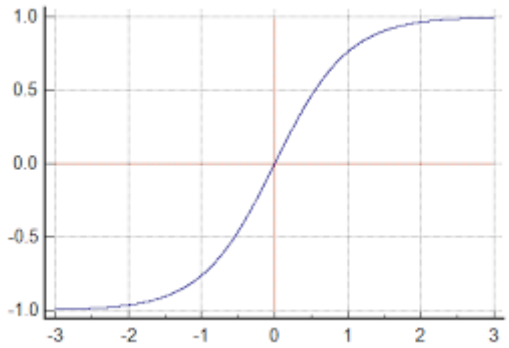
Bàsicament, si el valor que li arriben a les neurones es transformarà en un valor entre 0 i 1. Aquesta funció té molts avantatges entre les quals estan:

- Al ser una funció no-lineal ens permet esquivar el problema de la linealitat de les xarxes neuronals.
- La sortida està delimitada entre 0 i 1, i per tant pot ser interpretat com una probabilitat.

- Tendeix a portar les activacions a valors molt propers al final de la corba. Això vol dir que fa distincions clares en la predicció.
- Es fàcil de calcular computacionalment parlant.

Capa 2: Per a la segona capa de neurones tenim una funció tangent hiperbòlica:

L´expressió corresponent és la següent:



Il·lustració 26: Gràfica de la funció tangent hiperbòlica

És una funció semblant a la de la primera capa però la sortida està delimitada entre -1 i 1, enlloc de 0 i 1 com la sigmoid. Un dels principals avantatges que té respecte a la funció sigmoid és que la pendent de la corba és molt més pronunciada o el que és el mateix, que la derivada és més gran en el cas de la tangent. Això permet que en cada iteració els pesos de les neurones variïn més ràpid que amb la funció sigmoid. Els principal avantatges d´aquesta funció són:

- No es saturen tant fàcilment com les funcions sigmoid
- Té la derivada més gran i per tant, permet actualitzar els pesos més ràpidament en cada iteració.

Capa de Sortida: En aquesta capa tenim la funció Softmax:

$$P(y=j \mid \Theta^{(i)}) = \frac{e^{\Theta^{(i)}}}{\sum_{j=0}^{k} e^{\Theta_k^{(i)}}}$$

on $\Theta = w_0 x_0 + w_1 x_1 + \ldots + w_k x_k = \sum_{i=0}^{k} w_i x_i = w^T x$

Aquesta funció es fa servir com una distribució de probabilitat, és a dir, la suma de la probabilitat de totes les classes ha de ser 1. El nombre de neurones d´aquesta capa sempre serà igual a la quantitat de classes amb les que estem entrenant. En el cas dels experiments amb 2 classes tindrem 2 neurones de sortida i en el cas d´entrenar amb 17 classes tindrem 17 neurones de sortida. Al final en les neurones de sortida el que tindrem serà una probabilitat i això ens permetrà saber quina és la classe que té més possibilitats de ser correcte.

La funció de cost que farem servir en els experiments estarà donada per la següent equació:

$$Y * \log(y\_)$$

on **Y** son els valors reals dels coeficients dels àudios i **y_** són els valors predits en cada iteració a la sortida de l´última capa. A part, com hem explicat en capítols anteriors, farem servir gradients descendents per tal de minimitzar la funció de cost ja que és el mètode més efectiu.

Finalment, el nombre de neurones de les capes ocultes, el Learning-Rate i les iteracions que realitzarà la xarxa les podem anar variant i definir-les per a cada experiment. D´aquesta manera podrem saber quins valors inicials fan que la xarxa doni millors resultats.

3.3.3 Entrenament de la xarxa neuronal

El entrenament de la xarxa neuronal s´ha fet en els servidors de la UPC mitjançant una VPN[13]. Tots els arxius i la base de dades necessària s´han pujat mitjançant WinScp que fa servir el protocol SSH per transferir arxius.

Al demanar la execució del programa al servidor es pot indicar quina quantitat de recursos farà servir (CPUs, memòria RAM) i posa en cua la petició. Un cop el servidor hagi acabat amb totes les peticions de la cua, reservarà els recursos que li hem demanat i comença a executar el nostre codi.

3.3.4 Avaluació

Es fa amb la base de dades d´avaluació i es compara la classe que s´ha predit amb la classe real que indica el fitxer de referència. Un cop fet, es construeix la matriu de confusió i es fan els càlculs del precisió, recall i F1-Score tal i com s´ha explicat en els capítols anteriors.

[13] Virtual Private Network: Extensió d´una xarxa d´àrea local dintre d´una xarxa pública com internet

Capítol 4: Resultats

4. Resultats

En aquest capítol mostrarem els resultats obtinguts amb la xarxa neuronal plantejada en el capítol anterior. Els resultats els dividirem en 2 seccions: resultats fent servir 2 classes i resultats amb 17 classes.

Per als resultats amb 2 classes el procediment que s´ha seguit ha estat el següent:

- Assignar un valor a les iteracions de la xarxa, al Learning-Rate i al nombre de neurones per capa i executar el procés d´entrenament i classificació.
- Calcular la matriu de confusió amb 2 classes comparant els valors real i les prediccions i identificant els Verdaders Positius, Verdaders Negatius, Falsos Positius i Falsos Negatius (**Taula 1 del Apartat 2.1.3.1)**
- Calcular l´exactitud, precisió, recall i F1-Score **(Equació 1, Equació 2, Equació 3 i Equació 4 del apartat 2.1.3.2)**
- Variar els valors d´iteracions, Learning-Rate i nombre de neurones i tornar a repetir el procés.

El fet de realitzar diferents experiments variant els paràmetres de la xarxa ens servirà per veure quins son els valors inicials que ens permeten obtenir millors resultats.

Per als resultats amb 17 classes el procediment ha estat una mica diferent. Per als paràmetres de la xarxa agafarem els valors que han donat millors resultats en els múltiples experiments que hem fet amb 2 classes i els aplicarem directament. D´aquesta manera no necessitem fer molts experiments per trobar uns valors inicials adequats. El procediment per a 17 classes ha estat:

- Assignar un valor a les iteracions de la xarxa, al Learning-Rate i al nombre de neurones basat en els resultats que haurem obtingut en els experiments de 2 classes.
- Calcular la matriu de confusió amb múltiples classes **(Il-lustració 14 i grup d´Equacions 1 del apartat 2.1.3.1)**
- Calcular la precisió, recall i F1-Score per classe **(Equació 1, Equació 2, Equació 3 i Equació 4 del apartat 2.1.3.2)**

4.1 Resultats 2 classes

Passarem a posar els diferents resultats que s´han obtingut en els experiments i en cada cas indicarem amb quins valors de la xarxa s´han aconseguit.

Experiment 1

- **Iteracions:** 50
- **Neurones Capa 1:** 180
- **Neurones Capa 2:** 200
- **Learning-Rate:** 0,01

Matriu:

	0	**1**
0	**58**	307
1	255	**475**

Mètriques Obtingudes:

Exactitud	Precisió	Recall	F1-Score
0,49	0,19	0,16	0,17

Taula 8: Mètriques Experiment 1

Mètriques per classe:

Classe	Precisió	Recall	F1-Score	Nº Prediccions	Nº real àudios
S. Alarma	0,19	0,16	0,17	308	365
S. Vehicle	0,61	0,76	0,63	787	730

Taula 9: Mètriques per classe del experiment 1

Ja es pot veure que la descompensació de la base de dades provoca que les prediccions tendeixin a la classe vehicle. Com es pot veure a la **Taula 9,** les mètriques donen millors resultats per a la classe vehicle que per a la classe Alarma.

Experiment 2

En aquest experiment variarem el número de neurones per veure si té algun efecte notable en els resultats:

- **Iteracions:** 50
- **Neurones Capa 1:** 280
- **Neurones Capa 2:** 300
- **Learning-Rate:** 0,01

Matriu:

	0	**1**
0	**73**	292
1	318	**412**

Mètriques obtingudes:

Exactitud	**Precisió**	**Recall**	**F1-Score**
0,44	0,19	0,2	0,19

Taula 10: Mètriques Experiment 2

Mètriques per classe:

Classe	**Precisió**	**Recall**	**F1-Score**	**Nº Prediccions**	**Nº real àudios**
S. Alarma	0,19	0,2	0,19	391	365
S. Vehicle	0,59	0,56	0,57	704	730

Taula 11:Mètriques per classe del experiment 2

Els resultats no semblen millorar al haver variat el nombre de neurones de cada capa. El F1-Score continua sent semblant al anterior experiment.

Experiment 3

En aquest experiment variarem les iteracions i el Learning-Rate.

- **Iteracions:** 100
- **Neurones Capa 1:** 180
- **Neurones Capa 2:** 200
- **Learning-Rate:** 0,001

Matriu:

	0	1
0	**189**	176
1	200	**530**

Mètriques obtingudes:

Exactitud	Precisió	Recall	F1-Score
0,66	0,49	0,52	0,50

Taula 12: Mètriques Experiment 3

Mètriques per classe:

Classe	Precisió	Recall	F1-Score	Nº Prediccions	Nº real àudios
S. Alarma	0,49	0,52	0,50	389	365
S. Vehicle	0,75	0,73	0,74	706	730

Taula 13: Mètriques per classe del experiment 3

Els resultats semblen millorar notablement al augmentar les iteracions i disminuir el Learning-rate. Tot i així, sembla que continua donant millors resultats la classe vehicle que no pas la classe alarma.

Experiment 4

Augmentarem ara les iteracions per veure si continua millorant o es queda igual.

- **Iteracions:** 200
- **Neurones Capa 1:** 180
- **Neurones Capa 2:** 200
- **Learning-Rate:** 0,001

Matriu:

	0	1
0	**193**	172
1	257	**473**

Mètriques obtingudes:

Exactitud	Precisió	Recall	F1-Score
0,61	0,43	0,53	0,47

Taula 14: Mètriques Experiment 4

Classe	Precisió	Recall	F1-Score	Nº Prediccions	Nº real àudios
S. Alarma	0,43	0,53	0,47	450	365
S. Vehicle	0,73	0,65	0,69	645	730

Taula 15: Mètriques per classe del experiment 4

Els resultats semblen no haver variat molt respecte l´anterior experiment tot i haver augmentat el nombre d´iteracions.

Experiment Final

Per últim disminuirem el Learning-Rate un altre cop per veure si hi ha millora.

- **Iteracions:** 200
- **Neurones Capa 1:** 180
- **Neurones Capa 2:** 200
- **Learning-Rate:** 0,0005

Matriu:

	0	1
0	**216**	235
1	159	**571**

Mètriques obtingudes:

Exactitud	Precisió	Recall	F1-Score
0,67	0,58	0,48	0,53

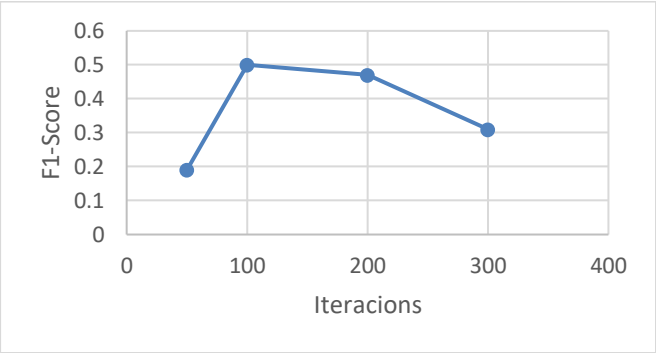
Taula 16: Mètriques Experiment Final

Classe	Precisió	Recall	F1-Score	Nº Prediccions	Nº real àudios
S. Alarma	0,58	0,48	0,53	375	365
S. Vehicle	0,71	0,78	0,74	806	730

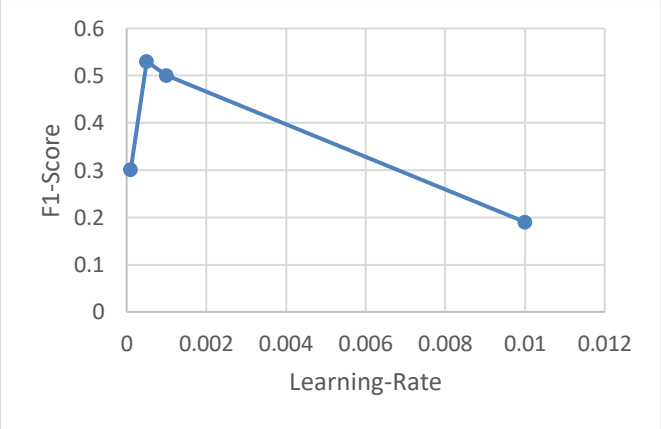
Taula 17: Mètriques per classe del experiment final

Podem veure que el F1-Score ha millorat una mica. Al final sembla que degut al desequilibri de la base de dades sempre obtenim millors resultats en la classe que té més àudios d´entrenament que no pas en l´altra.

Gràfiques 2 classes



Gràfica 1: Relació F1-score i iteracions



Gràfica 2: Relació F1-Score i Learning-Rate

Com es pot veure en la **Gràfica 1**, els valor d´iteracions que han donat millors resultats estan situats entre 100 i 200. En canvi, en la **Gràfica 2** podem veure que els millors valors per al Learning-Rate estan situats entre 0,0005 i 0.001. En certa manera ambdós valors estan relacionats. Com més petit és el Learning-Rate més gran hauran de ser les iteracions ja que sinó no s´aconseguirà minimitzar a temps la funció de cost i com més gran el Learning-Rate, més petit haurà de ser el valor de les iteracions. Cal tenir en compte també que a més iteracions i Learning-Rate més petit, més temps d´entrenament de la xarxa. La clau al final es trobar un terme mig pels 2 valors de manera que retorni bons resultats amb un temps d´execució raonable. En el nostre cas els millors resultats per a 2 classes han estat amb 200 iteracions i Learning-Rate de 0,0005 i ,per tant, aquests seran els valors que farem servir per executar el sistema per a 17 classes.

4.2 Resultats 17 classes

Aprofitant els experiments que hem realitzat amb 2 classes, configurarem la xarxa amb els valors que ens han donat millors resultats en el anterior apartat. D´aquesta manera ens evitarem haver de fer tants experiments. Així doncs, hem posat els següents valors per a realitzar l´entrenament per a 17 classes:

- **Iteracions:** 200
- **Neurones Capa 1:** 180
- **Neurones Capa 2:** 200
- **Learning-Rate:** 0,001

Matriu:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
0	**2**	3	0	0	2	0	0	0	0	0	0	1	0	0	0	0	32
1	1	**12**	0	0	0	0	4	0	0	0	0	9	2	0	12	0	7
2	1	1	**0**	0	0	7	12	8	0	0	0	15	0	0	0	0	4
3	4	0	0	**0**	1	1	2	2	0	0	0	4	5	1	1	3	0
4	3	0	0	0	**13**	21	0	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	1	**5**	5	4	0	0	0	0	1	0	0	0	0
6	4	1	0	0	3	4	**12**	4	0	0	0	5	3	1	1	0	2
7	1	0	0	0	2	5	17	**15**	0	0	0	11	2	1	0	0	0
8	2	1	0	0	1	1	2	0	**0**	5	0	29	11	0	1	0	6
9	0	0	0	0	0	3	0	0	0	**12**	0	36	2	0	0	1	3
10	0	3	0	0	0	2	1	1	0	16	**0**	21	9	1	0	3	2
11	0	0	0	0	0	0	0	0	0	0	0	**235**	0	0	0	0	0
12	0	0	0	0	3	0	0	0	0	0	0	34	**18**	1	0	0	4
13	0	0	0	0	0	0	0	0	0	0	0	23	12	**11**	10	0	0
14	0	0	0	0	0	0	0	0	0	0	0	49	21	2	**7**	1	0
15	0	0	0	0	0	0	1	0	0	0	0	32	13	1	1	**8**	0
16	56	0	0	0	0	0	0	0	0	0	0	27	2	1	0	0	**41**

Mètriques obtingudes:

Aquestes són les mètriques que hem obtingut del experiment comparades amb les del baseline proporcionat per l´organització:

Sistema	**Exactitud**	**Precisió**	**Recall**	**F1-Score**
Propi	0,36	0,27	0,23	0,22
Baseline	--	0,11	0,08	0,18

Taula 18: Mètriques obtingudes respecte el baseline

Com es pot veure les mètriques obtingudes són una mica millors que les del baseline tot i que hi moltíssim marge de millora encara.

Mirem ara les mètriques obtingudes per a cada classe:

Classe	Precisió	Recall	F1-Score	Nº Prediccions	Nº real àudios
B. Tren	0,03	0,05	0,04	74	40
B. Camió	0,57	0,26	0,36	21	47
A. cotxe	0	0	0	0	47
Marxa Enrere	0	0	0	0	24
S. Ambulància	0,5	0,34	0,40	26	38
S. Cotxe Policia	0,1	0,31	0,15	49	16
S. Camió Bombers	0,21	0,3	0,25	56	40
S. Defensa Civil	0,44	0,28	0,34	34	54
Crits	0	0	0	0	59
Bicicleta	0,36	0,21	0,27	33	57
Skate	0	0	0	0	59
Cotxe	0,44	1	0,61	531	235
Cotxe en moviment	0,18	0,3	0,23	102	60
Autobús	0,58	0,2	0,3	19	56
Camió	0,21	0,13	0,16	33	80
Motocicleta	0,5	0,14	0,22	16	56
Tren	0,41	0,32	0,36	101	127

Taula 19: Mètriques per classe

Es pot observar la mateixa tendència en la predicció en les classes de vehicles, sobretot en la classe cotxe. Realment el desequilibri en aquesta classe és molt gran respecte a les altres i d´aquí probablement la tendència a predir aquesta classe per sobre de les altres. A part també es veu que hi ha classes que no han estat predites mai, cosa que pot ser deguda als pocs àudios d´entrenament que tenien aquestes classes. Aprofundirem en aquest tema en les conclusions.

En la **taula 20** podem veure les classes que més s´han confós entre elles:

Classe predita	Classe real	Nº Confusions
B. Tren	Tren	56
Cotxe	Camió	49
Cotxe	Cotxe en moviment	34
Tren	B. Tren	32
Cotxe	Motocicleta	32

Taula 20: Major nombre de confusions entre classes

La classe cotxe es la que més confusions ha patit amb una grandíssima diferència. També tenim algun cas curiós com el de la “botzina del tren” i “tren” i al inrevés. Sembla que

aquestes 2 classes es confonen bastant entre elles. Escoltant algun àudios de la classe "botzina de tren", en molts casos es sent també el tren en moviment a part de la botzina i d´aquí potser la confusió.

En la **taula 20** podem veure un resum de les classes que han aconseguit millors resultats:

Classe	F1-Score
Cotxe	0,61
S. Ambulància	0,4
B.Camió	0,36
Tren	0,36
S. Defensa Civil	0,34
Autobús	0,3
Bicicleta	0,27
S. Camió Bombers	0,25
Cotxe en moviment	0,23
Motocicleta	0,22
Camió	0,16
S. Cotxe Policia	0,15
B.Tren	0,04
A.Cotxe	0
Marxa Enrere	0
Crits	0
Skate	0

Taula 21: Classificació segons F1-Score

Es pot veure que algunes classes tenen puntuació 0. Això vol dir bàsicament que la xarxa no ha predit cap àudio a aquella classe.

Capítol 5: Conclusions

5. Conclusions

Si alguna cosa es pot treure en clar de la realització d´aquest projecte es que tot el camp del Deep Learning i les xarxes neuronals és un tema molt complexa que es basa bàsicament en provar diferents combinacions d´estructures de xarxa fins que s'aconsegueix poc a poc anar millorant els teus resultats. No hi ha cap procediment que permeti aconseguir uns bons resultats de primeres ja que això es impossible degut a que depèn de moltes variables: complexitat de la base de dades, característiques de la xarxa neuronal, tipus de funcions que s'assignen a les neurones, paràmetres inicials de la xarxa, etc...

En els resultats que hem obtingut es pot veure com canvis no molt grans en els paràmetres inicials de la xarxa poden portar a resultats molt diferents. Al final es tracta de trobar una combinació de valors que funcioni amb la nostra base de dades que és al final el que hem fet en aquest projecte. Però com hem comentat, uns paràmetres que a nosaltres ens ha funcionat més o menys bé, a un altre persona amb una base de dades diferent potser ja no li funciona de la mateixa manera.

Tot i així hi ha molt marge de millora en la feina que hem fet per aconseguir que les prediccions finals siguin més encertades. A continuació proposarem una sèrie de propostes que podrien aconseguir que els resultats siguin millors:

- Una de les millores més clares que es podria realitzar seria fer servir les marques de temps proporcionades per l´organització del repte per a tallar els àudios pel lloc que ens interessa. Els clips d´àudio de 10 segons a part de l´esdeveniment sonor que el categoritza també contenen altres sons que no tenen res a veure. Aquests altres sons es poden considerar soroll per al propòsit d´aquest projecte per tant, tallant els àudios pels segons exactes on apareix l´àudio podríem millorar de forma considerable els resultats finals.

- Com hem explicat en anteriors capítols, el desequilibri de la base de dades pot suposar un problema. De fet, en els resultats obtinguts es pot veure que algunes classes tenien molts pocs àudios i això ha fet que no hagin estat predites moltes vegades. Per millorar els resultats s'hauria d´intentar tenir un nombre semblant d´àudios d´entrenament per a totes les classes. Degut a que en aquest projecte hem simulat una participació al repte DCASE, la normativa indica clarament que no es poden afegir àudios externs a la base de dades que ens proporcionen per tant l´única solució seria intentar eliminar àudios de les classes que en tenen molts (la classe cotxe sobretot).

- Una altra forma de lidiar amb el desequilibri de la base de dades sense necessitat d´eliminar àudios seria ponderar les classes. Bàsicament significa donar més importància a les classes amb poca presència dintre de la base de dades i donar menys importància a les classes amb molts àudios. Trobar una ponderació correcte

per a totes les classes pot fer millorar notablement els resultats finals. Si ens fixem amb els resultats obtinguts, la xarxa té molta tendència a predir la classe “cotxe” que és la que està més descompensada respecte a les demès amb diferència. Si li apliquem una ponderació per donar-li menys importància segurament la xarxa no farà tantes prediccions sobre aquesta classe.

A part de les millores comentades també seria una bona idea intentar canviar l´estructura de la xarxa que hem fet servir. S´ha fet servir una xarxa neuronal de capes ja que és una de les més “senzilles” per començar però hi ha tot un mon d´estructures de xarxes neuronals diferents amb les que experimentar i segur que alguna pot funcionar millor per al propòsit d´aquest projecte. Afegir més capes per fer una xarxa neuronal profunda, canviar la direcció del flux de dades a capes anteriors per a que la xarxa tingui memòria, canviar la funció de cost... les possibilitats són infinites.

Amb la realització d´aquest treball també ens hem adonat que es necessita molta potència de computació per a realitzar l´aprenentatge profund. Realment es un procés molt costós tant a nivell de recursos de la màquina com a nivell de temps. De fet, grups d´investigació que es dediquen a temps complet en aquest camp fan servir supercomputadors en paral·lel per a realitzar els entrenaments i tot i així un entrenament pot tenir un temps superior a un dia. Això ha estat un problema a l´hora de realitzar les primeres proves ja que aquestes s´han executat en un portàtil local arribant a saturar-lo per falta de recursos havent de reduir la mida de la base de dades per a poder realitzar proves en local.

Realment les possibilitats del Deep Learning són impressionants i segur que amb el pas dels anys s´anirà millorant i apareixeran noves formes d´utilitzar-lo. Auguro un bon futur per a aquesta tecnologia ja que sembla que cada cop es farà servir més en el nostre dia a dia.

Annex 1

Annex 1

En aquest annex estan els diferents codis que s´han construït per a realitzar aquest projecte.

1/ Codi que llegeix dels fitxers de referència i enumera quants àudios hi ha de cada classe, tant del set d´avaluació com del set d´entrenament.

```
import glob
import datetime
import os
import numpy as np

#Funció que extreu la etiqueta donant el id d´un àudio
def Exctract_Label(Link):
	Variable_Entrada=Link
	df = pd.read_csv('./Ground_Truth/Veritat_Training.csv')
	#print(df.loc[0,'Link':'Label'])
	len = df.shape[0]
	for i in range(0,len):
		Index = df.loc[i,'Links']
		Label = df.loc[i,'Label']
		if(Index==Link):
			Sortida=Label


	return Sortida

#Funció per recórrer els arxius *.wav y extreure els labels de cada àudio
def parse_audio_files(parent_dir,sub_dirs,file_ext="*.wav"):
	labels = np.zeros((17))
	contador = 0
	for label, sub_dir in enumerate(sub_dirs):
		for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):

			audio=fn.split('\\')[2]
			audio=audio[1:]
			Label = Exctract_Label(audio)
			labels[Label] = labels[Label] + 1
			contador = contador +1
			print(contador)

	return labels

#Definició de les carpetes on estan els àudios
parent_dir = './Audio'
tr_sub_dirs = ["Training"]
ts_sub_dirs = ["Evaluation"]
Clases_tr = parse_audio_files(parent_dir,tr_sub_dirs)
Clases_ts = parse_audio_files(parent_dir,ts_sub_dirs)

print(Clases_tr)
print(Clases_ts)
```

2/Programa que extreu les característiques, configura i crea la xarxa neuronal, la entrena i et retorna els resultats

```
#Configuració dels paràmetres de la xarxa neuronal
training_epochs = 150
n_hidden_units_one = 280
n_hidden_units_two = 300
learning_rate = 0.001


#Funció per extreure l´etiqueta mitjançant el link de l´àudio i el arxiu de referència
def Exctract_Label(Link,flag):
	Sortida = 22
	Variable_Entrada=Link
	if(flag == 0):
		df = pd.read_csv('./Ground_Truth/Veritat_Training.csv')
	else:
		df = pd.read_csv('./Ground_Truth/Veritat_Evaluation.csv')
	len = df.shape[0]
	for i in range(0,len):
		Index = df.loc[i,'Links']
		Label = df.loc[i,'Label']
		if(Index==Link):
			Sortida=Label


	return Sortida

#Funció per extreure les característiques d´un àudio passat com a paràmetre
def extract_feature(file_name):
   X, sample_rate = librosa.load(file_name)
   stft = np.abs(librosa.stft(X))
   mfccs = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T,axis=0)
   return mfccs

#Funció per recórrer la carpeta d´àudios *.wav y extreure les característiques i les etiquetes de cada un.
def parse_audio_files(parent_dir,sub_dirs,flag,file_ext="*.wav"):
	contador=0
	features, labels = np.empty((0,40)), np.empty(0)
	for label, sub_dir in enumerate(sub_dirs):
		for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):
			try:
				mfccs = extract_feature(fn)
			except Exception as e:
				print ("Error encountered while parsing file: ", fn)
				continue
			audio=fn.split('\\')[2]
			audio=audio[1:]
			if(Exctract_Label(audio,flag)==22):
				print('indeterminat')
			else:
				labels = np.append(labels,Exctract_Label(audio,flag))
				ext_features = np.hstack(mfccs)
				features = np.vstack([features,ext_features])
			contador = contador+1
			print(contador)
```

```
        return np.array(features), np.array(labels, dtype = np.int)

#Codifica cada classe amb un valor binari. Exemple amb 2 classes --> Classe 1: 01 Classe 2: 10
def one_hot_encode(labels):
        n_labels = len(labels)
        n_unique_labels = len(np.unique(labels))
        one_hot_encode = np.zeros((n_labels,n_unique_labels))
        one_hot_encode[np.arange(n_labels), labels] = 1
        return one_hot_encode

#Calcula la matriu de confusió comparant prediccions i valors reals. (17 classes)
def conf_matrix(true,pred,file):

        n_unique_labels = len(np.unique(true));
        suma = 0
        VP = 0
        FP = np.zeros((1,n_unique_labels))
        Fals_Pos = 0
        FN = np.zeros((1,n_unique_labels))
        Fals_Neg = 0
        Matrix = np.zeros((n_unique_labels,n_unique_labels))
        Prec_per_clase = np.zeros((1,n_unique_labels))

        for x in range(0,len(true)):
                if(true[x]==pred[x]):
                        Matrix[[true[x]],[pred[x]]] = Matrix[[true[x]],[pred[x]]] + 1

                elif(true[x]!=pred[x]):
                        Matrix[[true[x]],[pred[x]]] = Matrix[[true[x]],[pred[x]]] + 1

        for z in range(0,n_unique_labels):
                VP = VP + Matrix[[z],[z]]
                for y in range(0,n_unique_labels):
                        suma = suma + Matrix[[z],[y]]

                FN[0,z] = suma - Matrix[[z],[z]]
                Prec_per_clase[0,z] = Matrix[[z],[z]] / suma
                suma = 0

        suma = 0
        for v in range(0,n_unique_labels):

                for w in range(0,n_unique_labels):
                        suma = suma + Matrix[[w],[v]]

                FP[0,v] = suma - Matrix[[v],[v]]
                suma = 0



        print(Matrix)
        print('----------------------------------------------------------------------------------')
        file.write(str(Matrix) + '\n')
        file.write('----------------------------------------------------------------------------------' + '\n')
        file.write(str(Prec_per_clase))
```

```
        Precision = 0
        Recall = 0
        Fscore = 0

        return Precision,Recall,Fscore

#Estructura de carpetes per deixar els àudios
parent_dir = './Audio'
tr_sub_dirs = ["Training"]
ts_sub_dirs = ["Evaluation"]

#Fem l´extracció de característiques i labels
tr_features, tr_labels = parse_audio_files(parent_dir,tr_sub_dirs,0)
ts_features, ts_labels = parse_audio_files(parent_dir,ts_sub_dirs,1)

#Codifiquem les etiquetes
tr_labels = one_hot_encode(tr_labels)
ts_labels = one_hot_encode(ts_labels)

#Configuració de paràmetres de la xarxa
n_dim = tr_features.shape[1]
sd = 1 / np.sqrt(n_dim)
n_classes = 17

#Creem els tensors, un per les característiques i un altre per les etiquetes
X = tf.placeholder(tf.float32,[None,n_dim])
Y = tf.placeholder(tf.float32, shape=(None,n_classes))

#Primera capa de neurones
W_1 = tf.Variable(tf.random_normal([n_dim,n_hidden_units_one], mean = 0, stddev=sd))
b_1 = tf.Variable(tf.random_normal([n_hidden_units_one], mean = 0, stddev=sd))
h_1 = tf.nn.tanh(tf.matmul(X,W_1) + b_1)

#Segona capa de neurones
W_2 = tf.Variable(tf.random_normal([n_hidden_units_one,n_hidden_units_two],
mean = 0, stddev=sd))
b_2 = tf.Variable(tf.random_normal([n_hidden_units_two], mean = 0, stddev=sd))
h_2 = tf.nn.sigmoid(tf.matmul(h_1,W_2) + b_2)

#Capa de sortida
W = tf.Variable(tf.random_normal([n_hidden_units_two,n_classes], mean = 0, stddev=sd))
b = tf.Variable(tf.random_normal([n_classes], mean = 0, stddev=sd))
y_ = tf.nn.softmax(tf.matmul(h_2,W) + b)

init = tf.global_variables_initializer()

#Definim la funció de cost i el mètode per minimitzarla
cost_function = -tf.reduce_sum(Y * tf.log(y_))
optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cost_function)

correct_prediction = tf.equal(tf.argmax(y_,1), tf.argmax(Y,1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

cost_history = np.empty(shape=[1],dtype=float)
y_true, y_pred = None, None
```

```
print(datetime.datetime.now())
print("------------------------------------------------------------------------")

#Executem la xarxa
with tf.Session() as sess:
        sess.run(init)
        for epoch in range(training_epochs):
                _,cost = sess.run([optimizer,cost_function],feed_dict={X:tr_features,Y:tr_labels})
                cost_history = np.append(cost_history,cost)

        y_pred = sess.run(tf.argmax(y_,1),feed_dict={X: ts_features})
        y_true = sess.run(tf.argmax(ts_labels,1))
        print("------------------------------------------------------------------------")
        print("Test accuracy: ",round(sess.run(accuracy,
                feed_dict={X: ts_features,Y: ts_labels}),3))

#Treiem els resultats per pantalla i els escrivim en un arxiu de text
file = open('./Resultados/' + str(datetime.datetime.now().strftime('%Y-%m-%d_%H_%M_%S')) + '.txt','w')
file.write('' + '\n')
file.write(str(datetime.datetime.now()) + '\n')
file.write('Parámetros Actuales' + '\n')
file.write('' + '\n')
file.write('Training_epochs = ' + str(training_epochs) + '\n')
file.write('N_hidden_units_one = ' + str(n_hidden_units_one) + '\n')
file.write('N_hidden_units_two = ' + str(n_hidden_units_two) + '\n')
file.write('Learning_rate = ' + str(learning_rate) + '\n')
file.write('------------------------------------------------------------------------' + '\n')

p,r,f,s = precision_recall_fscore_support(y_true, y_pred, average="micro")
file.write('Accuracy: ' + str(round(p,3)) + '\n')
file.write('------------------------------------------------------------------------' + '\n')

Precision,Recall,Fscore = conf_matrix(y_true,y_pred,file)
file.write('------------------------------------------------------------------------' + '\n')
file.write('Precissió,Recall i F-Score per clase: ' + '\n')
file.write('\n')

Class_Names = ['B. Tren','B. Camió','Al. Cotxe','Marxa enrere','Sir. Ambulància','Sir. Cotxe Policia','Sir.
Bombers','Sir. Defensa Civil','Crits','Bicicleta','Skate','Cotxe','Cotxe en
moviment','Autobus','Camio','Motocicleta','Tren']

print(sk.metrics.classification_report(y_true, y_pred, target_names=Class_Names))
Report = sk.metrics.classification_report(y_true, y_pred, target_names=Class_Names)
file.write(Report + '\n')
```

Bibliografia

[1] *Detection and Classification of Acústics Scenes and events 2017* [En línia]. cs.tut.fi*, 2017.* Disponible a: /http://www.cs.tut.fi/sgn/arg/dcase2017/documents/dcase-2017-challenge-paper.pdf

[2] A.Burkov. *The Hundred-Page Machine Learning Book*. 2018.

[3] *Definición de sonido* [En línia]. Condeptodefinicion.de, 2011. Disponible a: http://conceptodefinicion.de/sonido/

[4] T.Rodriguez. *Como entender las claves del presente y el futuro de la inteligencia artificial* [En línia]. xataka.com, 2017. Disponible a: https://www.xataka.com/robotica-e-ia/machine-learning-y-deep-learning-como-entender-las-claves-del-presente-y-futuro-de-la-inteligencia-artificial

[5] A.González. *¿Que es Machine Learning?* [En línia]. Cleverdata.io, 2014. Disponible a: http://cleverdata.io/que-es-machine-learning-big-data/

[6] *A large-scale data set of manually annotated audio events* [En línia]. research.google.com. Disponible a: https://research.google.com/audioset/index.html

[7] K.Ping. *Accuracy, Precision, Recall or F1?* [En línia]. Towardsdatascience.com, 2018. Disponible a: https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9

[8] A.Saeed. *Feature Extraction from sound and classification using Neural Networks* [En línia]. Aqibsaeed.github.io, 2016. Disponible a: http://aqibsaeed.github.io/2016-09-03-urban-sound-classification-part-1/

[9] Y.Sakashita. M.Aono*. Acoustic Scene Classification by Ensemble of Spectograms based on Adaptative Temporal Divisions* [En línia]. Dcase.community, 2018. Disponible a: http://dcase.community/documents/challenge2018/technical_reports/DCASE2018_Sakashita_15.pdf

[10] *Computing Precision and Recall for Multi-Class Classification Problems* [En línia]. Test-analytics101.rxnlp.com, 2014. Disponible a: http://text-analytics101.rxnlp.com/2014/10/computing-precision-and-recall-for.html

[11] *Tipos de Machine Learning* [En línia]. heroesdeldato.com,2018. Disponible a: http://heroesdeldato.com/tipos-de-machine-learning-clasificacion-vs-regresion/

[12] S.Becker. M.Ackermann. S.Lapuschkin. K. Müller. W.Samek. *Interpreting and Explaining Deep Neural Networks for Classification of Audio Signals* [En línia]. arxiv.org, 2018. Disponible a: https://arxiv.org/pdf/1807.03418v1.pdf

[13] *Decision Functions* [En linea]. towardsdatascience.com,2017. Disponible a: https://towardsdatascience.com/machine-learning/home

[14] T.Giannakopoulos. A. Pikrakis. *Spectral-Flux* [En línia]. sciencedirect.com, 2014. Disponible a: https://www.sciencedirect.com/topics/engineering/spectral-flux

[15] M.Rouse. *Aprendizaje profundo (Deep Learning*) [En línia]. searchdatacenter.com, 2017. Disponible a: https://searchdatacenter.techtarget.com/es/definicion/Aprendizaje-profundo-deep-learning

[16] L.JiaKai. *Mean teacher convolution System for DCASE 2018 task 4* [En línia]. Dcase.community, 2018. Disponible a: https://dcase.community/documents/challenge2018/technical_reports/DCASE2018_Lu_19.pdf