UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria Informatica e dell'Automazione

TESI DI LAUREA

Metodi e tecnologie per il rilevamento di delfini in ambienti di pesca

Methods and technologies for dolphin detection in fishing environments

Relatore
Prof. David Scaradozzi

Candidato
Nicolas Bravi

Primo Correlatore
Prof. Francesco Di Nardo

Secondo Correlatore
Prof. Rocco De Marco

ANNO ACCADEMICO 2022-2023

Università Politecnica delle
Marche Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione
Via Brecce Bianche – 60131 Ancona (AN), Italy

Sommario

La tesi in oggetto aspira ad applicare principi di bioacustica congiuntamente ad avanzate tecniche di intelligenza artificiale, con lo scopo di rilevare la presenza di delfini in prossimità di imbarcazioni per la pesca attraverso l'utilizzo di una rete neurale convoluzionale per la classificazione e l'identificazione di una delle principali vocalizzazioni caratteristiche di questo cetaceo e cioè il fischio. In un ulteriore sviluppo, la rete neurale viene impiegata per l'etichettatura di file audio precedentemente registrati, generando un file di testo ove, lungo la barra temporale, sono marcati i fischi identificati. Nel contesto del documento, sono delineate le diverse fasi che hanno anticipato la concretizzazione di questo sistema, con un esame delle metriche di valutazione e delle tecniche selezionate, nonché delle motivazioni per le scelte operate. Per offrire una panoramica comparativa e garantire una comprensione approfondita del progetto, la rete neurale ideata è stata confrontata con due delle più riconosciute reti convoluzionali, specificamente elaborate per il problema della classificazione delle immagini. La capacità di decifrare tali suoni non solo potenzia la nostra comprensione riguardo le intricate strutture sociali, i modelli comportamentali e le tattiche di caccia dei delfini, ma apre la via allo sviluppo di metodi significativamente più efficaci volti alla loro conservazione e tutela. Questa tecnologia, pertanto, può fornire nuove prospettive riguardo la vita sociale dei delfini, le loro strategie per risolvere problemi e le loro competenze cognitive. Al termine della tesi, oltre a un'analisi conclusiva attinente ai risultati conseguiti, vengono prospettate considerazioni relative a potenziali sviluppi futuri, aprendo così nuove linee di ricerca e applicazioni pratiche che potrebbero ulteriormente approfondire e ampliare la nostra conoscenza e interazione con questi mammiferi marini.

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

Questa tesi si propone di sfruttare le moderne tecniche di intelligenza artificiale per il rilevamento della presenza di delfini tramite analisi acustica, attraverso il riconoscimento dei loro fischi. L'obiettivo principale di questo lavoro di ricerca è sviluppare ed addestrare una Rete Neurale Convoluzionale (CNN) in grado di riconoscere automaticamente e marcare nel tempo i fischi dei tursiopi (Tursiops truncatus), ovvero la specie di cetacei più presente nel mar Mediterraneo. Una possibile applicazione di questo metodo riguarda il rilevamento di tali delfini in contesti di pesca, un'area di fondamentale importanza e di questioni urgenti per la conservazione delle specie marine e la sostenibilità della pesca in mare [1-2]. In particolar modo, nella pesca a strascico il "foraggiamento del peschereccio" risulta una tecnica del tursiope nel bacino del Mediterraneo ben documentata [3-4]. Oltre alle possibili ripercussioni di sopravvivenza dovute alla loro cattura accidentale, si ha dalla parte del settore della pesca perdite economiche dovute dal danneggiamento delle reti e alla riduzione della quantità e qualità del pescato, dovute alla predazione [5-6]. Con la crescita esponenziale delle industrie della pesca, le tecniche e gli strumenti utilizzati per catturare il pesce sono diventati sempre più efficaci, ma purtroppo anche più dannosi per la fauna marina. I delfini, curiosi e spesso in cerca di cibo, si avvicinano alle reti da pesca per mangiare, rischiando di rimanere impigliati e danneggiando l'attrezzatura dei pescatori. Inoltre, molti subiscono accidentalmente lesioni a causa di tali interazioni e, in alcuni casi, questi muoiono. Quindi, questo conflitto non rappresenta solo una minaccia economica per i pescatori, ma solleva anche questioni etiche legate alla sopravvivenza dei delfini. Proprio per questo, per limitare l'interazione tra i cetacei e le attività di pesca professionale è nato il progetto Life DELFI [7] che coinvolge enti di ricerca, università, associazioni ambientaliste e aree marine protette nello studiare soluzioni e modelli per realizzare un equilibrio tra l'essere umano e l'ambiente marino. Lo scopo di questo elaborato è di offrire un contributo, con l'ausilio dell'intelligenza artificiale, a mitigare il crescente problema dell'interazione dei delfini con le reti da pesca nella speranza di trovare il più velocemente possibile una coesistenza equilibrata tra attività di pesca e la vita marina.

CAPITOLO 1

Intelligenza Artificiale per la salvaguardia dei delfini

In questo capitolo viene descritto il problema causato dalla continua interazione tra i delfini e le attività di pesca, con le sue conseguenze. Viene inoltre effettuato uno studio sulle possibili soluzioni per risolvere il problema.

1.1 Interazione delfini con attività di pesca

Il dilemma morale ed economico nato dall'interazione tra delfini e pescatori sta generando numerose discussioni nel tentativo di trovare una coesistenza pacifica tra uomo e natura. Le soluzioni attuali non rispettano adeguatamente le questioni etiche inerenti alla vita animale, si cerca dunque un sistema che non danneggi nessun soggetto coinvolto.

1.1.1 Dinamiche tra delfini e pescatori

I delfini manifestando una particolare intelligenza e socievolezza hanno da sempre catturato l'immaginazione umana. La loro capacità di comunicare attraverso vari tipi di vocalizzazioni ha stimolato numerose ricerche rivolte a decifrare il loro linguaggio e a comprendere meglio come essi percepiscono e interagiscono con l'ambiente circostante. Tuttavia, mentre antiche cronache descrivono rapporti armoniosi tra pescatori e delfini, come la pesca cooperativa o la cura dei delfini impigliati nelle reti [8-9], questo rapporto ha subito una trasformazione nel tempo fino a diventare una minaccia per i cetacei e un danno economico per i pescatori. Secondo quanto riportato nell'articolo [1], nelle interviste a centinaia di pescatori nel mar Adriatico tra Italia e Croazia è emerso che più della metà di essi ha avuto spiacevoli incontri con delfini almeno una volta durante la loro attività di pesca. Inoltre, molti di essi affermano come il numero di esemplari di cetacei sia aumentato notevolmente negli anni, in particolar modo la specie dei tursiopi (Tursiops truncatus), il cetaceo attualmente più presente nel mar Adriatico. Uno dei fenomeni più comune durante le interazioni viene chiamato "depredazione", ovvero il furto di cibo già

catturato [10]. In alcuni casi, i delfini hanno imparato a rubare il pesce direttamente dalle reti dei pescatori come mostrato in Figura 1, un comportamento che può portare a perdite economiche per i pescatori. Oltre a disperdere numerosi pesci, che naturalmente desiderano evitare di essere predati, i delfini non solo riducono la quantità delle catture, ma ne compromettono anche la qualità [5] (Figura 2). Infatti, tutto il pescato che viene morso smette di essere commestibile. In aggiunta, un altro danno economico per i pescatori è legato all'attrezzatura danneggiata, dovuto ai morsi inferti alle reti da pesca e alla cattura accidentale di questi cetacei. Questo evento, al di là dell'aspetto economico, solleva anche una questione morale. Infatti, molti delfini, venendo accidentalmente catturati nelle reti, non solo causano danni all'attrezzatura da pesca, che deve poi essere inevitabilmente sostituita o riparata, ma, rimanendo impigliati, subiscono lesioni e, nel peggiore dei casi, perdono la vita. Proprio a causa di quest'ultima motivazione, l'interazione tra delfini e pescatori è diventata un argomento di ampio dibattito, con implicazioni economiche per i pescatori e morali riguardanti la sopravvivenza di questa specie animale.

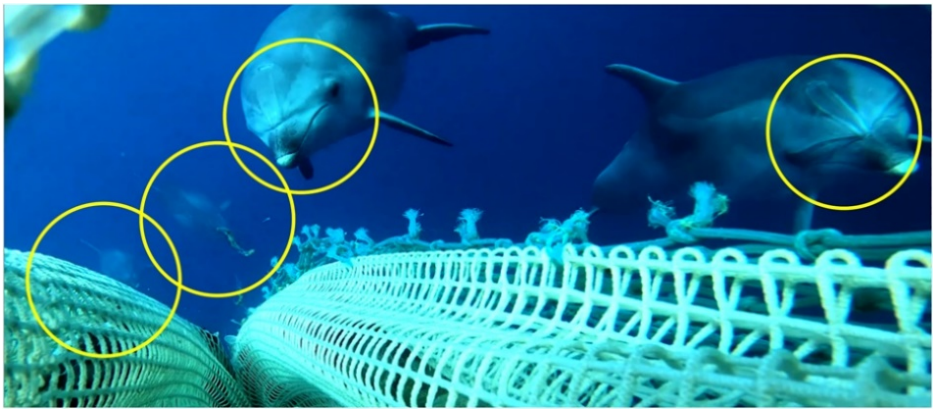

Figura 1. *Nei cerchi gialli delfini in fase di depradazione [10]*

Figura 2. *Resti di pesce mangiato da delfino [11]*

1.1.2 Misure di mitigazione

Alcuni pescatori hanno pensato di farsi giustizia da soli, provando a risolvere il problema nel più brusco dei modi, ovvero uccidendo i delfini intravisti durante le loro attività di pesca [3]. Effettivamente, potrebbe ridurre il problema dal punto di vista economico, ma per una questione etica e morale questo sistema non può essere accettabile. La soluzione ideale sarebbe quella di allontanare i delfini dalle zone di pesca, senza porli in pericolo di vita, da questo scopo nasce il progetto LIFE DELFI [7], coinvolgendo enti di ricerca, università, associazioni ambientaliste e aree marine protette nell'intento comune di sviluppare soluzioni e modelli di gestione sostenibili delle interazioni fra delfini e pesca. Una delle proposte fornite, suggerisce l'utilizzo di deterrenti acustici noti come pinger. I pinger sono dispositivi di dissuasione audio basati su un livello di sorgente acustica relativamente basso (in genere inferiore a 150 dB 1 μPa a 1 m) e operante da frequenze medio alte (2.5-10 kHz) con armoniche fino a frequenze più elevate [7]. Tramite l'utilizzo di idrofoni riescono a prelevare il segnale acustico sott'acqua, che poi viene analizzato per rilevare la presenza di un delfino. In tal caso i pinger emettono ultrasuoni interagendo specificamente con il comportamento acustico dei delfini alterando il loro orientamento, la loro capacità di cercare cibo e la loro comunicazione intra-specifica. Poiché gli idrofoni in commercio non permetterebbero al dispositivo di avere un prezzo accessibile a tutti i pescatori, negli articoli [12-13] vengono studiate delle alternative che consistono nella realizzazione di un idrofono a basso costo con caratteristiche accettabili. Nonostante l'esistenza di diversi tipi di vocalizzazione del delfino come ad esempio i click, in questo progetto vengono analizzati i fischi in quanto emessi molto frequentemente e con caratteristiche matematiche più definite. Tuttavia, è nato un interessante dibattito sui possibili metodi di identificazione del fischio e l'obiettivo è di selezionare una tecnica che riesca a raggiungere un'accuratezza accettabile.

1.2 Stato dell'arte

L'identificazione acustica di delfini può essere effettuata in più modi. Di seguito vengono analizzate le possibili soluzioni proposte in letteratura.

1.2.1 Passive Acoustic Monitoring

La Passive Acoustic Monitoring (PAM) [14-15-16] rappresenta un approccio rispettoso e non invasivo nei confronti dell'ambiente marino, fondamentale per la rilevazione acustica subacquea di cetacei. Questa metodologia specifica si basa sull'impiego e la sistemazione strategica di dispositivi acustici passivi marini, ad esempio gli idrofoni, posizionati con l'obiettivo di registrare i suoni provenienti dall'ambiente sottomarino circostante. Di conseguenza, si verifica la possibilità di monitorare i movimenti di cetacei in ampi tratti di mare aperto, quantificandone la presenza in determinate aree e osservando i loro comportamenti in un contesto naturale. Tale metodo non solo gioca un ruolo cruciale nello studio approfondito dei mammiferi marini, ma si rivela anche fondamentale nella mitigazione degli impatti negativi derivanti dall'essere umano, come le attività di pesca. Il principio su cui si basa la PAM involve l'analisi dettagliata di picchi di segnale, riuscendo a differenziare i suoni emessi dai cetacei da altri potenziali rumori marini. Una delle limitazioni di questo sistema può essere rappresentata dalla sovrapposizione tra le vocalizzazioni dei cetacei e suoni di fondo di ampiezza significativa. Per di più, considerando che le vocalizzazioni dei delfini sono notevolmente direzionali, esiste la concreta possibilità che gli stessi non vengano rilevati se sono orientati in una direzione opposta rispetto alla posizione degli idrofoni. Nonostante ciò, l'analisi PAM rimane una delle tecniche più utilizzate in tale contesto. Il software open-source "Pamguard" [17] costituisce uno degli esempi più noti in questo specifico settore, fornendo una piattaforma avanzata e dedicata per l'analisi PAM, focalizzata sulla rilevazione dei cetacei. In parallelo, i dispositivi POD, sviluppati dalla Chelonia Limited, si sono dimostrati strumenti inestimabili nel campo della ricerca marina [18]. Con una notevole autonomia che si attesta sui quattro mesi, sia in termini di durata della batteria che di capacità di archiviazione, questi dispositivi vengono posizionati strategicamente in mare con l'intenzione di registrare dati in maniera intelligente, escludendo quelli ritenuti non necessari. I dati raccolti sono poi sottoposti a un accurato processo di analisi mediante

un software specializzato che, attraverso l'analisi PAM, rileva e distingue con precisione i diversi tipi di cetacei presenti. Essi hanno trovato un'importante applicazione anche nel progetto SAMBAH (Static Acoustic Monitoring of the Baltic Sea Harbour Porpoise), che ha rappresentato un'iniziativa significativa nell'ambito della conservazione e dello studio dei delfini nel mar Baltico [19]. Per un periodo di due anni, 304 dispositivi C-POD sono stati posizionati come in Figura 3, in modo da occupare la più vasta area possibile del mar Baltico. I dati raccolti hanno offerto uno sguardo sulla distribuzione ed il comportamento dei delfini, fornendo allo stesso tempo informazioni vitali per sviluppare strategie di conservazione e protezione efficaci per questa specie in questo mare.

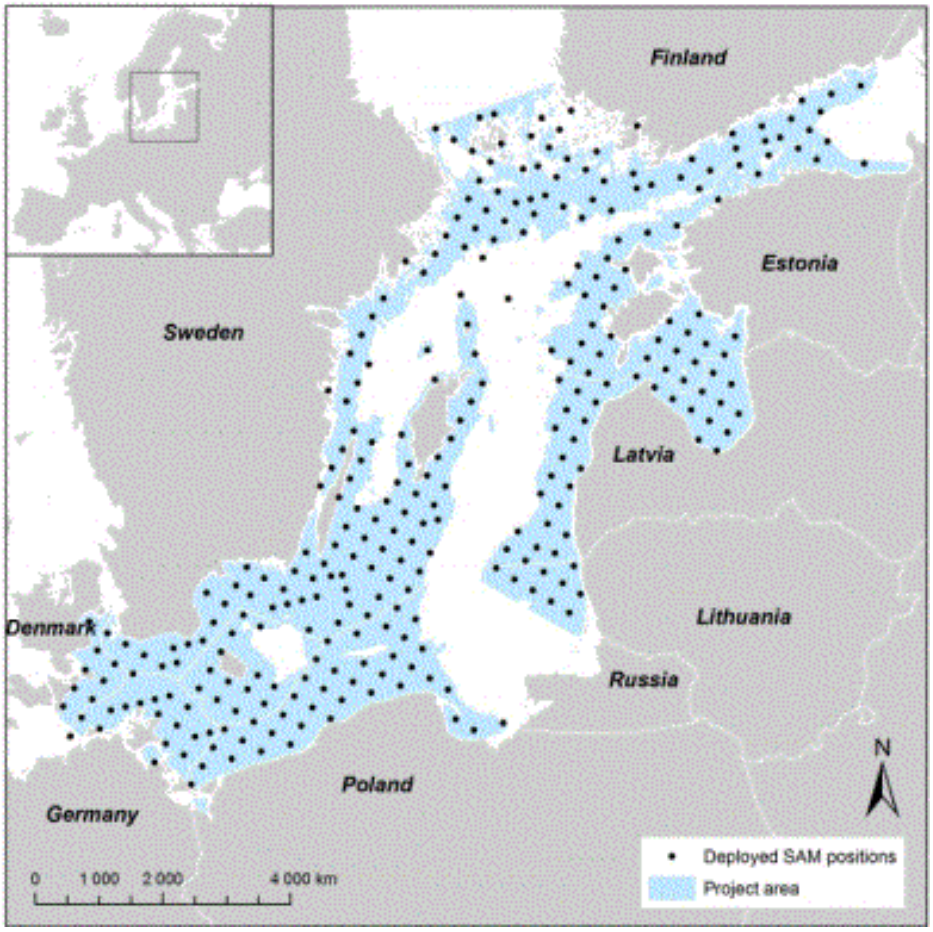

Figura 3. *SAMBAH, posizionamento C-POD*

1.2.2 Machine learning

L'applicazione di algoritmi di machine learning emerge come un'alternativa all'analisi PAM nel contesto del rilevamento acustico dei delfini. L'implementazione di queste tecniche di intelligenza artificiale risulta particolarmente pertinente in scenari nei quali il modello matematico del sistema non è predefinito o risulta impegnativo da derivare. Nonostante la risonanza consolidata dell'intelligenza artificiale nell'analisi delle immagini, la sua efficacia nel trattamento di dati audio direttamente sul segnale, in diversi contesti ha manifestato risultati meno immediati. Alcune ricerche propongono una metodologia che converte il segnale audio in una rappresentazione in frequenza, che può risultare utile anche per risolvere il problema della sovrapposizione di suoni di sottofondo, se emanati a frequenze differenti. Un esempio di identificazione acustica di specie animali è fornito dalla challenge proposta da HP sull'identificazione acustica delle vocalizzazioni degli uccelli cappuccino, suggerendo la via del machine learning [20]. Nelle soluzioni pubblicate, le registrazioni selezionate sono state trasformate in spettrogrammi, impiegati per l'addestramento di una rete neurale, la quale è stata successivamente utilizzata per analizzare lunghe registrazioni con l'obiettivo di quantificare le vocalizzazioni emesse da tale uccello. Similmente, un progetto open-source sulla classificazione delle vocalizzazioni di pipistrelli ha adottato un approccio parallelo [21]. Le registrazioni degli ultrasuoni, utilizzate dai pipistrelli per l'ecolocalizzazione e la caccia, vengono convertiti in spettrogrammi, per poi affidare la classificazione ad una rete neurale. Il rilevamento acustico dei delfini è strettamente paragonabile a questi esempi, specialmente con i pipistrelli in quanto hanno delle frequenze simili ai fischi del delfino e, inoltre, anch'essi possono essere potenzialmente rilevati tramite analisi PAM. Pertanto, il presente progetto intende perseguire la direzione del machine learning, esplorando l'applicabilità e l'efficacia delle tecniche di apprendimento automatico nell'analisi acustica delle vocalizzazioni dei delfini.

1.3 Reti neurali artificiali

Le reti neurali artificiali (ANN) sono sistemi di elaborazione computazionale che traggono ispirazione dai sistemi nervosi biologici, come il cervello umano. Le ANN sono principalmente composte da un gran numero di nodi computazionali interconnessi (chiamati neuroni), che lavorano in modo distribuito per apprendere collettivamente.

1.3.1 ANN e CNN

Le Reti Neurali Convoluzionali (CNN) sono un tipo particolare di rete neurale artificiale, progettate specificamente per elaborare dati con una topologia a griglia, quali le immagini. A differenza delle ANN tradizionali, le CNN utilizzano una particolare operazione chiamata "convoluzione" che permette loro di riconoscere pattern locali all'interno di queste griglie. Mentre una ANN potrebbe richiedere che le immagini vengano ridimensionate o trasformate in vettori monodimensionali per l'elaborazione, una CNN prende l'immagine come input in formato bidimensionale, o addirittura tridimensionale considerando i canali di colore, processando direttamente le strutture spaziali. Questa capacità rende le CNN particolarmente potenti per il riconoscimento di immagini, a cui spesso ne sono associate nonostante le loro applicazioni non si limitano in quel settore. Il funzionamento di una ANN si basa sulle reti neurali biologiche, composte da neuroni biologici Figura 4 che comunicano tra loro con impulsi elettrici e segnali chimici tramite sinapsi [22]. Un neurone artificiale Figura 5 è un modello matematico che ne prende spunto, riceve più input che vengono elaborati e produce un output tramite una funzione di attivazione non lineare, utilizzata come soglia.

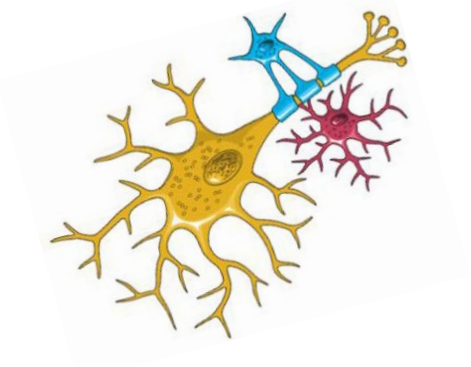

Figura 4. *Neurone Biologico*

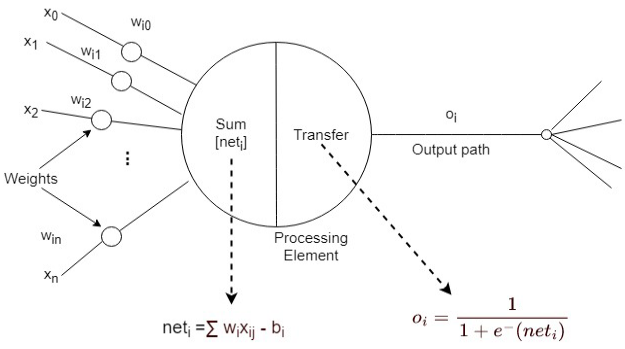

Figura 5. *Neurone Artificiale*

1.3.2 Campi di utilizzo

Le CNN sono uno dei pilastri fondamentali nell'ambito del deep learning, specialmente quando si tratta di dati strutturati come griglie, tipo le immagini. Le loro principali applicazioni sono:

- **Classificazione delle immagini:** Questa è probabilmente l'applicazione più conosciuta delle CNN. L'obiettivo è assegnare un'etichetta ad un'immagine da un set di categorie. Ad esempio, data un'immagine, determinare il tipo di animale rappresentato
- **Rilevamento di oggetti:** Il rilevamento di oggetti localizza e identifica più oggetti all'interno di un'immagine. Ad esempio, una foto potrebbe identificare sia un gatto che una palla e mostrare un rettangolo attorno a ciascuno di essi.
- **Segmentazione semantica:** Questa applicazione mira a classificare ogni singolo pixel di un'immagine. Per esempio, in un'immagine di una strada, può identificare le aree che sono strada, marciapiede, edifici, automobili, persone, ecc.
- **Analisi di video:** Le CNN possono essere utilizzate per riconoscere e classificare azioni o eventi in un video. Questo può essere utile in molte applicazioni come la sorveglianza, l'analisi sportiva o la categorizzazione di contenuti video.
- **Riconoscimento facciale:** Le CNN possono identificare e riconoscere volti specifici in immagini e video. Questo ha applicazioni in sicurezza, social media, fotografia e molte altre aree.
- **Applicazioni mediche:** Le CNN sono state utilizzate per identificare e classificare anomalie in immagini mediche, come radiografie, risonanze magnetiche ed altre tipi di scansioni.
- **Applicazioni non visive:** Anche se le CNN sono principalmente note per l'elaborazione di immagini, possono anche essere adattate per l'analisi di altri tipi di dati strutturati.

Questi sono solo alcuni esempi; l'adattabilità e l'efficacia delle CNN in vari compiti le rendono una delle architetture più popolari e utilizzate nell'ambito del deep learning e della visione artificiale.

1.3.3 Motivi per il loro utilizzo

L'importanza e l'efficacia della CNN in questi ambiti deriva dalla capacità di apprendere automaticamente le caratteristiche rilevanti dei dati, anziché richiedere che queste caratteristiche siano specificate manualmente da un modello matematico. Questo la rende estremamente potente per l'elaborazione di immagini e altre forme di dati strutturati come griglie. Nel caso del rilevamento del fischio dei delfini, una CNN può essere impiegata per le sue capacità di:

- **Alta Precisione:** le reti neurali possono apprendere le caratteristiche dei fischi dei delfini e distinguere questi suoni da altri rumori marini.
- **Automazione:** una volta addestrata, la rete neurale può analizzare grandi quantità di dati acustici in modo automatico, riducendo il bisogno di revisioni manuali.
- **Adattabilità:** con ulteriori dati, la rete neurale può continuare a perfezionarsi, migliorando la sua capacità di riconoscere i fischi in diverse condizioni.
- **Analisi Temporale:** possono identificare i fischi dei delfini in tempo reale o su lunghe registrazioni, fornendo informazioni sulla presenza e il comportamento dei delfini in un'area specifica.

CAPITOLO 2

Materiali e metodi

In questo capitolo vengono descritti i metodi e le tecniche utilizzate per la progettazione della rete neurale per il rilevamento dei fischi dei delfini, cercando di ottenere la massima accuratezza possibile.

2.1 Metodo

L'approccio metodologico adottato per questa ricerca si sviluppa in diverse fasi:

- **Raccolta dei dati:**
  - **Acquisizione:** Utilizzo di idrofoni per registrare suoni sottomarini in aree dove i delfini sono noti per essere attivi.
  - **Etichettatura:** Ascoltare le registrazioni e selezionare manualmente i segmenti di audio in cui si sentono i fischi dei delfini e i segmenti di audio con rumori totalmente differenti, memorizzandoli in due cartelle differenti assegnando così un'etichetta.
- **Pre-elaborazione dei dati:**
  - **Frammentazione:** Dividere le lunghe registrazioni in segmenti più brevi, ad esempio di un secondo ciascuno.
  - **Spettrogramma:** Convertire ogni segmento in uno spettrogramma tramite la FFT (Fast Fourier Transform), che rappresenta la distribuzione di frequenza del suono nel tempo.
  - **Filtraggio:** Filtrare opportunamente le frequenze interessate ed applicare eventuali operatori matematici per rendere più chiara la figura, facilitando il compito della rete neurale.

- **Preparazione dataset:**
  - **Caricamento immagini:** Caricare le immagini nelle dimensioni coincidenti con l'input richiesto dalla rete neurale, e normalizzandole.
  - **Train, validation, test:** Suddivisione del dataset in Train, Validation, Test, con la seconda che viene considerata facoltativa.
- **Architettura della rete neurale:**
  - Scegliere un'opportuna architettura per la rete neurale, definendone numero di layer, numero di filtri, funzioni di attivazione e tutti i parametri che ne riguardano.
- **Addestramento:**
  - Utilizzare il dataset di addestramento per allenare la rete selezionando i parametri adatti al tipo di architettura e dati adoperati.
  - Monitorare le prestazioni sul set di validazione per evitare l'overfitting.
- **Valutazione:**
  - Una volta addestrata, testare la rete sul dataset di test per valutarne le prestazioni in termini di precision, recall, accuracy e F1-Score, valutando inoltre la quantità di falsi negativi e falsi positivi, visibili anche nella Confusion Matrix.
- **Applicazione:**
  - **Utilizzo:** Per mettere in pratica la rete, preparare il materiale da passare come input e far partire la classificazione. Se si tratta di un numero sostanzioso di elementi in input, raggrupparli in batch in modo da ottenere risultati in tempi più brevi.
  - **Interpretazione:** Si può scegliere di far restituire alla rete un numero binario che si riferisce alla rispettiva classe oppure un valore in percentuale. Nell'ultimo caso, questo può essere interpretato per stabilire l'appartenenza ad una determinata classe, gestendo i falsi positivi ed i falsi negativi.

In conclusione, l'uso di reti neurali nel rilevamento di suoni specifici come il fischio dei delfini offre uno strumento potente per la ricerca marina e la conservazione degli ecosistemi marini. Con l'addestramento e l'ottimizzazione adeguati, queste reti possono fornire risultati precisi e tempestivi.

2.2 Acquisizione sperimentale

La fase della raccolta dati rappresenta un passaggio cruciale per l'analisi e la costruzione di tutte le elaborazioni successive. Il suo obiettivo principale è di creare un dataset adeguato che permetta alla rete neurale di apprendere efficacemente.

2.2.1 Acquisizione

Al fine di realizzare un sistema più dinamico possibile, il CNR IRBIM (Istituto per le Risorse Biologiche e le Biotecnologie Marine del Consiglio Nazionale delle Ricerche) in questi ultimi anni ha effettuato più registrazioni in zone, momenti temporali e posizionamenti differenti, in modo tale da acquisire registrazioni di popolazioni di delfini differenti sovrapposti a diversi tipi di rumore, come ecoscandagli, motore di imbarcazioni, ma anche interferenze con altre specie marine come, ad esempio, alcune specie di crostacei. Molte registrazioni sono state effettuate in diversi punti del mar Adriatico in momenti in cui era nota la presenza di delfini, ma per riuscire a collezionare una ampia quantità di vocalizzazioni di tursiope, il delfinario è il luogo in cui è possibile ottenerle facilmente e con un livello di rumore ambientale ridotto. Grazie alla disponibilità concessa da Costa Edutainment, sono stati effettuati campionamenti anche presso il delfinario Oltremare di Riccione che ospita diversi tursiopi nella "Laguna dei Delfini", costituita da diversi ambienti interconnessi, ma separabili in modo da confinare uno o gruppi di esemplari in una o più partizioni presenti in Figura 6. Durante i campionamenti, sono state anche tenute in conto le attività svolte dagli addestratori, in particolar modo per quanto riguarda la distribuzione del cibo.

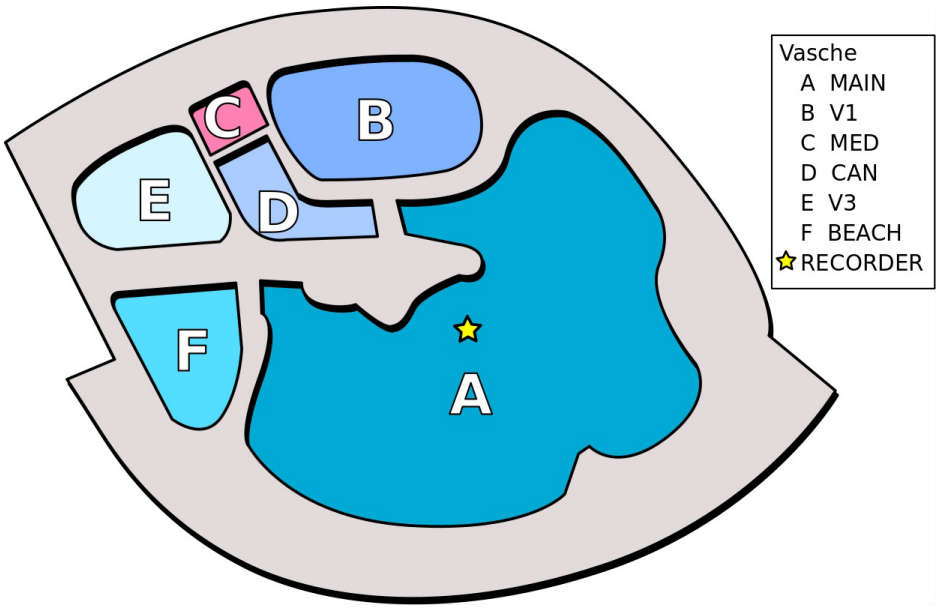

Figura 6. *"Laguna dei delfini" acquario Oltremare di Riccione*

2.2.2 Registrazione

Per questa attività è stato impiegato un registratore subacqueo autonomo UREC384K capace di campionare in modo continuativo con frequenza fino 384 kSps ed è equipaggiato con un idrofono Sensor Technology SQ26-08. L'UREC è un dispositivo di circa 35 cm di lunghezza e 13 cm di diametro comprensivo di protezione messa per riparare l'idrofono con un'autonomia superiore alle 72 ore. Le registrazioni vengono memorizzate in file wave con un sample rate di 192 kHz. In Figura 7 è presente il diagramma di Bode dell'idrofono Sensor Technology SQ26-08, che ne descrive il comportamento. Come è possibile notare, c'è una caduta di segnale tra i 30 kHz e i 40 kHz, ma ciò non è importante per il rilevamento dei fischi dei delfini, in quanto hanno frequenze più basse. Tuttavia, il registratore utilizzato è molto costoso ed il sistema finale comprensivo di esso sarebbe per cui messo in vendita ad un prezzo molto alto, accessibile

quindi a pochi. Gli articoli [12-13] mostrano come sia possibile realizzare un idrofono ad un costo di circa $10 con caratteristiche simili a idrofoni più costosi disponibili in commercio. Inoltre, come descritto nella tesi [23], si è creato un sistema di caratterizzazione acustica utilizzando un approccio diverso dall'UREC384K, cercando di migliorare la qualità degli idrofoni a basso costo realizzati fino ad ora.

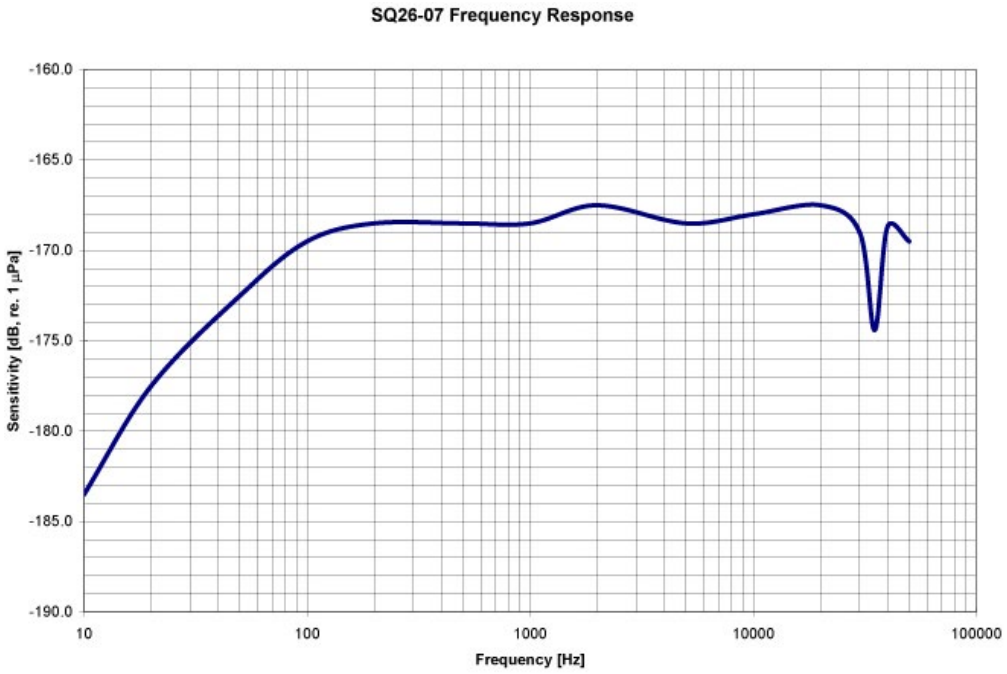

Figura 7. *Diagramma di Bode idrofono Sensor Technology SQ26-08*

2.2.3 Etichettatura

Una volta ottenute le registrazioni in file wav, gli audio sono stati processati da operatori PAM (Passive Acoustic Monitoring) qualificati del CNR IRBIM attraverso la modalità visuale e le vocalizzazioni rilevate contrassegnate attraverso il software Audacity. Al riconoscimento di ogni fischio del delfino, è stata inserita un'etichetta per dichiararne la presenza. L'obiettivo è quello di selezionare una discreta quantità di fischi e di rumori vari, salvando queste registrazioni in due cartelle differenti, in modo tale da poterne distinguere la classe, rumore o fischio. Tuttavia, facendo anche delle

sperimentazioni, si decide di effettuare una selezione dei fischi rilevati tramite l'analisi PAM, in quanto alcuni di questi potrebbero creare confusione alla rete neurale durante l'apprendimento, ottenendo un'accuratezza inferiore. Ad esempio, tutti i fischi di una durata troppo breve non vengono selezionati, come quelli sovrapposti a degli eventi acustici particolarmente rumorosi e quelli con un'ampiezza molto bassa che risultano quasi impercettibili. Per quanto concerne i rumori, selezionare solo delle registrazioni in momenti di tranquillità non sarebbe stato sufficiente per l'apprendimento della rete neurale, in quanto avrebbe identificato come delfino qualsiasi suono non silenzioso. Per questa motivazione sono stati memorizzati anche degli audio con eventi particolarmente rumorosi, riuscendo a rendere il sistema finale più dinamico. Dopo aver analizzato tutti i file wav selezionati tramite l'utilizzo del software Audacity, si ottiene un dataset di 508 audio di fischi e 455 audio di rumori, per un totale di 963 audio. In sintesi, i fischi memorizzati in questo dataset provengono da esemplari di delfini tursiopi differenti, in termini di sesso e popolazione, oltre al fatto che sono stati registrati in ambienti e dinamiche differenti. Questo dettaglio è fondamentale, perché come riportato da vari studi, ad esempio l'Articolo [24], i tursiopi emettono un fischio univoco, con cui riescono a identificarsi tra loro stessi. Ciò implica che avere un'enorme quantità di registrazioni, ma provenienti dagli stessi delfini, non generalizzerebbe il sistema, ottenendo una staticità che ovviamente non è desiderata. Oltre questa informazione, sono state effettuate delle statistiche sui fischi selezionati, poi prese in considerazione per alcune scelte effettuate e descritte nel capitolo 2.3 e 2.6. La durata media di un fischio è di 0.58 secondi, il fischio più lungo ha una durata di 1.7 secondi e, per scelta personale, il più corto ha una durata di 0.25 secondi, in quanto i più brevi sono stati scartati.

2.3 Pre-Processamento Dati

La fase di pre-processamento dei dati consiste nel migliorare la qualità del dataset, evidenziando al meglio le caratteristiche del fischio del delfino. Una corretta preparazione dei dati può fare la differenza per ottenere dei risultati molto efficienti.

2.3.1 Creazione spettrogramma

Ogni segnale audio è convertito in uno spettrogramma mediante la FFT (Fast Fourier Transform), utilizzando la libreria Numpy [25]. La FFT è fondamentale per tradurre un segnale dal dominio del tempo a quello della frequenza, permettendo così di analizzare la distribuzione delle frequenze presenti nel segnale e facilitando l'identificazione di caratteristiche specifiche. È stata adottata una finestra temporale di 2048 campioni, con una sovrapposizione (overlap) del 50%, quindi ogni segmento di 1024 campioni è condiviso con il precedente. La scelta della dimensione della finestra determina la lunghezza dei segmenti del segnale su cui verrà applicata individualmente la FFT. Per ridurre alcune perdite del segnale vengono utilizzate finestre di Hanning che attenuano le frequenze laterali, riducendo il fenomeno di "leakage" [26-27]. Il numero di punti utilizzati per la trasformata (nfft) è pari a 2048, ma viene selezionata solo la prima metà dei risultati, in quanto questa operazione restituisce dei valori complessi e nella seconda metà si avrebbero solo i valori complessi coniugati, causando una ridondanza. Si calcola l'ampiezza dei valori ottenuti e poi vengono convertiti in decibel (dB). Osservando gli spettrogrammi delle registrazioni appartenenti al dataset etichettate come delfini, si sceglie di selezionare solo le frequenze tra 5 kHz e 25 kHz. Dalla teoria dei segnali e degli elementi principali di elettronica, per fare ciò viene proposto generalmente un filtro passabanda. Tuttavia, osservando i diagrammi di Bode, Figura 8, è possibile osservare come anche parte delle frequenze desiderate vengano coinvolte nell'operazione di filtraggio, subendo una parziale attenuazione. Inoltre, i segnali che si trovano nelle vicinanze ma esterni all'intervallo stabilito, non verrebbero tagliati del tutto e, riferendosi sempre ai diagrammi di Bode, si può notare come l'ampiezza del segnale venga ridotta sempre di più con l'allontanarsi dall'intervallo prestabilito.

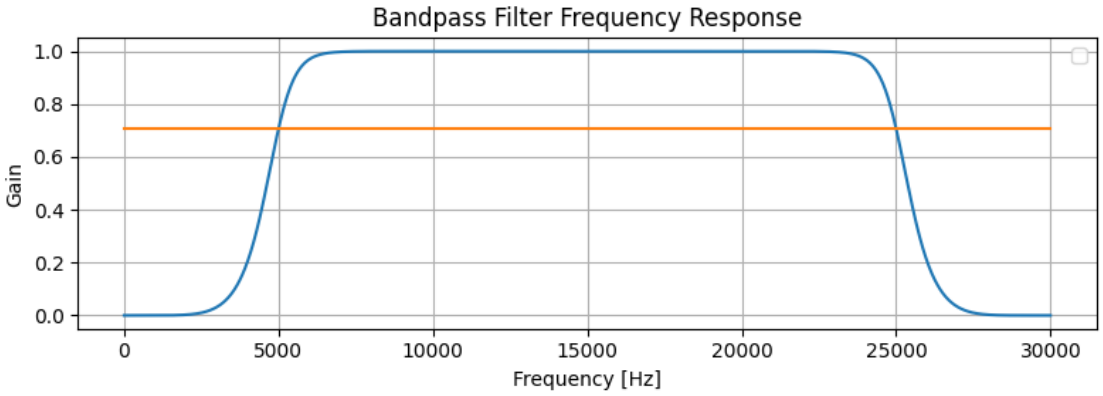

Figura 8. *Diagramma di Bode filtro passabanda 5 kHz -25 kHz*

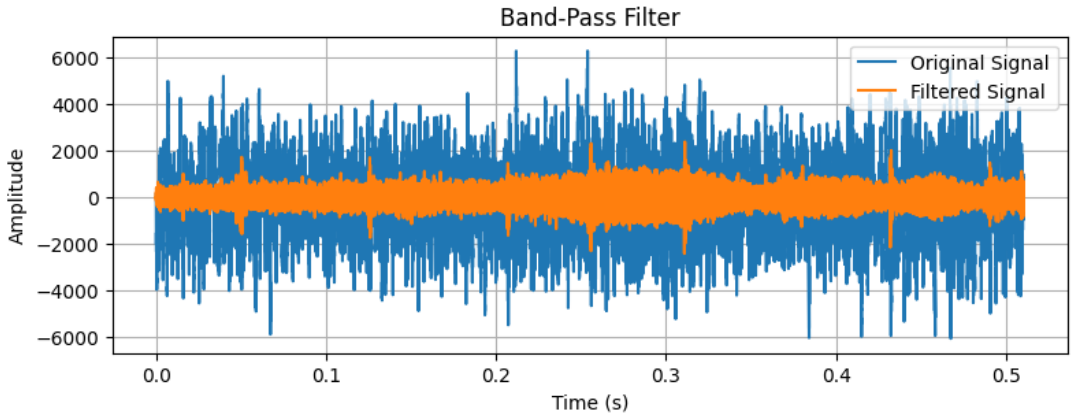

Figura 9. *Effetto filtro passabanda su un segnale*

L'opzione ideale prevede l'utilizzo di un filtro capace di effettuare un taglio netto delle frequenze interessate, ciò è possibile con una semplice selezione numerica. Dopo aver eseguito le diverse istruzioni per la creazione dello spettrogramma, ciò che si ottiene è un'immagine, rappresentata sotto forma di matrice. Per tagliare le frequenze desiderate, si decide di selezionare le righe della matrice che raffigurano i segnali tra i 5 kHz e i 25 kHz. Adottando questo metodo, i segnali destinati all'analisi non subiscono alcuna attenuazione.

Al fine di diminuire il tempo computazionale ed evitare una ridondanza di informazioni, gli spettrogrammi vengono generati in scala di grigio anzichè a colori (Figura 10). Sebbene vengano utilizzate immagini 2D, per questo sistema due dimensioni risultano pienamente adeguate.

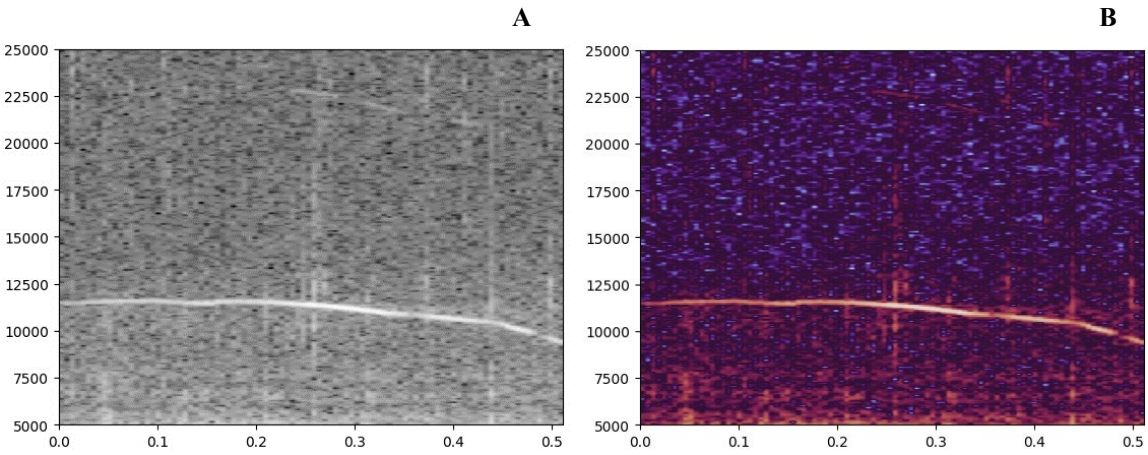

Figura 10. *Immagine A: Spettrogramma fischio in scala di grigi. Immagine B: Spettrogramma fischio in RGB.*

2.3.2 Frammentazione audio

Facendo riferimento alle statistiche, è stato deciso di catturare e frammentare le registrazioni in segmenti audio della durata di 0.6 secondi avanzando ogni 0.2 secondi, avendo quindi due terzi di sovrapposizione con l'immagine precedente, ovvero il 66.6%. Questo approccio viene adottato per tutte le registrazioni su cui viene poi applicata la rete neurale per l'identificazione del fischio del delfino. Tale processo permette di individuare approssimativamente l'inizio e la fine del fischio, ma, per maggiore accuratezza o scopi differenti, questi valori potrebbero essere modificati. Considerando ciò, si è deciso di effettuare lo stesso passaggio anche al dataset che successivamente viene utilizzato per l'addestramento della rete neurale, sulle registrazioni con una durata oltre un secondo. Tale scelta è stata effettuata sia per avere un dataset più numeroso, che per facilitare il funzionamento della rete neurale. Considerando il processamento di un fischio specifico di durata superiore al secondo all'interno di una registrazione, la frammentazione descritta in precedenza non analizza il fischio in maniera integrale, ma lo scompone in più

immagini, contenenti ognuna una parte significativa che viene successivamente analizzata. Questo implica che la rete neurale deve essere in grado di rilevare il fischio del delfino anche se mostrato a pezzi. Per cui, prendendo come esempio il fischio in Figura 11 della durata di un secondo, viene suddiviso in 3 frame di 0.6 secondi che traslano di 0.2 secondi rispetto al precedente come in Figura 12. Inoltre, per il dataset di addestramento viene utilizzato anche l'audio originale, per cui dal fischio sottostante si ricavano 4 spettrogrammi. Facendo questa frammentazione, dal dataset con 508 audio di fischi e 455 audio di rumori, si ottengono 596 immagini di fischi e 923 immagini di rumori, per un totale di 1519 spettrogrammi.

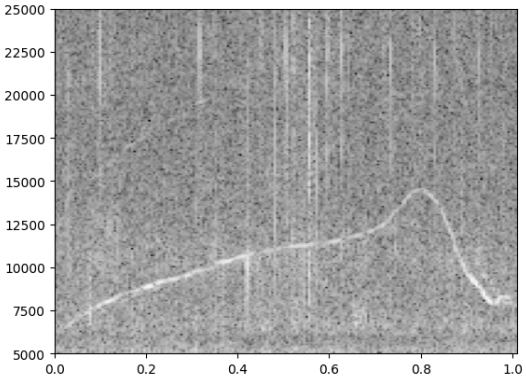

Figura 11. *Fischio della durata di 1 s.*

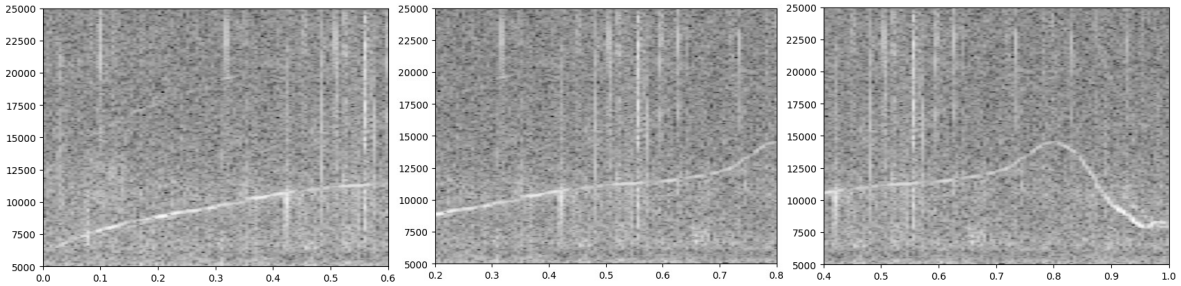

Figura 12. *Fischio di un secondo spezzettato in tre immagini da 600 ms. ciascuna, sovrapposte di 400 ms.*

2.3.3 Filtro immagine

Sebbene il fischio del delfino sia generalmente ben evidente nella maggior parte degli spettrogrammi, una rete neurale analizza i dettagli dell'intera immagine, inclusi il rumore di sottofondo ed altre caratteristiche che invece compaiono in primo piano. Queste ultime in particolare possono confondere la rete non permettendo alla stessa una corretta classificazione, restituendo di conseguenza falsi positivi e falsi negativi che ovviamente sono indesiderati. Pertanto, si è deciso di utilizzare il filtro Sobel, uno degli operatori di convoluzione più impiegati per l'identificazione dei bordi. Questo filtro si basa sull'approssimazione del gradiente dell'intensità dell'immagine che gioca un ruolo fondamentale nella rilevazione dei contorni perché sono spesso caratterizzati da rapidi cambiamenti nell'intensità dei pixel. Una grande ampiezza del gradiente indica un cambiamento significativo e, quindi, la presenza di un bordo. Il funzionamento di questo filtro si fonda su una matrice di convoluzione, anche conosciuta come kernel o maschera. Al fine di evidenziare sufficientemente le caratteristiche principali dell'immagine, si fa uso di una matrice 5x5, riportata in Figura 13.

-1	-2	0	2	1
-4	-8	0	8	4
-6	-12	0	12	6
-4	-8	0	8	4
-1	-2	0	2	1

Figura 13. *Matrice di convoluzione per filtro Sobel verticale*

Applicato ad uno spettrogramma, può essere particolarmente utile per rilevare eventi acustici orizzontali, come il fischio di un delfino in quanto, come possibile notare in Figura 14, esso appare come una linea orizzontale, rappresentando un suono che rimane approssimativamente costante in frequenza per un certo periodo di tempo. Il filtro Sobel verticalmente orientato evidenzia questi bordi orizzontali rilevando rapidi cambiamenti

nell'intensità da un pixel all'altro in direzione verticale. Una volta applicato il filtro, gli oggetti orizzontali, come il fischio del delfino, si manifestano come regioni di elevata intensità o "bordi", mentre gli oggetti verticali, come segnali a frequenze molto alte, vengono trascurati. Ciò significa che se in uno spettrogramma è presente un fischio sovrapposto a degli eventi acustici verticali molto rumorosi, quest'ultimi vengono rimossi mentre il primo viene evidenziato, come nel caso in Figura 15.

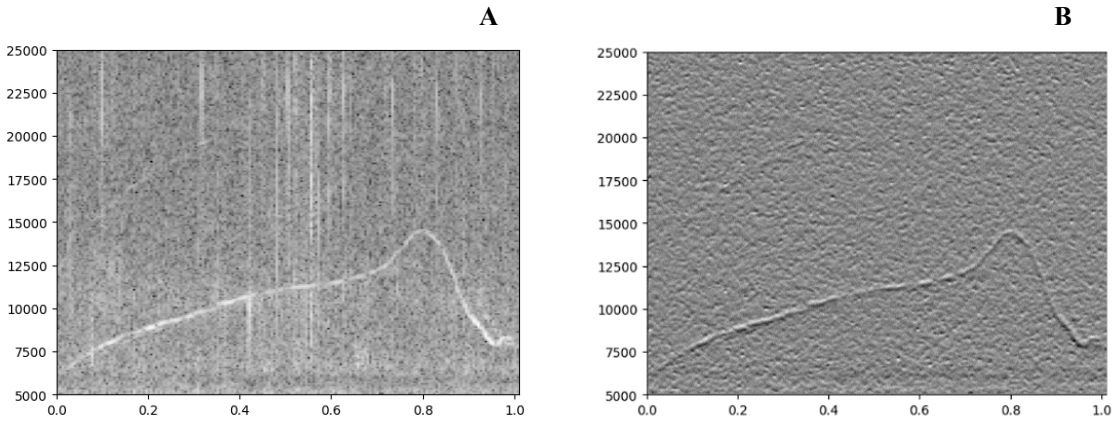

Figura 14. *Immagine A: Spettrogramma del fischio di un delfino. Immagine B: Stesso spettrogramma applicando il filtro Sobel.*

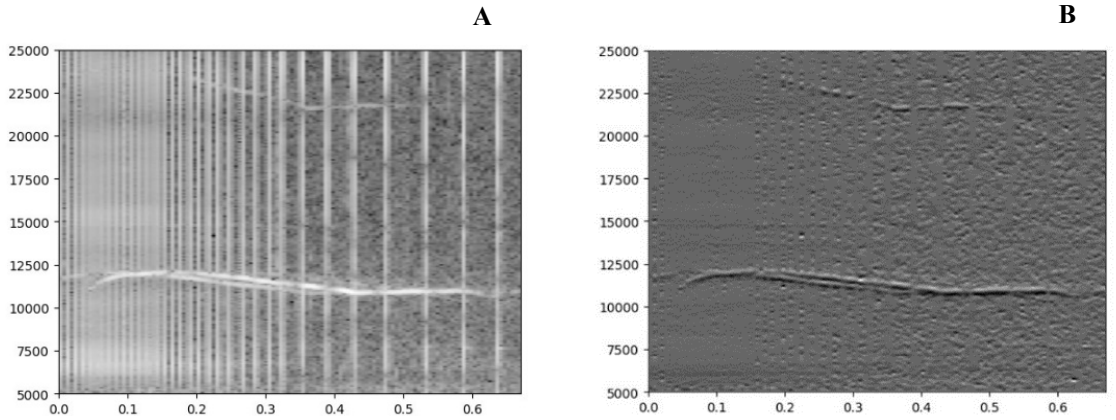

Figura 15. *Immagine A: Spettrogramma fischio delfino con molto rumore verticale. Immagine B: Stesso spettrogramma applicando il filtro Sobel.*

Tale passaggio ha permesso alla rete di capire meglio qual è l'oggetto da essere rilevato nell'immagine. Con questo filtro, le differenze tra le immagini appartenenti alla classe degli esempi negativi non sono molte, per cui è risultato opportuno addestrare la rete anche con dei rumori i cui spettrogrammi potrebbero essere confusi con il fischio di un delfino, come nella Figura 16.

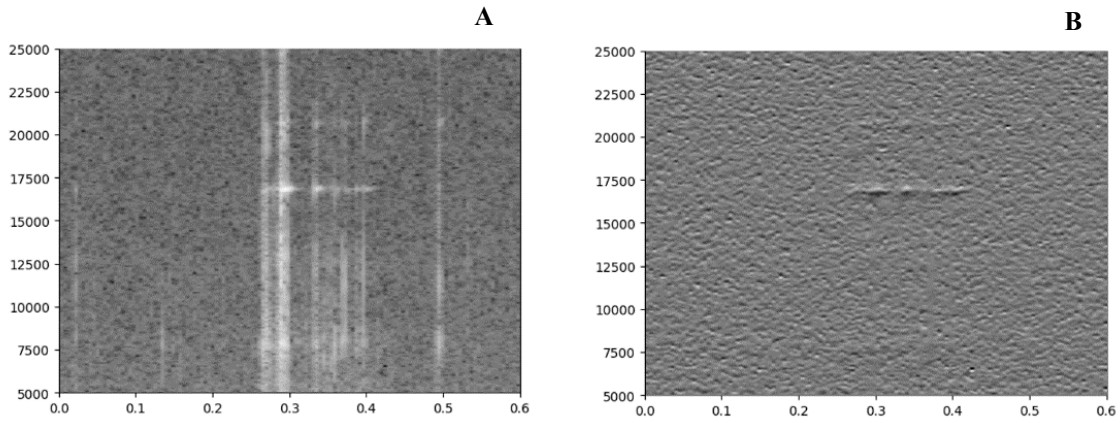

Figura 16. *Immagine A: Spettrogramma di un rumore orizzontale. Immagine B: Stesso spettrogramma applicando il filtro Sobel.*

2.4 Rete Neurale Profonda

Una rete neurale profonda DNN (Deep Neural Network) è un tipo di rete neurale artificiale composto da più strati di nodi (neuroni) interconnessi.

2.4.1 Rete Neurale Convoluzionale

Una rete neurale convoluzionale (CNN) è un'architettura di rete che impara direttamente dalle immagini. Una CNN è composta da più strati che hanno il compito di elaborare un input al fine di produrre un output, rilevando determinate caratteristiche come oggetti, bordi, testi ed altri dettagli. In questo, gioca un ruolo fondamentale il layer convoluzionale, ovvero uno dei blocchi principali di questo genere di reti. L'operazione principale di questo tipo di layer è quella di applicare un filtro a parti sovrapposte dell'input, producendo una mappa delle caratteristiche (feature map). Tale filtraggio avviene tramite una matrice di convoluzione (o kernel), normalmente di dimensioni 3x3 o 5x5 che viene fatta scorrere sull'intera area dell'input. Come mostrato in Figura 17**,** la convoluzione non riguarda direttamente l'intera immagine, ma vengono analizzati gruppi di pixel per estrarne le caratteristiche.

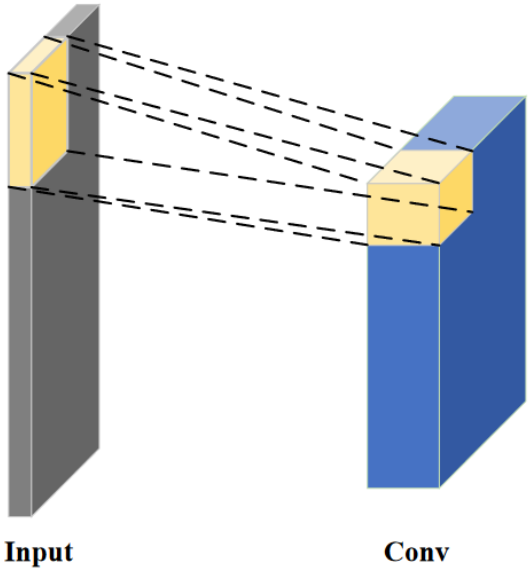

Figura 17. *Layer Convoluzionale*

Tutti i layer compresi tra il primo e l'ultimo vengono chiamati "layer nascosti" e la loro quantità rappresenta la profondità. Per questa motivazione, quando si utilizzano queste reti si parla di "deep learning" che rappresenta un ramo importante del machine learning. Come dimostrato nell'Articolo [28], la profondità della rete neurale convoluzionale con filtri piccoli (ad esempio di dimensioni 3x3) è direttamente proporzionale alla sua accuratezza per una classificazione di immagini. Questo dettaglio viene preso in considerazione per la scelta dell'architettura della rete neurale.

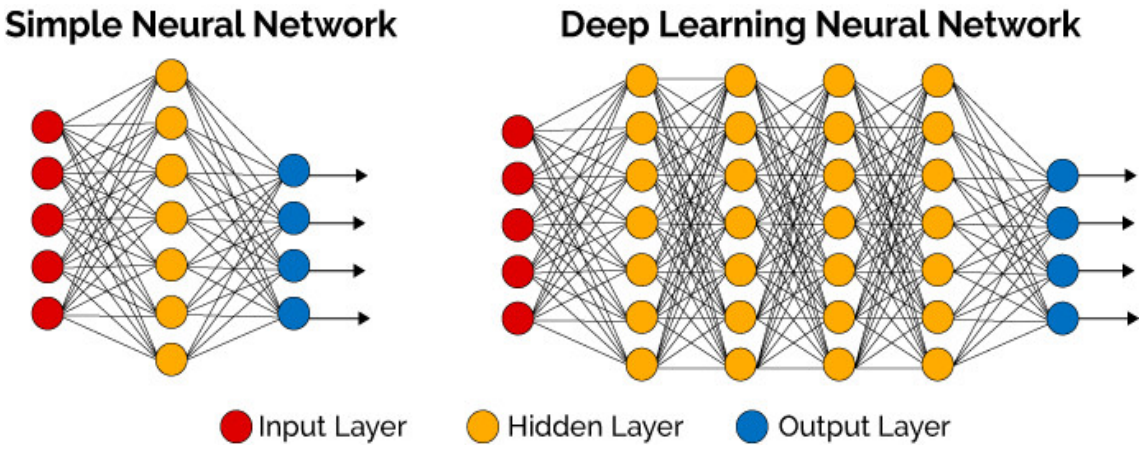

Figura 18. *Rete neurale profonda*

A seguito di un layer convoluzionale, è comune inserire quello che viene chiamato layer di pooling. Il suo scopo è quello di ridurre la dimensione dei dati, restituendo un unico neurone da un gruppo di neuroni ottenuti come output dal layer precedente [29]. Generalmente, i più utilizzati sono il "max pooling layer" ed il "average pooling layer" che, da un gruppo di pixel, prendono in considerazione rispettivamente il valore massimo e la media, come mostrato in Figura 19. Normalmente viene effettuato tramite una matrice 2x2 che viene fatta traslare, dimezzando la dimensione dell'input.

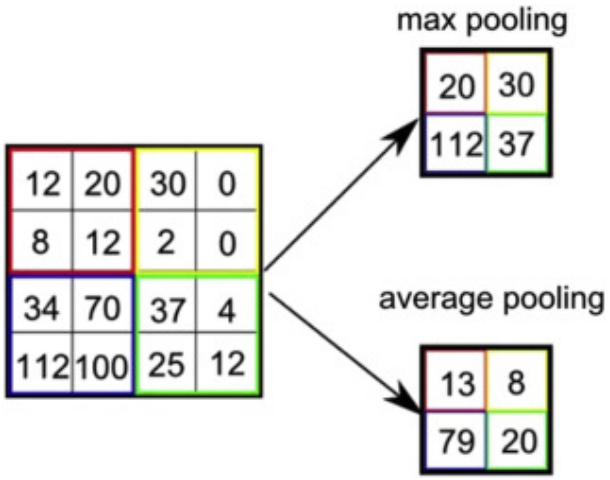

Figura 19. *Operazione di pooling*

2.4.2 Flatten e Dense

Utilizzando layer convoluzionali e di pooling, la dimensione dell'output si conserva, per cui, dando in input immagini 2D, alla fine si ha un output ugualmente 2D. Poiché l'uscita finale della rete neurale deve essere un solo numero, viene utilizzata l'operazione di "flatten", che converte l'input multidimensionale, come ad esempio un'immagine, in output unidimensionale, ovvero un vettore, spianando i dati in modo che possano essere elaborati dai layer successivi. Ad esempio, in un'immagine di dimensioni 3x3 pixel, il flatten prende questa matrice e la trasforma in un vettore di lunghezza 3*3=9. In questo modo, l'immagine viene rappresentata come un vettore piatto di pixel, che può quindi essere fornito ai layer densamente connessi o ad altri layer della rete neurale.

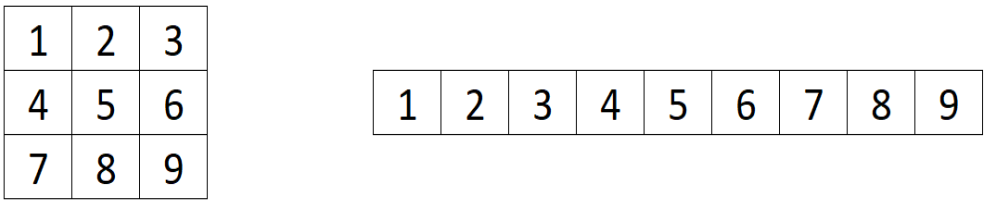

Figura 20. *Operazione di Flatten*

In sintesi, il layer Flatten è un componente essenziale nelle CNN quando si desidera collegare i layer convoluzionali ai layer completamente connessi convertendo una matrice multidimensionale in un vettore monodimensionale. Il layer completamente connesso, detto “dense layer”, collega ciascun neurone del layer di input a ciascun neurone del layer di output, creando quindi una connessione completa tra di essi. Il layer "dense," noto anche come "fully connected layer" o "layer completamente connesso," collega ciascun neurone nel layer di input a ciascun neurone nel layer di output, creando una connessione completa tra di essi.

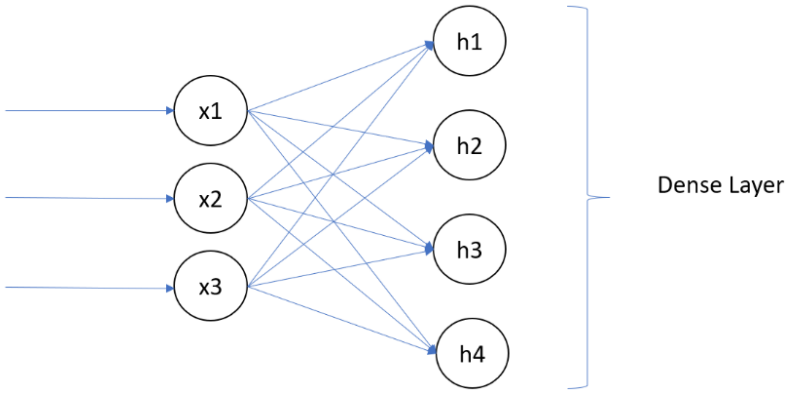

Figura 21. *Dense Layer*

2.4.3 Funzioni di attivazione

Una rete neurale artificiale può apprendere schemi complessi di dati con l'aiuto di funzioni di attivazione che hanno il compito di scegliere quali segnali inviare al neurone successivo [30]. Una delle sue caratteristiche più importanti è la capacità di aggiungere una non linearità alla rete neurale. La funzione ReLU o (Rectified Linear Unit) è la funzione di attivazione comunemente più utilizzata ed è definita come $f(y) = max(0, y)$. I principali motivi che rendono la ReLU la funzione di attivazione più popolare ed ampliamente utilizzata sono:

1. **Non linearità:** l'introduzione di non linearità nella rete neurale è fondamentale per consentire l'apprendimento da errori e l'adattamento a funzioni complesse.
2. **Tempo computazionale:** A differenza di altre funzioni di attivazione ReLU è computazionalmente efficiente perché richiede solo una semplice operazione di soglia a 0. Questo rende l'addestramento delle reti più veloce.
3. **Mitigazione del problema di scomparsa del gradiente:** Mentre altre funzioni hanno il problema della "scomparsa del gradiente", dove i gradienti possono diventare molto piccoli rallentando l'apprendimento, la ReLU in molti casi aiuta a mitigare questo problema.

Tuttavia, uno svantaggio della ReLU è il problema delle "unità morte", dove alcuni neuroni possono a volte non attivarsi mai portando a porzioni inutili nella rete.

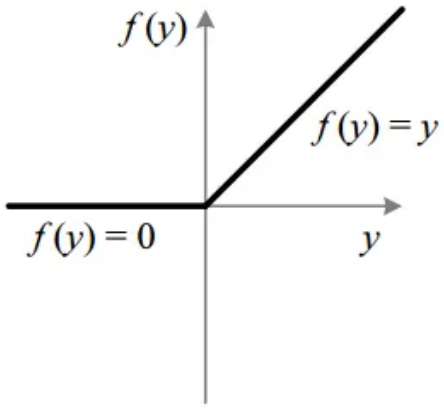

Figura 22. *Funzione di attivazione ReLu*

Un'altra funzione di attivazione è la "sigmoid". Tale funzione è definita come:

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

Le caratteristiche principali della sigmoid sono:

1. **Output tra 0 e 1:** La sigmoid prende un valore in input e lo trasforma in un valore tra 0 e 1, con output centrato in 0.5. Questa caratteristica la rende particolarmente utile per problemi di classificazione binaria, dove l'output può rappresentare una probabilità.
2. **Differenziabilità:** La sigmoid è una funzione continua e differenziabile, che la rende utilizzabile in algoritmi di ottimizzazione come la discesa del gradiente.
3. **Scomparsa del gradiente:** Un problema di questa funzione di attivazione è la "scomparsa del gradiente". Quando i valori di input sono molto grandi o molto piccoli, la derivata della funzione diventa molto vicina a zero. Ciò può causare che i pesi della rete aggiornino molto lentamente o smettano addirittura di aggiornarsi durante la fase di addestramento, rallentando questa fase.

A causa di alcuni di questi svantaggi, la funzione sigmoid è stata spesso sostituita da altre funzioni di attivazione come ReLU nelle reti neurali profonde. Tuttavia, la sigmoid è ancora utilizzata nell'output di reti per classificazione binaria.

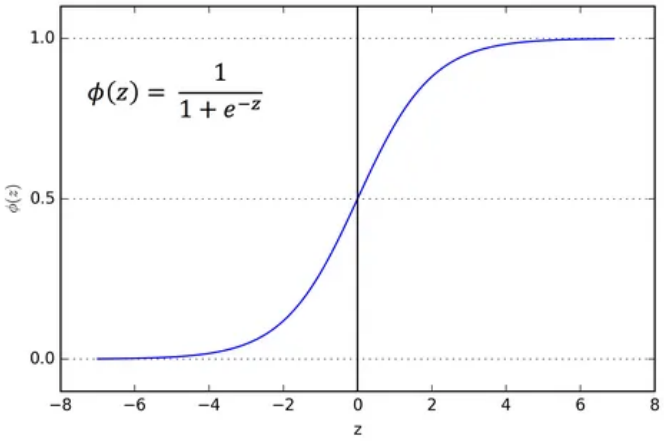

Figura 23. *Funzione di attivazione Sigmoid*

2.4.4 Architettura rete

La rete utilizzata è una rete neurale convoluzionale (CNN) selezionata tramite il framework Optuna. Inizia con un layer convoluzionale dotato di 32 filtri di dimensione 3x3 e una funzione di attivazione ReLU. Questo layer riceve in input immagini 2D di dimensione 224x224. Successivamente, c'è un layer di pooling che dimezza le dimensioni spaziali dell'output precedente in ogni direzione, attraverso l'operazione di max pooling con una finestra 2x2. Poi, per un totale di quattro volte consecutive, si ha un layer convoluzionale con 64 filtri 3x3 e attivazione ReLU, seguito da un layer di max pooling 2x2. Dopo questi layer convoluzionali e di pooling, la struttura dell'output viene appiattita attraverso un'operazione di Flatten, convertendo l'output 2D in un array. Infine, la rete ha un dense layer con 512 neuroni e funzione di attivazione ReLU, che si conclude con un altro dense layer con un solo neurone e funzione di attivazione Sigmoid, comunemente utilizzata per problemi di classificazione binaria. Non viene inserita una soglia per avere un valore binario come output poiché si desidera visualizzare il valore ottenuto dalla rete, compreso tra 0 e 1. Parlando in termini di percentuali, questo valore indica la probabilità di appartenenza alla classe 1, ovvero quella dei delfini. Si preferisce questo approccio in quanto si desidera avere la possibilità di interpretare i dati, potendo valutare anche eventuali combinazioni di risultati consecutivi.

Come citato all'inizio di questo paragrafo, la rete neurale è stata selezionata tramite Optuna che ha ottimizzato gli iperparametri una volta impostati i range di valori desiderati. Basandosi sulle esperienze ed esperimenti trovati in letteratura e prendendo in considerazione le osservazioni effettuate nei paragrafi precedenti, una possibile architettura generata e testata da Optuna è di questo tipo:

- Da 3 a 5 layer convoluzionali con un numero di filtri tra [16, 32, 64, 128, 256, 512, 1024], kernel 3x3, funzione di attivazione ReLU, seguiti ognuno da un maxPooling layer 2x2. Il primo layer ha un input di 224x224.
- Un'operazione di Flatten.
- Uno o due Dense layer con numero di neuroni compreso tra [64, 128, 256, 512, 1024], funzione di attivazione ReLU.
- Dense layer (output) con un neurone, funzione di attivazione Sigmoid.

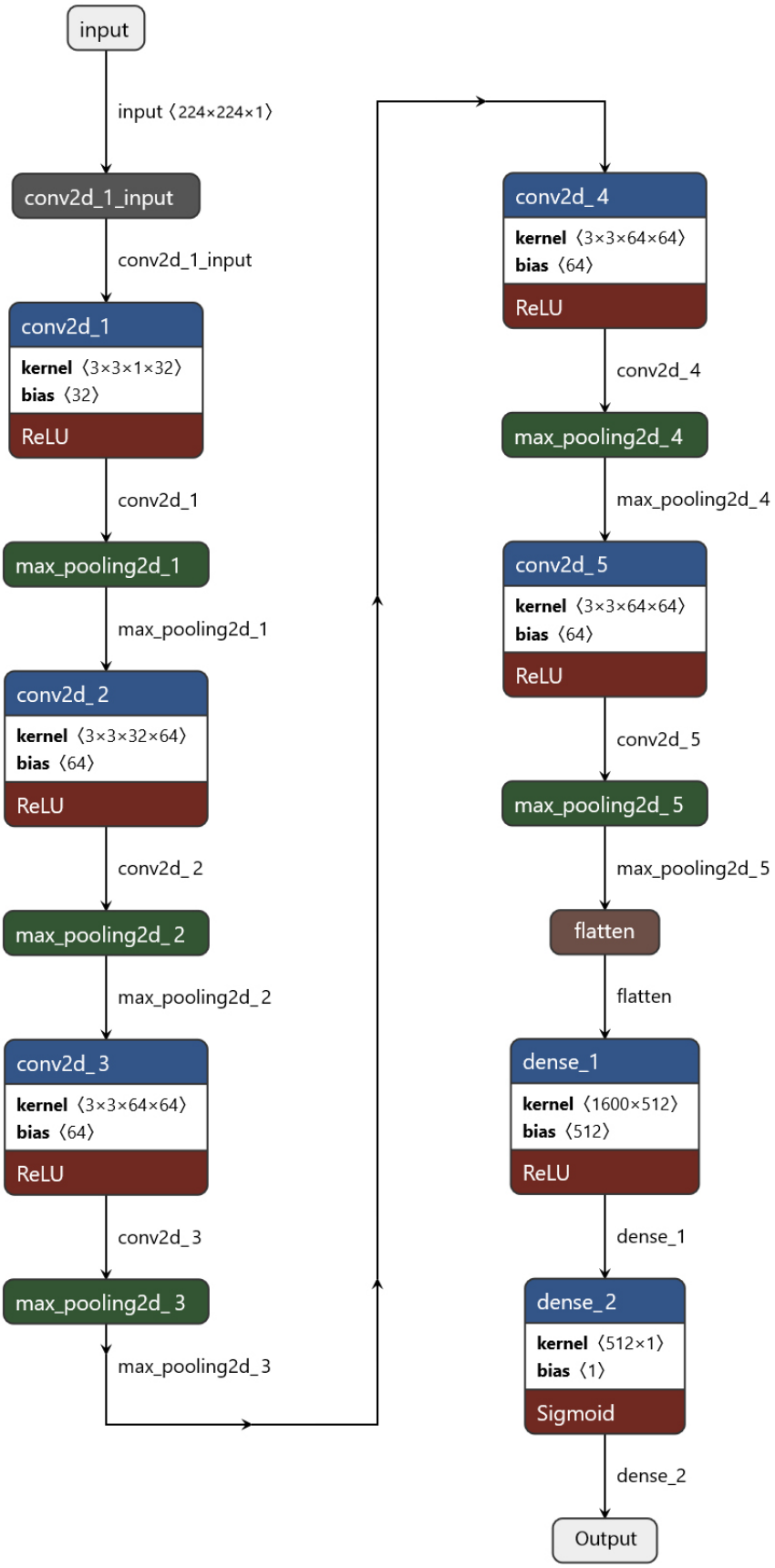

Figura 24. *Architettura CNN utilizzata*

2.5 Addestramento

In questa fase viene descritto il metodo e le scelte effettuate per l'addestramento della rete neurale.

2.5.1 Suddivisione Dataset

Dopo aver effettuato l'operazione di etichettatura e di filtraggio delle immagini, come specificato precedentemente si arriva ad ottenere 596 immagini di fischi e 923 immagini di rumori, per un totale di 1519 spettrogrammi memorizzati in due cartelle differenti. I file vengono caricati all'interno di un unico array, affiancati da un'etichetta 0 o 1 per indicarne la classe. Per cui, ogni elemento è costituito da un'immagine e in questo caso un numero binario. Al fine di addestrare la rete l'intero dataset viene suddiviso in train, validation e test, con il set di validation che è facoltativo. Il set di train generalmente è il più grande in quanto viene utilizzato per addestrare la rete, che impara basandosi su questi dati. Il set di validazione, non sempre necessario, viene utilizzato per valutare l'addestramento e può essere preso in considerazione per evitare l'overfitting e usare tecniche di ottimizzazione per l'allenamento. Invece, il set di test, solitamente il più piccolo, non viene mai considerato durante l'addestramento, ma viene utilizzato per valutare la rete, calcolando i valori dell'accuratezza e delle varie metriche di valutazione del modello. Poiché il dataset a disposizione è piuttosto piccolo, si sceglie di non utilizzare il set di valutazione. In tal caso, solitamente si utilizza l'80%-90% per il train ed il resto per il test.

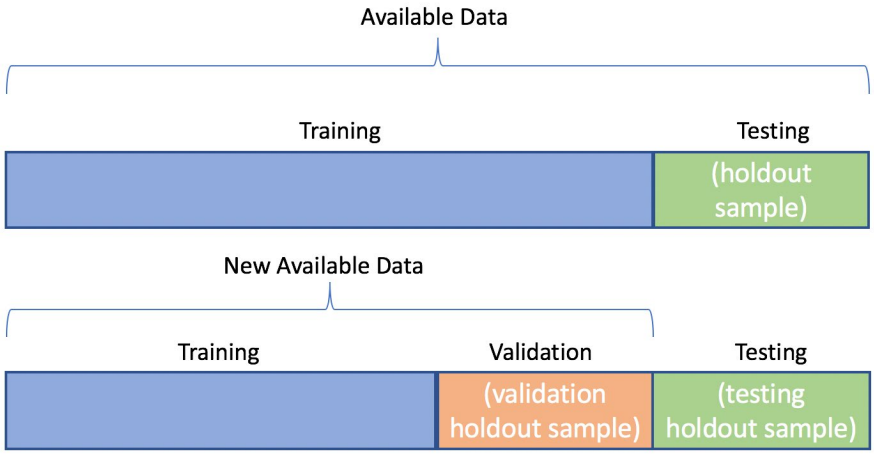

Figura 25. *Suddivisione dataset*

2.5.2 Fold Cross-Validation

Una volta stabilite le percentuali di suddivisione tra train e test, i file selezionati per ogni set vengono scelti in modo casuale. Inoltre, potrebbero esserci alcuni esempi di immagini nel dataset che possono risultare complicate alla rete e possono essere la causa di qualche falso positivo e falso negativo, specialmente se sono nel dataset di test, oppure possono peggiorare la qualità e quindi l'accuratezza se fanno parte del dataset di train. Per cui, molto probabilmente si avrebbero dei risultati differenti se i file scelti per il train ed il test cambiassero. È possibile rispondere a questo quesito con la K-Fold Cross-Validation [31-32], specialmente utile quando si ha un dataset abbastanza piccolo. Questa tecnica consiste nell'addestrare K modelli della stessa architettura e con la stessa modalità, ma con dataset diversi. Viene suddiviso l'intero dataset in K parti dette fold, utilizzando K-1 fold per l'addestramento di un modello e la restante per il test. Per ogni modello, viene utilizzata una fold di test differente come mostrato in Figura 26 che indica il funzionamento di una 10-Fold Cross-Validation.

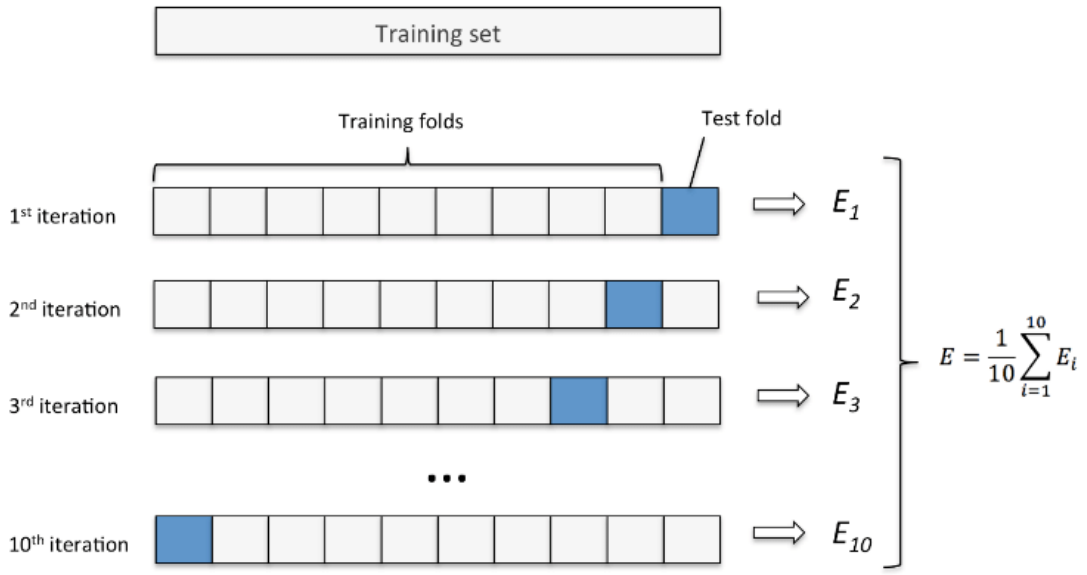

Figura 26. *10-Fold Cross-Validation*

2.5.3 Learning Rate

Il "learning rate" o tasso di apprendimento è un iperparametro che determina quanto rapidamente un modello di apprendimento automatico si adatta ai dati durante l'addestramento [33]. La scelta del learning rate viene effettuata con le seguenti caratteristiche:

- **Dimensione del Passo:** Un learning rate elevato potrebbe causare passi troppo grandi nel processo di ottimizzazione con il rischio di andare in overfitting. D'altra parte, un learning rate troppo basso potrebbe causare passi molto piccoli, rendendo l'addestramento abbastanza lento.

- **Decadimento:** È possibile utilizzare tecniche di "decadimento del learning rate" dove il learning rate diminuisce nel tempo. Questo può aiutare l'algoritmo a fare passi grandi all'inizio quando si è lontani dal minimo e passi più piccoli avvicinandosi al minimo.

- **Adaptive Learning Rate:** Ci sono algoritmi di ottimizzazione come Adam, Adagrad e RMSprop che adattano il learning rate durante l'addestramento. Questi metodi regolano il learning rate in base alla storia dei gradienti.

In sintesi, la scelta del learning rate determina quanto rapidamente ed efficacemente un modello si adatta ai dati durante l'addestramento influenzando la qualità e la velocità del processo di addestramento come rappresentato in Figura 27.

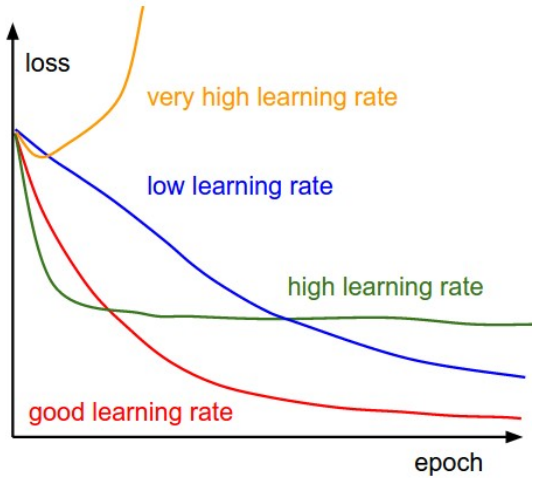

Figura 27. *Variazione learning rate*

2.5.4 Adam

Il controllore "Adam" (Adaptive Moment Estimation) è un algoritmo di ottimizzazione e viene impiegato come un'alternativa alla discesa del gradiente per aggiornare i pesi della rete neurale basandosi sui dati di addestramento [34]. La particolarità di Adam sta nella sua capacità di adattare i tassi di apprendimento per ciascun peso della rete traducendosi in prestazioni notevolmente migliori nei problemi con gradienti molto rumorosi o limitati. Adam unisce le migliori proprietà di altri algoritmi di ottimizzazione, come AdaGrad e RMSprop rendendolo efficace in molte applicazioni.

2.5.5 Early Stopping

L'early stopping è una tecnica utilizzata durante l'addestramento di reti neurali per evitare l'overfitting. Il suo scopo è di verificare se la performance generalmente sul set di validazione smette di migliorare o inizia a peggiorare per un certo numero di epoche consecutivo, in tal caso interrompe l'addestramento [35]. Il "certo numero di epoche" è definito come "pazienza". Inoltre, consente di salvare il modello nel punto in cui la metrica selezionata aveva il valore migliore prima di iniziare a peggiorare. In pratica l'early stopping permette di avere modelli che generalizzano meglio su nuovi dati evitando l'overfitting e risparmiando tempo e risorse computazionali.

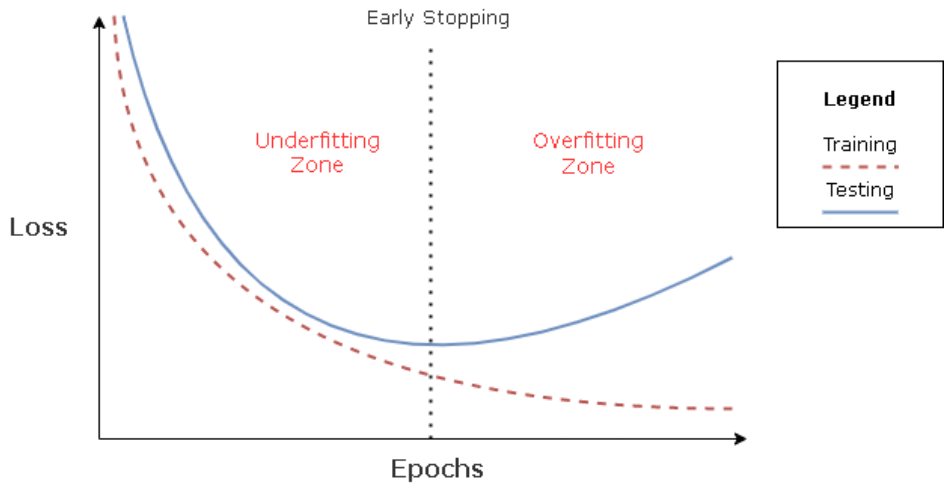

Figura 28. *Early Stopping*

2.5.6 Peso delle classi

Il "peso delle classi" nelle reti neurali si riferisce all'importanza assegnata a ciascuna classe durante la fase di addestramento. Quando si addestra una rete neurale su un set di dati, può capitare che alcune classi siano rappresentate da molti più esempi rispetto ad altri portando di fatto a predire in modo eccessivo la classe maggioritaria a scapito delle classi minoritarie. Un'altra situazione che può capitare è quella di voler forzare volutamente la rete a predire una o più classi rispetto ad altre. Per affrontare questo problema, si possono utilizzare i pesi delle classi. In pratica, durante l'addestramento si assegnano dei pesi mirati per ciascuna classe aiutando la rete a focalizzare di più l'apprendimento in proporzione ai pesi assegnati. La determinazione del peso esatto per ciascuna classe può dipendere dal problema specifico e dalla distribuzione dei dati. Per il problema di rilevamento di delfini, si preferisce avere uno sbilanciamento sui falsi negativi piuttosto che sui falsi positivi. Per questa motivazione, si decide di dare un peso di 1.5 alla classe dei rumori ed un peso di 1.0 alla classe dei fischi.

2.6 Etichettatura

In questa sezione viene illustrata la strategia per il posizionamento delle etichette, dichiarando inizio e fine del fischio del delfino dando una interpretazione ai risultati ottenuti dalla rete neurale.

2.6.1 Interpretazione Risultati

Poiché ogni frame di 200 millisecondi viene processato tre volte dalla rete neurale (avendo frame di 600 millisecondi), teoricamente la rete dovrebbe essere in grado di restituire almeno tre valori in percentuale superiori al 50% di fila. Facendo le eventuali considerazioni, per diminuire il numero di falsi positivi e non aumentare troppo il numero di falsi negativi, sono state effettuate delle scelte analizzando i valori ottenuti, selezionando quali gruppi di numeri devono essere considerati come delfini. Dopo aver effettuato più tentativi su diverse registrazioni, si è deciso di prendere come positivi due possibili combinazioni di percentuali:

1. Almeno due percentuali sopra al **70%** di fila;
2. Una percentuale sopra il **90%** preceduta o che precede un'altra almeno sopra il **50%**.
3. Una percentuale sopra il **99.5%.**

Tali valori sono stati scelti analizzando le risposte della rete neurale, utilizzando quindi un approccio data-driven. Infatti, si nota come quasi tutti i fischi rilevati, abbiano una percentuale sopra il 95%. A volte può accadere di avere una percentuale sopra il 99.5% affiancata da valori che non oltrepassano nemmeno il 10%. L'ultimo caso non è molto coerente con la considerazione fatta in precedenza, ovvero che ogni fischio viene analizzato almeno in tre frame diversi. Tuttavia, si tratta di una percentuale troppo alta per non essere presa come positiva, quindi si decide di considerare un valore del genere come fischio.

2.6.2 Posizionamento Etichette

L'interpretazione dei dati descritta nel paragrafo precedente riguarda una trasformazione binaria personalizzata delle percentuali, ottenute dalla rete neurale. Una volta ottenuti tali valori, è necessario reinterpretarli per definire temporalmente l'inizio e la conclusione del fischio del delfino. Utilizzando ancora un approccio data-driven, basandosi quindi sulle sperimentazioni, ma affidandosi anche alla logica, si decide di procedere nel seguente modo: il fischio inizia 100 ms dopo il primo rilevamento, mentre termina 300 ms dopo la prima immagine in cui non viene più rilevato il fischio. Queste decisioni sono state prese tenendo anche in considerazione che si desidera selezionare l'intero fischio, cercando di non escluderne l'inizio o la fine. Come è possibile osservare in Figura 29, questa rete non sarebbe in grado di contare il numero di fischi, perché se alcuni di essi fossero molto ravvicinati, questi verrebbero identificati come un unico fischio. In realtà, considerando il vero scopo di questo progetto, tale problematica si rileva trascurabile, in quanto il vero obiettivo è di riuscire a rilevare la presenza di un delfino. Questo dettaglio viene quindi sorvolato.

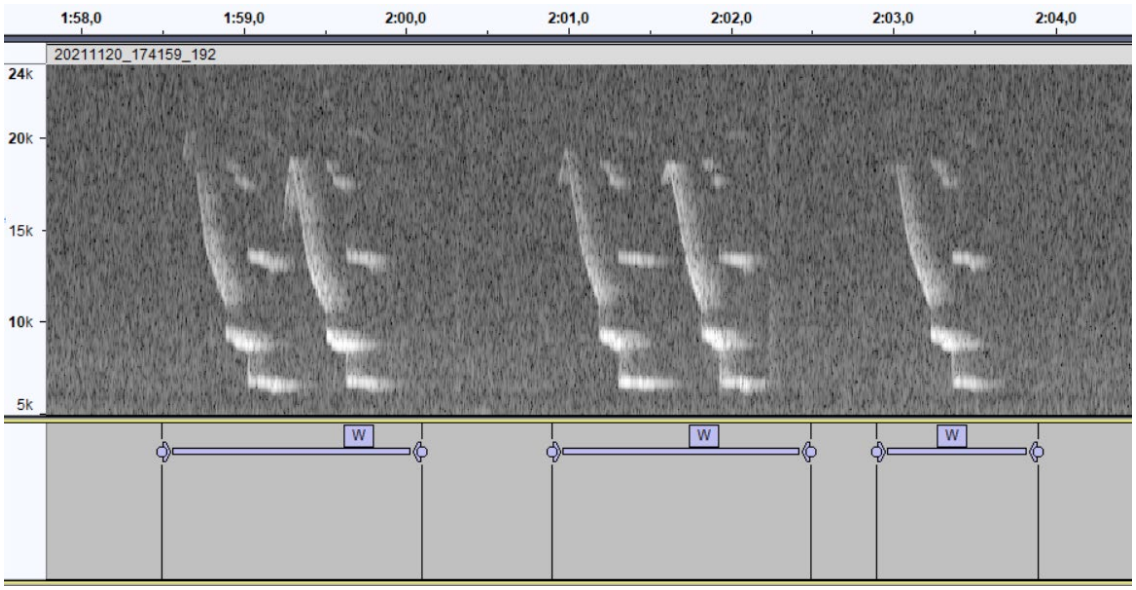

Figura 29. *Posizionamento etichette*

2.7 Strumenti

In questa sezione si espongono gli strumenti hardware e software utilizzati per la realizzazione di questo sistema.

2.7.1 Keras

Keras è una libreria open-source di deep learning scritta in Python. È stata progettata per consentire una sperimentazione veloce con reti neurali e si distingue per la sua facilità d'uso e la sua modularità. La filosofia di base dietro Keras è di essere user-friendly, modulare ed estensibile. Dal suo sviluppo è diventato uno degli strumenti di deep learning più popolari e ampiamente adottati nella comunità di ricerca e nello sviluppo. Tuttavia, con l'evoluzione e la crescita di TensorFlow, Keras è stato incorporato in TensorFlow stesso come interfaccia di alto livello. Ciò che rende Keras particolarmente attraente per molti sviluppatori e ricercatori è la sua semplicità. In questo progetto è stata adoperata la versione 2.12.0 per la progettazione di modelli di reti neurali [36].

2.7.2 TensorFlow

TensorFlow è un framework open source di apprendimento automatico sviluppato da Google [37]. È diventato uno degli strumenti più popolari e ampiamente utilizzati nel campo del deep learning e dell'apprendimento automatico grazie alle sue capacità potenti e flessibili. Questa architettura permette di sfruttare appieno le risorse hardware per accelerare i calcoli, come la GPU. TensorFlow supporta una vasta gamma di algoritmi e modelli di apprendimento automatico, rendendolo adatto a una varietà di compiti, da problemi di classificazione a quelli di regressione. Al suo interno contiene anche la libreria Keras, impiegata per la progettazione della rete neurale. Per questo progetto è stata utilizzata la versione 2.12.0.

2.7.3 Optuna

Optuna è un framework open source progettato per ottimizzare gli iperparametri dei modelli di machine learning. Nel funzionamento di Optuna, inizialmente si definisce lo spazio di ricerca per gli iperparametri [38-39]. Questo implica di stabilire quali iperparametri si desidera ottimizzare e il loro possibile range o set di valori. Una volta che lo spazio di ricerca è stabilito, Optuna sperimenta sistematicamente diverse combinazioni di iperparametri. Questa combinazione viene poi utilizzata per addestrare e valutare un modello. Basandosi sui valori delle metriche di valutazione di una rete neurale e facendo riferimento sul registro storico dei tentativi, Optuna decide quali iperparametri testare nella successiva iterazione. Infine, Optuna è progettato per essere compatibile e integrarsi facilmente con molte delle popolari librerie di machine learning come TensorFlow e Keras. Questo lo rende uno strumento versatile, adatto a una vasta gamma di applicazioni e uno degli strumenti di riferimento per l'ottimizzazione di iperparametri nel mondo del machine learning. La versione utilizzata è la 3.3.0.

2.7.4 PyCharm

PyCharm è un ambiente di sviluppo integrato (IDE) creato da JetBrains, specificamente progettato per Python, ma supporta anche altri linguaggi di programmazione, anche per sviluppi di siti web. Una delle sue principali caratteristiche è l'evidenziazione della sintassi e l'autocompletamento intelligente durante la scrittura del codice. Gestire progetti più grandi o lavorare in team richiede anche una gestione efficace delle versioni del codice. PyCharm ha un'ottima integrazione con sistemi di controllo versione come Git, facilitandone il salvataggio di backup e lo storico del lavoro. Esistono due versioni di PyCharm: la Community Edition, che è gratuita e open source, e la Professional Edition, a pagamento, che offre caratteristiche aggiuntive, in particolare per lo sviluppo web e scientifico. Per questo progetto è stata utilizzata la versione 2023.1.1 della Community Edition. In definitiva, PyCharm è un IDE ricco di funzionalità che mira a migliorare la produttività degli sviluppatori Python, offrendo un set completo di strumenti in un unico pacchetto.

2.7.5 Audacity

Audacity è un software open-source per la registrazione e l'editing audio. Lanciato nel 1999 da Dominic Mazzoni e Roger Dannenberg e reso open source nel 2000 [40] è diventato uno degli strumenti audio gratuiti più popolari e ampiamente utilizzati a livello globale. Con esso è possibile registrare audio, convertire vecchi nastri e dischi in registrazione digitale, nonché editare file audio di vari formati. Una delle caratteristiche distintive di Audacity è la sua interfaccia a tracce. Inoltre, gli utenti possono visualizzare ogni traccia audio in modo visivo, il che consente una manipolazione precisa e dettagliata. Ciò è possibile sia con la visualizzazione della forma d'onda che dell'analisi in frequenza del segnale, utilizzata per l'analisi PAM manuale e per il controllo dello spettrogramma in specifici momenti temporali. La sua natura open-source ha anche portato una comunità attiva di sviluppatori e utenti che contribuiscono costantemente con nuove funzionalità, miglioramenti e supporto per gli utenti. La versione utilizzata per questo progetto è la 3.3.2.

2.7.6 Numpy

NumPy è una libreria per il linguaggio di programmazione Python, specializzata nell'elaborazione di array e matrici numeriche [25]. La sua efficienza e flessibilità nel gestire operazioni matematiche su grandi volumi di dati è dovuta all'utilizzo di funzioni ottimizzate scritte in linguaggio C, superando di gran lunga l'elaborazione basata su liste o cicli Python puri in termini di velocità. Ciò rende NumPy ideale per operazioni matematiche complesse, analisi dei dati, elaborazione del segnale e molte altre applicazioni dove la velocità e la precisione sono cruciali. Questo è il motivo del suo impiego in questo progetto per la gestione di grandi liste di dati e per la creazione degli spettrogrammi in scala di grigio, tramite la Fast Fourier Transform (FFT) applicata ai segnali audio memorizzati su file wave. La versione utilizzata è la 1.23.5.

2.7.7 OpenCV

La Open Source Computer Vision Library, detta OpenCV, è un framework open source progettato specificamente per l'elaborazione delle immagini. Fondata da Intel nel 2000, OpenCV si è affermata come uno degli strumenti essenziali per gli specialisti di Computer Vision [41]. È conosciuta per la sua versatilità, dato che supporta una vasta gamma di operazioni, dall'elaborazione di immagini come il ridimensionamento e il filtraggio, a tecniche più avanzate come la rilevazione e il tracciamento di oggetti. Grazie alla sua natura open source, OpenCV beneficia anche di miglioramenti e aggiornamenti continui da parte della sua vasta comunità. In questo progetto, è stata utilizzata la versione 4.8.0 per il pre-processamento dei dati, effettuando un filtraggio alle immagini al fine di evidenziarne le caratteristiche principali.

2.7.8 Computer utilizzato

Il computer utilizzato per la progettazione di questo software e la ricerca è un laptop HP Pavilion 15-eg con processore i7-1165G7 octa-core di undicesima generazione con un clock di 2.8 GHz. La memoria RAM installata è di 16 GB ed il sistema operativo è Windows 10 a 64 bit. La GPU, che grazie a TensorFlow è stata sfruttata, è una Tiger Lake GT2 con un clock di 400 MHz e la scheda video integrata ha una frequenza di aggiornamento di 60 KHz.

CAPITOLO 3

Risultati e discussione

In questo capitolo vengono mostrati i risultati ottenuti dalle sperimentazioni descritte nel capitolo 2, facendo un confronto con reti più complesse e analizzando i possibili sviluppi futuri.

3.1 Reti testate

Tramite le metriche di valutazione descritte di seguito, viene testata la qualità della rete realizzata ed i risultati ottenuti vengono riportati in tabella. Infine, vengono provate anche le reti ResNet50 e VGG16, famose per problemi di classificazione di immagini.

3.1.1 Metriche di valutazione

Per la valutazione delle reti neurali, è stato riservato un set di test, costituito da tutti i dati che non sono stati utilizzati per la fase di addestramento. Questi vengono utilizzati per la verifica del corretto funzionamento del modello che deve processarli, estraendo poi un risultato binario ottenuto tramite una soglia impostata al 50%. Poiché sono note le classi di appartenenza di tutti i file, a seguito del processamento è possibile dichiarare il numero di falsi positivi, falsi negativi, veri positivi e veri negativi, rappresentabili con la Confusion Matrix. Questi dati vengono utilizzati per ricavare i valori in percentuale di alcune metriche di valutazione, ovvero Precision, Recall, Accuratezza, F1-score [42]. Come è possibile osservare dalle Formule (2, 3) la Precision rappresenta il bilancio dei falsi positivi, mentre la Recall rappresenta il bilancio dei falsi negativi, più questi valori si avvicinano al 100% meno errori sono stati effettuati. L'F1-score rappresenta la media di Precision e Recall, mentre l'accuratezza, indica la percentuale di elementi classificati correttamente.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$F1\_Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

TP = True Positive TN = True Negative
FP = False Positive FN = False Negative

Nonostante di norma venga fatto riferimento specialmente all'accuratezza, in questo caso si dà maggiore importanza ai valori di Precision e Recall, in particolare la prima. Come già spiegato nel Capitolo 2, fornendo una motivazione di alcune scelte effettuate, si preferisce avere più falsi negativi che falsi positivi. Per tale motivazione, si cerca una rete neurale con il valore di Precision più alto possibile, senza però trascurare le altre metriche di valutazione, in particolar modo l'accuratezza.

3.1.2 Valutazione rete

L'architettura della CNN utilizzata è descritta nel paragrafo 2.4.4 e, come specificato nel Capitolo 2.5, per l'addestramento è stata utilizzata la 10-Cross Fold Validation, in cui ogni modello è stato allenato con un numero di epoche massimo di 200 con pazienza impostata a 15 sull'accuratezza tramite Early Stopping, compilatore Adam con learning rate settato a 0.0001, funzione di perdita Binary Cross Entropy, metrica accuratezza, peso della classe dei rumori pari a 1.5 e peso della classe dei fischi pari a 1.0, il tutto con batch size di 32. Tutte le reti ottenute sono state valutate con le metriche di valutazione descritte precedentemente e le migliori sono state testate su registrazioni acquisite all'acquario Oltremare di Riccione, però assenti di etichette ottenute con analisi PAM manuale. Dopo aver calcolato i valori di Precision, Recall, Accuratezza, F1-score e MSE per ogni fold, al fine di ottenere una valutazione generale viene effettuata una media, presente nella Tabella 1. Come è possibile notare, alcune fold hanno dei valori leggermente più deludenti, mentre altre, come le fold 4 e 8, hanno il massimo dell'accuratezza, ovvero il 100%. Tuttavia, come già specificato, per la valutazione del modello bisogna tenere conto della media dei valori di tutte le fold addestrate. Per cui, sul dataset utilizzato si può affermare che questa rete ha un'accuratezza del 99.3%.

Fold	Precision	Recall	Accuracy	F1-score
1	1.000	0.970	0.987	0.985
2	0.981	0.981	0.987	0.981
3	1.000	0.982	0.993	0.991
4	1.000	1.000	1.000	1.000
5	1.000	0.985	0.993	0.992
6	1.000	0.983	0.993	0.991
7	1.000	0.986	0.993	0.993
8	1.000	1.000	1.000	1.000
9	0.979	0.979	0.987	0.979
10	1.000	0.983	0.993	0.992
Average	**0.996**	**0.985**	**0.993**	**0.990**

Tabella 1. *Risultati 10-Fold Cross-Validation*

3.1.3 ResNet50 e VGG16

Come trovato in letteratura [43-44-45], per problemi di classificazione di immagini tramite deep learning, le reti neurali ResNet50 e la VGG16 hanno dimostrato in alcuni esperimenti di avere dei rendimenti nettamente migliori, anche in casi di classificazione con più classi. Tuttavia, applicate a questo dataset hanno reso un risultato negativo, andando in Overfitting in entrambi i casi. Poiché entrambe le reti hanno un'architettura complessa e il dataset di training non è sufficientemente grande, i modelli non hanno avuto modo di imparare a distinguere le classi. Entrambe, sono state valutate con il set di test utilizzando le metriche di valutazione. Tutte le immagini sono state classificate come rumori, ottenendo Precision, Recall e F1-score nulli e un'accuratezza del 57%. Poiché i risultati ottenuti con VGG16 e ResNet50 sono diversi da quelli in letteratura, si è deciso di verificare l'andamento dell'addestramento per la rete ResNet50, che è risultata essere la più performante. Tale verifica è stata effettuata suddividendo il dataset totale in train al 70%, validazione al 20% e test al 10%. Il set di validazione, come spiegato nel paragrafo 2.5.1, viene utilizzato per valutare l'addestramento. Avanzando con le epoche si ha una crescita della val_loss ed una riduzione del valore di loss Figura 30-A. Entrambi i valori dovrebbero diminuire sempre di più cercando di tendere a zero, invece si allontanano tra di loro indicando un Overfitting [46-47]. Inoltre, il valore dell'accuratezza di addestramento è molto alto, ciò significa che la rete non ha imparato a generalizzare il problema, ma impara a memorizzare le immagini nel set di train, infatti il valore di val_accuracy resta costante senza nessun miglioramento Figura 30-B.

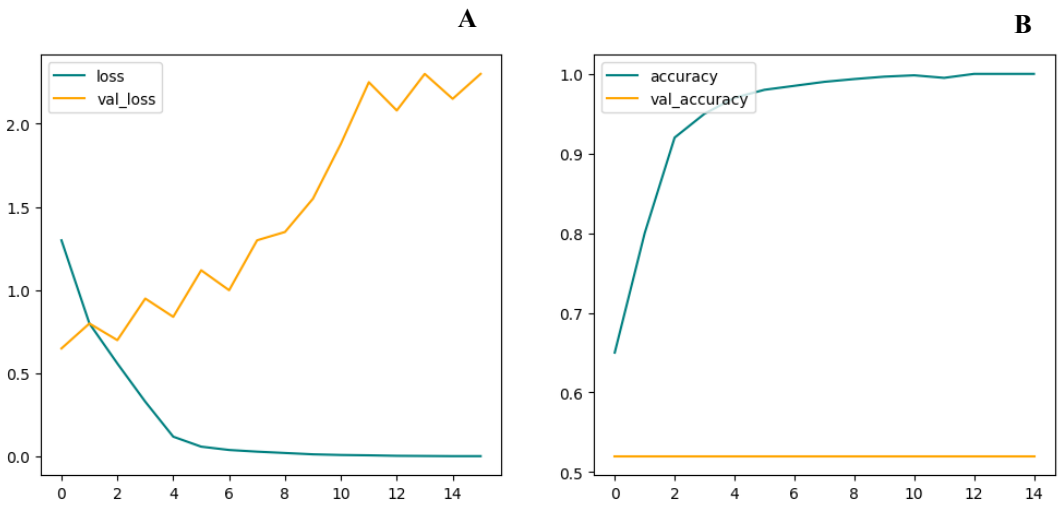

Figura 30. *Immagine A: Andamento accuracy e val_accuracy per ResNet50. Immagine B: Andamento loss e val_loss per ResNet50.*

3.2 Nuovo dataset

Dopo aver selezionato il modello di rete neurale migliore, è stato testato su nuove registrazioni con delfini diversi da quelli presenti nei file utilizzati per l'addestramento, inseriti nel dataset. In totale sono 99 file wave della durata da 5 minuti ciascuno per un totale di 8 ore e 15 minuti. Per ogni registrazione, si è seguita la procedura illustrata nel Capitolo 2. Sono stati creati gli spettrogrammi in finestre di 0.6 secondi traslando ogni 0.2 secondi, con una sovrapposizione totale di 0.4 secondi tra un'immagine e l'altra. Di conseguenza, sono stati generati 1498 spettrogrammi in scala di grigio, successivamente elaborati tramite filtro Sobel per evidenziare gli eventuali fischi. Dopo la creazione di tutte le immagini, queste vengono raggruppate in batch da 64, in quanto la rete neurale ha bisogno di meno tempo computazionale per analizzare le reti in gruppi piuttosto che singolarmente. La rete neurale processa poi i batch, assegnando a ogni immagine una percentuale di appartenenza alla classe dei delfini, risultando in una lista di 1498 percentuali al termine dell'elaborazione. Successivamente, questa lista viene analizzata da un algoritmo descritto al paragrafo 2.6.1 per interpretare i risultati ottenuti della rete neurale, selezionando quindi le combinazioni di percentuali desiderate per dichiarare la presenza di un delfino, generando di conseguenza un'etichetta appropriata. Tutte le etichette vengono poi salvate all'interno di un file di testo nominato come la registrazione correlata. Per ogni fischio rilevato, il file riporta il momento di inizio, di fine e un'etichetta.

Non è stata ancora effettuata una ground truth di queste registrazioni, per cui non è attualmente possibile ottenere delle statistiche a riguardo, ma osservando le etichette inserite automaticamente e analizzando gli errori commessi dalla rete, è stato possibile individuare quali sono le limitazioni del sistema. Alcuni fischi abbastanza evidenti, specialmente di alcuni delfini in stato di cattività, non vengono rilevati dalla rete neurale, dando quindi dei falsi negativi, come in Figura 31. Escluse alcune eccezioni, i falsi positivi presenti nelle etichette sono dovuti ad altre vocalizzazioni di delfini, detti Buzz. Quest'ultimo è ottenuto da un segnale di un delfino ad una frequenza talmente alta da generare l'effetto in Figura 32 mostrandolo come un insieme di linee orizzontali sovrapposte e non come linee verticali, per cui invece di essere eliminato dal Sobel verticale, viene evidenziato. Controllando le percentuali restituite dalla rete neurale,

alcuni di questi buzz arrivano ad essere classificati come fischi anche al 100%, ciò significa che il modello non è stato in grado di imparare che queste vocalizzazioni non sono dei fischi. Tuttavia, come detto precedentemente, quasi tutti i falsi positivi ottenuti sono dati da queste vocalizzazioni. Inoltre, come desiderato, la rete genera più falsi negativi che falsi positivi.

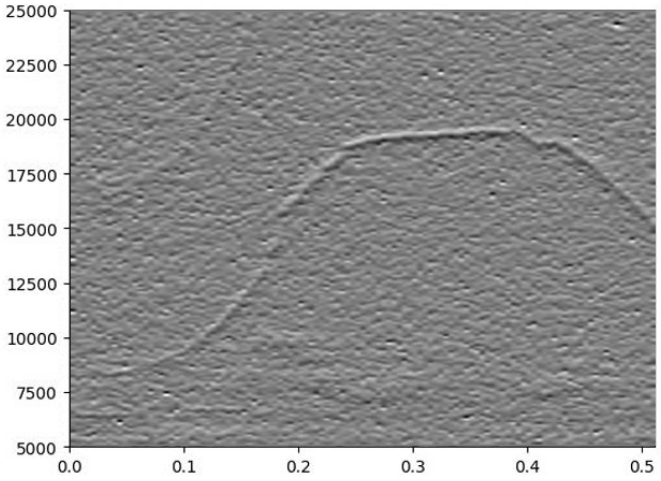

Figura 31. *Fischio di delfino in stato di cattività non rilevato (falso negativo)*

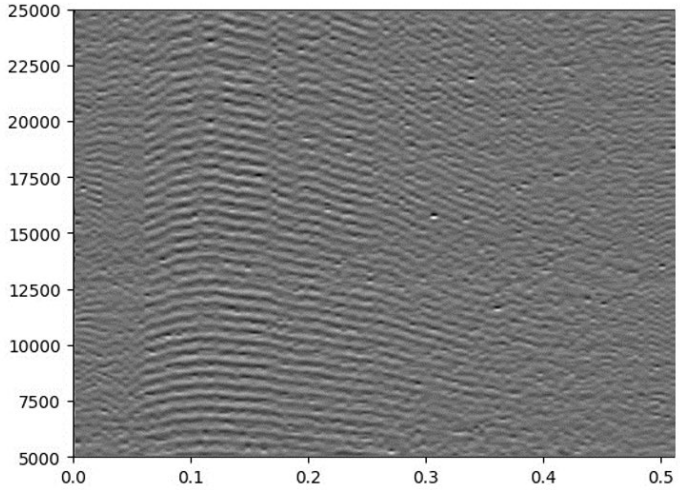

Figura 32. *Buzz di delfino riconosciuto come fischio (falso positivo)*

3.3 Sviluppi futuri

In base ai risultati ottenuti, sono stati ipotizzati dei possibili sviluppi futuri per ampliare le possibilità di ridurre il problema dell'interazione delfini con i pescatori.

3.3.1 Real time

L'obiettivo di questa tesi è di contribuire alla realizzazione di un dispositivo in grado di ridurre l'interazione tra i delfini e le attività di pesca, per farlo è necessario che questo sistema funzioni in real time. Dunque, per uno sviluppo futuro si potrebbe fare in modo che durante una registrazione in tempo reale vengano passati alcuni secondi di audio alla rete neurale continuamente fino al momento della rilevazione di un delfino, a quel punto potrebbe essere emesso il segnale acustico per far allontanare questi cetacei. Per fare ciò, è necessario spostare il software su un microcontrollore con delle caratteristiche hardware sufficienti per un fluido funzionamento di questo sistema.

3.3.2 Ampliamento dataset

Un'espansione accurata e significativa del dataset potrebbe aiutare la rete neurale a generalizzare il problema, rendendo il modello più robusto e affidabile nell'affrontare scenari diversi e non incontrati durante la fase di addestramento. Inoltre, con un dataset sufficientemente grande, potrebbero essere riprovate le reti ResNet50 e VGG16, che hanno dimostrato di essere più performanti in caso di ampi dataset per classificazione di immagini [43-44-45] in prestazioni di alto livello in competizioni e benchmark, come quelli proposti dalla sfida ImageNet. Questo, riuscirebbe a ridurre il numero di falsi positivi e di falsi negativi.

3.3.3 Doppia rete neurale per buzz

Per risolvere il problema dei Buzz, identificati come fischi di delfini, si potrebbe progettare una seconda rete neurale in grado di riconoscere un fischio o un Buzz. Al momento in cui viene identificato un delfino con la rete principale, può essere utilizzata questa nuova rete per distinguere un fischio da un Buzz. Questo consentirebbe di inserire le etichette non solo dei fischi, ma anche di questi particolari segnali.

3.3.4 Modelli di movimento dei delfini

Utilizzare la rete neurale per comprendere i modelli di movimento dei delfini può essere utile per mappare aree di mare stabilendo zone di esclusione. Si potrebbero delineare aree in cui la pesca è vietata o limitata durante periodi in cui si prevede una loro maggiore presenza. Questo approccio proattivo può ridurre significativamente il rischio di interazione tra delfini e imbarcazioni da pesca. Parallelamente, le zone di precauzione potrebbero essere stabilite in aree dove, pur essendo presente un certo livello di attività dei delfini, la pesca può continuare con certe restrizioni magari integrando le reti neurali con sistemi di tracciamento in tempo reale. Si potrebbero fornire avvisi immediati alle imbarcazioni in prossimità di delfini in modo da dare la possibilità agli equipaggi di modificare le loro rotte o tecniche di pesca, minimizzando ulteriormente il rischio di interazione.

Conclusioni

Il presente lavoro di tesi descrive un processo che ha permesso di sviluppare e validare una rete neurale focalizzata sull'identificazione del fischio dei delfini, esplorando e applicando metodologie derivanti dal campo dell'intelligenza artificiale alla bioacustica marina. I risultati ottenuti si distinguono per un grado di accuratezza eccezionalmente elevato e confermano che l'approccio metodologico adottato è stato ampiamente adeguato e proficuo, conducendo a esiti che possono essere qualificati come pienamente soddisfacenti. Infatti, mediante l'applicazione di tecniche consolidate, quali la 10-Fold Cross-Validation, accompagnate dall'interpretazione dell'output della rete neurale, si è raggiunta una sorprendevole accuratezza totale del 99.3%, eccellendo nelle aspettative. Grazie all'introduzione di apposite e specifiche soglie, questo approccio consente anche di identificare la presenza dei delfini nella scala temporale, identificando così un punto di partenza per un processo finalizzato all'allontanamento automatico di questi cetacei, scoraggiandone l'interazione con le attività di pesca. È importante sottolineare che il modello proposto è stato validato attraverso l'uso di un singolo dataset, il che, pur mostrando risultati con accuratezza molto elevata, limita la generalità delle conclusioni ottenute. Tuttavia, le prestazioni evidenziate dal modello e la robustezza dei risultati conseguiti mostrano grandi potenzialità e apportano un contributo significativo al campo di studio, offrendo un solido punto di partenza per ulteriori ricerche e sviluppi futuri. In conclusione, questo studio mostra come la tecnologia e la scienza possano lavorare insieme per capire meglio e proteggere i delfini, aprendo la strada ad un futuro dove umani e vita marina possano coesistere in modo più armonioso e rispettoso.

Bibliografia

[1] A. Petetta *et al.*, "Fishers' Perception on the Interaction between Dolphins and Fishing Activities in Italian and Croatian Waters," *Diversity*, vol. 15, no. 2, p. 133, Jan. 2023, doi: 10.3390/d15020133.

[2] G. Bearzi, "In Cetaceans of the Mediterranean and Black Seas: State of Knowledge and Conservation Strategies," in *Interactions between Cetacean and Fisheries in the Mediterranean Sea*, vol. 9, Monte Carlo, Monaco: ACCOBAMS, pp. 1–20, 2002.

[3] B. L. Chilvers and P. J. Corkeron, "Trawling and bottlenose dolphins' social structure," *Proceedings of the Royal Society B: Biological Sciences*, vol. 268, no. 1479, pp. 1901–1905, Sep. 2001, doi: 10.1098/rspb.2001.1732.

[4] S. Bonizzoni, S. Hamilton, R. R. Reeves, T. Genov, and G. Bearzi, "Odontocete cetaceans foraging behind trawlers, worldwide," *Reviews in Fish Biology and Fisheries*, vol. 32, no. 3, pp. 827–877, Apr. 2022, doi: 10.1007/s11160-022-09712-z.

[5] G. Lauriano, L. Caramanna, M. Scarnò, and F. Andaloro, "An overview of dolphin depredation in Italian artisanal fisheries," *Journal of the Marine Biological Association of the United Kingdom*, vol. 89, no. 5, pp. 921–929, Apr. 2009, doi: 10.1017/s0025315409000393.

[6] D. Rocklin, M.-C. Santoni, J. Culioli, J. -a. Tomasini, D. Pelletier, and D. Mouillot, "Changes in the catch composition of artisanal fisheries attributable to dolphin depredation in a Mediterranean marine reserve," *Ices Journal of Marine Science*, vol. 66, no. 4, pp. 699–707, Mar. 2009, doi: 10.1093/icesjms/fsp036.

[7] "CNR life delfi – ridurre le interazioni dei cetacei con le attività di pesca," https://lifedelfi.eu/

[8] D. A. Russell, "The Loeb Plutarch - Harold Cherniss and William C. Helmbold: Plutarch, Moralia. Vol. xii. With an English translation. (Loeb Classical Library.) Pp. xii+590. London: Heinemann, 1957. Cloth, 15*s*. net.," *Classical Review*, vol. 9, no. 3, pp. 246–247, Dec. 1959, doi: 10.1017/s0009840x00173342.

[9] S. Bonizzoni, N. B. Furey, and G. Bearzi, "Bottlenose dolphins (*Tursiops truncatus*) in the north-western Adriatic Sea: Spatial distribution and effects of trawling," *Aquatic Conservation-marine and Freshwater Ecosystems*, vol. 31, no. 3, pp. 635–650, Oct. 2020, doi: 10.1002/aqc.3433.

[10] F. Di Nardo, R. De Marco, A. Lucchetti, and D. Scaradozzi, "A WAV file dataset of bottlenose dolphin whistles, clicks, and pulse sounds during trawling interactions," *Scientific Data*, vol. 10, no. 1, Sep. 2023, doi: 10.1038/s41597-023-02547-8.

[11] G. Lauriano, C. M. Fortuna, G. Moltedo, and G. N. Di Sciara, "Interactions between common bottlenose dolphins (Tursiops truncatus) and the artisanal fishery in Asinara Island National Park (Sardinia): assessment of catch damage and economic loss," *The Journal of Cetacean Research and Management*, vol. 6, no. 2, pp. 165–173, Mar. 2023, doi: 10.47536/jcrm.v6i2.780.

[12] R. De Marco *et al.*, "The development of a Low-Cost hydrophone for passive acoustic monitoring of dolphin's vocalizations," *Remote Sensing*, vol. 15, no. 7, p. 1946, Apr. 2023, doi: 10.3390/rs15071946.

[13] R. De Marco *et al.*, "A low-cost approach in acoustic monitoring of dolphin presence," *2022 IEEE International Workshop on Metrology for the Sea; Learning to Measure Sea Health Parameters (MetroSea)*, Oct. 2022, doi: 10.1109/metrosea55331.2022.995093

[14] T. A. Marques *et al.*, "Estimating animal population density using passive acoustics," *Biological Reviews*, vol. 88, no. 2, pp. 287–309, Nov. 2012, doi: 10.1111/brv.12001.

[15] G. La Manna, M. Manghi, G. Pavan, F. Lo Mascolo, and G. Sarà, "Behavioural strategy of common bottlenose dolphins (*Tursiops truncatus*) in response to different kinds of boats in the waters of Lampedusa Island (Italy)," *Aquatic Conservation-marine and Freshwater Ecosystems*, p. n/a, May 2013, doi: 10.1002/aqc.2355.

[16] W. Zimmer, *Passive acoustic monitoring of cetaceans*. 2011. doi: 10.1017/cbo9780511977107.

[17] MTC Media, "PAMGuaRd | Passive Acoustic Monitoring |PAM Open Source Software - PAMGuard," Jun. 29, 2001. https://www.pamguard.org/

[18] "Chelonia Limited manufactures automatic cetacean monitoring equipment," https://www.chelonia.co.uk/

[19] "SAMBAH – Static Acoustic Monitoring of the Baltic Sea Harbour Porpoise," *http://www.sambah.org/*

[20] "Z by HP Unlocked Challenge 3 - Signal Processing," *Kaggle*, Feb. 17, 2022. https://www.kaggle.com/datasets/kenjee/z-by-hp-unlocked-challenge-3-signal-processing

[21] "For AI Wildlife Species Bat Detector," https://hackaday.io/project/169854-ai-wildlife-species-bat-detector/details

[22] "Neurons in Neural Networks," *https://www.baeldung.com/cs/neural-networks-neurons*

[23] E. Marziali, "Studio ed implementazione di un sistema di caratterizzazione acustica per idrofoni low-cost," UNIVPM Ancona, 2023.

[24] V. M. Janik and L. S. Sayigh, "Communication in bottlenose dolphins: 50 years of signature whistle research," *Journal of Comparative Physiology A-neuroethology Sensory Neural and Behavioral Physiology*, vol. 199, no. 6, pp. 479–489, May 2013, doi: 10.1007/s00359-013-0817-7.

[25] "NumPy," https://numpy.org/

[26] P. Podder, T. Z. Khan, M. H. Khan, and M. Rahman, "Comparative performance analysis of Hamming, Hanning and Blackman Window," *International Journal of Computer Applications*, vol. 96, no. 18, pp. 1–7, Jun. 2014, doi: 10.5120/16891-6927.

[27] S. Gupta and A. Panghal, "Performance Analysis of FIR Filter Design by Using Rectangular, Hanning and Hamming Windows Methods," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 6, Jun. 2012.

[28] K. Simonyan, "Very deep convolutional networks for Large-Scale image recognition," *arXiv.org*, Sep. 04, 2014. https://arxiv.org/abs/1409.1556

[29] Y. Zhang, K. Muhammad, and C. Tang, "Twelve-layer deep convolutional neural network with stochastic pooling for tea category classification on GPU platform," *Multimedia Tools and Applications*, vol. 77, no. 17, pp. 22821–22839, Feb. 2018, doi: 10.1007/s11042-018-5765-3.

[30] C. Zhang and P. C. Woodland, "DNN speaker adaptation using parameterised sigmoid and ReLU hidden activation functions," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016, doi: 10.1109/icassp.2016.7472689.

[31] M. W. Browne, "Cross-Validation methods," *Journal of Mathematical Psychology*, vol. 44, no. 1, pp. 108–132, Mar. 2000, doi: 10.1006/jmps.1999.1279.

[32] D. Berrar, "Cross-Validation," in *Elsevier eBooks*, pp. 542–545, 2019, doi: 10.1016/b978-0-12-809633-8.20349-x.

[33] A. W Senior, "An empirical study of learning rates in deep neural networks for speech recognition," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing.*, May 2013, doi: 10.1109/icassp.2013.6638963.

[34] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, "AdaM Optimization Algorithm for wide and deep neural network," *Knowledge Engineering and Data Science*, vol. 2, no. 1, p. 41, Jun. 2019, doi: 10.17977/um018v2i12019p41-46.

[35] L. Prechelt, "Early Stopping — But When?," in *Lecture Notes in Computer Science*, pp. 53–67, 2012, doi: 10.1007/978-3-642-35289-8_5.

[36] "Keras: Deep Learning for humans," *https://www.tutorialspoint.com/keras/keras_introduction.htm*

[37] "TensorFlow," *https://www.tensorflow.org/*

[38] T. Akiba, "OpTUNA: a next-generation Hyperparameter Optimization Framework," *arXiv.org*, Jul. 25, 2019. https://arxiv.org/abs/1907.10902

[39] "Optuna - A hyperparameter optimization framework," https://optuna.org/

[40] "Audacity," *https://www.audacityteam.org/about/credits/*

[41] A. Zelinsky, "Learning OpenCV---Computer Vision with the OpenCV Library (Bradski, G.R. et al.; 2008)[On the Shelf]," *IEEE Robotics & Automation Magazine*, vol. 16, no. 3, p. 100, Sep. 2009, doi: 10.1109/mra.2009.933612.

[42] D. Powers, "Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation," *arXiv (Cornell University)*, vol. 2, no. 1, pp. 37–63, Dec. 2011, doi: 10.48550/arXiv.2010.16061.

[43] L. Wen, X. Li, and L. Gao, "A transfer convolutional neural network for fault diagnosis based on ResNet-50," *Neural Computing and Applications*, vol. 32, no. 10, pp. 6111–6124, Feb. 2019, doi: 10.1007/s00521-019-04097-w.

[44] S. Mascarenhas and M. Agarwal, "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification," in *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*, Bengaluru, India. Available: https://doi.org/10.1109/centcon52345.2021.9687944

[45] D. Theckedath and R. R. Sedamkar, "Detecting affect states using VGG16, RESNEt50 and SE-RESNet50 networks," *SN Computer Science*, vol. 1, no. 2, Mar. 2020, doi: 10.1007/s42979-020-0114-9.

[46] Y. Xue, "An Overview of Overfitting and its Solutions," *Journal of Physics*, vol. 1168, p. 022022, Feb. 2019, doi: 10.1088/1742-6596/1168/2/022022.

[47] "Overfitting in Machine Learning: What It Is and How to Prevent It," *https://elitedatascience.com/overfitting-in-machine-learning*

Sitografia

Figura 1 Ref. Articolo [10]

Figura 2 Ref. Articolo [11], foto di Ginevra Moltedo

Figura 3 http://www.sambah.org/

Figura 4 www.baeldung.com/cs/neural-networks-neurons

Figura 5 www.baeldung.com/cs/neural-networks-neurons

Figura 7 https://datascience.eu/it/visione-artificiale/reti-neurali-convoluzionali-il-modo-eli5/

Figura 17 https://www.sciencedirect.com/science/article/abs/pii/B9780323900324000146

Figura 18 https://hackernoon.com/log-analytics-with-deep-learning-and-machine-learning-20a1891ff70e

Figura 19 https://datascience.eu/it/visione-artificiale/reti-neurali-convoluzionali-il-modo-eli5/

Figura 21 https://medium.comnerd-for-tech/layers-in-neural-network-90d48a5a42fb

Figura 25 https://algotrading101.com/learn/train-test-split/

Figura 26 https://www.researchgate.net/figure/Repeated-K-Fold-Cross-Validation-A-repeated-10-fold-CV-was-applied-The-10-fold-CV-works_fig1_328798891

Figura 27 https://cs231n.github.io/neural-networks-3/

Figura 28 https://towardsdatascience.com/the-million-dollar-question-when-to-stop-training-deep-learning-models-fa9b488ac04d

Ringraziamenti

A conclusione di questo lavoro accademico, mi è doveroso ringraziare tutti coloro che hanno avuto un ruolo significativo in questi tre anni, dentro e fuori il contesto universitario.

Innanzitutto, vorrei ringraziare il Prof. David Scaradozzi per avermi dato la possibilità di lavorare a questo progetto stimolante e affascinante, nella speranza che la soluzione descritta in questa tesi possa contribuire a salvare la vita dei cari amici delfini. Ringrazio il Dott. Francesco Di Nardo per tutti i consigli preziosi, il sostegno e l'empatia dimostrata durante questo percorso di lavoro e ricerca. Ringrazio inoltre il Dott. Rocco De Marco per la sua dedizione ed i suoi insegnamenti sulla comunicazione di questa specie marina simpatica e curiosa.

Dedico questa tesi e il più importante dei ringraziamenti a mia madre Roberta, mio padre Pietro e mio fratello Samuel, per il sostegno, l'affetto, la vicinanza in ogni istante. Grazie a tutta la mia famiglia per non avermi fatto mancare nulla, avermi sempre sostenuto in qualsiasi tipo di iniziativa e per aver sempre creduto in me.

Ringrazio tutti i colleghi con cui ho dei ricordi indelebili e con cui ho trascorso giornate intere a studiare e a sperimentare in laboratorio, arrivando anche a chiusura. Grazie a tutti i miei amici, dai più vicini ai più distanti, che non smettono mai di farmi divertire e con cui ho condiviso dei momenti indimenticabili.

Infine, ringrazio tutte le persone che prenderanno questo traguardo come un motivo per riunirsi e festeggiare.