UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMA
ZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

RETI NEURALI CONVOLUZIONALI PER LA CLASSIFICAZIONE DI SUONI AMBIENTALI

Relatore: Prof. Nicola Bellotto

Laureando: Federico Guidolin

ANNO ACCADEMICO 2023 – 2024

Data di laurea 19/03/2024

Ringraziamenti

Per prima cosa vorrei ringraziare la mia famiglia, innanzitutto i miei genitori, per avermi dato la possibilità di seguire questo percorso ed avermi fornito l'aiuto necessario ad affrontare ogni momento di difficoltà, e mio fratello e mia sorella per aver rappresentato un punto di sfogo e distrazione.

Poi vorrei ringraziare tutte le persone che mi sono state vicine , prima fra tutti ringrazio Elisa, per essere sempre stata al mio fianco durante questo percorso ed aver illuminato le giornate più buie con il suo sorriso, e ringrazio Pietro, per avermi aiutato con la sua allegria e spensieratezza a vedere il lato positivo delle cose anche nelle situazioni più difficili.

Infine ringrazio il Prof. Nicola Bellotto per l'opportunità e per il supporto fornito nella realizzazione di questo progetto.

Padova, Marzo 2024

Federico Guidolin

Sommario

La classificazione di suoni ambientali è una componente fondamentale in numerose situazioni quotidiane; grazie ai recenti sviluppi delle tecnologie di deep learning è stato possibile sviluppare sistemi di classificazione sempre più robusti ed efficienti. L'obiettivo che questa tesi si propone è quello di progettare ed implementare un sistema di classificazione di suoni ambientali basato sulle reti neurali convoluzionali, riconosciute come uno dei metodi più affidabili in ambito di sistemi di classificazione. L'utilizzo di questa tipologia di reti, infatti, permette l'estrazione e l'analisi di caratteristiche gerarchiche dai segnali audio particolarmente efficienti per il riconoscimento di suoni. Il metodo di sviluppo include l'utilizzo di un dataset di suoni ambientali e l'implementazione di una rete neurale convoluzionale realizzata per catturare al meglio le caratteristiche estratte dai vari segnali audio. Il modello realizzato è quindi allenato e ottimizzato attraverso un processo di sperimentazione, ponendo particolare attenzione alla regolazione degli iperparametri della rete. Il modello è successivamente valutato attraverso una fase di test della rete, mostrando le potenzialità delle reti convoluzionali applicate in ambiti di classificazione di suoni. Infine, dopo un'analisi dei risultati ottenuti, viene effettuato un resoconto del lavoro svolto evidenziando le limitazioni del progetto e le possibili direzioni dello sviluppo futuro.

Indice

1 Introduzione **1**

1.1 Motivazioni 1

1.2 Obiettivi 2

2 Stato dell'Arte **3**

2.1 Concetti Fondamentali 3

2.1.1 Convolutional Neural Network 4

2.2 Revisione della letteratura 5

2.2.1 Preelaborazione del segnale 6

2.2.2 Estrazione delle caratteristiche 6

2.2.3 Metodi di classificazione di suoni 9

2.2.4 Principali applicazioni di reti convoluzionali 10

3 Strumenti e Metodologie **13**

3.1 Architettura della rete 13

3.1.1 Motivazioni 13

3.1.2 Caratteristiche strutturali 13

3.2 Ambiente di sviluppo 14

3.3 Dataset di suoni ESC-50 16

4 Implementazione **17**

4.1 Preelaborazione del segnale 17

4.1.1 Estrazione delle caratteristiche 17

4.1.2 Data augmentation 19

4.2 Calibrazione degli iperparametri 19

5 Risultati sperimentali **23**

5.1 Sperimentazioni 23

5.1.1 Struttura di partenza 23

5.1.2 Secondo modello 25
5.1.3 Terzo modello 26
5.2 Risultati finali 28
5.2.1 Cross-Validation 28
5.2.2 Matrice di confusione 29

6 Conclusioni **33**
6.1 Riassunto del lavoro svolto 33
6.2 Criticità e limitazioni 33
6.3 Considerazioni per lo sviluppo futuro 34

Bibliografia **37**

Elenco delle figure

2.1 Semplice modello matematico per un neurone [4] 3
2.2 Esempio grafico del processo di convoluzione [5] 4
2.3 Esempio grafico di un'operazione di max pooling con filtri di dimensione 2x2 [5] 5
2.4 Modello tipico di un sistema di riconoscimento di suoni [1] 5

3.1 Struttura tipica di una CNN 15

4.1 Passaggi di trasformazione dalla forma d'onda allo spettrogramma in scala mel. 18

5.1 Grafici della perdita e dell'accuracy di training e di validation del primo modello 24
5.2 Grafici della perdita e dell'accuracy di training e di validation del secondo modello 26
5.3 Grafici della perdita e dell'accuracy di training e di validation del terzo modello 27
5.4 Risultati ottenuti dai modelli proposti. 28
5.5 Risultati ottenuti da ciascuna fase della Cross-validation. 29
5.6 Matrice di confusione della Cross-Validation 30

Capitolo 1

Introduzione

1.1 Motivazioni

Il suono è una vibrazione che si propaga come onda acustica attraverso un corpo elastico e viene percepito dagli organi uditivi. Il riconoscimento dei suoni che ci circondano costituisce una componente fondamentale della vita quotidiana di ogni persona e gioca un ruolo essenziale nelle attività umane più basilari, come comunicazione, sicurezza e intrattenimento. Questo ha portato ad un aumento della domanda di sistemi di riconoscimento di suoni automatici robusti ed efficienti. Nei primi anni di ricerca tali sistemi sono stati suddivisi nella letteratura in: sistemi di riconoscimento vocale (speech recognition systems) e sistemi di riconoscimento non vocale (non-speech recognition systems). Mentre la ricerca in ambito di riconoscimento vocale ha ricevuto attenzione consistente nel corso di diversi decenni passati, lo sviluppo di sistemi di riconoscimento non vocale si è intensificato solo negli ultimi due decenni, grazie alle numerose applicazioni che nel corso del tempo sono aumentate e si sono gradualmente diversificate. Tra le applicazioni più importanti possiamo trovare esempi come audio sorveglianza, riconoscimento di eventi sonori e riconoscimento di suoni ambientali [1].

Tuttavia, sebbene le tecnologie di riconoscimento automatizzato siano in continuo progresso, la letteratura evidenzia anche alcune criticità riguardo lo sviluppo di sistemi di riconoscimento di suoni [2]:

- la dimensione dei segnali audio utilizzati durante lo sviluppo, infatti per rappresentare un segnale audio di breve durata sono necessarie diverse migliaia di valori;

- la disponibilità insufficiente di dati sia dal punto di vista qualitativo, occorrono infatti conoscenze avanzate per ottenere dei dati coerenti utilizzabili nello sviluppo, che da quello quantitativo, vista la mancanza di quantità sufficienti di dati affidabili e ben organizzati.

1.2 Obiettivi

Possiamo quindi definire quali sono gli obiettivi che questo progetto di tesi vuole raggiungere:

- Progettare un sistema di classificazione ponendo particolare attenzione alle varie scelte di progettazione.
- Implementare un sistema di classificazione di suoni ambientali basato su Convolutional Neural Network che riesca a classificare suoni appartenenti a categorie predefinite in modo consistente.
- Analizzare il processo di implementazione del sistema riportandone dati e risultati ed evidenziando errori di percorso e conseguenti soluzioni adottate.

Per riassumere, questa tesi si pone come obiettivo la progettazione e l'implementazione di un sistema che riesca ad effettuare la classificazione delle categorie di suoni utilizzate.

Nel Capitolo 2 verranno introdotti i concetti fondamentali utili a comprendere le metodologie di implementazione utilizzate in questo lavoro, insieme ad un'analisi delle attuali tecniche di riconoscimento più efficienti.

Nel Capitolo 3 verranno presentate le scelte legate all'architettura della rete, alle librerie software utilizzate per l'implementazione e al dataset utilizzato.

Nel Capitolo 4 verrà descritto il design del sistema di riconoscimento evidenziando le principali scelte implementative effettuate.

Nel Capitolo 5 sarà proposto un resoconto sulla fase sperimentale del progetto mostrando i risultati ottenuti.

Infine nel Capitolo 6 verrà proposto un resoconto del lavoro svolto mostrando obiettivi raggiunti, criticità e considerazioni per il lavoro futuro.

Capitolo 2

Stato dell'Arte

2.1 Concetti Fondamentali

Artificial Neural Network (ANN) è un'espressione utilizzata per identificare un modello computazionale che simuli l'attività del cervello umano dal punto di vista dell'elaborazione di informazioni [3]. La struttura di un'ANN è caratterizzata da unità, o nodi, connesse fra di loro attraverso una serie di collegamenti, ciascun collegamento è associato ad un peso che determina intensità e segno della connessione. La somma pesata degli input di un'unità viene poi processata utilizzando una funzione di attivazione che produce l'output, le funzioni di attivazione tipicamente usate sono funzioni di soglia e funzioni logistiche [4].

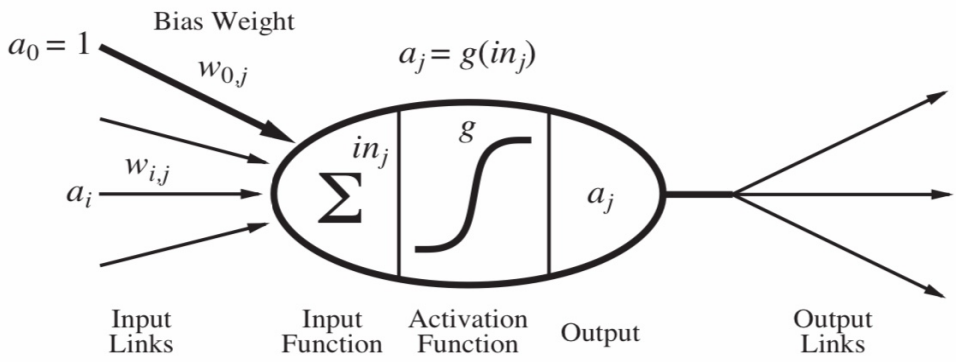

Figura 2.1: Semplice modello matematico per un neurone [4]

Il Deep Learning è una branca del Machine Learning e dell'Intelligenza Artificiale che si basa sull'utilizzo delle ANN per il riconoscimento automatico di pattern nei dati attraverso l'elaborazione e la rappresentazione di dati ad alto livello. In ambito di Deep Learning le ANN utilizzate prendono il nome di Deep Neural Networks (DNN) e sono caratterizzate da una struttura composta da una serie di strati di nodi, detti layer, suddivisi in: input layer, che accettano

segnali e dati dall'esterno, output layer, che realizzano l'output del sistema, e hidden layers, utilizzati per estrarre caratteristiche di livello progressivamente più alto a partire dai dati.

2.1.1 Convolutional Neural Network

Convolutional Neural Network (CNN) è una tipologia di DNN principalmente progettata per analizzare dati di tipo bidimensionale. La parte fondamentale di una CNN è il convolutional layer, che contiene un insieme di filtri, detti kernel, di dimensione variabile ma inferiore a quella dell'input che scorrono sulla superficie dell'input stesso creando una matrice delle caratteristiche (feature map). Ciascuna casella della matrice è ottenuta come risultato del prodotto scalare fra il filtro e una parte della matrice di input che abbia la stessa dimensione del filtro, ottenendo così una matrice la cui dimensione è uguale al numero delle possibili posizioni del filtro sull'input, come si può vedere dalla Figura 2.2. L'output di un convolutional layer è ottenuto impilando le feature map generate dai vari filtri applicati [5].

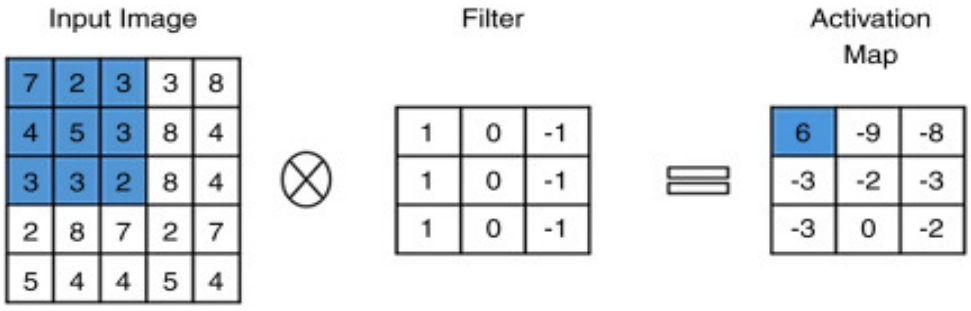

Figura 2.2: Esempio grafico del processo di convoluzione [5]

Ogni convolutional layer è inoltre caratterizzato da una funzione di attivazione che viene applicata all'output di ciascuna unità, possono essere semplici funzioni lineari oppure funzioni più complesse che introducono una componente di non-linearità nella rete, utile ad aumentare la robustezza e la stabilità. La funzione di attivazione più comune è la funzione ReLU (Rectified Linear Unit) che imposta a zero tutti i valori negativi, lasciando invariati quelli positivi.

Una limitazione della feature map è che la matrice memorizza la posizione precisa delle caratteristiche dell'input, per tanto una modifica nella posizione di tali caratteristiche comporterebbe una modifica anche nella feature map, rendendo meno efficace la rete. Una soluzione a questa problematica è il pooling layer, che ridimensiona le feature map generando un segnale con risoluzione inferiore che mantiene le informazioni strutturali più importanti, eliminando alcuni dettagli superflui e aumentando il livello della robustezza della rete. L'operazione di pooling opera su ogni feature map, che viene divisa in settori non sovrapposti di dimensioni uguali, a ciascun settore viene poi applicata l'operazione di pooling specificata; le più comuni sono: max pooling, l'output dell'operazione è il valore massimo del settore considerato come si può

vedere nell'esempio grafico fornito nella Figura 2.3, e average pooling, l'output dell'operazione è la media di tutti i valori presenti nel settore.

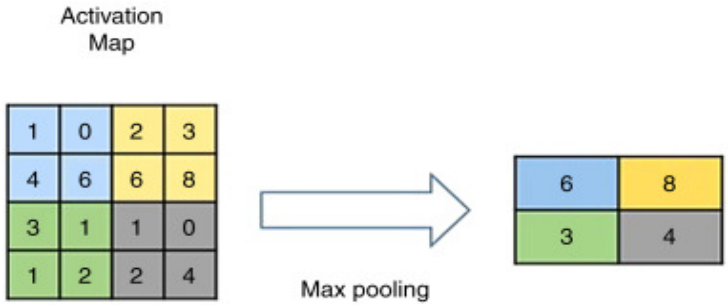

Figura 2.3: Esempio grafico di un'operazione di max pooling con filtri di dimensione 2x2 [5]

Infine la rete si conclude con uno o più dense layer, detti anche fully-connected layer, in cui, a differenza dei convolutional layer, ciascun nodo è connesso e riceve un input da ogni nodo del layer precedente; collegano i convolutional layers al layer di output e servono per la classificazione dei dati ricevuti in input.

2.2 Revisione della letteratura

I problemi di pattern recognition in ambito di riconoscimento di suoni sono caratterizzati da una struttura generale, rappresentata in Figura 2.4, che rimane la stessa nella maggior parte delle applicazioni, è composta da tre fasi principali: preelaborazione del segnale, estrazione delle caratteristiche e classificazione.

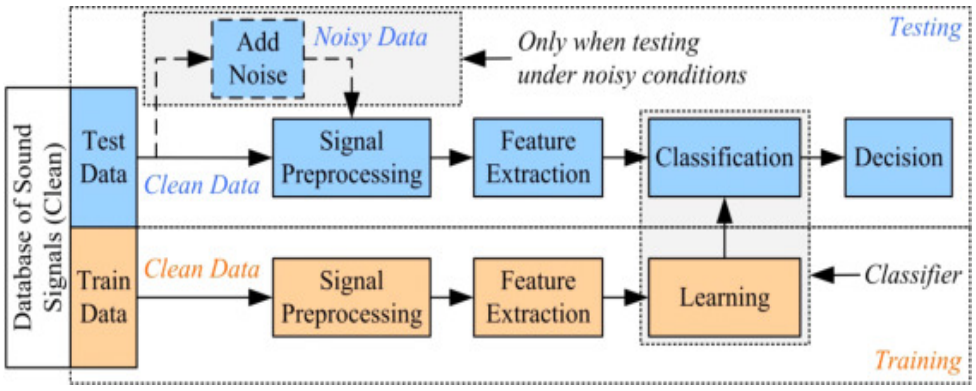

Figura 2.4: Modello tipico di un sistema di riconoscimento di suoni [1]

2.2.1 Preelaborazione del segnale

Con preelaborazione del segnale si intende l'insieme di operazioni che devono essere eseguite sul segnale audio e che permettono di adattare e semplificare il segnale per eseguire le successive elaborazioni. Le operazioni di preelaborazione più comuni sono:

- segmentazione del segnale, cioè una suddivisione del segnale audio in segmenti di dimensione minore che possono essere parzialmente sovrapposti;
- applicazione di una funzione finestra per attenuare il segnale, la funzione più comune è la finestra di Hamming;
- utilizzare differenti frequenze di campionamento che, a seconda del sistema utilizzato, possono aumentare la precisione della rete o semplificarla.

Esistono inoltre sistemi che provano a rendere opzionale la fase di preelaborazione [6] che, in alcuni casi, rischia di rendere il sistema poco flessibile; tuttavia nella maggioranza delle applicazioni la preelaborazione è una fase fondamentale dello sviluppo del sistema.

2.2.2 Estrazione delle caratteristiche

L'estrazione delle caratteristiche è una fase di estrema importanza per lo sviluppo di un sistema di riconoscimento poichè la performance del sistema dipende dalla qualità delle caratteristiche che vengono estratte, che a loro volta determinano quali informazioni e proprietà sono disponibili durante la fase di classificazione. Dovrebbero catturare proprietà invarianti nella stessa categoria di suoni e variabili tra categorie differenti. I principali attributi che distinguono un suono sono [7]:

- durata, tempo che trascorre fra l'inizio e la fine del segnale;
- intensità, rappresenta la percezione umana di quanto un suono è forte o debole;
- altezza, definito come l'attributo intensivo della sensazione uditiva in base al quale un suono può essere ordinato su una scala che va dal debole al forte;
- timbro, è definito come l'attributo della sensazione uditiva che rende l'ascoltatore in grado di giudicare che due suoni non identici, presentati in modo simile e con lo stesso volume e altezza, sono dissimili.

L'estrazione delle caratteristiche audio tenta quindi di catturare gli attributi sopra citati più adatti al dominio applicativo [8].

Caratteristiche temporali

Le caratteristiche temporali vengono estratte direttamente dal segnale audio grezzo senza alcuna trasformazione e pertanto hanno una bassa complessità. Vengono suddivise in tre gruppi in base a ciò che viene descritto dalla caratteristica:

- zero-crossing, permette una stima della frequenza dominante e del baricentro spettrale;
- ampiezza, rappresenta l'inviluppo temporale del segnale audio, semplice da calcolare ma possiede espressività limitata;
- potenza, la potenza è il quadrato medio di un segnale, per essere impiegata come caratteristica viene considerata la sua radice.

Caratteristiche della frequenza

Per ottenere caratteristiche appartenenti al dominio della frequenza è necessario applicare delle trasformazioni al segnale audio, i metodi più popolari sono la trasformata di Fourier e l'auto-correlazione; altri metodi utilizzati meno frequentemente includono la trasformata del coseno e la trasformata wavelet.

- autoregressione, consiste in una previsione del valore di ogni campione da parte di un predittore lineare, basandosi su una combinazione lineare dei precedenti valori;
- decomposizione adattiva tempo-frequenza, fornisce una risoluzione in frequenza che varia con la risoluzione temporale, questa categoria è basata sulla trasformata wavelet e su trasformate correlate;
- short-time Fourier transform (STFT), genera valori reali e complessi che rappresentano rispettivamente le componenti di frequenza e fase del segnale.

Caratteristiche della percezione

Alcune caratteristiche del suono possiedono un significato semantico nel contesto della percezione umana dell'udito, vengono suddivise in base alla qualità uditiva che descrivono:

- luminosità, caratterizza la distribuzione spettrale delle frequenze e descrive se un segnale è dominato da frequenze basse o alte;
- tonalità, proprietà del suono che permette di distinguere segnali di tipo rumoroso, che hanno spettro continuo, da segnali tonali, che hanno spettro lineare;
- armonia, proprietà che permette di distinguere segnali periodici da segnali non periodici;

Caratteristiche cepstrali

Le caratteristiche cepstrali sono rappresentazioni livellate in frequenza dello spettro logaritmico di magnitudo e catturano caratteristiche di timbro e tonalità. Consentono l'applicazione della metrica euclidea come misura della distanza grazie alla loro base ortogonale che facilita i confronti di somiglianza. Le caratteristiche cepstrali si suddividono in tre gruppi:

- banca dei filtri percettivi, le caratteristiche di questo gruppo si basano sulla trasformata di Fourier del logaritmo della magnitudine dello spettro del segnale originale;
- modello uditivo avanzato, le caratteristiche di questo gruppo si basano su un modello uditivo progettato per rappresentare fedelmente i processi fisiologici dell'udito umano;
- autoregressione, le caratteristiche di questo gruppo sono rappresentazioni cepstrali che si basano sull'analisi predittiva lineare.

Altre caratteristiche

Esistono poi altre categorie di caratteristiche che vengono utilizzate come supporto in particolari applicazioni, come il riconoscimento di suoni di tipo musicale. Tra queste troviamo:

- caratteristiche della frequenza di modulazione, identificano informazioni riguardo la modulazione in bassa frequenza di segnali audio. Suoni modulati provocano sensazioni uditive diverse nel sistema uditivo umano: basse frequenze producono la sensazione uditiva della forza di fluttuazione mentre alte frequenze producono la sensazione uditiva di ruvidità;
- caratteristiche di autodominio, le caratteristiche di questo gruppo rappresentano informazioni a lungo termine contenute in segmenti sonori che durano diversi secondi, questo porta a grandi quantità di dati con bassa espressività che rischiano di non essere adatti per ulteriore elaborazione.

Caratteristiche per CNN

Le reti convoluzionali sono state principalmente progettate per ricevere come input dati di tipo bidimensionale. Un segnale audio, invece, è un segnale monodimensionale formato da un insieme di valori, dell'ordine delle migliaia, che rappresentano l'ampiezza dell'onda sonora, ovvero l'intensità del suono. Si possono rappresentare graficamente come una forma d'onda, ovvero una funzione del tempo che descrive l'intensità del suono in ogni istante. Per poter utilizzare i segnali audio in una rete convoluzionale è necessario effettuare una trasformazione che traduca il segnale monodimensionale in un'immagine bidimensionale. Le tecniche principali per effettuare questa trasformazione sono due:

- spettrogramma, ovvero la rappresentazione dell'intensità del suono in funzione di frequenza e tempo, per ottenere uno spettrogramma è necessario suddividere il segnale in sottointervalli di dimensione costante e applicare a ciascuno di essi la trasformata di Fourier, assemblando i vari risultati si ottiene lo spettrogramma;

- mel-frequency cepstral coefficients (MFCC), i coefficienti che compongono il mel-frequency cepstrum, ovvero la rappresentazione dello spettro di potenza di un suono basato su una trasformata lineare del coseno di uno spettro di potenza logaritmico su una scala melodica non lineare di frequenza [9].

Sebbene le reti convoluzionali siano progettate per ricevere come input dati bidimensionali esistono delle applicazioni in cui si utilizza una rete monodimensionale in grado di operare sui file audio grezzi in modo da evitare l'intero processo di estrazione delle caratteristiche [10].

2.2.3 Metodi di classificazione di suoni

Dopo la fase di estrazione delle caratteristiche avviene la classificazione dei segnali audio in categorie differenti. I metodi di classificazione più utilizzati si possono suddividere in due gruppi: metodi di machine learning e metodi di deep learning [11].

Metodi di machine learning

- Support Vector Machine (SVM), si tratta di un insieme di metodi che differiscono per il kernel utilizzato e si suddividono in: binario, lineare, polinomiale e gaussiano. L'obiettivo delle SVM è quello di trovare un iperpiano che separi al meglio i dati di diverse classi rappresentati come punti in uno spazio multidimensionale. E' il classificatore basato su machine learning più popolare ed è stato ampiamente utilizzato in diversi studi in ambito di riconoscimento di suoni ambientali.

- K-Nearest Neighbour (k-NN), un dato viene classificato in base alla classe di maggioranza o al valore medio dei k dati più vicini nello spazio delle caratteritiche, il numero k dipende dal tipo di dati considerati, generalmente un valore maggiore permette una maggiore robustezza al rumore ma rende i confini tra le classi meno distinti.

- Hidden Markov Model (HMM), modello basato su un processo di Markov in cui gli stati sono nascosti, i risultati osservabili sono collegati agli stati nascosti e le transizioni tra gli stati sono governate dalle probabilità.

- Gaussian Mixture Model (GMM), un modello probabilistico che presuppone che tutti i dati siano generati da una miscela di un numero finito di distribuzioni gaussiane con

parametri sconosciuti, ciascuna distribuzione gaussiana cattura un sottoinsieme di dati e il modello combina queste distribuzioni per descrivere la distribuzione generale dei dati.

Metodi di deep learning

- Recurrent Neural Network (RNN), si tratta di un tipo di ANN caratterizzata da flusso bi-direzionale delle informazioni tra i vari layer. Questo permette alle RNN di creare cicli nella rete che mantengono uno stato di memoria degli input precedenti. Le RNN sono particolarmente soggette al problema del gradiente evanescente che avviene quando i gradienti usati per aggiornare la rete diventano eccessivamente piccoli o ”svaniscono”.

- Long-Short Term Memory Network (LSTM), si tratta di una tipologia particolare di RNN che tenta di risolvere il problema del gradiente evanescente utilizzando celle di memoria specializzate e meccanismi di gate per migliorare le prestazioni sull'immagazzinamento di informazioni sul lungo periodo, questo le rende particolarmente potenti in ambiti in cui il mantenimento della memoria è importante.

2.2.4 Principali applicazioni di reti convoluzionali

Con lo sviluppo delle metodologie basate sulle reti convoluzionali e il loro utilizzo efficiente in diversi ambiti quali la computer vision, il riconoscimento vocale, la modellazione del linguaggio, e altre aree correlate, è stato dimostrato che le architetture basate su CNN superano i metodi più convenzionali in diversi ambiti. Pertanto, negli ultimi anni, le reti convoluzionali sono state applicate al riconoscimento di suoni ambientali [12]. La maggior parte di queste applicazioni si basa sull'utilizzo dei tre dataset più importanti in ambito di classificazione di suoni ambientali, ovvero: ESC-50, descritto nel Paragrafo 3.3, ESC-10, un sottoinsieme del precedente, e UrbanSound8k [13], una raccolta di 8732 registrazioni audio suddivise in 10 classi.

La prima applicazione di CNN a sistemi di classificazione di suoni ambientali si ha in [14], ottenendo delle accuracy di 80.5% ESC-10, 64.9% ESC-50, 73.7% UrbanSound8k, che hanno superato le prestazioni dei metodi tradizionali. Successivamente sono state realizzate numerose implementazioni differenti di sistemi basati su CNN che propongono modifiche nella struttura della rete, nell'estrazione delle caratteristiche e nella preelaborazione dei dati.

In primo luogo è stata dimostrata l'importanza della fase di data augmentation [15], in particolare in mancanza di quantità sufficienti di dati etichettati, migliorando le prestazioni delle reti proposte. L'aggiunta di questa fase ha permesso di raggiungere un'accuracy del 79% sul dataset UrbanSound8k, e di suggerire come delle operazioni di data augmentation specifiche per ogni classe, possano ulteriormente migliorare le prestazioni della rete.

Successivamente sono state proposte alcune implementazioni caratterizzate dall'utilizzo di CNN monodimensionale, in cui non vengono estratte caratteristiche a partire dall'input ma vengono analizzate le forme d'onda. Il risultato migliore è stato ottenuto in [16], dove l'accuracy ottenuta sul dataset UrbanSound8k è stata di 89%.

Alcune implementazioni utilizzano una combinazione di più reti con caratteristiche diverse, un primo esempio si ha in [17] in cui sono state utilizzare due reti convoluzionali, di cui la prima utilizza spettrogrammi log-mel come caratteristica, ottenendo come risultati: 81.1% su ESC-50, 91.4% su ESC-10, 90.2% su UrbanSound8k; mentre la seconda caratterizzata da una struttura end-to-end, che utilizza direttamente le forme d'onda delle registrazioni audio, ottenendo come risultati: 65.8% su ESC-50, 85.2% su ESC-10, 87.7% su UrbanSound8k. La struttura finale è stata ottenuta attraverso la fusione delle due reti proposte, basata sulla teoria Dempster-Shafer [18], che permette di utilizzare i vantaggi di entrambe le rappresentazioni utilizzate dalle due reti migliorando le prestazioni complessive; i risultati ottenuti sono stati: 83.1% su ESC-50, 92.6% su ESC-10, 92.2% su UrbanSound8k. Un altro esempio di implementazione basata sullo stesso principio si ha in [19], in questo caso le due reti proposte presentano la stessa struttura a livello dei layer ma operano su caratteristiche diverse: la prima utilizza un insieme costituito da spettrogramma log-mel, cromagramma, contrasto spettrale e caratteristiche del centroide tonale; mentre la seconda utilizza MFCC al posto dello spettrogramma mantenendo invariate le caratteristiche restanti. I risultati ottenuti dalle due reti sono stati rispettivamente di 95.2% e 95.3% su UrbanSound8k mentre la rete finale, composta dalla fusione delle due reti, utilizzando la teoria Dempster-Shafer, è stato di 97.2% su UrbaSound8k.

Un ulteriore metodologia per migliorare le prestazioni della rete attraverso la data augmentation è stata proposta in [20]. L'operazione proposta, infatti, consiste nell'aumentare in modo significativo il numero di file a disposizione per migliorare le prestazioni dei modelli considerati. Questo approccio è stato dapprima testato su due reti convoluzionali create da zero, per dimostrarne l'efficacia, ed è stato poi applicato ad alcuni modelli pre-addestrati attraverso tecniche di transfer learning. I risultati ottenuti dimostrano come l'approccio proposto migliori considerevolmente le prestazioni dei modelli considerati, in particolare i risultati ottenuti dalle due reti sono stati: 93.5% su ESC-10, 96.1% su ESC-50 e 95.05% su UrbanSound8k per la prima rete, 95.5% su ESC-10, 95.13% su ESC-50 e 87.08% su UrbanSound8k per la seconda rete; i risultati migliori ottenuti dai modelli pre-addestrati sono stati: 99.04% su ESC-10, 99.49% su UrbanSound8k ottenuti dalla rete ResNet-152 [21] e 97.57% su ESC-50 ottenuto dalla rete DenseNet-161 [22]. Questi ultimi risultati rappresentano lo Stato dell'Arte attuale in ambito di sistemi di riconoscimento di suoni ambientali [20].

Nella Tabella 2.1 vengono elencati i metodi di sviluppo di sistemi di riconoscimento basati su CNN presentati negli studi citati, ordinati in base all'accuracy, per mettere a confronto le

varie metodologie.

Articolo	Metodo	ESC-10	ESC-50	US8k
[14]	2-D CNN	80.5%	64.9%	73.7%
[15]	2-D CNN	-	-	79%
[16]	1-D CNN	-	-	89%
[17]	DS-CNN	92.6%	83.1%	92.2%
[19]	DS-CNN	-	-	97.2%
[20]	CNN Transfer Learning	99.04%	97.57%	99.49%

Tabella 2.1: Risultati principali ottenuti negli articoli presentati.

Capitolo 3

Strumenti e Metodologie

3.1 Architettura della rete

3.1.1 Motivazioni

La struttura principale del modello proposto è basata su una rete neurale convoluzionale, le cui caratteristiche sono già state introdotte nel Capitolo 2. Sebbene le reti convoluzionali siano state progettate principalmente per il riconoscimento di immagini e per la computer vision, già dalle prime applicazioni in ambito di riconoscimento di suoni ambientali sono stati riscontrati risultati promettenti [14], che sono stati confermati e migliorati da diversi studi successivi [12], [15], [23]. I vantaggi principali che una CNN presenta sono [24]:

- connessioni locali, i nodi dei convolutional layers sono connessi solo ad una parte dei nodi del layer precedente, aumentando l'efficienza della rete riducendo il numero di parametri;
- condivisione dei pesi, i filtri sono applicati allo stesso modo su ogni parte dell'immagine, riducendo ulteriormente il numero di parametri e permettendo alla rete di riconoscere pattern invarianti rispetto alla posizione;
- diminuzione della dimensionalità, i pooling layer permettono di ottenere una versione sommaria dell'input che mantiene però tutte le caratteristiche fondamentali eliminando invece quelle superflue.

3.1.2 Caratteristiche strutturali

Come introdotto nel Capitolo 2, l'architettura di una rete convoluzionale è caratterizzata dalla presenza di diversi layer che appartengono a 3 categorie principali: convolutional layer, pooling layer e dense layer. La progettazione di una rete neurale è un procedimento sperimentale basato

sull'euristica; il numero di layer appartenenti a ciascuna categoria e le caratteristiche relative, come la funzione di attivazione o la dimensione dei filtri, sono componenti che devono essere determinate sperimentalmente valutando le prestazioni della rete rispetto ai vari cambiamenti apportati.

La struttura di una rete convoluzionale è suddivisa in due sezioni principali, come si può osservare nella Figura 3.1. La prima sezione si occupa dell'estrazione delle caratteristiche dalle immagini ed è composta da un input layer, che ha il compito di ricevere i dati di input e adattarli alla struttura dei layer successivi, e da una serie di convolutional layer che possono essere alternati a pooling layer. Il risultato della fase di convoluzione e pooling è un vettore di feature map costruito su tre dimensioni, per poter essere ulteriormente elaborato dai layer successivi viene impiegato un flatten layer che trasforma i dati ricevuti in un vettore monodimensionale adatto ad essere elaborato dai dense layer. Infine la seconda sezione è costituita da una serie di fully-connected layer che eseguono la classificazione delle caratteristiche estratte, in particolare l'ultimo dense layer svolge il ruolo di layer di output, la cui dimensione dipende dal numero di classi di dati da classificare, e serve a compiere la predizione finale. Il layer di output riceve come input un vettore monodimensionale di valori reali chiamati logit, o attivazioni, che possono assumere un valore compreso fra $(-\infty, +\infty)$. Tuttavia questi valori non possono essere interpretati direttamente per la classificazione dell'input, per tanto è necessario applicare la funzione di attivazione softmax. Essa scala i valori del vettore di logit, ottenendo come risultato un vettore di valori compresi fra 0 e 1 che corrispondono alle probabilità che l'input di partenza appartenga ad una determinata classe.

$$\text{softmax}(Z_i) = \frac{\exp(Z_i)}{\sum_n \exp(Z_j)} = \frac{e_i^Z}{e_1^Z + e_2^Z + ... + e_n^Z}$$

Nella formula qui mostrata dove Z_i rappresenta l'i-esimo valore del vettore di logit, la trasformazione applicata dalla funzione consiste nell'applicare la funzione esponenziale al valore Z_i per poi normalizzare tale valore, ottenendo così come risultato un numero compreso fra 0 e 1.

3.2 Ambiente di sviluppo

Come ambiente di sviluppo è stato utilizzato Google Colab, una piattaforma online che permette di scrivere ed eseguire codice in python attraverso l'utilizzo di un browser. Si tratta di un ambiente particolarmente adatto allo sviluppo in ambiti di machine learning e deep learning, selezionato per avere a disposizione un ambiente di facile utilizzo che permetta l'accesso alle versioni più recenti di python e delle librerie utilizzate.

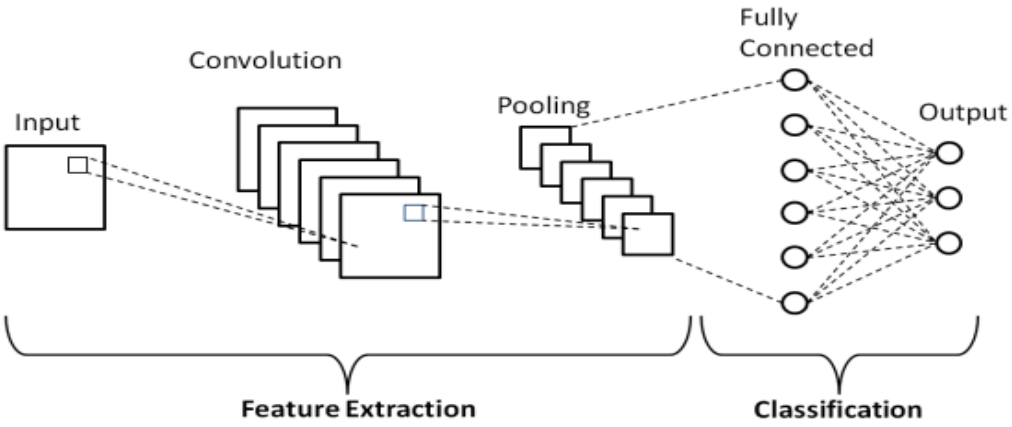

Figura 3.1: Struttura tipica di una CNN

Tensorflow

Tensorflow è un'interfaccia per esprimere algoritmi di machine learning e un'implementazione per eseguire tali algoritmi [25]. Si tratta di una delle piattaforme più utilizzate in ambiti di sviluppo di algoritmi di machine e deep learning, viste le numerose funzioni che mette a disposizione per la gestione e l'elaborazione di dati e le diverse librerie che permettono di operare sui dati su più livelli di astrazione. Inoltre una delle più importanti caratteristiche dell'interfaccia è la documentazione che, oltre alle informazioni principali riguardanti funzioni e strutture dati, contiene una grande quantità di tutorial e modelli pre-addestrati, facilitando l'approccio allo sviluppo di un sistema di riconoscimento.

In particolare è stata utilizzata la libreria Keras [26], una libreria di alto livello che trova largo impiego in applicazioni che riguardano le reti neurali in ambito di deep learning, ed è particolarmente adatta per applicazioni con una quantità limitata di dati. La sua funzione principale è quella di fornire delle funzioni che permettono una creazione rapida ed efficiente di una rete neurale facilitando la gestione della struttura e degli iperparametri.

Librosa

Librosa [27] è una libreria di python per l'analisi e l'elaborazione di audio e musica, si tratta di una delle librerie più usate in ambiti di classificazione di suoni in quanto fornisce numerose funzioni per l'estrazione delle caratteristiche di un segnale e per l'analisi nei domini del tempo e della frequenza. Inoltre fornisce funzioni utili alla manipolazione della struttura del segnale e alla gestione di caricamento e visualizzazione dei segnali.

3.3 Dataset di suoni ESC-50

Una componente fondamentale da tenere in considerazione durante la fase di progettazione di un sistema di riconoscimento di suoni ambientali è il dataset, ovvero l'insieme di dati che si intende utilizzare per le fasi di training e test della rete. Per lo sviluppo di questo lavoro è stato utilizzato come dataset di riferimento il dataset ESC-50 [28], che consiste in 2000 registrazioni etichettate di suoni ambientali, suddivise equamente in 50 classi, ciascuna registrazione ha una durata di 5 secondi e un formato uniformato (campionamento a 44.1 kHz, canale singolo, compresso con Ogg Vorbis a 192 kbits/s). Le classi possono essere suddivise in 5 categorie principali, contenenti 10 classi per categoria:

- suoni di animali;
- paesaggi sonori naturali e suoni dell'acqua;
- suoni umani non vocali;
- suoni domestici;
- suoni urbani.

Questo dataset è stato favorito rispetto ad altri dataset comunemente utilizzati in ambito di classificazione di suoni ambientali, come ESC-10 e UrbanSound8K, vista l'ampia varietà di classi di suoni che mette a disposizione, che permette di addestrare la rete a riconoscere suoni di tipologie differenti per validare possibili applicazioni in diversi ambiti. Tuttavia una delle carenze principali di questo dataset è il limitato numero di file per ciascuna classe, questo è dovuto al costo considerevole dell'annotazione manuale e dalla decisione di mantenere un equilibrio stretto tra le varie classi nonostante la carenza di file audio appartenenti ad alcune classi particolari.

Capitolo 4

Implementazione

4.1 Preelaborazione del segnale

La preelaborazione del segnale è la prima fase che deve essere affrontata durante lo sviluppo di un sistema di riconoscimento di suoni, si tratta di una fase fondamentale in quanto definisce la struttura dei dati che verranno poi elaborati dalla rete.

4.1.1 Estrazione delle caratteristiche

La componente principale della preelaborazione del segnale è l'estrazione delle caratteristiche. Esistono numerose tipologie di caratteristiche che possono essere prese in considerazione in un sistema di riconoscimento, per lo sviluppo di questo progetto è stato scelto lo spettrogramma in scala log-mel (logaritmica-melodica). Questa particolare tipologia di spettrogramma è stata ampiamente utilizzata per lo sviluppo di sistemi di riconoscimento di suoni e viene considerata una delle caratteristiche più robuste per questo tipo di applicazioni [12], ottenendo risultati migliori e più consistenti rispetto ad altri metodi comunemente utilizzati in questi ambiti [29].

Riprendendo quanto introdotto nel Capitolo 2, i segnali audio sono costituiti da un vettore monodimensionale di valori che rappresentano l'intensità del suono in funzione del tempo pertanto per poter essere elaborati da una CNN si estrae lo spettrogramma dal segnale che costituisce una rappresentazione bidimensionale. Lo spettrogramma viene ottenuto applicando la trasformata veloce di Fourier ad una segmentazione del segnale originale e le risultanti vengono poi assemblate per ottenere un'immagine bidimensionale; si ottiene così una funzione che esprime sia frequenza che intensità in funzione del tempo. Lo spettrogramma si configura come una mappa di calore in cui sull'asse delle ascisse è situato il tempo, sull'asse delle ordinate la frequenza e ad ogni punto viene associato un colore che rappresenta l'intensità del suono in quell'istante a quella determinata frequenza. Lo spettrogramma può inoltre essere rappresentato

utilizzando diverse scale di riferimento; una particolare rappresentazione consiste nell'utilizzare la scala mel, ovvero una trasformata logaritmica della frequenza del segnale. Questa scala è costruita in modo che distanze uguali sulla scala siano percepite dall'udito umano come differenze uguali dell'intensità del suono, in questo modo vengono evidenziate nello spettrogramma le componenti percepite maggiormente dall'udito che possono essere identificate più facilmente dalla rete. Lo spettrogramma log-mel è stato estratto utilizzando le funzioni fornite dalla libreria Librosa, nella figura 4.1 si possono vedere i passaggi che portano all'ottenimento di uno spettrogramma log-mel a partire dalla forma d'onda.

Le impostazioni utilizzate per l'estrazione sono le seguenti:

- tasso di campionamento = 22050;
- lunghezza della finestra di trasformata di Fourier = 2048;
- lunghezza della finestra di salto = 1024;
- numero di bande mel = 128.

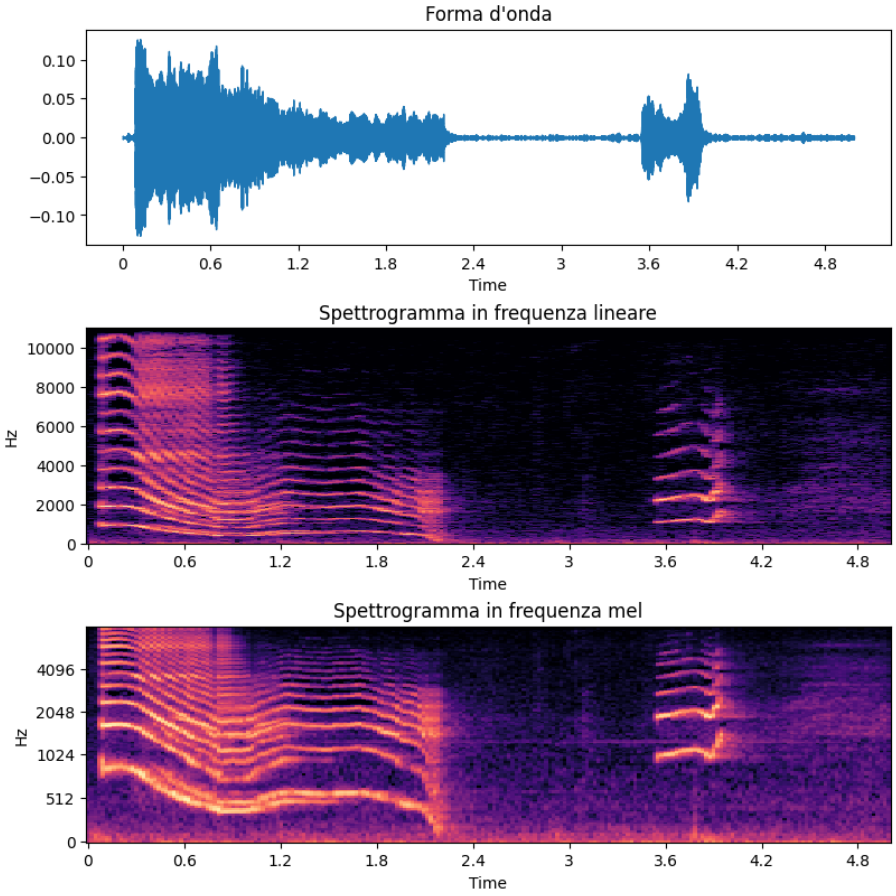

Figura 4.1: Passaggi di trasformazione dalla forma d'onda allo spettrogramma in scala mel.

4.1.2 Data augmentation

Una delle maggiori criticità che lo sviluppo di sistemi di riconoscimento di suoni ambientali presenta è l'assenza di quantità sufficiente di dati etichettati che possano essere utilizzati per il training delle reti neurali. Per limitare questo problema una delle soluzioni più ricorrenti è la fase di data augmentation. Essa consiste nel generare artificialmente nuovi dati applicando delle modifiche ai segnali audio di partenza, aumentando così il numero di elementi appartenenti a ciascuna classe. Questa pratica svolge un ruolo fondamentale nel migliorare le performance della rete, aumentando la robustezza e la capacità di generalizzazione del sistema [15]. Le operazioni eseguite durante la fase di data augmentation sono: traslazione dell'immagine lungo gli assi x e y, di un fattore casuale compreso fra 0 e 0.1 e zoom sull'immagine, di un fattore casuale compreso fra 0 e 0.1.

4.2 Calibrazione degli iperparametri

Gli iperparametri di una rete neurale sono un insieme di impostazioni che vengono fissate prima delle fasi di training e test e definiscono il comportamento e le performance della rete. Il processo di calibrazione degli iperparametri si basa quasi esclusivamente sull'euristica in quanto necessita di diverse sperimentazioni con varie combinazioni di iperparametri; l'obiettivo è quello di trovare la combinazione migliore che minimizzi l'errore di training prevenendo, allo stesso tempo, l'overfitting della rete. L'overfitting è uno dei rischi più significativi di questo processo e dipende dall'equilibrio fra complessità del modello e generalizzazione; è necessario, infatti, che la rete presenti un livello di complessità sufficiente ad ottenere delle prestazioni considerevoli ma che allo stesso tempo non perda la capacità di generalizzare le caratteristiche nei dati utilizzati per il training. Il rischio è quello di incontrare dei pattern ripetuti nei dati di training che non sono caratteristiche generali del tipo di dato che si sta analizzando, facendo perdere alla rete la capacità di riconoscere i dati che appartengono ad una determinata categoria senza presentare tali pattern non generali.

Learning rate

Il learning rate è uno degli iperparametri più importanti della rete, determina la dimensione del passo con cui la funzione di perdita scende verso il suo minimo assoluto, cioè definisce la quantità con cui i pesi vengono aggiornati. Ha un valore che varia da 0.0 a 1.0. Valori ridotti richiedono un tempo di training maggiore dovuto ai piccoli cambiamenti che vengono effettuati e rischiano di bloccare la funzione ad un minimo locale; valori elevati, al contrario, riducono il tempo di training ma rischiano che la funzione di perdita superi il minimo senza riuscire a raggiungere un valore ottimale. Il learning rate può essere impostato come costante oppure può

essere associato a delle funzioni che ne modificano il valore con l'avanzare del training della rete, in particolare si utilizzano funzioni che regolano il learning rate in base al numero di epoche di training trascorse oppure in base al valore attuale della funzione di perdita.

Ottimizzatori

L'ottimizzatore è una funzione o un metodo utilizzato per minimizzare l'errore della funzione di perdita massimizzando quindi l'efficienza della rete. Esistono diversi ottimizzatori che possono essere utilizzati da una rete neurale, che differiscono a seconda della funzione che rappresentano: i più conosciuti sono Stochastic Gradient Descent, Momentum e Adam [30].

Stochastic Gradient Descent (SGD) è un algoritmo che aggiorna i pesi della rete basandosi sulla derivata della funzione di perdita, i pesi vengono aggiornati dopo il calcolo della funzione di perdita su ogni dato utilizzato per il training. Poichè i parametri del modello sono aggiornati di frequente i pesi hanno una varianza elevata. Esistono poi alcune variazioni dello SGD come il Mini-Batch SGD che suddivide il dataset di training in sottoinsiemi di dimensione costante, detti batch, e aggiorna i pesi del modello dopo il calcolo della funzione di perdita su ogni insieme.

Momentum è un ottimizzatore ideato per ridurre l'elevata varianza del SDG. Si basa sull'utilizzo di una nuova variabile chiamata termine di momentum, utilizzata insieme al gradiente calcolato rispetto ad ogni peso, per aggiornare questi ultimi. Questa variabile è una media dei gradienti e serve per accumulare i gradienti passati guidandone la discesa verso il minimo della funzione di perdita; può essere interpretata come la velocità con cui l'ottimizzatore scende verso il minimo che aiuta a smorzare le oscillazioni nel processo di ottimizzazione. Una variazione di questo ottimizzatore è il Nesterov Momentum, che prima aggiorna i pesi utilizzando una frazione del termine di momentum precedente e poi calcola il gradiente della funzione di perdita utilizzando i nuovi pesi. Il termine di momentum può favorire la corretta predizione dei nuovi pesi da utilizzare per calcolare il gradiente in modo più efficace aiutando la funzione a convergere al minimo più velocemente.

Adam è un ottimizzatore che utilizza sia la media che la varianza del gradiente: la prima viene utilizzata in modo simile al termine di momentum, supportando l'ottimizzatore a mantenere la stessa direzione nella discesa verso il minimo; la seconda, invece, aiuta l'ottimizzatore a scalare il learning rate per ciascun parametro in base alla varianza del gradiente. Adam è considerato uno degli ottimizzatori più potenti in termini di tempo di discesa verso il minimo ma presenta costi computazionali più elevati.

Funzione di perdita

In ambito di machine learning e deep learning una funzione di perdita è una funzione utilizzata per valutare una possibile soluzione al problema di classificazione che si cerca di risolvere,

rappresenta infatti il costo dell'inaccuratezza della classificazione: più alto è il valore della funzione e più è inaccurata la previsione. In generale l'obiettivo di un algoritmo di classificazione è quello di trovare una funzione che riesca a predire al meglio le etichette di un dato insieme di input. Tuttavia a causa delle componenti probabilistiche, eventuali componenti rumorose nel segnale e incompletezza dei dati è possibile generare diverse classificazioni a partire da uno stesso input. Lo scopo è quindi quello di trovare quale sia la classificazione che si avvicina di più all'input basandosi sul criterio di massima verosimiglianza, ovvero trovare la classificazione che abbia una distribuzione delle previsioni più vicina possibile alla distribuzione dei dati di input [30].

La funzione di perdita più utilizzata è la cross-entropy che definisce un modello probabilistico coerente per dati discreti su K classi [31]. In altre parole misura la differenza fra le distribuzioni di input e di previsione. Ad ogni dato di previsione viene assegnato un insieme di valori che rappresentano le probabilità che il dato stesso appartenga ad una determinata classe, queste probabilità sono confrontate con l'effettiva probabilità di appartenenza che ha valore 1 in caso di appartenenza e 0 in caso di non appartenenza. Viene quindi assegnato un punteggio di penalità basato sulla differenza fra le due probabilità: maggiore è la differenza e maggiore è il punteggio ottenuto dalla previsione. Un modello che riesca a predire correttamente tutti i dati nelle classi di appartenenza otterrebbe un punteggio pari a 0.0.

Altri iperparametri

- Inizializzazione dei pesi, consiste nell'assegnare ai pesi della rete valori iniziali casuali piccoli da utilizzare come punto di partenza per l'ottimizzatore, variando i pesi di partenza è possibile ottenere risultati diversi con differenti caratteristiche di prestazione.

- Epoche di training, le epoche di training definiscono il numero di volte che l'algoritmo itera attraverso l'intero dataset, che può essere definito come costante oppure possono essere utilizzate delle tecniche che interrompono il training della rete quando si presentano delle condizioni particolari. Ad esempio quando la funzione di perdita non registra diminuzioni significative entro un certo numero di iterazioni.

- Layer della rete, ovvero l'architettura della rete che include tipologia, numero e caratteristiche dei vari layer come funzioni di attivazione e numero di unità.

- Dropout, si tratta di una tipologia di layer che viene inserita nella rete dopo i convolutional e dense layer. La loro funzione è quella di impostare a 0 in modo casuale il valore di una frazione, specificata dal dropout rate, dei pesi dell'input successivo, questo permette di aumentare il livello di generalizzazione della rete.

Capitolo 5

Risultati sperimentali

In questo Capitolo vengono mostrati i momenti più significativi del processo di sviluppo della rete, ponendo particolare attenzione sulla calibrazione degli iperparametri e i vari effetti che variazioni delle caratteristiche della rete possono avere sulle prestazioni.

5.1 Sperimentazioni

Per effettuare le varie sperimentazioni il dataset di file audio viene convertito in un dataset di spettrogrammi utilizzando il processo di estrazione delle caratteristiche descritto nel Capitolo 4. Successivamente è stato suddiviso in due sottoinsiemi, training set di 1600 su 2000 elementi e validation set di 400 su 2000 elementi; il validation set è stato poi ulteriormente diviso a metà in modo da utilizzarne una parte per la fase di test della rete. Tutte le suddivisioni sono avvenute mantenendo costante il rapporto fra dati per classe e dati totali.

Per aumentare l'efficienza e la velocità della fase di training del modello agli spettrogrammi viene applicata un'operazione di ridimensionamento, portando la dimensione da (128,216) a (64,108).

Come ottimizzatore è stato selezionato Adam, con un learning rate iniziale di 0.001.

Infine è stato implementato l'Early Stopping che consiste nell'interruzione della fase di training quando non vengono rilevate diminuzioni sufficienti nella funzione di perdita, con una pazienza di 15 epoche.

5.1.1 Struttura di partenza

La prima struttura della rete, e anche la più semplice, rappresenta il punto di partenza del processo di sviluppo. Questa rete è priva di alcune caratteristiche importanti e viene presentata per mostrare alcune problematiche legate all'assenza di alcuni iperparametri fondamentali. La struttura è composta da:

- input layer, con ridimensionamento;
- convolutional layer, 32 filtri 3x3, attivazione ReLU;
- max pooling layer, filtri di dimensione 2x2;
- convolutional layer, 64 filtri 3x3, attivazione ReLU;
- max pooling layer, filtri di dimensione 2x2;
- flatten layer;
- dense layer, 128 unità, attivazione ReLU;
- output dense layer, 50 unità, attivazione Softmax.

	Training	Validation	Test
Accuracy	0.92	0.32	0.17
Loss	0.3	13.04	4.25

Tabella 5.1: Prestazioni del primo modello

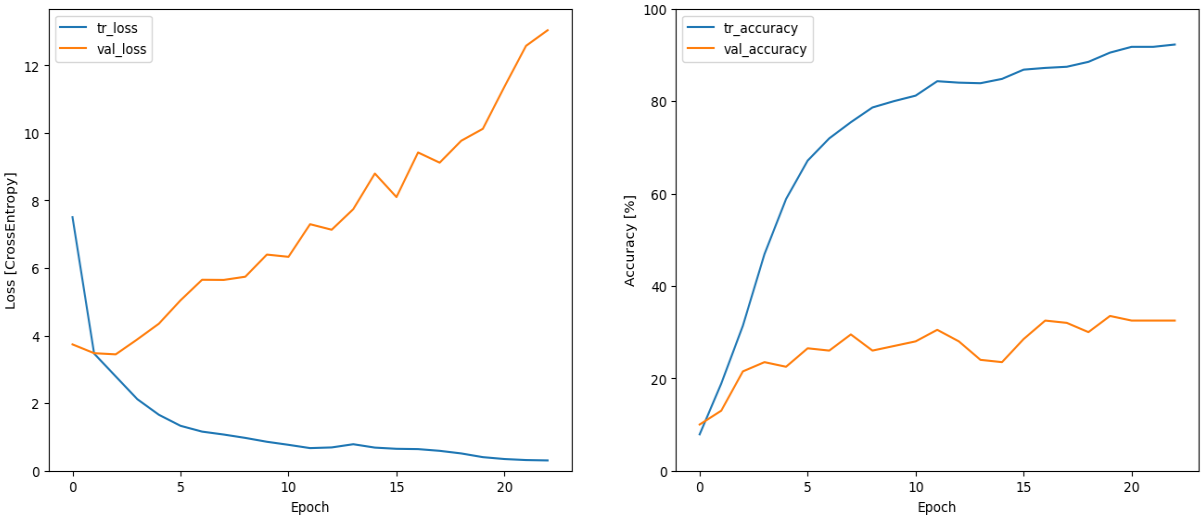

Figura 5.1: Grafici della perdita e dell'accuracy di training e di validation del primo modello

In questo modello l'Early Stopping è avvenuto dopo 23 epoche. Questo modello è caratterizzato da un forte overfitting, infatti si possono notare l'ampia differenza fra l'accuracy di training rispetto a quelle di validation e test, e quella della funzione di perdita che raggiunge un valore ottimale nella fase di training ma rimane elevato nelle fasi di test e soprattutto in quella

di validation. Questo fenomeno si presenta in ambito di deep learning quando una rete neurale presenta una complessità troppo elevata per i dati che sta analizzando non riuscendo a generalizzare correttamente le caratteristiche analizzate. Alcune soluzioni possibili per risolvere questa problematica sono l'utilizzo di un dataset più ampio o la diminuzione della complessità della rete.

5.1.2 Secondo modello

Nel secondo modello sono state introdotte le regolarizzazioni assenti nel primo utili a prevenire l'overfitting, in particolare sono stati aggiunti: l'operazione di data augmentation, il dropout e una funzione per determinare il learning rate che ne dimezza il valore ogni 30 epoche fermandosi a 0.0001.

- input layer, con ridimensionamento;
- convolutional layer, 32 filtri 3x3, attivazione ReLU;
- max pooling layer, filtri di dimensione 2x2;
- convolutional layer, 64 filtri 3x3, attivazione ReLU;
- max pooling layer, filtri di dimensione 2x2;
- dropout layer, rate di 0.25;
- flatten layer;
- dense layer, 128 unità, attivazione ReLU;
- dropout layer, rate di 0.25;
- output dense layer, 50 unità, attivazione Softmax.

	Training	Validation	Test
Accuracy	0.55	0.51	0.46
Loss	1.50	1.80	1.77

Tabella 5.2: Prestazioni del secondo modello

In questo modello l'Early Stopping è avvenuto dopo 127 epoche. Questa seconda struttura presenta un risultato decisamente migliore del primo, ottenendo delle percentuali di accuracy e

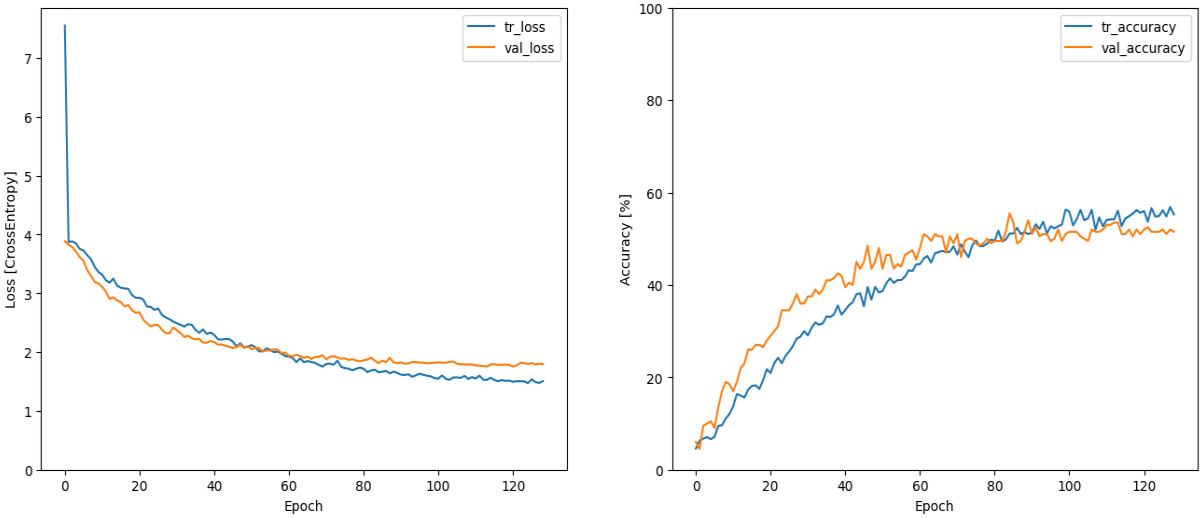

Figura 5.2: Grafici della perdita e dell'accuracy di training e di validation del secondo modello

validation molto simili tra loro annullando quasi completamente l'overfitting. E' quindi possibile notare come le tecniche di regolarizzazione adottate migliorano considerevolmente le capacità della rete di identificare e riconoscere le caratteristiche estratte dai dati di input, permettendo alla rete di generalizzare in modo corretto le informazioni trovate riconoscendo correttamente i dati di validation non analizzati in precedenza. Tuttavia le prestazioni non raggiungono livelli elevati a causa della bassa complessità della rete.

5.1.3 Terzo modello

Il terzo modello corrisponde alla forma definitiva della rete neurale, include alcune variazioni rispetto alla seconda utili ad aumentare la complessità della rete migliorandone le prestazioni. La struttura è composta da:

- input layer, con ridimensionamento;
- convolutional layer, 32 filtri 3x3, attivazione ReLU, L2 regolarizzazione = 0.001;
- convolutional layer, 64 filtri 3x3, attivazione ReLU, L2 regolarizzazione = 0.001;
- max pooling layer, filtri di dimensione 2x2;
- convolutional layer, 64 filtri 3x3, attivazione ReLU, L2 regolarizzazione = 0.001;
- max pooling layer, filtri di dimensione 2x2;
- dropout layer, rate di 0.25;

- flatten layer;
- dense layer, 128 unità, attivazione ReLU;
- dropout layer, rate di 0.25;
- dense layer, 128 unità, attivazione ReLU;
- dropout layer, rate di 0.25;
- output dense layer, 50 unità, attivazione Softmax;

	Training	Validation	Test
Accuracy	0.65	0.63	0.59
Loss	1.21	1.53	1.48

Tabella 5.3: Prestazioni del terzo modello

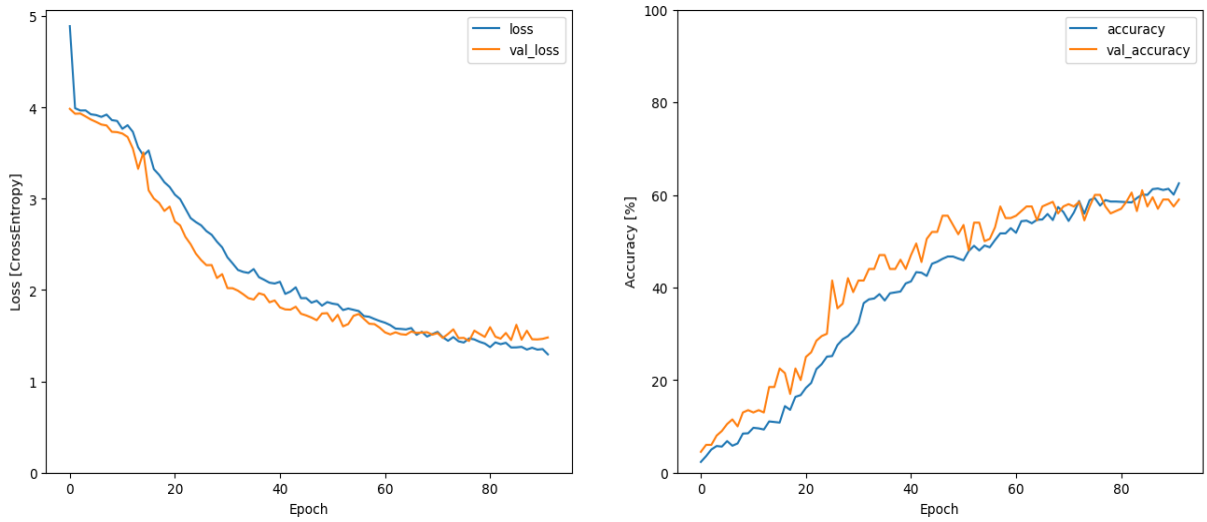

Figura 5.3: Grafici della perdita e dell'accuracy di training e di validation del terzo modello

In questo modello l'Early Stopping è avvenuto dopo 98 epoche. Il terzo modello presenta prestazioni migliori del secondo grazie alla maggior complessità della rete, in particolare sono state effettuate due modifiche principali:

1. modificata la struttura della fase di convoluzione, con l'aggiunta di un nuovo convolutional layer che ha permesso alla rete di aumentare la capacità di astrazione sui dati;

2. aggiunto un secondo dense layer, migliorando la capacità di classificazione della rete.

Inoltre nella fase di convoluzione è stato implementato un nuovo metodo di regolarizzazione per limitare l'overfitting della rete. Questo metodo, identificato dal parametro L2 dei layer, aggiunge un valore di penalità ai pesi dei convolutional layer basato sulla somma dei quadrati dei pesi; vengono favoriti pesi di dimensione ridotta che aiutano a prevenire l'overfitting rendendo la rete più robusta a piccole variazioni dei dati di input.

La Figura 5.4 mette a confronto i risultati ottenuti dai tre modelli proposti.

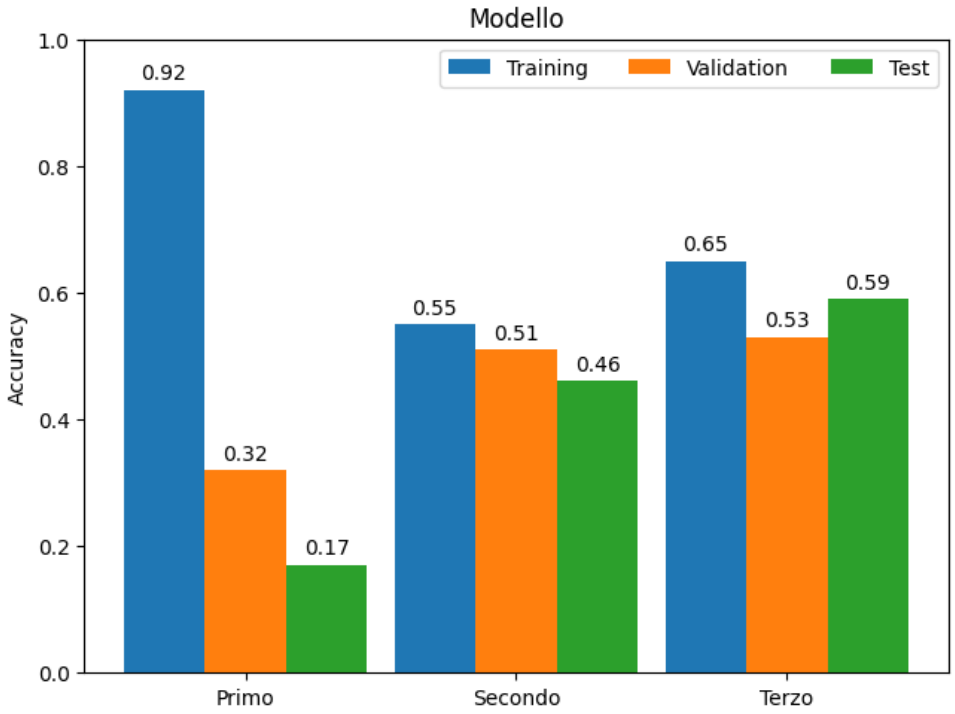

Figura 5.4: Risultati ottenuti dai modelli proposti.

5.2 Risultati finali

5.2.1 Cross-Validation

Il modello finale ottenuto durante la fase di sviluppo è stato valutato utilizzando la tecnica del Cross-validation. Questa tecnica consiste nel suddividere il dataset di partenza in sottoinsiemi di dimensione uguale, che vengono utilizzati a rotazione in modo da eseguire il processo di training della rete utilizzando come training set, validation set e test set sottoinsiemi di dati sempre differenti. Questa metodologia permette di verificare correttamente la capacità di generalizzare del modello selezionato eliminando le componenti di casualità dovute alla suddivisione iniziale ca-

suale dei dati nei vari sottoinsiemi e all'inizializzazione casuale dei pesi di partenza, queste due caratteristiche infatti rischiano di portare a risultati diversi dalle prestazioni reali del modello.

Per effettuare la Cross-validation il dataset è stato suddiviso in 5 sottoinsiemi, ciascuno di 400 elementi, il training è stato quindi eseguito 5 volte, utilizzando 4 sottoinsiemi per il training e dividendo a metà il quinto per i dataset di validation e test. I risultati di ciascuna fase sono rappresentati nella Figura 5.5. Le performance complessive del procedimento sono state ricavate dalla media delle accuracy di test delle 5 ripetizioni, che si possono trovare nella Tabella 5.4.

	Training	Validation	Test
Accuracy	0.73	0.65	0.66

Tabella 5.4: Accuracy medie ottenute dalla Cross-Validation.

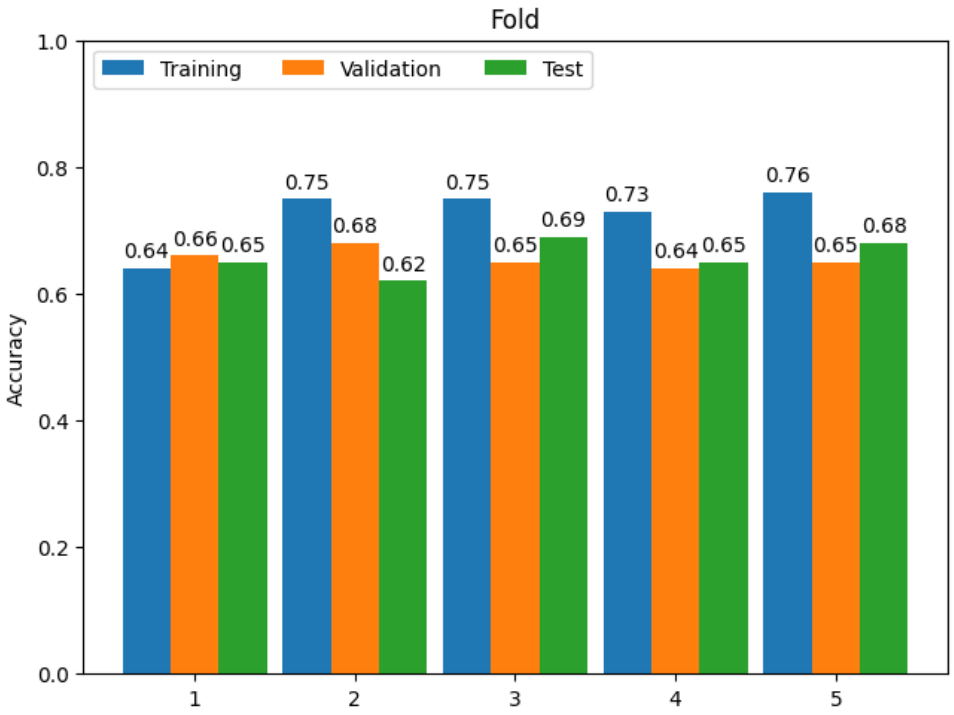

Figura 5.5: Risultati ottenuti da ciascuna fase della Cross-validation.

5.2.2 Matrice di confusione

Infine è stata ricavata una matrice di confusione, mostrata in Figura 5.6, per rappresentare i risultati delle previsioni ottenute dalla fase di Cross-Validation. Durante le varie fasi di test sono stati valutati 1000 suoni complessivi.

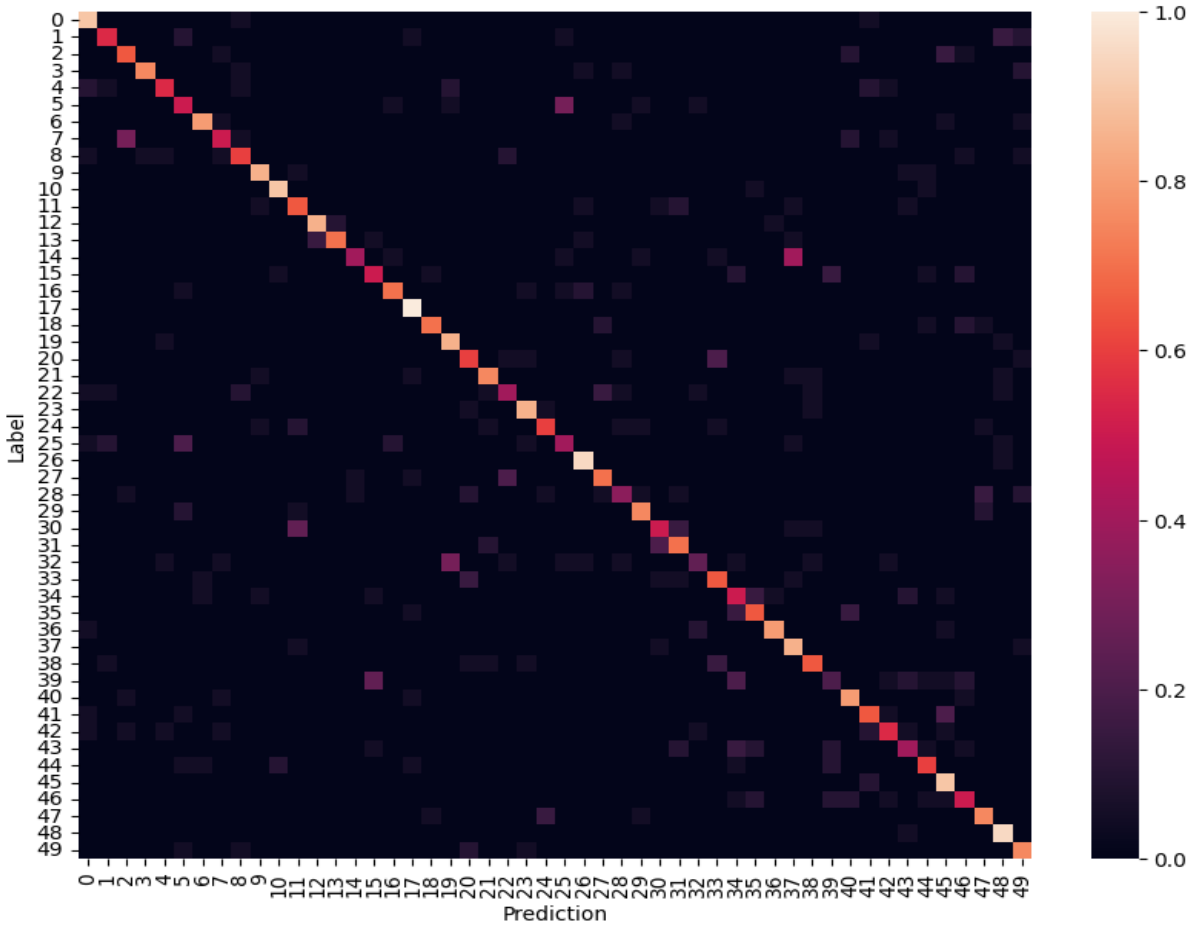

Figura 5.6: Matrice di confusione della Cross-Validation

Dalla matrice di confusione è possibile ricavare informazioni riguardo le classi che vengono identificate con precisione maggiore dal modello, ovvero: 17, scroscio dell'acqua accuracy 1.0, 26, risate con accuracy 0.95, 48 fuochi d'artificio con accuracy 0.95, 0, abbaiare di cani con accuracy 0.9, 10 pioggia con accuracy 0.9, 45, treni con accuracy 0.9.

Inoltre è possibile identificare quali sono state le classi che hanno ottenuto scarsi risultati osservando in corrispondenza di quali classi vengono registrate le previsioni errate. In particolare le classi che hanno ottenuto i risultati peggiori sono state:

1. 39, rottura del vetro, con un'accuracy di 0.2, predetta principalmente come 15, gocciolio dell'acqua, e 34, apertura di una lattina;

2. 32, digitazione su tastiera, con un'accuracy di 0.25, predetta principalmente come 6, tempesta;

3. 28, russare, con un accuracy di 0.35, che non è stata predetta da una classe in particolare ma tra le previsioni errate troviamo classi come 20, pianto di bambino, 49, aereo, e 49,

fuochi d'artificio;

4. 14, cinguettio di uccelli, con un accuracy di 0.4, predetta principalmente come 37, sveglia;
5. 25, passi, con un accuracy di 0.4, predetta principalmente come 5, gatto, e con probabilità minori con altre classi come 1, gallo, e 16, vento.

Infine è possibile eseguire le medie delle accuracy ottenute per ogni raggruppamento di classi, utile ad identificare quale sia la tipologia di suono che viene riconosciuta con prestazioni migliori dal modello:

1. paesaggi sonori naturali e suoni dell'acqua 0.725;
2. suoni urbani 0.685;
3. suoni di animali 0.665;
4. suoni umani non vocali 0.635;
5. suoni domestici 0.575.

Da questi risultati si può notare che il gruppo dei suoni domestici presenta la media più bassa. Questo può essere dovuto al fatto che i suoni contenuti in questo gruppo sono caratterizzati da forme d'onda, e quindi anche da spettrogrammi, con caratteristiche presenti in suoni appartenenti a categorie diverse e che vengono attribuite maggiormente a questi ultimi. Al contrario il gruppo che ha ottenuto la media più alta, paesaggi sonori naturali, presenta delle forme d'onda con caratteristiche più definite che vengono isolate e riconosciute più facilmente dal sistema.

Capitolo 6

Conclusioni

6.1 Riassunto del lavoro svolto

Per riassumere, questo progetto aveva come obiettivo principale quello di progettare ed implementare un sistema di riconoscimento di suoni ambientali basato su reti neurali convoluzionali, che riuscisse a classificare suoni appartenenti ad un insieme pre-determinato di classi con delle prestazioni soddisfacenti. Il primo passo nello sviluppo di questo lavoro è stato la scelta del dataset ESC-50 come base per l'implementazione della rete, seguito da un'nalisi della letteratura per selezionare l'architettura più adatta, in questo caso le reti convoluzionali. Successivamente è stata effettuata la fase di progettazione della rete, durante la quale sono state effettuate le scelte riguardanti il processo di preelaborazione dei dati ed estrazione delle caratteristiche, insieme all'inizializzazione degli iperparametri. Infine attraverso un processo di sperimentazione sono state analizzate le prestazioni della rete apportando eventuali modifiche e migliorie adatte a ottimizzare i risultati, per poi valutare l'architettura finale attraverso un processo di cross-validation. Il risultato finale ottenuto da questo progetto di sviluppo è una rete in grado di classificare registrazioni audio con accuracy del 66%.

6.2 Criticità e limitazioni

Confrontando il risultato ottenuto con quelli riportati nel paragrafo 2.2.4 è possibile notare che le prestazioni ottenute sono difficilmente paragonabili allo stato dell'arte attuale in ambito di classificazione di suoni ambientali. Questo è dovuto ad una serie di limitazioni che si sono riscontrate durante la realizzazione del progetto.

Il fattore che più ha limitato la fase di sviluppo è stato il tempo di training della rete. Questa componente dipende direttamente dalla complessità della rete, su cui influiscono principalmente la tipologia e il numero di layer che compongono la struttura, la dimensione dei filtri applicati

e la quantità di regolarizzazione che viene effettuata. Inoltre è da tenere in considerazione la quantità di risorse messe a disposizione dall'ambiente di sviluppo selezionato, Google Colab, che nonostante i vantaggi presentati, fornisce delle risorse variabili correlate ad una serie di fattori indipendenti dall'utente e che quindi possono non permettere sempre un'esecuzione efficiente del training. Nel caso presentato la componente temporale ha avuto una forte influenza su diversi aspetti dello sviluppo, a causa dei tempi prolungati riscontrati nell'esecuzione delle fasi di training e validation. Infatti durante la fase di calibrazione degli iperparametri è stato utilizzato un approccio di tipo *"trial and error"*, che ha comportato ad ogni riconfigurazione degli iperparametri, eseguita manualmente, la riesecuzione dell'intera fase di training della rete, rendendo la sperimentazione un processo particolarmente dispendioso in termini di tempo.

Un'altra limitazione può essere individuata nel processo di data augmentation, l'aumento della quantità di dati a disposizione è infatti uno dei metodi migliori per aumentare le prestazioni della rete. Tuttavia anche la quantità di dati influisce direttamente sul tempo di training soprattutto quando la dimensione del dataset di partenza viene moltiplicata per un fattore elevato; il training infatti analizza tutti i dati messi a disposizione e richiede un tempo sempre maggiore con l'aumentare della disponibilità di elementi.

6.3 Considerazioni per lo sviluppo futuro

A fronte delle problematiche sorte durante lo sviluppo di questo progetto, sono diverse le modifiche e le migliorie che possono essere apportate per migliorarne le prestazioni.

Una prima considerazione può essere effettuata sulla componente hardware, infatti il tempo di training della rete è fortemente collegato all'ambiente di sviluppo, per cui utilizzare un ambiente con una maggiore disponibilità di risorse di calcolo permetterebbe di semplificare il processo di calibrazione degli iperparametri portando ad uno sviluppo più rapido e ad un conseguente incremento delle prestazioni, dovuto soprattutto al maggior numero di sperimentazioni possibili.

Per quanto riguarda invece la componente software, si possono fare diverse considerazioni riguardo possibili modifiche che possono essere apportate. In primo luogo utilizzare delle tecniche di preelaborazione dei dati più complesse tra cui possiamo trovare operazioni come l'aggiunta di segnali rumorosi di sottofondo, che permettono una maggior robustezza verso le registrazioni di qualità inferiore, o diversi approcci per quanto riguarda la data augmentation, che possono consistere in un ulteriore aumento dei dati prodotti utilizzando le stesse tecniche oppure nell'utilizzo di diverse configurazioni di modifiche da apportare all'immagine. Un'altra considerazione possibile riguarda l'utilizzo di algoritmi specializzati per la calibrazione degli iperparametri, gran parte dei metodi che vengono proposti infatti basano questa parte dello svi-

luppo su calibrazioni di tipo manuale, tuttavia esistono delle tecniche, come *grid search* o *random search*, che permettono di affidare l'analisi dello spazio degli iperparametri ad un algoritmo in grado di individuare delle possibili configurazioni di particolare rilevanza.

In conclusione, sebbene i risultati ottenuti siano in linea con gli obiettivi prefissati all'inizio del progetto, sono diversi gli interventi che, sia dal lato hardware che da quello software, porterebbero ad un incremento delle prestazioni generali del sistema.

Bibliografia

[1] R. V. Sharan e T. J. Moir, «An overview of applications and advancements in automatic sound recognition,» *Neurocomputing*, vol. 200, pp. 22–34, 2016, ISSN: 0925-2312.

[2] O. O. Abayomi-Alli, R. Damaševičius, A. Qazi, M. Adedoyin-Olowe e S. Misra, «Data Augmentation and Deep Learning Methods in Sound Classification: A Systematic Review,» *Electronics*, vol. 11, n. 22, 2022, ISSN: 2079-9292.

[3] Y.-c. Wu e J.-w. Feng, «Development and application of artificial neural network,» *Wireless Personal Communications*, vol. 102, pp. 1645–1656, 2018.

[4] S. Russell e P. Norvig, *Artificial Intelligence: A Modern Approach* (Always learning). Pearson, 2016, ISBN: 9781292153964.

[5] S. Mostafa e F.-X. Wu, «Chapter 3 - Diagnosis of autism spectrum disorder with convolutional autoencoder and structural MRI images,» in *Neural Engineering Techniques for Autism Spectrum Disorder*, A. S. El-Baz e J. S. Suri, cur., Academic Press, 2021, pp. 23–38, ISBN: 978-0-12-822822-7.

[6] E. Steinfath, A. Palacios-Muñoz, J. R. Rottschäfer, D. Yuezak e J. Clemens, «Fast and accurate annotation of acoustic signals with deep neural networks,» *eLife*, vol. 10, R. L. Calabrese, S. R. Egnor e T. Troyer, cur.,

[7] I. Rida, *Feature Extraction for Temporal Signal Recognition: An Overview*, 2018.

[8] D. Mitrovic, M. Zeppelzauer e C. Breiteneder, «Features for Content-Based Audio Retrieval.,» *Advances in Computers*, vol. 78, pp. 71–150, gen. 2010.

[9] *Mel-frequency cepstrum - Wikipedia*, `https://en.wikipedia.org/wiki/Mel-frequency_cepstrum`.

[10] M. G. Ragab, S. J. Abdulkadir, N. Aziz, H. Alhussian, A. Bala e A. Alqushaibi, «An Ensemble One Dimensional Convolutional Neural Network with Bayesian Optimization for Environmental Sound Classification,» *Applied Sciences*, vol. 11, n. 10, 2021.

[11] A. Bansal e N. K. Garg, «Environmental Sound Classification: A descriptive review of the literature,» *Intelligent Systems with Applications*, vol. 16, p. 200 115, 2022, ISSN: 2667-3053.

[12] M. R. Ahmed, T. I. Robin e A. A. Shafin, «Automatic Environmental Sound Recognition (AESR) Using Convolutional Neural Network.,» *International Journal of Modern Education & Computer Science*, vol. 12, n. 5, 2020.

[13] J. Salamon, C. Jacoby e J. P. Bello, «A Dataset and Taxonomy for Urban Sound Research,» in *Proceedings of the 22nd ACM International Conference on Multimedia*, New York, NY, USA: Association for Computing Machinery, 2014, ISBN: 9781450330633.

[14] K. J. Piczak, «Environmental sound classification with convolutional neural networks,» in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[15] J. Salamon e J. P. Bello, «Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification,» *IEEE Signal Processing Letters*, vol. 24, n. 3, pp. 279–283, 2017.

[16] S. Abdoli, P. Cardinal e A. Lameiras Koerich, «End-to-end environmental sound classification using a 1D convolutional neural network,» *Expert Systems with Applications*, vol. 136, pp. 252–263, 2019, ISSN: 0957-4174.

[17] S. Li, Y. Yao, J. Hu, G. Liu, X. Yao e J. Hu, «An Ensemble Stacked Convolutional Neural Network Model for Environmental Event Sound Recognition,» *Applied Sciences*, vol. 8, n. 7, 2018, ISSN: 2076-3417.

[18] G. Shafer, *A Mathematical Theory of Evidence*. Princeton: Princeton University Press, 1976, ISBN: 9780691214696.

[19] Y. Su, K. Zhang, J. Wang e K. Madani, «Environment Sound Classification Using a Two-Stream CNN Based on Decision-Level Fusion,» *Sensors*, vol. 19, n. 7, 2019, ISSN: 1424-8220.

[20] Z. Mushtaq, S.-F. Su e Q.-V. Tran, «Spectral images based environmental sound classification using CNN with meaningful data augmentation,» *Applied Acoustics*, vol. 172, p. 107 581, 2021, ISSN: 0003-682X.

[21] K. He, X. Zhang, S. Ren e J. Sun, «Deep Residual Learning for Image Recognition,» in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[22] G. Huang, Z. Liu, L. van der Maaten e K. Q. Weinberger, «Densely Connected Convolutional Networks,» in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] Y. A. Al-Hattab, H. F. Zaki e A. A. Shafie, «Rethinking environmental sound classification using convolutional neural networks: optimized parameter tuning of single feature extraction,» *Neural Computing and Applications*, vol. 33, n. 21, pp. 14 495–14 506, 2021.

[24] Z. Li, F. Liu, W. Yang, S. Peng e J. Zhou, «A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects,» *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, n. 12, pp. 6999–7019, 2022.

[25] M. Abadi, A. Agarwal, P. Barham et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, Software available from tensorflow.org, 2015. indirizzo: `https://www.tensorflow.org/`.

[26] F. Chollet et al., *Keras*, `https://keras.io`, 2015.

[27] B. McFee, C. Raffel, D. Liang et al., «librosa: Audio and music signal analysis in python,» in *Proceedings of the 14th python in science conference*, vol. 8, 2015, pp. 18–25.

[28] K. J. Piczak, «ESC: Dataset for environmental sound classification,» in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[29] M. Huzaifah, *Comparison of Time-Frequency Representations for Environmental Sound Classification using Convolutional Neural Networks*, 2017.

[30] I. Goodfellow, Y. Bengio e A. Courville, *Deep Learning*. MIT Press, 2016, `http://www.deeplearningbook.org`.

[31] E. Gordon-Rodriguez, G. Loaiza-Ganem, G. Pleiss e J. P. Cunningham, «Uses and Abuses of the Cross-Entropy Loss: Case Studies in Modern Deep Learning,» J. Zosa Forde, F. Ruiz, M. F. Pradier e A. Schein, cur., ser. Proceedings of Machine Learning Research, vol. 137, PMLR, 2020.