



# POLITECNICO DI BARI

Dipartimento di Ingegneria ELETTRICA e dell'INFORMAZIONE  
**Corso di Laurea in Ingegneria Informatica e dell'Automazione**

---

Tesi di Laurea in BIOINFORMATICA

**IMPLEMENTAZIONE E VALIDAZIONE DI UNA RETE NEURALE  
CONVOLUZIONALE COME STRUMENTO DI SUPPORTO DIAGNOSTICO  
NELLA MALATTIA DI ALZHEIMER MEDIANTE IMMAGINI DI RISONANZA  
MAGNETICA**

Relatore:  
**Prof. Ing. Stefano Mazzoleni**

Laureando:  
**Cristian Saracino**



<b>Capitolo 1 – Introduzione.....</b>	<b>1</b>
<b>1.1 La malattia di Alzheimer .....</b>	<b>3</b>
<b>1.1.1 Descrizione e sintomatologia .....</b>	<b>3</b>
<b>1.1.2 Fattori di rischio e diagnosi.....</b>	<b>5</b>
<b>1.1.3 Tipi di immagini MRI e caratteristiche rilevanti per la malattia di Alzheimer .....</b>	<b>6</b>
<b>1.2 Reti Neurali Convoluzionali (CNN).....</b>	<b>8</b>
<b>1.2.1 Architettura delle CNN.....</b>	<b>8</b>
<b>1.2.2 Funzioni di attivazione.....</b>	<b>18</b>
<b>1.2.3. Applicazioni delle CNN nell'elaborazione di immagini MRI .....</b>	<b>22</b>
<b>Capitolo 2 – Materiali e metodi .....</b>	<b>24</b>
<b>2.1 Dataset OASIS 1 .....</b>	<b>24</b>
<b>2.2 Preprocessing delle immagini.....</b>	<b>27</b>
<b>2.3. Architettura della rete CNN implementata .....</b>	<b>31</b>
<b>2.4 Parametri di addestramento e validazione del modello.....</b>	<b>37</b>
<b>2.5 Implementazione.....</b>	<b>38</b>
<b>2.5.1 Scelta della piattaforma e degli strumenti di sviluppo.....</b>	<b>38</b>
<b>2.5.2. Implementazione del modello CNN .....</b>	<b>40</b>
<b>2.5.3. Ottimizzazione e tuning degli iperparametri .....</b>	<b>40</b>
<b>2.5.4. Valutazione delle performance del modello .....</b>	<b>42</b>
<b>Capitolo 3 – Risultati .....</b>	<b>44</b>
<b>Capitolo 4 – Discussione .....</b>	<b>47</b>
<b>Conclusioni .....</b>	<b>50</b>
<b>Bibliografia .....</b>	<b>51</b>
<b>Appendice .....</b>	<b>53</b>

## **Capitolo 1 – Introduzione**

L'intelligenza artificiale (IA) è l'abilità di un sistema informatico di poter apprendere, elaborare l'informazione e risolvere problemi, in modo simile al funzionamento dell'intelligenza umana.

Negli ultimi anni, ha assunto un ruolo sempre più centrale in numerosi settori, tra cui l'industria, la domotica, la gestione aziendale e, in particolare, la medicina. In ambito medico-sanitario, gli algoritmi di IA forniscono un ottimo supporto per l'identificazione delle patologie, per la gestione delle terapie e per operazioni di monitoraggio, tanto che negli Stati Uniti d'America sono per il momento novecentocinquanta le applicazioni di IA approvate dall'ente regolatore FDA, che certifica la loro sicurezza ed efficacia [1].

L'obiettivo di questa tesi di laurea è quello di sviluppare e validare un modello di Deep Learning (DL) basato su una rete neurale convoluzionale (CNN), finalizzato alla classificazione delle immagini di risonanza magnetica (MRI) cerebrale in quattro categorie: a) assenza di demenza, b) stadio molto lieve di demenza, c) stadio lieve di demenza e d) stadio moderato di demenza.

Lo scopo è quello fornire uno strumento di supporto automatico all'attività diagnostica del neurologo, capace di identificare precocemente la malattia di Alzheimer, con particolare attenzione alle fasi iniziali. In questi stadi, la diagnosi clinica può risultare più complessa, poiché spesso richiede l'analisi comparativa di più immagini acquisite nel tempo per rilevare cambiamenti sottili e progressivi. Il sistema proposto mira a ridurre questa difficoltà, aiutando a riconoscere i primi segnali della patologia in modo più rapido e preciso, supportando così il processo decisionale del medico.

Il modello sarà sviluppato utilizzando la piattaforma Google Colab, un ambiente di cloud computing che offre accesso a risorse computazionali avanzate attraverso macchine remote, disponibili a fronte di un abbonamento. Per la realizzazione del modello, si farà uso di Inception V3, un modello pre-addestrato noto per le sue prestazioni nell'analisi delle immagini. Inception V3 sarà ulteriormente personalizzato con l'aggiunta di strati specifici, in modo da adattare il modello alla classificazione multiclasse richiesta, permettendo di distinguere tra i quattro stadi di demenza (assenza, molto lieve, lieve e moderata). Questo approccio consentirà di sfruttare i vantaggi del trasferimento di apprendimento, migliorando l'efficacia del modello nella classificazione delle immagini di risonanza magnetica cerebrale.

La tesi di laurea è organizzata secondo la seguente struttura. Nel primo capitolo viene presentata la descrizione clinica della malattia di Alzheimer, una patologia neurodegenerativa che colpisce specifiche aree cerebrali. Vengono discusse le attuali tecniche diagnostiche e come le immagini MRI possano essere utilizzate per identificare e valutare lo stato della malattia. Inoltre, vengono presentate le CNN, in termini di architetture funzionali e della loro applicazione alle immagini di risonanza magnetica (MRI).

Successivamente vengono presentate le caratteristiche del dataset utilizzato, gli algoritmi di pre-processing adottato per le MRI, l'architettura scelta per raggiungere l'obiettivo della tesi e di come verranno valutate le metriche del modello.

Inoltre, viene descritta la strategia di implementazione, in termini di piattaforme utilizzate e di implementazione del modello, e sarà effettuata la stima delle metriche a seguito di un processo di ottimizzazione di parametri e iperparametri.

Nel terzo capitolo saranno presentati i risultati. Nel quarto e ultimo capitolo si analizzeranno i risultati ottenuti e verranno contestualizzati rispetto alle necessità reali mediche, inoltre verranno introdotte possibili modifiche per sviluppi futuri.

## **1.1 La malattia di Alzheimer**

### **1.1.1 Descrizione e sintomatologia**

La demenza è una condizione neurologica che comporta una perdita delle funzioni cognitive di un individuo, quali il pensiero, la memoria e il ragionamento, e delle abilità relative alle attività della vita quotidiana, tali da renderlo non più autonomo.

Una percentuale significativa dei casi di demenza è causata dalla malattia di Alzheimer, ossia una patologia neurodegenerativa che comporta un progressivo processo di distruzione delle cellule nervose, in particolare nelle zone dell'ippocampo e nella corteccia cerebrale, ossia le zone le cui funzioni riguardano la memoria a breve e a lungo termine, il pensiero e la coscienza.

La causa all'origine della malattia di Alzheimer sembrerebbe essere un'alterazione nella metabolizzazione della proteina APP, mediante cui si viene a formare una sostanza neurotossica, nota come beta-amiloide, che si accumula nel cervello fino a formare delle vere e proprie placche, che comportano inizialmente un'interruzione dell'attività neuronale, successivamente una perdita delle connessioni tra i neuroni e infine la loro atrofia.

Man mano che le placche si espandono, aree sempre più vaste del cervello vengono colpite e iniziano a restringersi.

Oltre all'accumulo delle placche di beta-amiloide tra i neuroni, si vengono a formare accumuli di proteine tau all'interno del neurone stesso. Infatti, i neuroni sono supportati internamente dai microtubuli, ossia strutture in grado di aiutare con l'approvvigionamento dei nutrienti dal corpo cellulare agli assioni e i dendriti, che vengono stabilizzati attraverso legami con le proteine tau. Nella malattia di Alzheimer, tuttavia, cambiamenti chimici significativi comportano un distaccamento delle proteine tau dai microtubuli, che iniziano a legarsi fra loro formando veri e propri ammassi di proteine all'interno della cellula nervosa, noti come grovigli neurofibrillari. Questi ammassi bloccano il sistema di trasporto dei nutrienti con conseguente danneggiamento delle comunicazioni sinaptiche tra neuroni.

Persone affette dalla malattia di Alzheimer manifestano dapprima problematiche di perdita di memoria, inizialmente in forma leggera per poi arrivare a episodi invalidanti per l'individuo, successivamente si associano solitamente altri sintomi quali:

- disturbi nella comunicazione e impoverimento del vocabolario;
- difficoltà nell'esecuzione delle attività giornaliere;
- disorientamenti spaziali e temporali;
- allucinazioni;
- disturbi del sonno.

È possibile che i primi sintomi della malattia si manifestino anche decenni dopo la sua comparsa [2].

Il decorso della malattia, secondo il Clinical Dementia Rating (CDR), ossia una scala di valori sviluppata da John C. Morris e dei suoi colleghi della Washington University School of Medicine può essere suddiviso in tre stadi. Per poter effettuare la classificazione un professionista andrà a valutare diversi aspetti motori-cognitivi,

a cui assegnerà un punteggio, ovvero sei classi quali: la memoria, l'orientamento, il giudizio e la soluzione dei problemi, le attività sociali, la casa ed il tempo libero, la cura personale.

CDR™ Scoring Table

Subject Initials \_\_\_\_\_

CLINICAL DEMENTIA RATING (CDR™):		0	0.5	1	2	3
		Impairment				
		None 0	Questionable 0.5	Mild 1	Moderate 2	Severe 3
Memory	No memory loss or slight inconsistent forgetfulness	Consistent slight forgetfulness; partial recollection of events; "benign" forgetfulness	Moderate memory loss; more marked for recent events; deficit interferes with everyday activities	Severe memory loss; only highly learned material retained; new material rapidly lost	Severe memory loss; only fragments remain	
Orientation	Fully oriented	Fully oriented except for slight difficulty with time relationships	Moderate difficulty with time relationships; oriented for place at examination; may have geographic disorientation elsewhere	Severe difficulty with time relationships; usually disoriented to time, often to place	Oriented to person only	
Judgment & Problem Solving	Solves everyday problems & handles business & financial affairs well; judgment good in relation to past performance	Slight impairment in solving problems, similarities, and differences	Moderate difficulty in handling problems, similarities, and differences; social judgment usually maintained	Severely impaired in handling problems, similarities, and differences; social judgment usually impaired	Unable to make judgments or solve problems	
Community Affairs	Independent function at usual level in job, shopping, volunteer and social groups	Slight impairment in these activities	Unable to function independently at these activities although may still be engaged in some; appears normal to casual inspection	No pretense of independent function outside home	Appears well enough to be taken to functions outside a family home	Appears too ill to be taken to functions outside a family home
Home and Hobbies	Life at home, hobbies, and intellectual interests well maintained	Life at home, hobbies, and intellectual interests slightly impaired	Mild but definite impairment of function at home; more difficult chores abandoned; more complicated hobbies and interests abandoned	Only simple chores preserved; very restricted interests, poorly maintained	No significant function in home	
Personal Care	Fully capable of self-care		Needs prompting	Requires assistance in dressing, hygiene, keeping of personal effects	Requires much help with personal care; frequent incontinence	

Score only as decline from previous usual level due to cognitive loss, not impairment due to other factors.

Copyright 2001 by Washington University in St. Louis, Missouri.  
All rights reserved.

*Figura 1 Tabella relativa a Clinical Dementia Rating [3, 4].*

**Very Mild Stage (CDR = 0.5):** In questa fase risulta dubbia la presenza di demenza: si hanno episodi sporadici, non significativi di perdita di memoria, un normale orientamento nello spazio, capacità di pensiero nella norma e una vita autonoma.

**Mild Stage (CDR = 1):** in questa fase della malattia, l'individuo potrebbe mantenere una certa indipendenza e riuscire a svolgere autonomamente le attività quotidiane più semplici. Tuttavia, potrebbero verificarsi episodi di lapsus verbali e difficoltà a richiamare parole familiari o eventi recenti. Tra le difficoltà più comuni vi sono la fatica nel ricordare i nomi di persone conosciute da poco, problemi organizzativi e di pianificazione, oltre a una tendenza a dimenticare informazioni appena lette. In alcuni casi, potrebbero manifestarsi episodi di disorientamento spaziale.

**Moderate Stage (CDR = 2):** Gli episodi di perdita di memoria diventano sempre più gravi: le informazioni apprese di recente vengono rapidamente dimenticate, mentre i ricordi più consolidati possono essere ancora

mantenuti. Si manifestano disorientamenti spaziali e temporali, accompagnati da una perdita significativa della capacità di risolvere problemi. L'autonomia è fortemente compromessa, consentendo all'individuo di svolgere solo compiti molto semplici. È necessaria un'assistenza costante per attività quotidiane come vestirsi e mantenere una buona igiene personale.

Severe Stage (CDR = 3): In questa fase rimangono solo piccoli frammenti di ricordi, spesso legati all'infanzia. L'individuo perde completamente l'autonomia, con frequenti episodi di incontinenza e una totale incapacità di risolvere problemi. È considerato troppo compromesso per svolgere qualsiasi attività al di fuori dell'ambiente domestico o assistito.

### 1.1.2 Fattori di rischio e diagnosi

Meno del 5% dei casi è causato da condizioni di familiarità, ossia la presenza di geni alterati che riescono ad essere trasmessi alla prole. Quali varianti nei geni della presenilina 1 (*PS1*) sul cromosoma 14, della presenilina 2 (*PS2*) sul cromosoma 1 o della proteina precursore della beta amiloide (*APP*) sul cromosoma 21. Queste tipologie di casi hanno insorgenza precoce rispetto ai casi spontanei, anche prima dei 40 anni. Il restante 95% dei casi, ossia quelli naturali che non hanno nessun legame con l'ereditarietà, si manifestano prevalentemente in età presenile e in modo sporadico nella popolazione [2].

I fattori di rischio modificabili sono legati allo stile di vita: fumo, alimentazione inadeguata e scarsa attività fisica. Questi elementi contribuiscono all'insorgenza di condizioni come ipercolesterolemia, ipertensione e obesità, così come i traumi cerebrali pregressi, che sembrano aumentare il rischio di sviluppare la malattia. Al contrario, una maggiore scolarizzazione, una rete sociale solida e il mantenimento di un'attività fisica e mentale regolare sono associati a una riduzione del rischio [5].

Per poter effettuare la diagnosi della malattia di Alzheimer, non esistono test definitivi, solitamente si prosegue per esclusione dopo aver effettuato un attento esame delle condizioni psico-fisiche.

Le principali tappe del processo diagnostico sono:

- Anamnesi e visita medica: vengono poste delle domande ai parenti del soggetto su possibili difficoltà nell'eseguire attività quotidiane o su problematiche di autonomia generali, quali la pianificazione e la gestione del denaro. Il paziente verrà sottoposto a una visita neurologica dove verranno testate memoria, attenzione e capacità di eseguire un set di istruzioni. Viene solitamente anche utilizzato un piccolo esame noto come Mini-Mental State Examination (MMSE), che consiste nel sottoporre il malato a domande del tipo: "Che giorno è? In che città ci troviamo? Come si chiama questo?".
- Esami di laboratorio: è opportuno un test di laboratorio di sangue e urine per escludere ulteriori patologie che potrebbero causare sintomi simili alla demenza o di malattie che potrebbero causare un fattore di rischio per la comparsa dell'Alzheimer.
- Esami strumentali:

- Risonanza **magnetica (RM/MRI)**: Fornisce immagini molto dettagliate della struttura del cervello. Confrontando le immagini ottenute a distanza di alcuni mesi, è possibile osservare i cambiamenti in specifiche aree cerebrali.
- TAC (**Tomografia assiale computerizzata**): Misura lo spessore di alcune parti del cervello, che tende ad assottigliarsi rapidamente nei pazienti affetti da Alzheimer.
- SPECT (**Tomografia computerizzata a emissione di fotone singolo**): Valuta il flusso sanguigno nel cervello, che risulta ridotto nei pazienti con Alzheimer a causa della diminuzione dell'attività delle cellule nervose.
- PET (**Tomografia a emissione di positroni**): Principalmente utilizzata nei centri di ricerca, permette di rilevare cambiamenti nel funzionamento cerebrale, come un utilizzo anomalo del glucosio da parte del cervello nei pazienti con Alzheimer [6].

Per il momento non esistono cure definitive per la malattia di Alzheimer; le terapie moderne mirano a un rallentamento del decorso e a fornire aiuti per la gestione dei sintomi:

- inibitori della colinesterasi (donepezil, rivastigmina e galantamina);
- antagonisti del recettore NMDA (memantina);
- trattamenti non farmacologici (come la stimolazione cognitiva).

Si sono fatti molti passi in avanti nel corso degli ultimi 15 anni, in particolare per la produzione di anticorpi che possano andare a contrastare lo sviluppo di beta amiloide.

Nel 2021 la FDA ha approvato il primo anticorpo anti-amiloide per il trattamento della malattia di Alzheimer, aducanumab, e lo scorso gennaio ne è stato approvato un secondo, lecanemab. Più recentemente, un terzo anticorpo, Donanemab, ha dimostrato in protocolli di ricerca di fase 3 di essere efficace nel rimuovere l'amiloide cerebrale e nel rallentare la progressione dei sintomi cognitivi in maniera significativa [7].

La ricerca continua a esplorare nuove possibilità. Ad esempio, è stato osservato che i pazienti con artrite reumatoide hanno una minore incidenza di Alzheimer, probabilmente grazie all'uso prolungato di farmaci antinfiammatori, il che suggerisce che questi farmaci potrebbero avere un ruolo nel contrastare lo sviluppo della malattia [6].

### **1.1.3 Tipi di immagini MRI e caratteristiche rilevanti per la malattia di Alzheimer**

La risonanza magnetica (MRI) è una procedura non invasiva che permette di ottenere immagini tridimensionali dettagliate del corpo umano. Viene utilizzata principalmente per diagnosticare e monitorare condizioni che riguardano i tessuti molli, come muscoli, legamenti e, in particolare, il cervello. Nel caso del cervello, le scansioni MRI sono in grado di distinguere tra materia bianca e materia grigia, risultando utili per la rilevazione di tumori, aneurismi e altre patologie. Rispetto ad altre tecniche, come le radiografie a raggi X, la MRI ha il vantaggio di non emettere radiazioni ionizzanti, rendendola più sicura per esami frequenti.

Il principio di funzionamento della MRI si basa sulla generazione di un potente campo magnetico che allinea i protoni del corpo umano lungo le sue linee di campo. Successivamente, una corrente a radiofrequenza, che ha una frequenza compresa tra 3 KHz e 300 GHz, viene applicata al corpo del paziente, perturbando

l'allineamento dei protoni. Quando la corrente viene interrotta, i protoni tornano gradualmente alla loro posizione iniziale di allineamento con il campo magnetico. Il tempo di riallineamento e la quantità di energia rilasciata variano a seconda del tipo di molecola. Queste differenze vengono rilevate dall'apparecchio e tradotte in immagini dettagliate che consentono ai medici di distinguere tra diversi tipi di tessuti in base alle loro proprietà magnetiche [8].

L'analisi mediante risonanza magnetica (MRI) rappresenta un utile strumento per l'identificazione precoce della malattia. Le immagini ottenute dallo scan permettono di rilevare diversi indicatori patologici, come la riduzione del volume dell'ippocampo, che avviene già nelle fasi iniziali. Inoltre, si possono osservare perdite di volume nella corteccia cerebrale, in particolare nelle aree temporali e parietali, accompagnate da un'espansione dei ventricoli e dei solchi cerebrali, conseguente all'atrofia progressiva delle cellule nervose [9]. Sebbene le immagini MRI non siano sufficienti per poter effettuare la diagnosi, possono essere un ottimo strumento di supporto per escludere ulteriori possibili patologie ed evidenziare aree del cervello che potrebbero essere danneggiate dall'Alzheimer.

## 1.2 Reti Neurali Convoluzionali (CNN)

### 1.2.1 Architettura delle CNN

Il sistema nervoso è composto da unità cellulari chiamate neuroni, i quali rivestono un ruolo fondamentale nello sviluppo delle funzioni cognitive, tra cui il pensiero e il ragionamento, attraverso la trasmissione di impulsi elettrici. Il neurone costituisce l'elemento base del cervello umano e ha la capacità di ricevere segnali elettrici tramite le proprie estensioni chiamate dendriti. Se la somma pesata dei segnali in ingresso supera una soglia specifica di attivazione, il neurone genera un impulso elettrico che si propaga lungo l'assone.

Ogni assone è connesso ai dendriti di un altro neurone, consentendo così la trasmissione di segnali sotto forma di potenziali d'azione, comunemente noti come spike. Ciascun neurone è in grado di stabilire connessioni con un numero variabile di altri neuroni, tipicamente compreso tra 1.000 e 10.000.

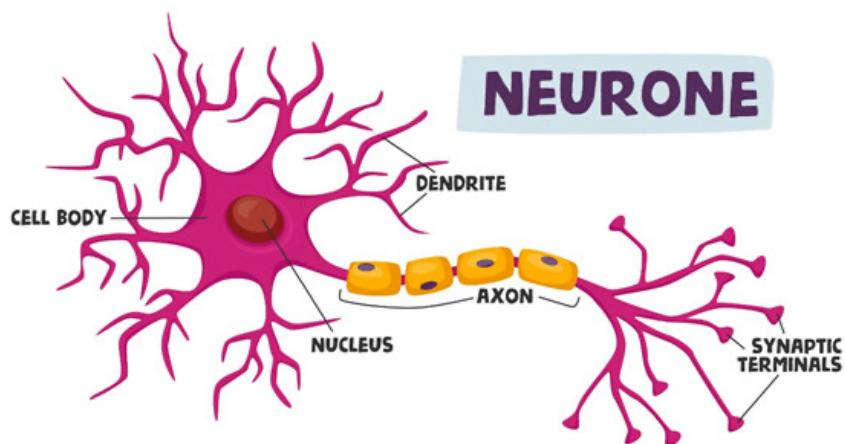
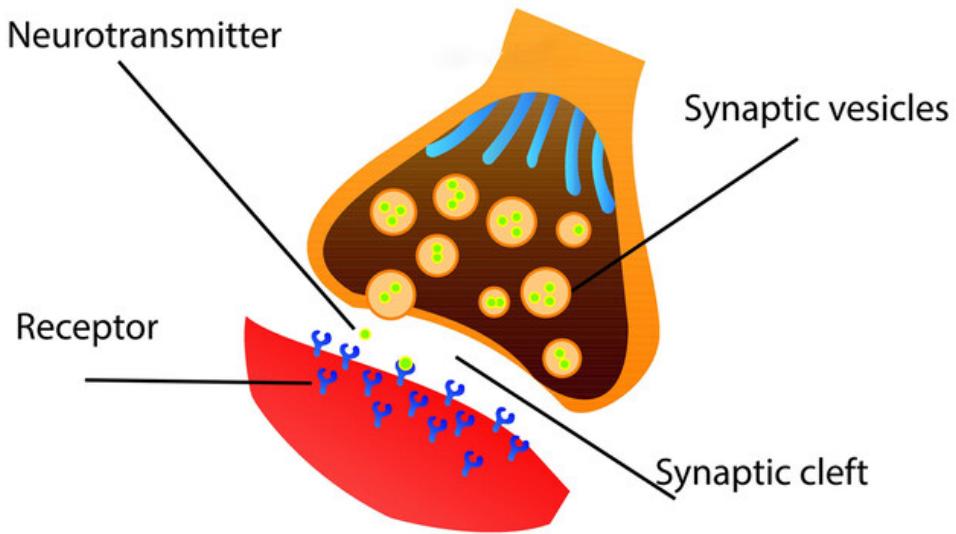


Figura 2. Descrizione grafica del singolo neurone (fonte: <https://mood-disorders.co.uk/glossary/what-are-neurones>)

La propagazione dei potenziali d'azione è resa possibile dall'apertura di canali ionici specifici per il sodio e il potassio, situati sulla membrana neuronale. Quando il processo di depolarizzazione supera una determinata soglia, si genera un effetto a cascata che provoca una rapida variazione del potenziale di membrana, il quale passa da circa -70 mV a +30 mV per un tempo dell'ordine di un millisecondo. Questo meccanismo rappresenta la base della trasmissione degli impulsi elettrici all'interno della rete neuronale.

I neuroni sono connessi tra loro attraverso sinapsi, che possono essere di due tipi:

- Sinapsi elettriche, nelle quali l'impulso nervoso si propaga direttamente dalla cellula presinaptica a quella postsinaptica;
- Sinapsi chimiche, in cui la cellula presinaptica converte il potenziale d'azione in un rilascio di neurotrasmettitori. Questi si legano ai recettori presenti sulla membrana della cellula postsinaptica, generando un nuovo segnale elettrico.

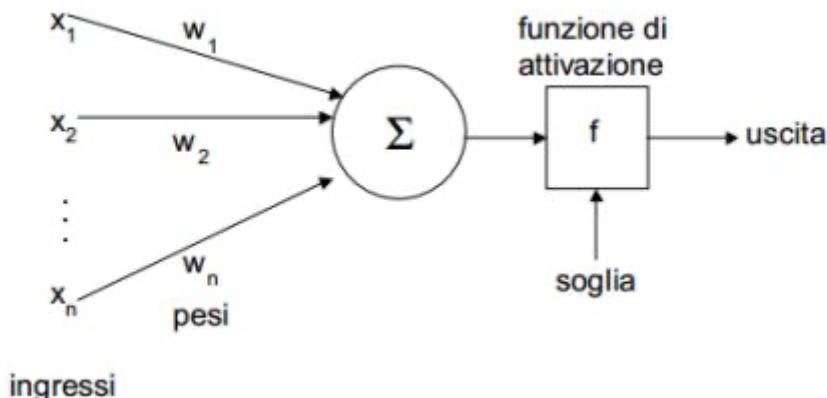


*Figura 3. Descrizione grafica di una sinapsi (fonte: <https://mood-disorders.co.uk/glossary/what-are-neurones>)*

Questo processo di trasmissione delle informazioni può essere modellato matematicamente. Il neurone artificiale è una rappresentazione semplificata che imita il comportamento del neurone biologico. A ciascun input del neurone artificiale è associato un peso, che può avere valore positivo o negativo, a seconda che stimoli o inibisca l'attivazione del neurone. Se la somma pesata degli input supera la soglia di attivazione, il neurone genera un'uscita. Il bias regola la predisposizione del neurone all'attivazione, modificando il valore di soglia.

L'algoritmo del neurone artificiale può essere descritto nei seguenti passaggi:

1. Caricare i valori degli input  $x_i$  e dei relativi pesi  $w_i$ .
2. Calcolare la somma degli input pesati  $\sum w_i x_i$ .
3. Applicare il risultato della somma pesata alla funzione di attivazione  $g(\sum w_i x_i)$ .
4. L'output del neurone è il risultato della funzione di attivazione  $y = g(\sum w_i x_i + b)$ .



*Figura 4. Archittetura di una rete neurale artificiale (fonte: <https://meccanicaefonderia.it/diagnostica-e-reti-neurali-artificiali/>)*

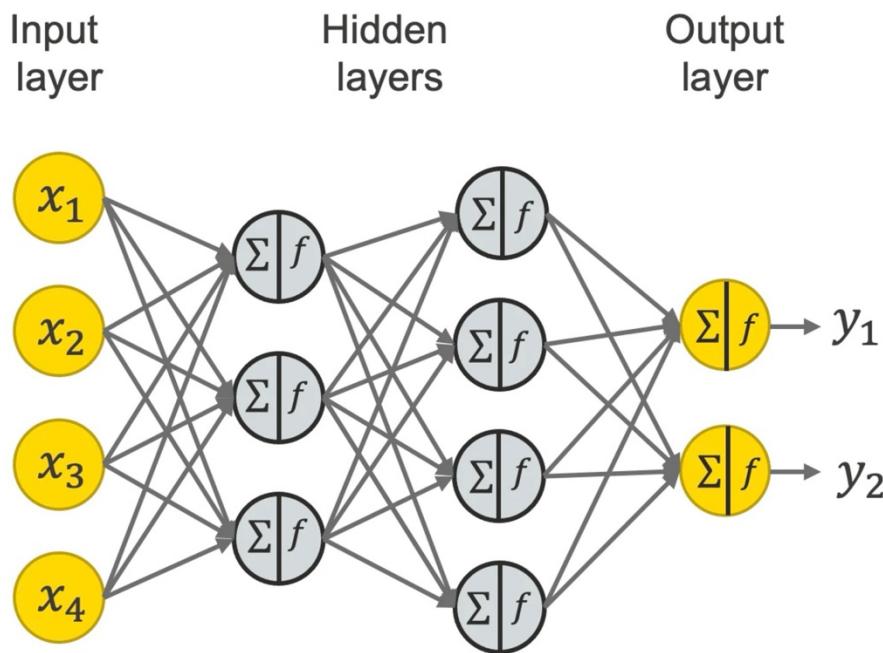
Una volta definiti questi neuroni artificiali, è possibile costruire strutture a grafo orientato, note come reti neurali, costituite da nodi, i neuroni artificiali, ed archi, i pesi sinaptici. Queste reti sono formate da uno strato di input, eventuali strati nascosti e uno strato di output.

I vantaggi nell'utilizzo di reti neurali sono i seguenti:

- Facilità di costruzione, è sufficiente creare copie di neuroni artificiali per poi collegarli tra loro;
- Elevato parallelismo, che permette a grandi moli di dati di essere processati in tempi relativamente brevi;
- Tolleranza al rumore, poiché l'output riesce a rimanere stabile anche in presenza di rumore, imprecisione e incompletezza nei dati di input;
- Adatte per implementare problemi che non chiedono risposte accurate, ma risposte approssimate con un basso grado di errore.

Gli svantaggi sono i seguenti:

- Alto consumo di risorse computazionali e di tempo per l'addestramento della rete;
- Non è possibile, a differenza della programmazione tradizionale, capire perché la rete restituisce un risultato specifico;
- Necessità di un dataset sufficientemente grande e vario per l'addestramento;
- Non è possibile avere la certezza a priori che un problema sarà risolto;



*Figura 5. Esempio di una rete neurale completamente connessa (fonte: <https://www.knime.com/blog/a-friendly-introduction-to-deep-neural-networks>)*

La programmazione della rete permette di definirne la struttura, mentre la fase di apprendimento permette ad essa di capire come comportarsi con gli input, dato che all'inizio la rete non ha alcuna forma di conoscenza, che verrà accumulata mediante la variazione dei pesi attraverso algoritmi di ottimizzazione.

L'addestramento della rete può distinguersi principalmente in:

- Apprendimento supervisionato;
- Apprendimento non supervisionato.

Nell'apprendimento supervisionato viene presentato alla rete un training set, preparato da un supervisore esterno, composto da dati etichettati, ossia dati dal valore di output associato noto.

In un processo di apprendimento supervisionato:

1. La rete utilizza l'input per determinare un output.
2. L'errore viene determinato come differenza tra l'output e l'output atteso.
3. La rete modifica i pesi in base all'errore cercando di minimizzarlo e commetterà sempre meno errori.

Nell'apprendimento non supervisionato invece si fornisce un training set di dati non etichettati e si utilizzano particolari algoritmi che permettano alla rete di rilevare schemi di somiglianza sulla base delle loro caratteristiche.

La rete neurale più semplice è il perceptron, un classificatore binario. Questo modello è composto da uno strato di ingresso e uno strato di uscita, in cui ogni neurone dello strato di input è collegato a ciascun neurone dello strato di output. Non ci sono connessioni tra neuroni all'interno dello stesso strato. Le unità di uscita utilizzano una funzione a soglia a gradino per determinare la classe a cui appartiene l'input.

L'algoritmo di addestramento andrà a manipolare la pendenza della retta di separazione fin quando le due classi non saranno correttamente separate. Tuttavia, nel caso di classi non linearmente separabili, un percepitrone semplice non sarà sufficiente. In questi casi, si ricorre al percepitrone multistrato (Multilayer Perceptron, MLP), che include uno strato di input, uno strato di output e almeno uno strato nascosto nel livello intermedio. Questo strato nascosto permette alla rete di apprendere rappresentazioni più complesse e di gestire regioni non linearmente separabili, migliorando così la capacità di classificazione del modello.

L'attività del percepitrone si basa principalmente su due meccanismi:

- Un algoritmo di ottimizzazione, come la discesa del gradiente;
- Un algoritmo di addestramento, come la propagazione all'indietro.

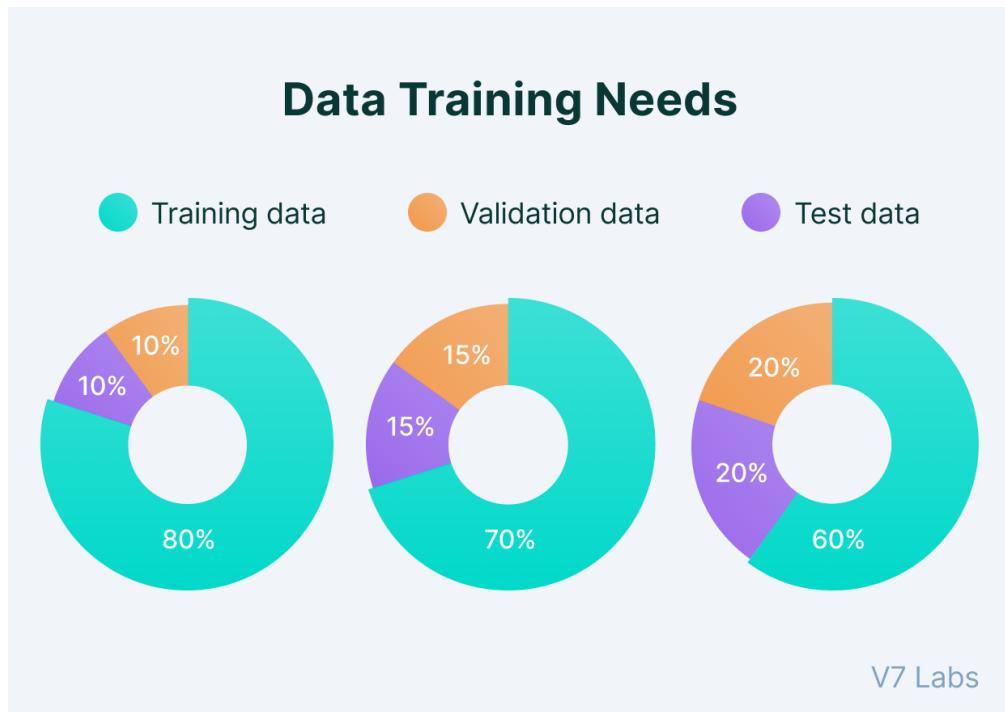
La discesa del gradiente è un algoritmo utilizzato per ottimizzare i parametri di una rete neurale, cioè i pesi, al fine di minimizzare la funzione di costo che misura l'errore tra le uscite attese e quelle predette dal modello. Geometricamente, il gradiente di una funzione a più variabili rappresenta la direzione di massimo accrescimento della funzione. Quindi, procedendo contro gradiente si individua la direzione in cui la funzione diminuisce più rapidamente, permettendo di ridurre l'errore e migliorare le prestazioni del modello.

La propagazione all'indietro o backpropagation, è un algoritmo di apprendimento delle reti neurali, organizzato in due fasi: una fase in avanti e una fase all'indietro. Nella fase in avanti, l'input viene propagato in avanti all'interno della rete, per generare una predizione. La predizione viene poi confrontata con il valore noto di output associato a quell'input per ottenere un errore. Nella fase di propagazione all'indietro, viene trasmesso l'errore così ottenuto per aggiornare i pesi.

Per garantire un efficace addestramento del modello, è essenziale disporre di un dataset statisticamente rappresentativo. Tale dataset viene suddiviso in tre insiemi principali:

- **training set**, o dataset di addestramento: rappresenta circa l'80% del dataset generale e contiene immagini etichettate che vengono utilizzate per addestrare la rete. Durante questo processo, il modello apprende regolando i pesi attraverso il meccanismo di propagazione all'indietro al fine di minimizzare la funzione di costo. Un training set sufficientemente ampio è fondamentale per permettere alla rete di riconoscere anche piccole differenze tra immagini simili, migliorando così la capacità del modello di distinguere i pattern rilevanti per il compito specifico.
- **validation set**, o dataset di validazione: questo insieme di immagini etichettate ha lo scopo di valutare la capacità del modello di generalizzare. Durante l'addestramento, il validation set viene utilizzato per monitorare le performance della rete su dati non inclusi nel training set, e per eseguire il tuning degli iperparametri (come il tasso di apprendimento, il numero di layer o la dimensione del batch), ossia tutti quei parametri che non vengono modificati dall'addestramento della rete, ma che vengono decisi a priori. L'obiettivo è evitare fenomeni di overfitting, ossia un eccessivo adattamento ai dati di addestramento che riduce la capacità del modello di generalizzare su nuovi dati. D'altro canto, è importante prevenire anche l'underfitting, in cui il modello non riesce a cogliere adeguatamente le informazioni dai dati, risultando in un'eccessiva semplificazione del problema.

- **testing set**, o dataset di test: viene utilizzato una volta completato l'addestramento del modello per valutare in modo oggettivo le sue performance finali. Il test set include dati mai visti prima dal modello e consente di misurare la capacità di generalizzazione su nuove immagini, attraverso metriche di valutazione come accuratezza, precisione, recall e F1-score. Questo passaggio è cruciale per verificare l'efficacia del modello nel risolvere il problema per cui è stato progettato.



*Figura 6. Esempio di proporzioni tra dataset (fonte: <https://www.v7labs.com/blog/train-validation-test-set>)*

Una rete neurale convoluzionale (CNN) è un particolare tipo di rete neurale, specializzata nel riconoscimento di immagini e segnali audio. Un'immagine può essere rappresentata come una matrice di valori, chiamati pixel, che forniscono informazioni sull'intensità della luce in ciascun punto dell'immagine. Nel caso di immagini in scala di grigi, l'immagine è rappresentata da una singola matrice, dove ogni elemento descrive l'intensità del grigio per un determinato pixel.

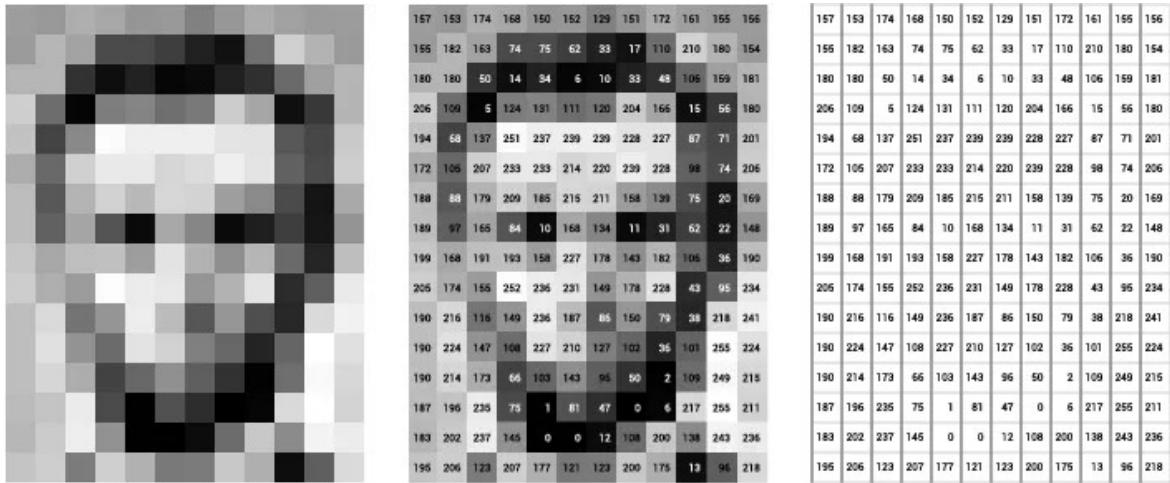


Figura 7. Forma matriciale di un'immagine in scala di grigi (fonte:

<https://medium.com/thedepthub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175>)

Per le immagini a colori, invece, solitamente nel formato RGB (Red, Green, Blue), l'immagine è composta da tre canali distinti: uno per il rosso, uno per il verde e uno per il blu. Ogni canale è rappresentato da una matrice separata, e ciascun elemento della matrice indica l'intensità del colore corrispondente per un dato pixel.

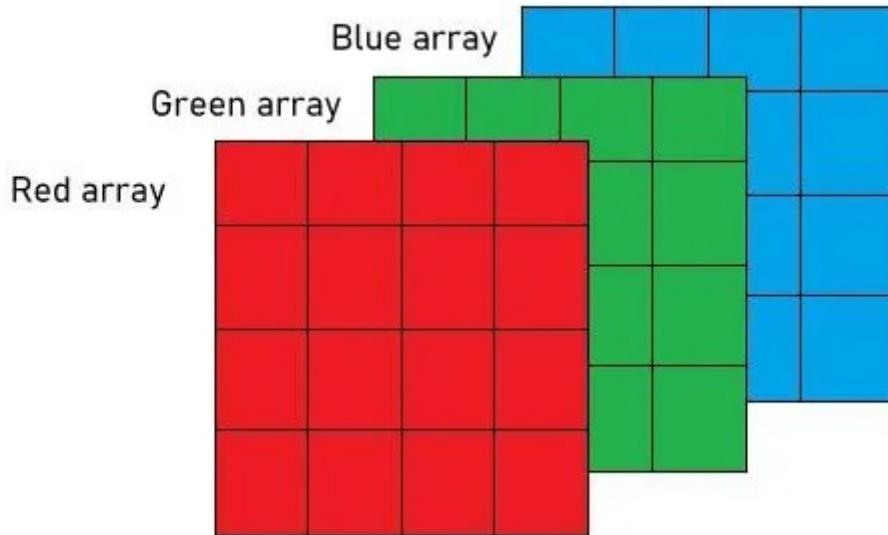


Figura 8. Forma matriciale di un'immagine RGB (fonte: <https://www.geeksforgeeks.org/matlab-rgb-image-representation/>)

Una CNN è formata da due parti principali: la prima si occupa dell'estrazione automatica delle caratteristiche rilevanti dell'immagine (features), come bordi, colori, texture e forme; la seconda parte, costituita da strati completamente connessi (fully connected), è responsabile dell'elaborazione finale e della classificazione dell'immagine sulla base delle feature estratte.

## Convolution Neural Network (CNN)

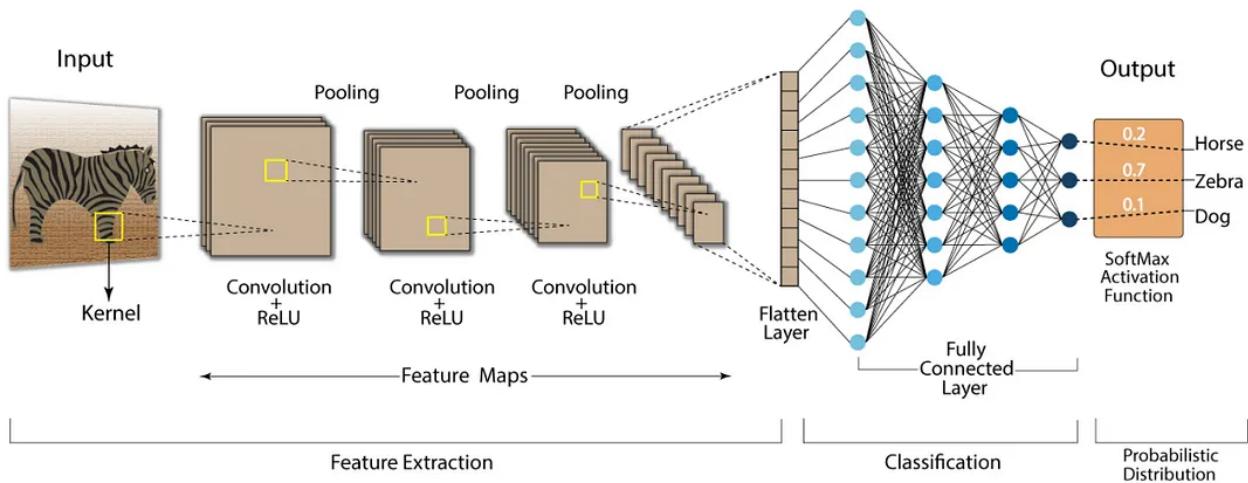


Figura 9. Esempio di una struttura di una CNN (fonte: <https://developersbreach.com/convolution-neural-network-deep-learning/>)

Il primo componente di una rete neurale convoluzionale (CNN) è lo strato di convoluzione. La convoluzione è un'operazione matematica che combina due funzioni, producendo una terza funzione che descrive il grado di sovrapposizione tra le funzioni coinvolte. Nel caso continuo la convoluzione è espressa dall'integrale:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau$$

Mentre nel caso discreto la convoluzione è espressa da:

$$(f * g)[n] = \sum_{k=-\infty}^{\infty} f[k]g[n - k]$$

Per eseguire questo processo, si utilizzano filtri specializzati, chiamati kernel. Questi sono piccole matrici di valori che, scorrendo sull'immagine, generano una nuova matrice nota come feature map. Durante il movimento del kernel sull'immagine di input, viene calcolata la somma dei prodotti dei valori sovrapposti in ogni posizione. Questo procedimento consente di estrarre caratteristiche significative dall'immagine, rappresentate successivamente nella feature map. Prendendo come esempio un kernel 2x2:

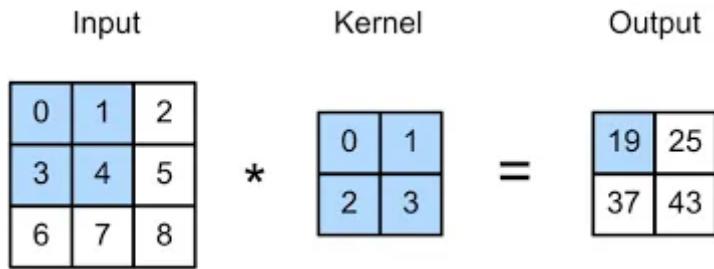


Figura 10 Esempio di convoluzione tra un input 3x3 e un kernel 2x2 (fonte:

<https://medium.com/thedepthub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175>)

Si posiziona il kernel in alto a sinistra dell'input e si effettua la somma dei prodotti dei valori attualmente sovrapposti:

$$- \quad (0 \times 0) + (1 \times 1) + (3 \times 2) + (4 \times 3) = 19$$

Si esegue uno spostamento del kernel di un pixel e si esegue la medesima operazione:

$$- \quad (1 \times 0) + (2 \times 1) + (4 \times 2) + (5 \times 3) = 25$$

Dopo aver completato la prima riga si sposta il kernel alla riga inferiore e si ripete il processo:

$$- \quad (3 \times 0) + (4 \times 1) + (6 \times 2) + (7 \times 3) = 37$$

Si esegue un ulteriore spostamento di un pixel:

$$- \quad (4 \times 0) + (5 \times 1) + (7 \times 2) + (8 \times 3) = 43$$

La matrice di output così ottenuta è conosciuta come feature map.

Nel caso delle immagini RGB, vengono utilizzati kernel per ogni canale, poiché alcune caratteristiche potrebbero risultare più evidenti in un canale rispetto agli altri. Ogni strato convoluzionale dispone di un numero definito di kernel, e ciascuno di questi genera una feature map separata. Di conseguenza, l'insieme di tutte le feature map generate costituisce l'output completo dello strato convoluzionale.

Il kernel è caratterizzato anche da uno stride, un parametro che determina di quanti pixel il kernel si sposta sull'immagine di input ad ogni passo di convoluzione. Un valore maggiore di stride riduce le dimensioni dell'output, consentendo un'attenzione particolare alle caratteristiche globali dell'immagine; questo approccio è utile per rilevare pattern e informazioni generali, ma potrebbe comportare la perdita di dettagli più fini. Al contrario, un valore minore di stride porta a dimensioni spaziali dell'output più ampie, preservando così dettagli più fini e locali. In questo modo, la rete è in grado di catturare informazioni più dettagliate sulle caratteristiche presenti nell'immagine.

Un ulteriore parametro particolarmente significativo è il padding, che consiste nell'aggiungere pixel attorno ai bordi dell'immagine. Questo è importante perché i pixel situati ai bordi vengono analizzati un numero inferiore di volte rispetto a quelli posizionati centralmente. Di conseguenza, le caratteristiche che si concentrano lungo i bordi potrebbero non ricevere l'attenzione adeguata durante l'elaborazione. Pertanto, l'aggiunta di padding è consigliata quando si desidera catturare informazioni significative situate ai bordi dell'immagine, assicurando che tali caratteristiche siano adeguatamente considerate nel processo di convoluzione. Il padding può anche essere asimmetrico rispetto ai bordi dell'immagine, ad esempio maggiore nel lato destro rispetto a quello sinistro.

Le dimensioni dell'output, nel caso di padding simmetrico, si calcola secondo la formula:

$$\text{Output Size} = \frac{\text{Input size} + 2 \times \text{Padding} - \text{Kernel Size}}{\text{Stride}} + 1$$

Dove il “2xPadding” si riferisce a un padding applicato a entrambi i lati orizzontali, o verticali, dell'immagine, mentre il “+1” tiene conto della posizione di partenza del kernel.

Disposto dopo lo strato di convoluzione, si trova lo strato di pooling. Lo scopo di questo strato è di ridurre la dimensionalità dell'output di convoluzione, preservando il più possibile le features importanti. Riduzione che viene effettuata attraverso lo stesso meccanismo di scorrimento di un kernel, in questo caso vuoto, lungo l'immagine, con la differenza fondamentale che non viene eseguita l'operazione di convoluzione ma vengono eseguite funzioni matematiche come il “Max pooling” o il “Average pooling”. Queste operazioni consentono di estrarre i valori massimi o medi da sotto-aree dell'immagine, semplificando così la rappresentazione dei dati. L'operazione di pooling non modifica in alcun modo il numero di canali del dato, in quanto il pooling viene effettuato per ognuno di essi.

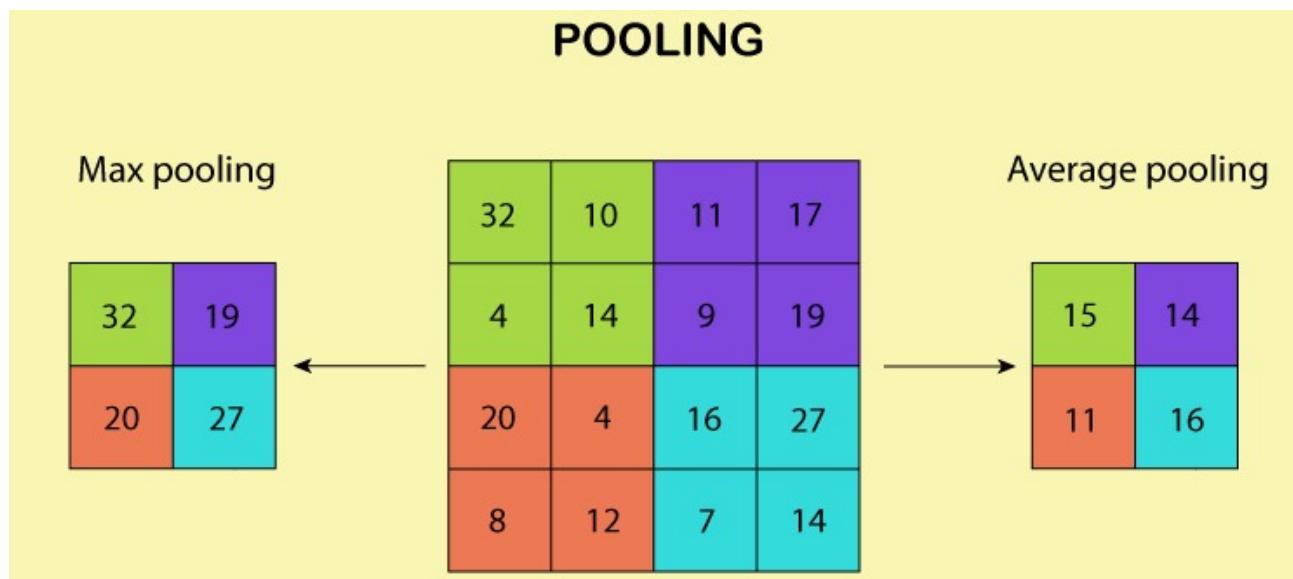


Figura 11. Esempio di max pooling e average pooling (fonte:

<https://medium.com/@prathammodi001/convolutional-neural-networks-for-dummies-a-step-by-step-cnn-tutorial-e68f464d608f>)

A seguito dello strato di pooling, si trova lo strato di flattening, il cui obiettivo è trasformare l'output di pooling, costituito da un insieme di feature map ridotte, in un array monodimensionale. Le feature maps hanno una struttura del tipo Base x Altezza x Profondità, dove Base x Altezza rappresenta la densità di pixel di ciascuna feature map e Profondità indica il numero totale di feature map generate. Questa trasformazione è fondamentale in quanto l'array monodimensionale, è il formato compatibile con gli strati completamente connessi che seguiranno.

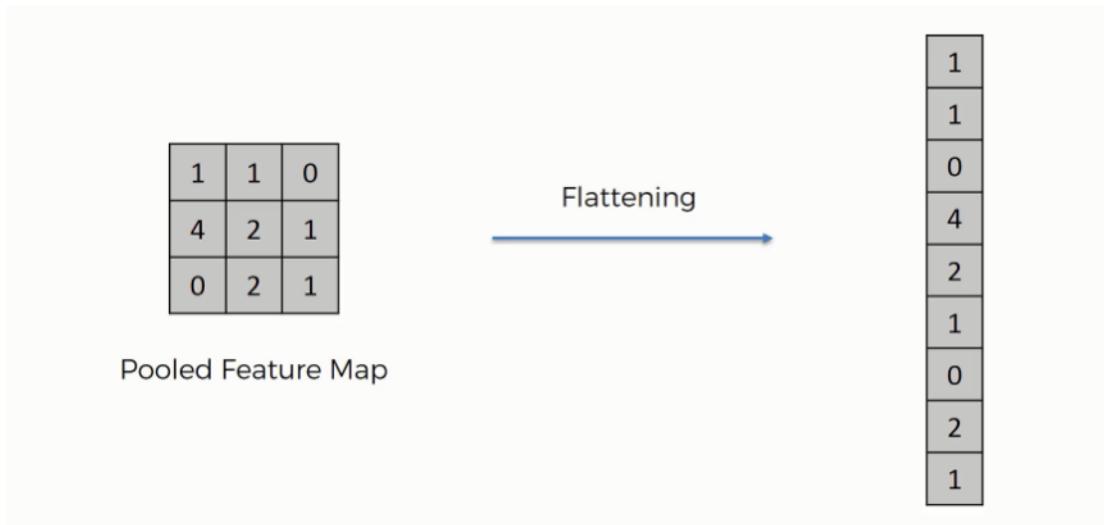


Figura 12 Flattening di una feature map fonte: (<https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening>)

Gli strati completamente connessi infine si occupano della logica della rete neurale e attraverso il processo di automiglioramento sono in grado di calibrare i propri pesi in maniera tale da minimizzare la funzione di costo scelta ed eseguire la classificazione sulla base delle features estratte precedentemente.

[10]

### 1.2.2 Funzioni di attivazione

Come spiegato in precedenza la rete neurale si basa su delle funzioni di attivazione per poter associare un output ad ogni somma pesata di input.

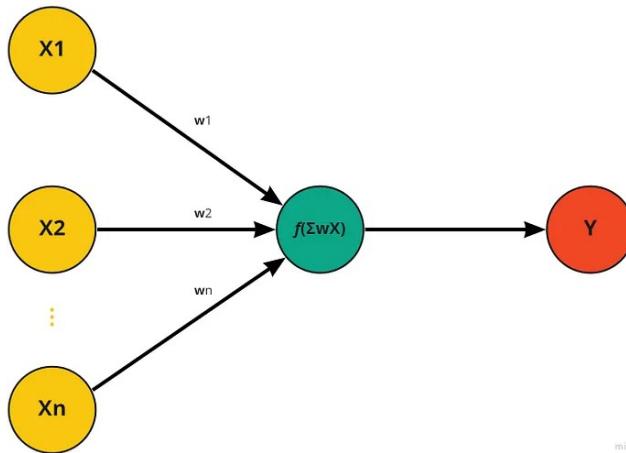


Figura 13 Rappresentazione di un neurone artificiale (fonte: <https://medium.com/@paolopiacentil/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>)

La funzione di attivazione  $f()$  viene applicata alla  $\sum x_i w_i$  tale che l'output sia  $Y = f(\sum x_i w_i)$

Tra le funzioni di attivazione più utilizzate, ci sono:

La **funzione a soglia**, genera un output pari ad 1 nel caso in cui la somma pesata degli input sia maggiore del valore di soglia scelto, altrimenti restituisce zero.

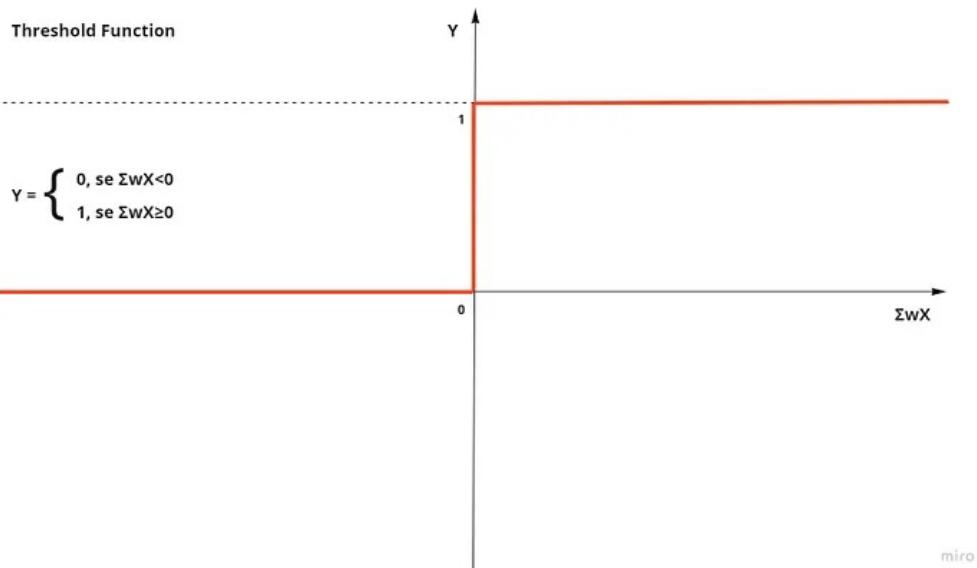


Figura 14 Funzione a soglia (fonte: <https://medium.com/@paolopiacentil/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>)

$$u(x): \mathbb{R} \rightarrow \mathbb{R}$$

$$u(x) = \begin{cases} 0, & x < \text{valore di soglia} \\ 1, & x \geq \text{valore di soglia} \end{cases}$$

È utilizzata principalmente per modellare problemi di classificazione binaria, anche se non è utilizzata davvero, in quanto:

- Non utilizzabile per classificazioni multiclassse
- Non derivabile a causa del punto di discontinuità in corrispondenza del valore di soglia, ciò comporta delle criticità nell'algoritmo di discesa del gradiente, e dunque un'impossibilità nell'aggiornamento dei pesi mediante backpropagation.

La **funzione sigmoide** è una funzione di attivazione non lineare che restituisce valori nell'intervallo continuo compreso tra 0 e 1. Questa caratteristica la rende particolarmente adatta per applicazioni di classificazione binaria. L'output della funzione può essere interpretato come una probabilità, poiché assume valori compresi tra 0 e 1. Confrontando questo valore di probabilità con una soglia predefinita, è possibile classificare i dati di input in due categorie distinte.

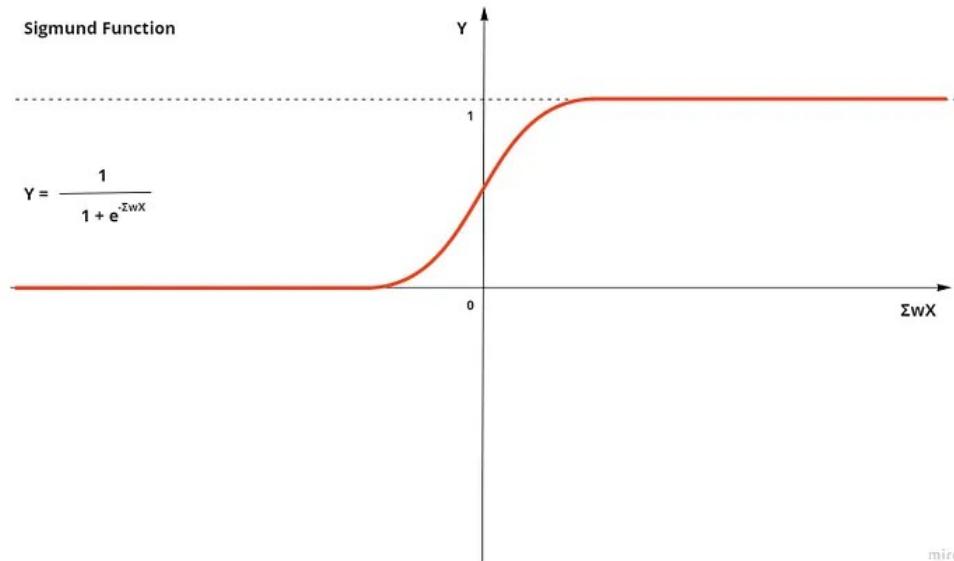


Figura 15 Funzione sigmoide (fonte: <https://medium.com/@paolopiacentil/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>)

$$\sigma(x) : \mathbb{R} \rightarrow \mathbb{R}$$

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

Ormai non più tanto utilizzata in quanto:

- Funzione non centrata nello zero: Gli output della sigmoide sono sempre positivi o nulli. Questo può creare squilibri durante il processo di discesa del gradiente, poiché gli aggiornamenti dei pesi potrebbero andare in direzioni incoerenti, rallentando la convergenza o complicando l'ottimizzazione complessiva del modello.
- Scomparsa del gradiente: per valori molto lontani dallo zero la funzione tende a saturare, portando il gradiente ad avvicinarsi allo zero con conseguente rallentamento o arresto dell'apprendimento.

La **funzione rettificatrice** o ReLU, è la funzione di attivazione non lineare più utilizzata: essa restituisce 0 nel caso in cui la somma pesata degli input sia minore o uguale a zero, altrimenti in caso di valore positivo restituisce il valore stesso della somma pesata.

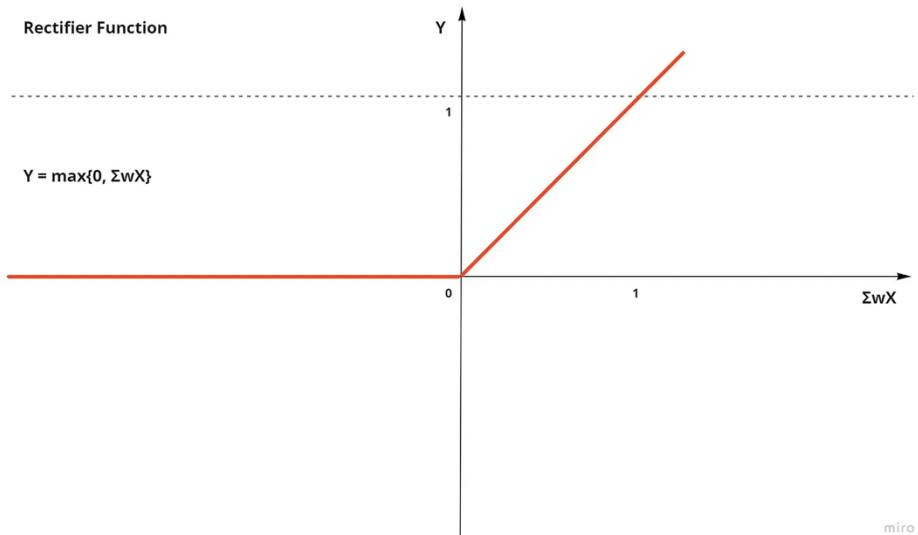


Figura 16 Funzione rettificatrice (fonte: <https://medium.com/@paolopiacenti/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>)

$$ReLU(x) : \mathbb{R} \rightarrow \mathbb{R}$$

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

$$ReLU(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

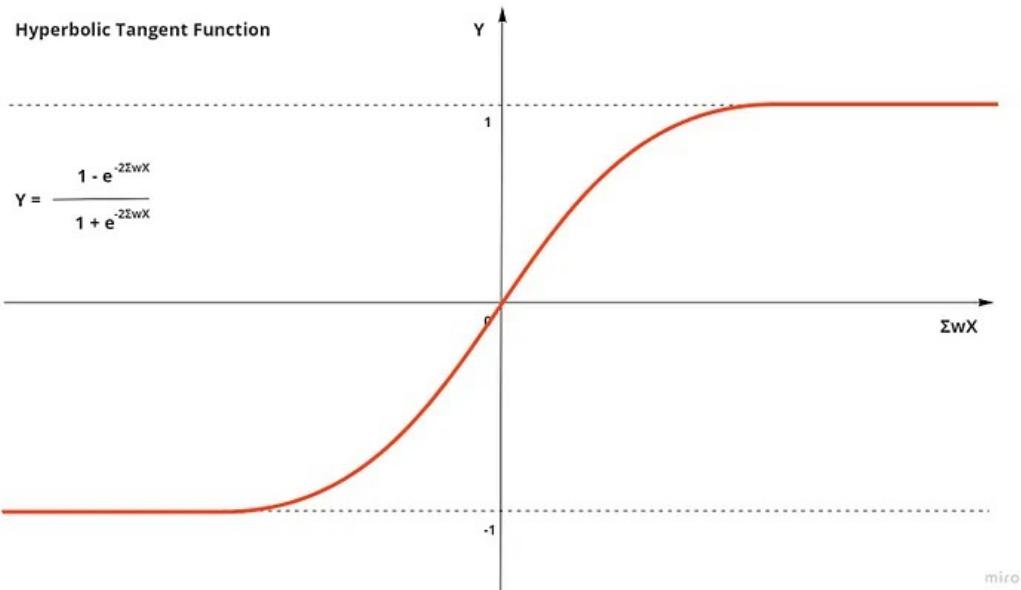
La ReLU non soffre del problema della scomparsa del gradiente, rendendola particolarmente adatta per l'addestramento di reti neurali profonde. Inoltre, la sua semplicità computazionale la rende preferibile rispetto a funzioni di attivazione più complesse, come quelle esponenziali o trigonometriche. Grazie alla sua struttura semplice, la ReLU può essere calcolata molto rapidamente, un aspetto cruciale quando si lavora con reti neurali di grandi dimensioni, dove l'efficienza computazionale è fondamentale per velocizzare il processo di addestramento.

Tuttavia, anch'essa presenta problematiche quali:

- Morte dei neuroni: si verifica quando alcuni neuroni, a causa degli aggiornamenti dei pesi, smettono di attivarsi correttamente, restituendo sempre zero come output. Ciò accade quando i pesi di un neurone sono modificati in modo tale che l'input del neurone risulti sempre negativo prima dell'applicazione della funzione ReLU. Poiché la ReLU restituisce zero per valori negativi, l'output del neurone diventa costantemente zero, causando un gradiente nullo durante il backpropagation. Di conseguenza, il neurone smette di imparare, rimanendo inattivo e, di fatto, "morto".
- Funzione non centrata nello zero.

La **tangente iperbolica** (tanh) ha un comportamento simile alla funzione sigmoide, ma con una differenza significativa: i suoi output sono compresi nell'intervallo  $[-1, 1]$ , rendendola centrata rispetto allo zero. Questo è un vantaggio importante rispetto alla sigmoide, poiché la tanh permette ai valori negativi di influenzare le attivazioni, contribuendo a un migliore bilanciamento nella rete. In altre parole, la centratura intorno a zero facilita una più equilibrata distribuzione degli output, il che aiuta a evitare aggiornamenti sbilanciati dei pesi durante il processo di discesa del gradiente.

Tuttavia, la funzione tanh soffre anch'essa del problema della scomparsa del gradiente. Quando gli input sono molto grandi o molto piccoli, la funzione satura e gli output si avvicinano rispettivamente a 1 o -1. In queste regioni saturate, il gradiente si avvicina a zero, il che rende difficile per la rete aggiornare correttamente i pesi e continuare ad apprendere in modo efficace. Questo fenomeno rallenta notevolmente l'addestramento, soprattutto in reti profonde, dove i segnali si attenuano man mano che si propagano all'indietro [11].



*Figura 17 Funzione tangente iperbolica (fonte: <https://medium.com/@paolopiacenti/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>)*

$$\tanh(x) : \mathbb{R} \rightarrow \mathbb{R}$$

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

### 1.2.3. Applicazioni delle CNN nell'elaborazione di immagini MRI

L'utilità principale di una CNN è la sua capacità di estrarre in maniera automatica le features di un'immagine, attraverso il processo di convoluzione tra l'immagine di input e i kernel, che permettono alla rete di apprendere pattern visivi complessi come bordi, texture e forme.

Nel caso di risonanze magnetiche dell'area del cervello, le CNN possono essere utilizzate per classificare i pazienti in categorie, sulla base di determinate features rilevate. Le classificazioni possono essere binarie, del tipo soggetto sano e soggetto malato, oppure multiclass, ad esempio suddividere i soggetti sulla base della gravità e dell'avanzamento della malattia. Possono essere utilizzate anche per procedure di segmentazione automatica, ossia il processo con cui un'immagine viene suddivisa in regioni con caratteristiche simili come colore e intensità, col fine di isolare regioni di interesse come potrebbero essere tumori, ippocampo e corteccia cerebrale.

Ad esempio, nello studio [12], gli autori hanno presentato una procedura per l'utilizzo dei dati della risonanza magnetica per classificare i tumori cerebrali in gruppi distinti. Il modello di deep learning ha raggiunto un'accuratezza del 96,13% nel compito di identificare tumori come gliomi, meningiomi o tumore all'ipofisi. Gli autori dello studio [13] hanno basato i loro modelli per la rilevazione dei tumori cerebrali sui pre-addestrati Inception V3, ResNet-50 e VGG-16. ResNet-50 è risultato essere il migliore con un'accuratezza del 95%. Gli autori dello studio [14] hanno proposto un approccio all'estrazione automatica delle features relative alla classificazione dei tumori al cervello da un dataset di MRI, usando kernel 3x3. Il nuovo modello ha raggiunto un'accuratezza del 94,74%.

Lo studio [15] è riuscito a basare il proprio modello su ResNet-50 per la classificazione dei casi di Alzheimer sul dataset di MRI noto come ADNI, con un range di accuratezza tra 85.7% e 99%.

## **Capitolo 2 – Materiali e metodi**

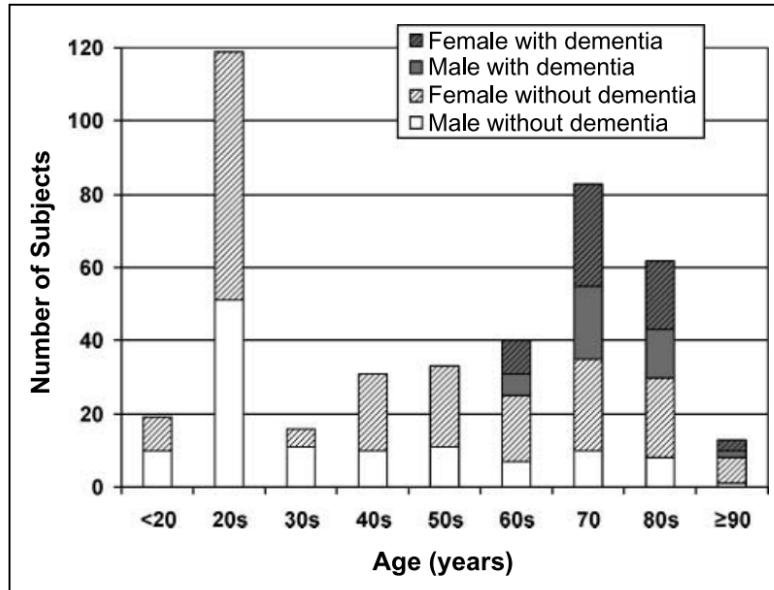
### **2.1 Dataset OASIS 1**

Il dataset utilizzato per questo progetto di tesi è l’OASIS (Open Access Series of Imaging Studies), ovvero il risultato di un progetto effettuato con lo scopo di realizzare immagini di risonanza dell’area cerebrale che potesse essere gratuitamente disponibile per la comunità scientifica. Fu realizzato dalla Washington University nel 2010, successivamente alla selezione di 416 individui con età compresa tra i 18 e i 96 anni, di cui 100 individui con diagnosi di malattia di Alzheimer tra lo stadio molto lieve e lo stato moderato, classificati secondo la scala CDR. I soggetti sono stati selezionati da un campione molto più ampio di individui che hanno partecipato a studi sulle MRI presso la Washington University, sulla base delle seguenti caratteristiche: soggetti destrimano, con disponibilità di almeno tre immagini pesate in T1, che abbiano ricevuto una valutazione clinica recente nel caso degli anziani [16].

I soggetti più giovani sono stati reclutati dalla comunità della Washington University, mentre quelli più anziani, con o senza demenza, sono stati reclutati tramite l’Alzheimer Disease Research Center (ADRC) della stessa università. Questi ultimi sono venuti a conoscenza dell’ADRC principalmente attraverso i media o il passaparola; di questi, circa l’80% ha contattato autonomamente il centro, mentre il resto è stato indirizzato dai propri medici.

Tutti i partecipanti sono stati sottoposti a uno screening per verificarne l’idoneità alla ricerca. I soggetti giovani e di mezza età sono stati intervistati da un tecnico qualificato prima della scansione, per raccogliere informazioni sulla loro storia medica e sull’uso di sostanze psicoattive. Gli anziani, di età pari o superiore a 60 anni, hanno invece ricevuto una valutazione clinica completa presso l’ADRC. Sono stati esclusi dallo studio i soggetti con forme di demenza diverse dall’Alzheimer, come la demenza vascolare o l’afasia primaria progressiva, oltre a chi presentava malattie neurologiche o psichiatriche, gravi traumi cranici, una storia di ictus clinicamente rilevante o l’uso di farmaci psicotropi. Anche i soggetti con evidenti anomalie anatomiche rilevate tramite risonanza magnetica, come grandi lesioni o tumori, sono stati esclusi. Tuttavia, sono stati inclusi i partecipanti con alterazioni cerebrali legate all’età, come leggere atrofie. Le acquisizioni di risonanza magnetica sono state generalmente effettuate entro un anno dalla valutazione clinica dei soggetti (media = 105 giorni, intervallo = 0-314 giorni). Tuttavia, tre soggetti affetti da malattia di Alzheimer (AD) sono stati sottoposti a scansione dopo un intervallo leggermente più lungo (635, 449 e 443 giorni). Il set di dati finale comprende un totale di 416 partecipanti. Inoltre, 20 soggetti ventenni sono stati sottoposti a una seconda sessione di imaging entro 90 giorni dalla prima, per fornire un parametro utile alla valutazione dell’affidabilità delle scansioni. Per ciascun soggetto sono state effettuate 3-4 risonanze magnetiche pesate in T1, acquisite in un’unica sessione utilizzando uno scanner Siemens Healthineers da 1.5T, presso Erlangen, Germania. Per ridurre al minimo i movimenti della testa, sono state utilizzate imbottiture e una maschera facciale termoplastica. Inoltre, ai partecipanti sono state fornite cuffie per facilitare la comunicazione durante la scansione, e una capsula di vitamina E è stata posizionata sul lato sinistro della fronte per garantire un riferimento chiaro dell’orientamento anatomico. I parametri di scansione sono stati opportunamente regolati per migliorare il contrasto tra la sostanza bianca e quella grigia del cervello. Il campione finale, dunque, è

costituito da 218 individui con età compresa tra i 18 e i 59 anni e 198 soggetti con età compresa tra i 60 e i 96 anni. Ciascun gruppo include approssimativamente un egual numero di maschi e femmine. Per quanto riguarda gli anziani, 98 hanno un CDR = 0 e dei restanti 100 anziani: 70 hanno un CDR = 0.5, 28 con un CDR=1 e 2 con un CDR=2



*Figura 18. Distribuzione del dataset OASISI al variare dell'età*

**Table 2.** Age and Diagnosis Characteristics of the Data Set

Age Group	Total n	Without Dementia			With Dementia					
		n	Mean	Male	Female	n	Mean	Male	Female	CDR 0.5/1/2
<20	19	19	18.53	10	9	0		0	0	0/0/0
20s	119	119	22.82	51	68	0		0	0	0/0/0
30s	16	16	33.38	11	5	0		0	0	0/0/0
40s	31	31	45.58	10	21	0		0	0	0/0/0
50s	33	33	54.36	11	22	0		0	0	0/0/0
60s	40	25	64.88	7	18	15	66.13	6	9	12/3/0
70s	83	35	73.37	10	25	48	74.42	20	28	32/15/1
80s	62	30	84.07	8	22	32	82.88	13	19	22/9/1
≥90	13	8	91.00	1	7	5	92.00	2	3	4/1/0
Total	416	316		119	197	100		41	59	

*Tabella 1. Tabella riassuntiva del dataset OASIS*

Per ogni soggetto i file di scansione sono stati convertiti dal formato IMA proprietario di Siemens nel formato Analyze 7.5 a 16 bit utilizzando un programma di conversione personalizzato. Le caratteristiche facciali sono state rimosse dalle immagini utilizzando il software del fMRI Data Center, al fine di non rendere riconoscibili i soggetti partecipanti. Nei 12 casi in cui il software ha lasciato una parte eccessiva del viso oppure ha tagliato in modo invasivo l'area cranica, la deidentificazione è avvenuta attraverso l'utilizzo del software fornito dal Morphometry Biomedical Informatics Research Network. Questo metodo registra l'immagine in un atlante in cui le caratteristiche facciali e il tessuto cerebrale sono stati etichettati a mano. A tutti i voxel nell'immagine

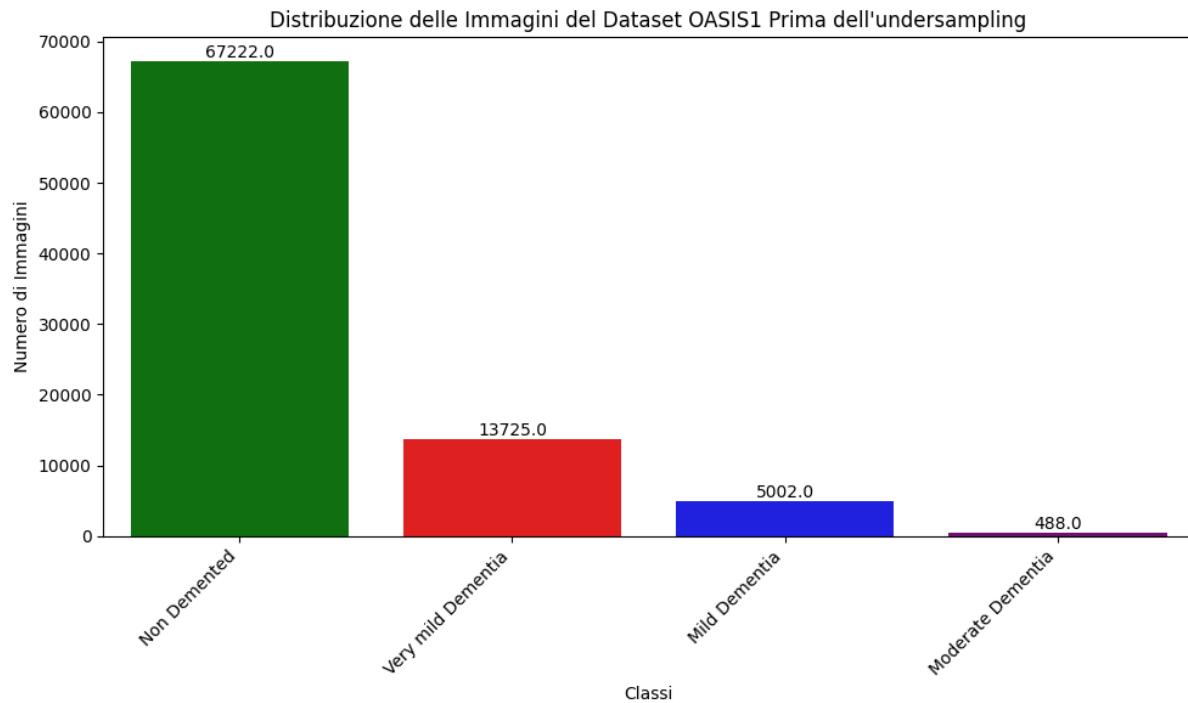
che hanno una probabilità nulla di appartenere a zone cerebrali e con probabilità non nulla di appartenere alla faccia viene assegnato un valore nullo di intensità. Infine, sono state eseguite correzioni di intensità nelle zone di non omogeneità a causa di un campo magnetico non uniforme.

Tutte le immagini sono state attentamente analizzate per rilevare problemi di acquisizione o errori di processamento. Ogni immagine tridimensionale è stata esaminata attraverso le sue slice lungo gli assi principali. Alle immagini è stato poi assegnato un punteggio di qualità da 1 a 3, dove 1 indica immagini di alta qualità, 2 segnala difetti minori, e 3 rappresenta immagini con difetti significativi, causati da rumore elettronico, distorsioni dovute a apparecchi dentali o un posizionamento errato della testa. Le immagini di livello 3 sono state eliminate dal dataset, mentre alcune immagini di qualità borderline sono state mantenute per fornire un intervallo realistico della qualità di acquisizione. Dal dataset ufficiale sono state eliminate 10 singole scansioni e 15 sessioni complete per bassa qualità.

Il dataset OASIS è distribuito nel formato NiFTI (Neuroimaging Informatics Technology Initiative), sviluppato dal National Institute of Mental Health e dal National Institute of Neurological Disorders and Stroke intorno al 2002. Ogni file NiFTI è costituito da due parti: un header in formato .hdr e l'immagine effettiva in formato .img. Per addestrare la rete neurale in questo progetto, è stato necessario convertire questi file NiFTI in formati 2D come il jpg. Grazie all'integrazione nativa di Kaggle su Google Colab, è stato utilizzato il dataset "OASIS Alzheimer's Detection", disponibile su Kaggle, una piattaforma che offre accesso a migliaia di dataset open source. Per la creazione di questo dataset, è stata impiegata la libreria FSL (FMRIB Software Library) per convertire i file NiFTI in file jpg. In particolare, l'immagine 3D originale è stata suddivisa lungo l'asse di profondità in 256 slice, delle quali sono state selezionate solo quelle comprese tra la sessantesima e la centosessantesima. Questa selezione è stata effettuata per escludere le regioni troppo superficiali e terminali del cervello, concentrandosi così sulle aree più significative ai fini della rilevazione dell'Alzheimer. Poiché da ciascuna immagine 3D sono state estratte 61 slice, il dataset totale si compone di 86.400 immagini 2D con risoluzione 496x248, per un totale di 1.23GB di dati, che sono stati organizzati nelle seguenti cartelle: No Dementia, Very Mild Dementia, Mild Dementia e Moderate Dementia [17].

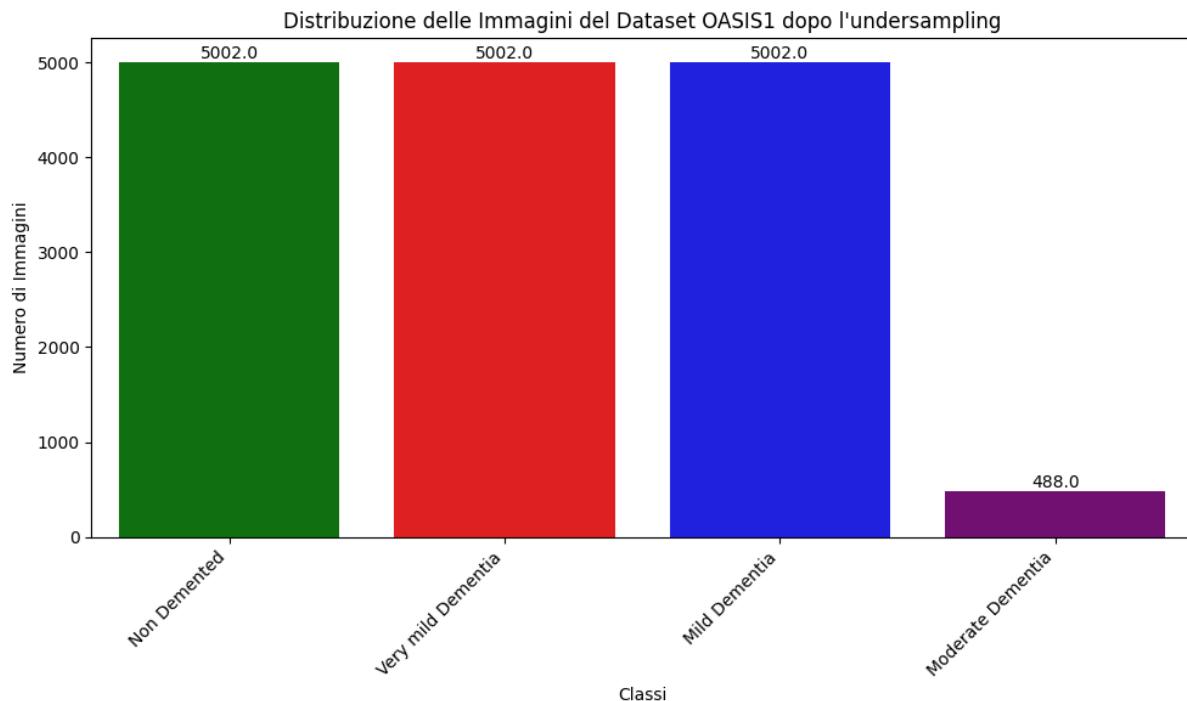
## 2.2 Preprocessing delle immagini

Il dataset utilizzato risulta essere troppo sbilanciato: principalmente costituito da immagini di pazienti privi di demenza, con una piccola minoranza di immagini rappresentanti la malattia. Se il dataset fosse stato utilizzato in questa maniera, il modello sarebbe stato troppo tarato nel riconoscimento dei pazienti negativi, rispetto al riconoscimento dei pazienti positivi alla malattia. Inoltre, il caricamento nella memoria di circa 80 mila immagini avrebbe richiesto l'utilizzo di macchine particolarmente prestanti, non disponibili per questo progetto.



*Figura 19. Distribuzione delle immagini del Dataset OASIS 1 prima dell'undersampling*

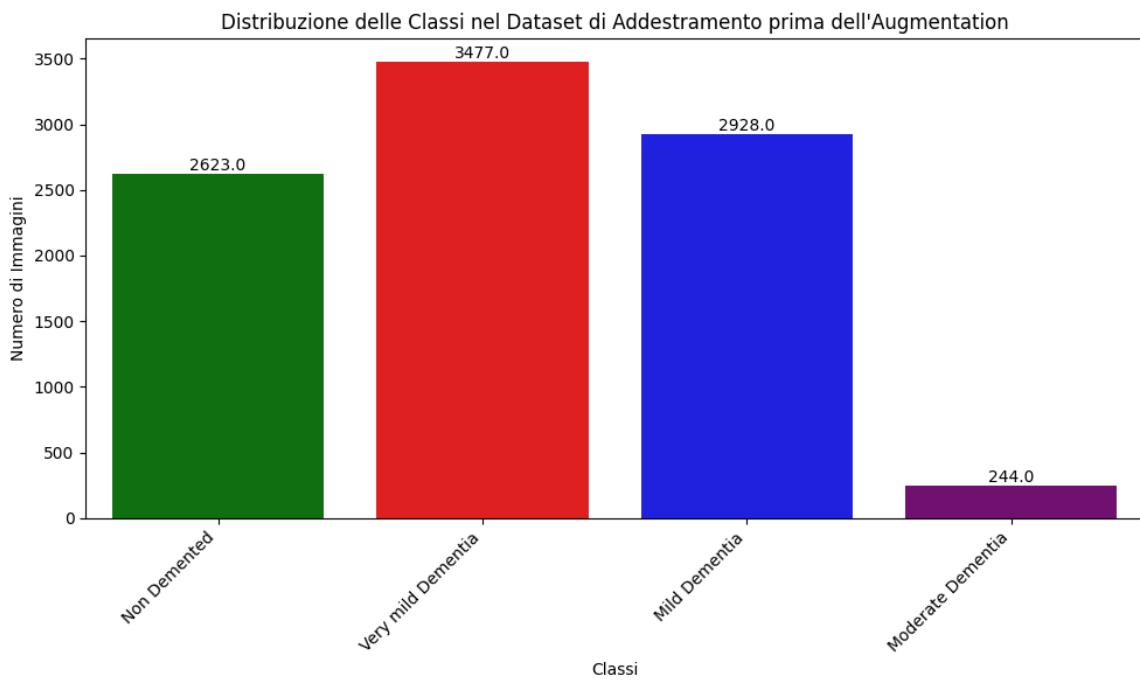
Col fine di risolvere questa problematica è stata effettuata l'operazione di undersampling, ovvero un'operazione di riduzione delle classi maggioritarie del dataset, in maniera tale da renderle bilanciate.



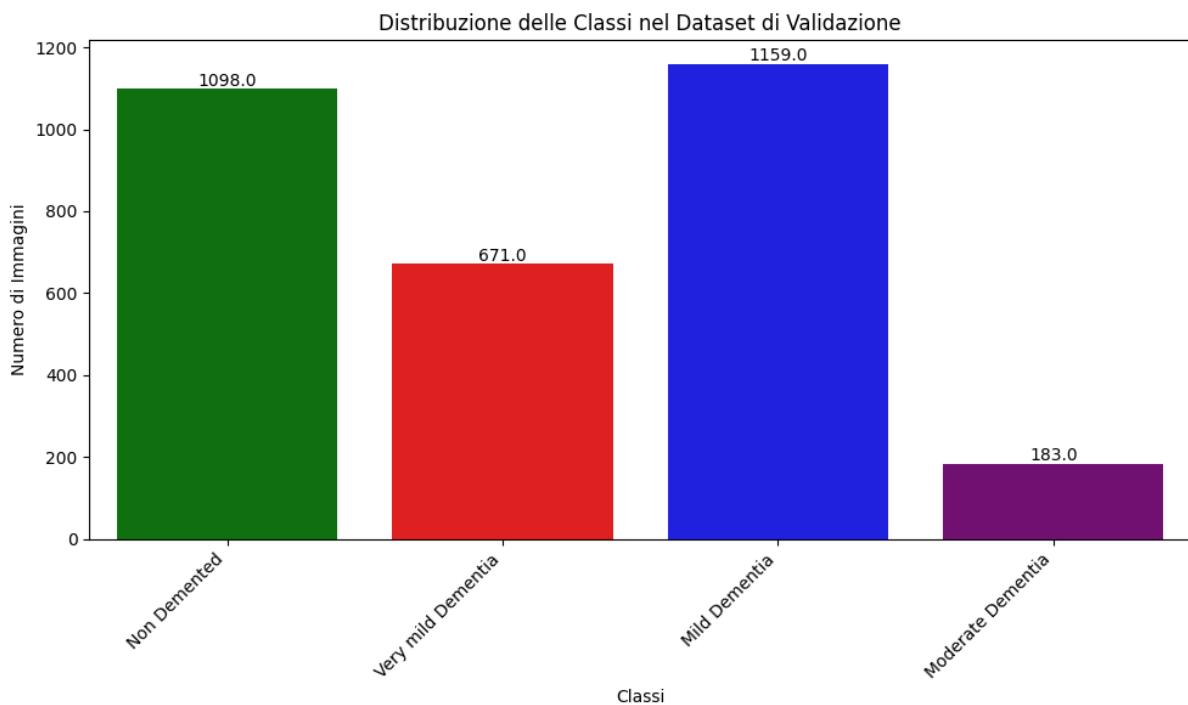
*Figura 20 Distribuzione delle immagini del Dataset OASIS1 dopo l'undersampling*

Le immagini così fornite sono state distribuite tra training set, validation set e test set. Il tutto è stato eseguito facendo un'attenta considerazione: tutte le slice di ciascuna immagine sarebbero dovute finire nello stesso set, poiché se così non fosse stato il modello sarebbe stato addestrato, validato e verificato sulle componenti delle stesse immagini. Ciò avrebbe portato a delle metriche falsate, con un altissimo grado di accuratezza delle predizioni. Ciò è stato realizzato mediante la funzione `GroupShuffleSplit` della libreria `sklearn.model_selection`. La distribuzione è stata fatta nella seguente modalità:

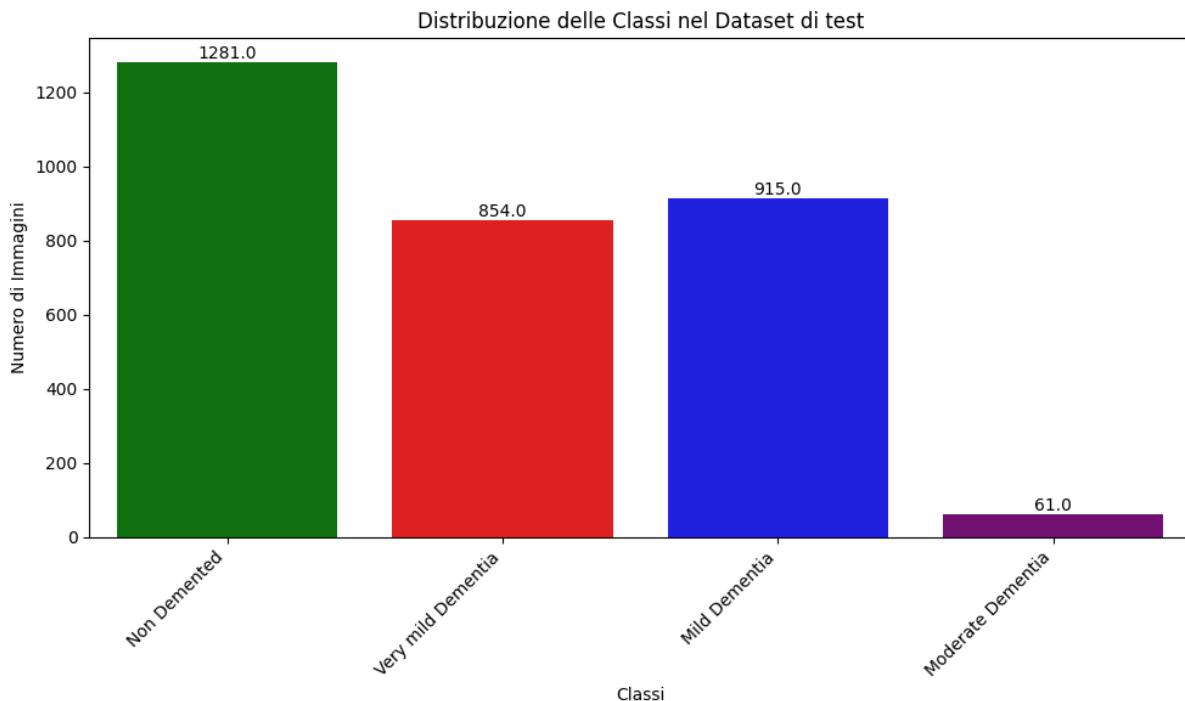
- 60% dei dati per il training set;
- 20% dei dati per il validation set;
- 20% dei dati per il test set.



*Figura 21. Distribuzione delle classi nel dataset di Addestramento prima dell'Augmentation*



*Figura 22 Distribuzione delle classi nel dataset di validazione*



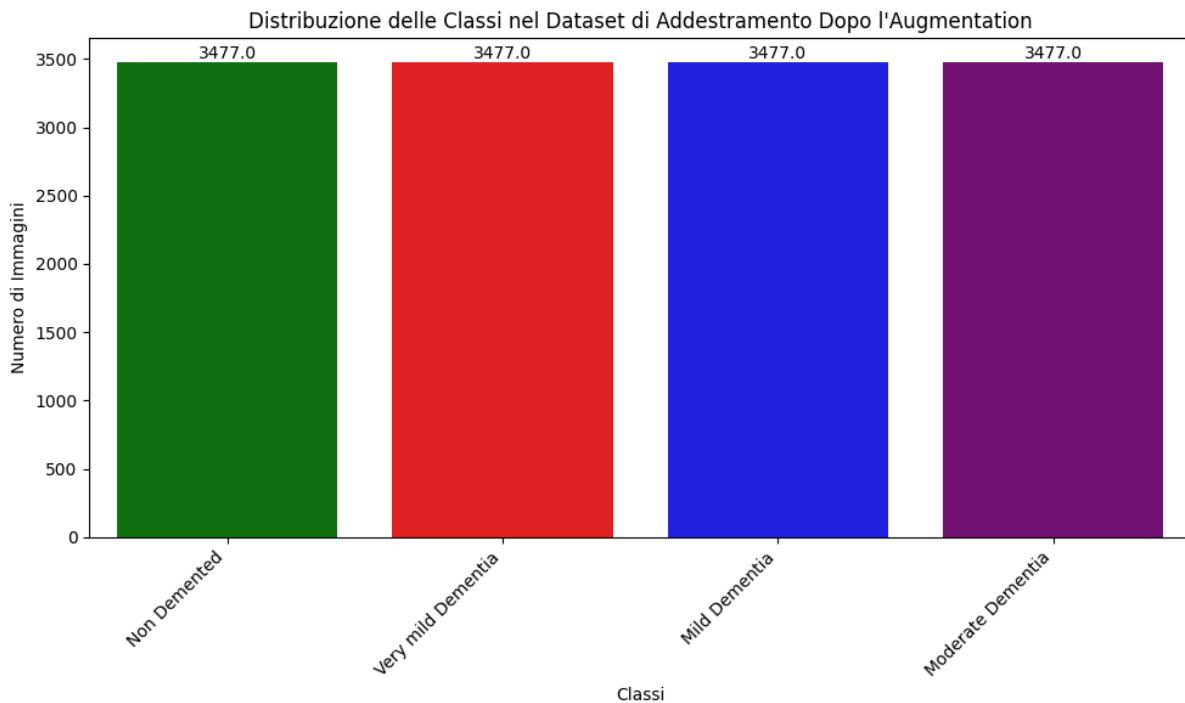
*Figura 23 Distribuzione delle classi nel dataset di test*

Per garantire un modello il più possibile bilanciato, è stata applicata la tecnica di Data Augmentation al dataset di training. Questo processo ha permesso di aumentare il numero di immagini appartenenti alle classi minoritarie attraverso diverse manipolazioni delle immagini originali, tra cui:

- Variazioni della luminosità entro un range del 5%;
- Zoom delle immagini fino a un massimo del 5%;
- Ribaltamenti orizzontali;
- Traslazioni verticali entro un range del 5%;
- Traslazioni orizzontali entro un range del 5%.

Queste trasformazioni hanno consentito di arricchire il dataset, migliorando la capacità del modello di riconoscere correttamente le classi.

Infine, per ognuno dei tre dataset è stato applicato un rescaling del valore assunto dai pixel mediante rapporto con il valore 255, ossia il massimo valore rappresentabile all'interno del pixel, in maniera tale da ottenere valori di intensità nell'intervallo [0,1] migliorando in questo modo anche la convergenza degli algoritmi di propagazione all'indietro e discesa del gradiente. Per poter effettuare queste operazioni sono state utilizzate le librerie tensorflow.keras.preprocessing.image e imblearn.over\_sampling per ottenere i tools ImageDataGenerator e SMOTE.



*Figura 24 Distribuzione delle classi nel dataset di addestramento dopo l'augmentation*

### 2.3. Architettura della rete CNN implementata

La rete neurale convoluzionale utilizzata si basa sul modello "Inception V3", sviluppato da Google e pubblicato nel 2016. Questo modello è stato pre-addestrato su ImageNet, un ampio dataset contenente circa un milione di immagini suddivise in 1000 classi. Grazie al meccanismo del transfer learning, è possibile riutilizzare il modello Inception V3 per creare un nuovo modello specifico per la classificazione dell'Alzheimer. Il transfer learning permette di trasferire la conoscenza accumulata da Inception V3, sotto forma di pesi, sfruttando i meccanismi di identificazione già consolidati nel modello, riducendo così i tempi di addestramento e migliorando la precisione anche su nuovi task.

Il funzionamento di Inception V3 è basato sui seguenti principi:

- Fattorizzazione in convoluzioni più piccole
- Fattorizzazione in convoluzioni asimmetriche
- Riduzione efficiente delle feature map
- Utilizzo di un classificatore ausiliario

La fattorizzazione in convoluzioni che sfruttano kernel di dimensioni ridotte permette di ridurre il numero di parametri della rete senza alterarne l'efficienza. Piuttosto che usare un filtro  $5 \times 5$ , con  $5 \times 5 = 25$  parametri, è utile utilizzare due livelli con filtri  $3 \times 3$ , con  $3 \times 3 + 3 \times 3 = 18$  parametri.

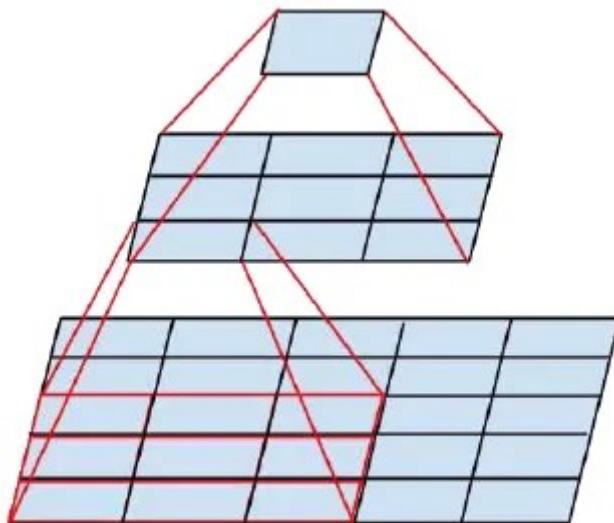


Figura 25 Fattorizzazione in convoluzioni più piccole (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)

In questo modo il numero di parametri della rete viene ridotto del 28%. Sulla base di questa tecnica viene definita la prima tipologia di modulo di questo modello: il blocco Inception A.

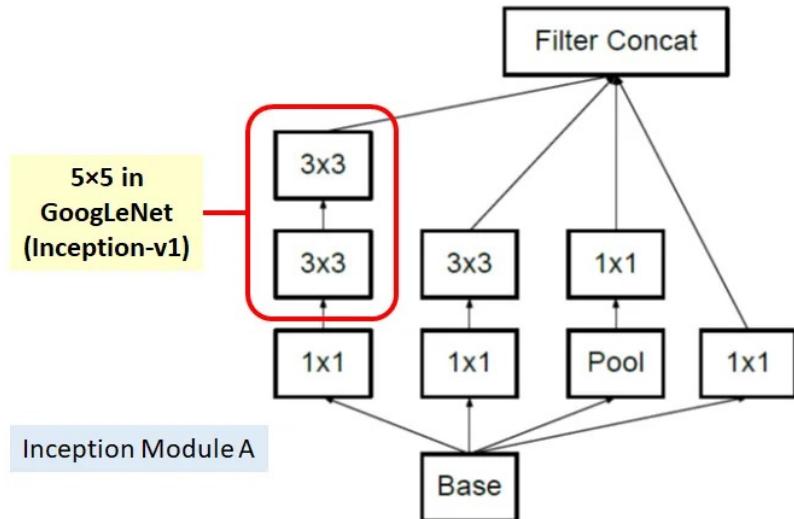


Figura 26 Inception modulo A (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)

Esso utilizza quattro percorsi distinti per eseguire convoluzioni di dimensioni diverse. Questi percorsi lavorano in parallelo, e i risultati vengono concatenati lungo l'asse dei canali di output, formando una rappresentazione complessiva ricca di informazioni. I quattro percorsi principali sono:

- **Convoluzione 1x1:**

Questo percorso esegue una semplice convoluzione 1x1 sull'immagine di input, riducendo il numero di canali mantenendo le informazioni spaziali. La convoluzione 1x1 è usata spesso nelle reti Inception per ridurre la dimensionalità e i costi computazionali.

- **Convoluzione 1x1 seguita da convoluzione 5x5:**

In questo percorso, si esegue una convoluzione 1x1 per ridurre la dimensionalità, seguita da una convoluzione 5x5. La convoluzione 5x5 cattura caratteristiche su una scala spaziale più ampia rispetto alla 1x1, permettendo di comprendere meglio il contesto globale dell'immagine.

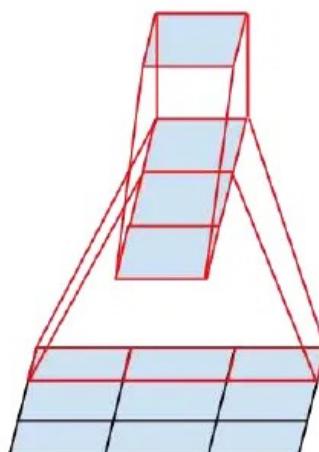
- **Convoluzione 1x1 seguita da due convoluzioni 3x3:**

Invece di eseguire una singola convoluzione 5x5, questo percorso usa una strategia di fattorizzazione: una convoluzione 1x1 viene seguita da due convoluzioni 3x3 consecutive. Questo approccio riduce il numero di parametri rispetto a una singola convoluzione 5x5, pur mantenendo la capacità di catturare informazioni su una scala spaziale simile.

- **Pooling medio seguito da convoluzione 1x1:**

Questo percorso esegue un average pooling sull'immagine di input, che riduce le dimensioni spaziali ma conserva le informazioni essenziali. Dopo il pooling, viene eseguita una convoluzione 1x1 per ridurre la dimensionalità dei canali e combinare le informazioni globali estratte con le altre convoluzioni.

La fattorizzazione in convoluzioni asimmetriche viene utilizzata anch'essa per la riduzione dei parametri: invece che usare filtri 3x3 con  $3 \times 3 = 9$  parametri, si utilizza un primo filtro 1x3 seguito da un secondo filtro 3x1, così che si abbiano  $1 \times 3 + 3 \times 1 = 6$  parametri; il 33% in meno rispetto al caso precedente.



*Figura 27 Fattorizzazione in convoluzioni asimmetriche (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)*

Nel caso in cui invece fossero stati utilizzati due filtri 2x2, ci sarebbero stati  $2 \times 2 \times 2 = 8$  parametri; solo l'11% in meno. Sulla base di questa tecnica vengono definiti il secondo e il terzo modulo del modello: il blocco Inception B e il blocco Inception C.

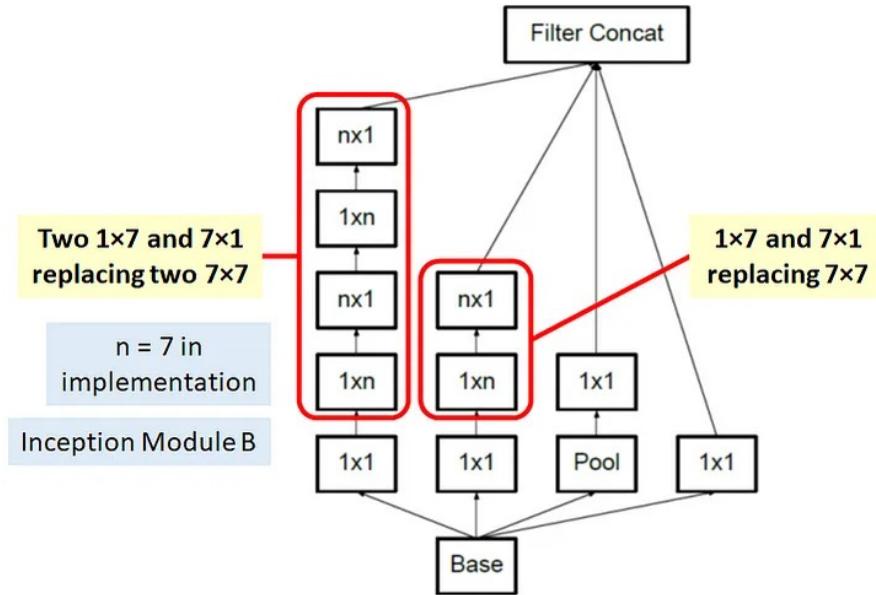


Figura 28 Inception modulo B (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)

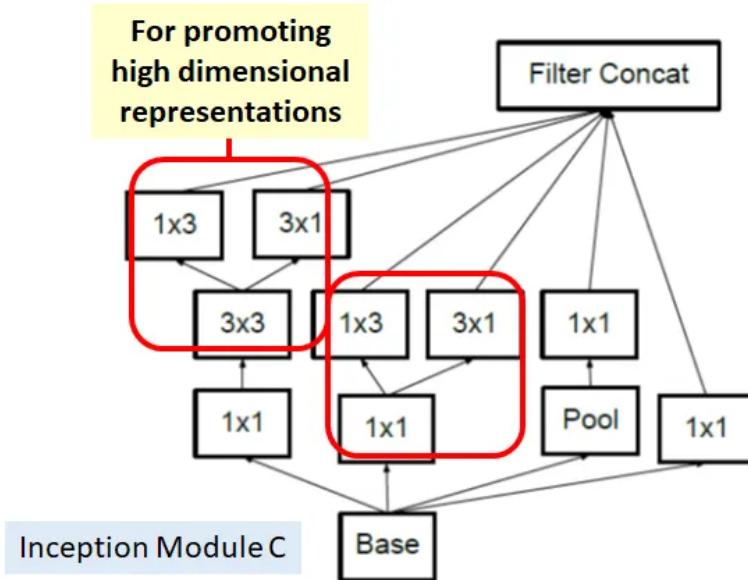


Figura 29 Inception modulo C (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)

Attraverso l'utilizzo di questi moduli, è possibile la costruzione di una rete neurale con meno parametri, che sia più resistente all'overfitting.

Inception V3 utilizza un classificatore ausiliario, ossia una rete neurale secondaria che viene interposta tra i livelli di una rete più grande durante il processo di addestramento. Il suo scopo principale è quello di fungere da regolarizzatore, ovvero componenti che aggiungono una penalità alla funzione di perdita della rete, per disincentivare l'apprendimento di pattern troppo dettagliati, col fine di preservare il modello dallo stato di overfitting e dunque la capacità del modello di generalizzare.

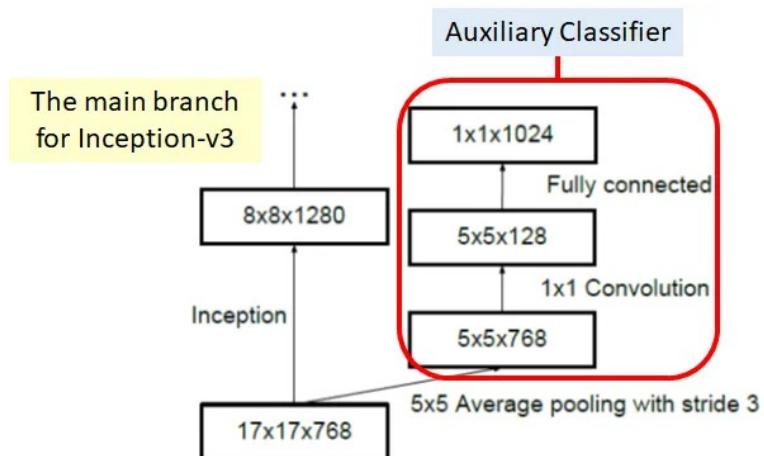
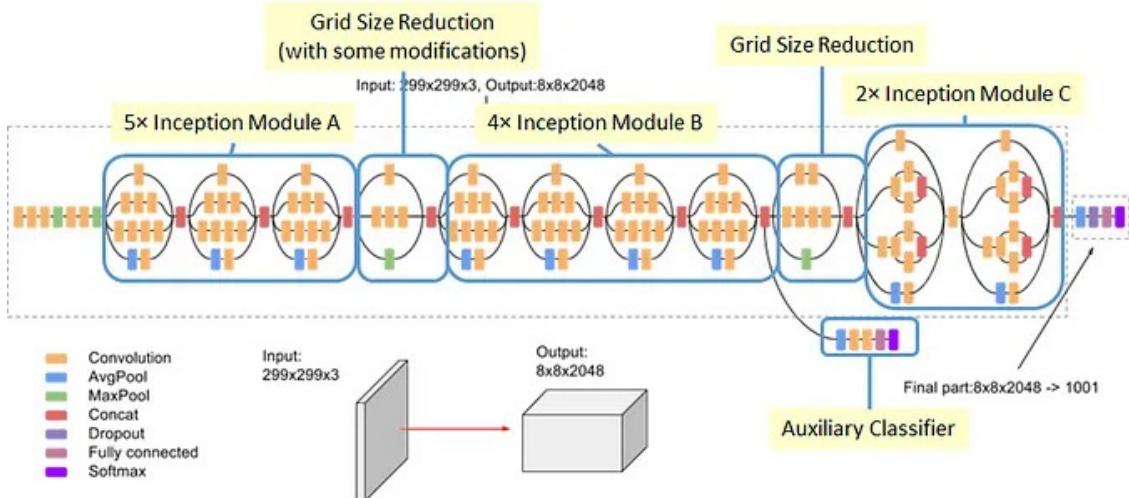


Figura 30 Classificatore ausiliario (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)

Tutti i concetti finora analizzati si condensano nell'architettura di Inception V3.



*Figura 31 Schema della rete Inception v3 (fonte: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>)*

A questa struttura di partenza è stato aggiunto uno strato di Global Average Pooling, per ottenere da un numero n di feature map, un vettore monodimensionale di n elementi che potesse essere utilizzato successivamente negli strati completamente connessi. Infatti, questa tipologia di operazione effettua la media aritmetica tra ogni valore della i-esima feature map, che poi verrà salvato nella posizione i-esima di un vettore monodimensionale di output. Infine sono presenti due strati completamente connessi, ove il primo con 1024 neuroni e funzione di attivazione “ReLU” si occuperà di combinare tutte le feature finora estratte, per poter aumentare la conoscenza della rete, dopodiché la rete si concluderà con l'ultimo strato completamente connesso avente 4 neuroni, pari al numero di classi in cui verranno classificati i dati, con funzione di attivazione “Softmax“ che ricevuto un vettore di numeri reali, fornirà un vettore contenente la distribuzione di probabilità che quel dato appartenga ad ogni classe, con conseguente classificazione delle previsioni [18,19].

## 2.4 Parametri di addestramento e validazione del modello

Per l'addestramento di una rete neurale sono fondamentali due classi di specifiche: i parametri e gli iperparametri. I parametri rappresentano variabili interne al modello, che quest'ultimo modifica autonomamente durante il processo di addestramento, come i pesi. Mentre gli iperparametri vengono definiti a priori dal professionista che si occupa della gestione del modello, tra di essi ci sono:

- Il tasso di apprendimento;
- Il numero di epoche;
- La dimensione dei batch;

Il tasso di apprendimento quantifica la dimensione del passo percorso durante lo spostamento contro gradiente e dunque quantifica quanto velocemente i parametri del modello si aggiornano durante l'addestramento. Lo scopo, infatti, dell'algoritmo di discesa del gradiente è quello di fornire il verso con cui modificare i pesi della rete col fine di minimizzare la funzione di costo del modello, e dunque l'errore commesso durante le previsioni. Questa variazione avverrà ad ogni epoca di una quantità proporzionale al tasso di apprendimento. La scelta del tasso di addestramento deve essere ben ponderata: valori troppo alti porteranno a passi di grande entità durante l'aggiornamento dei parametri, con il rischio che il modello salti oltre il minimo della funzione di perdita, rendendo difficile la convergenza, mentre valori troppo piccoli potrebbero rendere l'aggiornamento dei pesi troppo lento, con conseguente tempo di convergenza elevato verso il minimo della funzione di costo; inoltre, un tasso di apprendimento molto basso potrebbe portare il modello a rimanere intrappolato in un punto di minimo locale, rendendo così stagnante la progressione delle sue metriche.

Il numero di epoche invece quantifica il numero di volte in cui l'algoritmo di addestramento visionerà tutti i dati del dataset di addestramento. Durante il processo di addestramento, il dataset verrà diviso in lotti, noti come batch, e per ciascuno di essi verrà effettuato l'aggiornamento dei parametri basato sul gradiente calcolato su quel gruppo di dati. Una volta che tutti i batch saranno stati gestiti, si concluderà l'epoca e il modello potrà ripartire ad elaborare il dataset di addestramento fino all'esaurimento delle epoche a disposizione. Anche il numero di epoche deve essere calibrato con cura e spesso è necessario sperimentare per trovare il valore ottimale. Un numero di epoche particolarmente elevato potrebbe rendere il processo di addestramento particolarmente lungo, e a causa dell'eccessiva elaborazione del dataset di addestramento potrebbe verificarsi una situazione di overfitting del modello, che si è adattato così tanto ai dati campione da non riuscire più a generalizzare sui nuovi dati. Un numero di epoche particolarmente ridotto invece potrebbe non essere sufficiente al modello per raggiungere una buona conoscenza dei dati, con conseguente situazione di underfitting, per cui il modello non è stato in grado di apprendere correttamente dai dati e a risultare dunque inefficace nel fare previsioni sul dataset di addestramento o su nuovi dati.

La dimensione dei batch permette di rendere gestibili dataset troppo grandi per essere elaborati in un unico passaggio. Batch di piccole dimensioni comportano un minore utilizzo della memoria e forniscono una maggiore frequenza di aggiornamento dei pesi, rendendo l'algoritmo in grado di adattarsi più velocemente alle nuove informazioni; tuttavia, il rumore introdotto da piccoli insiemi di dati potrebbe rendere la convergenza

instabile e sarebbe richiesto un numero di epoch più elevato per raggiungere un buon grado di generalizzazione.

La validazione del modello, basata sul dataset di validazione, rende possibile l'ottimizzazione degli iperparametri della rete, sulla base di fattori quali il tempo di convergenza, l'andamento della funzione di costo e l'efficacia nelle previsioni. La funzione di costo misura quanto il modello si discosta, con le sue previsioni, dai valori veri. Lo scopo di un processo di addestramento è proprio quello di incrementare la conoscenza della rete, tale da migliorare l'efficacia delle previsioni e dunque minimizzare la funzione di costo. Il suo monitoramento aiuta a capire come il modello si sta comportando con i dati nuovi. Ad esempio, se la funzione dovesse decrescere sul dataset di addestramento e dovesse crescere sul dataset di validazione, in modo duraturo, allora il modello si starebbe adattando eccessivamente sui dati di addestramento, perdendo la capacità di generalizzare. Le funzioni di costo più utilizzate sono, per quanto riguarda problemi di classificazione:

- Binary Cross-Entropy Loss, per la classificazione binaria;
- Categorical Cross-Entropy Loss, per classificazioni multclasse.

Essendo il modello del progetto un classificatore, in grado di suddividere i dati in quattro categorie, allora verrà utilizzata la Categorical Cross-Entropy Loss.

## 2.5 Implementazione

### 2.5.1 Scelta della piattaforma e degli strumenti di sviluppo

A causa dell'eccessiva potenza computazionale richiesta per l'addestramento di questa rete, è stato necessario l'utilizzo della piattaforma di cloud computing Google Colab, in grado di rendere disponibili per gli utenti abbonati, delle macchine con elevate risorse computazionali, utilizzabili da remoto. Essa, infatti, rende possibile l'esecuzione di codice python direttamente sul browser, senza la necessità di effettuare installazioni, per via della sua compatibilità con i Jupyter Notebook. Oltre ciò, dispone di un'integrazione nativa della piattaforma Kaggle e ciò rende semplice il caricamento di migliaia di dataset pubblicati, anche di grandi dimensioni, sulla memoria della macchina utilizzata. Gli script scritti in Python possono essere suddivisi in sezioni distinte per eseguire il codice in più fasi, facilitando così la gestione e la comprensione del programma. Google Colab rende disponibili diverse configurazioni possibili. Per lo sviluppo del modello in esame è stata selezionata la macchina con le seguenti caratteristiche: 83.5GB di memoria RAM di sistema, 40GB di memoria RAM GPU e 112.6GB di memoria del disco rigido. La scheda video utilizzata in questa configurazione è la “Nvidia A100 GPU”, dal costo di circa 17000 euro.

Le librerie utilizzate per la realizzazione di questa rete CNN sono:

- tensorflow, una libreria open source per l'apprendimento automatico, che rende disponibili moduli sperimentati e ottimizzati.
- keras, una libreria open source specializzata per il deep learning. Nacque inizialmente come una libreria a sé stante, ma venne poi integrata in tensorflow come API di alto livello predefinita per la costruzione e l'addestramento di reti neurali all'interno di TensorFlow.

- os, per la gestione del file system messo a disposizione dalla macchina remota: permette di spostare, copiare, rinominare, eliminare file attraverso linee di codice python. È stata utilizzata per l'accesso alle cartelle del dataset e per l'operazione di undersampling.
- numpy, libreria che rende possibili metodi di alto livello per interagire efficientemente con dati disposti in matrici di grandi dimensioni o array.
- pandas, utilizzata per l'analisi di dati. Mette a disposizione strutture dati e metodi efficienti per manipolare tabelle numeriche. È stata utilizzata per la divisione del dataset originale in dataset di addestramento, validazione e test.
- sklearn, utilizzata per l'accesso alla funzione GroupShuffleSplit, utile per poter effettuare la suddivisione tra dataset, facendo sì che immagini appartenenti allo stesso gruppo, in questo caso tutte le slice di ciascuna immagine, venissero smistate insieme. Ciò è stato fatto per effettuare un corretto addestramento del dataset, che altrimenti sarebbe stato addestrato su varianti delle immagini che sarebbero poi state utilizzate per la sua validazione e testing, portando allo sviluppo di false metriche.
- tqdm, una libreria utilizzata per visualizzare l'esecuzione di iterazioni attraverso una barra di progresso.
- matplotlib, per la realizzazione di grafici descrittivi come gli istogrammi e anche per rappresentare graficamente l'andamento delle metriche del modello.
- seaborn, una libreria di python che potenzia gli strumenti di data visualization del modulo matplotlib. Aggiunge funzionalità quali gestione dei colori e dell'ordine delle classi nell'istogramma.

### 2.5.2. Implementazione del modello CNN

L'implementazione del modello di base Inception V3 è avvenuta mediante Keras: sono stati importati i pesi del pre-addestramento sul dataset ImageNet e il modello è stato importato senza gli strati superiori (`include_top = False`), in quanto sarebbero stati poi adattati per la classificazione in esame. Inoltre, è stata definita la dimensione dell'input a 496x248, ovvero la risoluzione delle immagini MRI costituenti il dataset OASIS. Successivamente, è stato aggiunto uno strato di Global Average Pooling per ridurre la dimensionalità dell'output convoluzionale e dopo questo, sono stati aggiunti uno strato denso intermedio con 1024 neuroni e attivazione ReLU, seguito da uno strato denso finale con 4 neuroni e attivazione softmax per la classificazione in 4 categorie.

```
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(496, 248, 3))
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
predictions = Dense(4, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
```

### 2.5.3. Ottimizzazione e tuning degli iperparametri

Gli iperparametri della rete sono stati calibrati attraverso un processo di sperimentazione iterativa. Durante questa fase i valori di tasso di apprendimento, dimensione dei batch e il numero di epoche sono stati testati per identificare la configurazione che ottimizzava le prestazioni del modello sul set di validazione. È stato utilizzato l'ottimizzatore “Adamax” per l'aggiornamento dei pesi della rete. I pesi vengono aggiornati secondo la seguente equazione:

$$\omega_t^i = \omega_{t-1}^i - \frac{\eta}{v_t + \epsilon} \cdot m_t$$

dove:

$$\begin{aligned} m_t &= \frac{m_t}{1 - \beta_1^t} \\ v_t &= \max(\beta_2 \cdot v_{t-1}, |G_t|) \\ m_t &= \beta_1 m_{t-1} + (1 - \beta_1) G \\ G &= \nabla_{\omega} C(\omega_t) \end{aligned}$$

In queste equazioni  $\eta$  rappresenta il tasso di apprendimento,  $\omega_t$  sono i pesi al passo  $t$ ,  $C(\cdot)$  è la funzione di costo,  $\nabla_{\omega} C(\omega_t)$  è il gradiente dei pesi  $\omega_t$ .  $\beta_i \in [0,1]$  è utilizzata per selezionare la quantità di informazione dei precedenti aggiornamenti che viene mantenuta al passo attuale.  $m_t$  è il primo momento (media del gradiente), che cattura la direzione media del gradiente, mentre  $v_t$  è il secondo momento (varianza non centrata del gradiente), che cattura la scala o la variabilità del gradiente.

Questo algoritmo di ottimizzazione imposta come tasso di apprendimento iniziale  $\eta = 0.01$  che attraverso un processo adattivo viene modificato in base alla magnitudine dei gradienti. Sono inoltre state utilizzate due

funzioni che basandosi sull'andamento della funzione di costo hanno modificato l'andamento degli iperparametri in maniera dinamica durante l'addestramento della rete. Esse sono “ReduceLROnPlateau” e “EarlyStopping”. Entrambe operano sulla base dell'andamento della funzione di costo: la prima riduce il learning rate con un certo fattore moltiplicativo dopo un certo numero di epochhe in cui la funzione di costo non presenta miglioramenti, mentre la seconda interrompe il processo di apprendimento dopo un certo numero di epochhe in cui la funzione di costo non presenta miglioramenti e ripristina i pesi all'epoca in cui la loss ha raggiunto il valore minimo. Nel caso attuale la ReduceLROnPlateau riduce il learning rate di un ordine di grandezza, utilizzando come fattore moltiplicativo 0.1, dopo tre epochhe in cui la funzione di loss smette di migliorare. Mentre la EarlyStopping ,dopo un periodo di pazienza di dieci epochhe in cui la loss smette di descrescere, interrompe il processo di addestramento e ripristina i pesi per ripristinare il valore minimo. La dimensione dei batch, invece, dopo un certo numero di prove è stata scelta pari a 32, in quanto questo è stato il valore che ha permesso di raggiungere valori più alti possibili di accuratezza sul dataset di validazione.

#### 2.5.4. Valutazione delle performance del modello

Per valutare le performance finali del modello si utilizza il dataset di test. Le previsioni effettuate su questo dataset permetteranno la costruzione della matrice di confusione, ovvero una matrice le cui righe rappresentano le classi vere dei dati, mentre le colonne rappresentano le classi predette dal modello. Ogni cella  $C_{ij}$  nella matrice mostra quante volte un campione appartenente alla classe  $i$  è stato predetto come appartenente alla classe  $j$ .

		CLASSI PREVISTE		
		classe 1	classe 2	classe 3
CLASSI EFFETTIVE	classe 1			
	classe 2			
	classe 3			

Figura 32 Matrice di confusione. fonte: (<https://www.andreaminini.com/ai/machine-learning/matrice-di-confusione>)

Le previsioni effettuate sul dataset di test rientrano nelle seguenti casistiche:

- Veri positivi (TP): sono le istanze appartenenti a una classe specifica che sono state correttamente identificate come appartenenti a quella classe dal modello.
- Veri negativi (TN): sono le istanze che non appartengono a una classe specifica e sono state correttamente classificate dal modello come non appartenenti a quella classe.
- Falsi positivi (FP): sono le istanze non appartenenti a una classe specifica che sono state erroneamente identificate come appartenenti a quella classe.
- Falsi negativi (FN): sono le istanze appartenenti a una classe specifica che sono state erroneamente identificate come non appartenenti a quella classe.

La prima delle metriche per la valutazione delle performance è l'accuratezza, ovvero il rapporto tra le istanze che sono state correttamente classificate e il numero totale di dati. Relativamente alla matrice di confusione, l'accuratezza corrisponderà al rapporto tra la somma delle istanze collocate sulla diagonale principale della matrice e il numero totale di dati.

$$\text{Accuratezza} = \frac{\sum_{i=1}^n C_{ii}}{\text{Numero totale di dati}} = \frac{\sum_{i=1}^n (TP_i + TN_i)}{\text{Numero totale di dati}}$$

La seconda metrica è la precisione, ovvero il rapporto tra le istanze correttamente classificate come positive e la somma delle istanze classificate correttamente come positive e quelle erroneamente classificate come positive considerando una classe alla volta. Considerando la classe  $i$ -esima verranno riconosciuti come veri

positivi le istanze appartenenti alla classe i-esima che vengono riconosciute come tali, mentre come falsi positivi tutti quei dati appartenenti alle altre classi che vengono erroneamente riconosciuti come appartenenti alla classe i-esima. Serve a identificare il tasso con cui un dato identificato come positivo risulta essere un vero positivo.

$$\begin{aligned} \text{Precisione}_i &= \frac{TP_i}{TP_i + FP_i} \\ \text{Precisione media} &= \frac{\sum_{i=1}^n \text{Precisione}_i}{n} \end{aligned}$$

La terza metrica è la sensibilità, ovvero il rapporto tra i veri positivi e la somma di tutti i dati realmente positivi, ovvero i veri positivi e i falsi negativi. Rappresenta il tasso con cui un dato realmente positivo viene identificato come vero positivo. Nel caso di un test diagnostico stabilisce la statistica con cui una persona positiva ad una malattia viene correttamente riconosciuta come tale.

$$\begin{aligned} \text{Sensibilità}_i &= \frac{TP_i}{TP_i + FN_i} \\ \text{Sensibilità media} &= \frac{\sum_{i=1}^n \text{Sensibilità}_i}{n} \end{aligned}$$

La quarta metrica è la specificità, ovvero il rapporto tra i veri negativi e la somma di tutti i dati realmente negativi, ovvero i veri negativi e i falsi positivi. Rappresenta la proporzione con cui un dato realmente negativo viene identificato come vero negativo. Nel caso di un test diagnostico stabilisce la statistica con cui una persona negativa ad una malattia viene correttamente riconosciuta come tale.

$$\begin{aligned} \text{Specificità}_i &= \frac{TN_i}{TN_i + FP_i} \\ \text{Specificità media} &= \frac{\sum_{i=1}^n \text{Specificità}_i}{n} \end{aligned}$$

L'ultima metrica in esame invece sarà F1 score, ovvero la media armonica tra la precisione del modello e la sua sensibilità. Permette di determinare se un test è particolarmente bilanciato su questi due parametri. Un alto valore di F1 score indica che il modello è efficace nel riconoscere le istanze positive senza generare troppi falsi positivi o falsi negativi.

$$F1 Score = 2 \frac{\text{Precisione} \cdot \text{Sensibilità}}{\text{Precisione} + \text{Sensibilità}}$$

## Capitolo 3 – Risultati

Durante il processo di addestramento la funzione ReduceLROnPlateau ha ridotto il tasso di apprendimento di quattro ordini di grandezza durante l'esecuzione di 19 epoche, passando da  $\eta = 0.001$  a  $\eta = 10^{-7}$ . La funzione di EarlyStopping invece ha interrotto il processo di apprendimento all'epoca 19, in quanto per 10 epoche non ci sono stati miglioramenti sulla funzione di costo ed ha ripristinato i pesi all'epoca 9, ovvero l'epoca in cui la funzione di loss ha raggiunto il valore minimo di  $val\_loss = 0.2036$  sul dataset di validazione e un valore di  $loss = 0.027$  sul dataset di addestramento. All'epoca 9 si è ottenuto un valore di accuratezza sul dataset di addestramento pari a  $accuracy = 0.9927$  e un valore di accuratezza sul dataset di validazione pari a  $val\_accuracy = 0.9486$ . Il processo di addestramento è durato 12 minuti e 58 secondi.

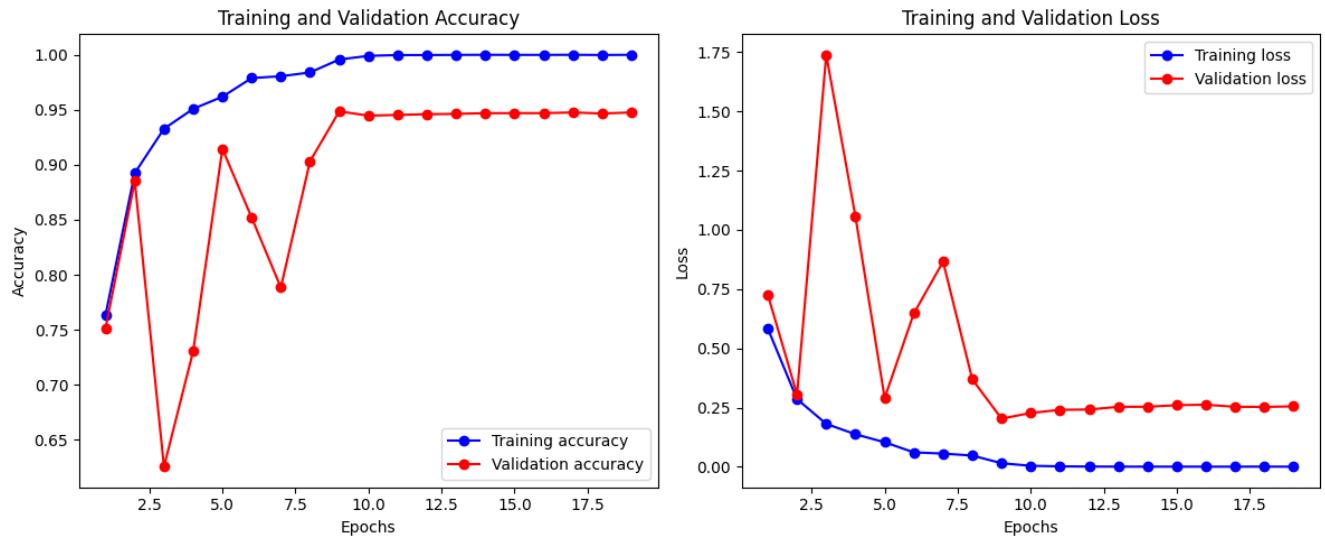


Figura 33 Andamento delle funzioni accuratezza e costo rispetto al dataset di addestramento e di validazione

Successivamente è avvenuto il processo di testing del modello, che si è prolungato per 2.84 secondi e ha dato luogo a un valore di accuratezza pari a  $test\_accuracy = 0.9460$ , un valore di costo pari a  $test\_loss = 0.2082$  e la seguente matrice di confusione.

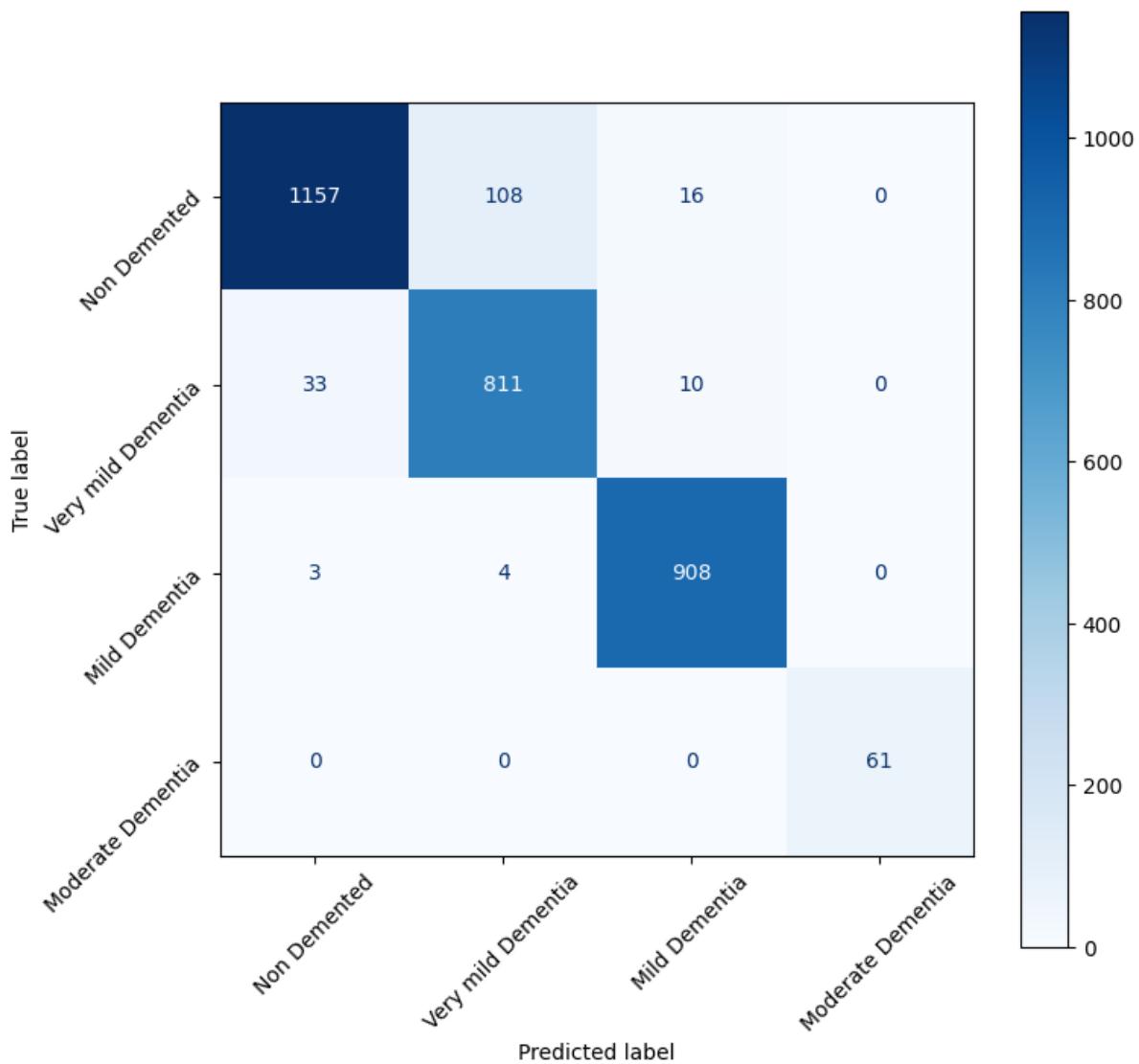


Figura 34 Matrice di confusione ottenuta dopo il test del modello

Si procede ora con il calcolo delle metriche per la valutazione del modello.

$$\text{Accuratezza} = \frac{\sum_{i=1}^n TP_i}{\text{Numero totale di dati}} = \frac{2937}{3111} = 94.4\%$$

$$\text{Precisione}_{\text{Non Demented}} = \frac{TP}{TP+FP} = \frac{1157}{1157 + 37 + 3} = \frac{1157}{1197} = 96.65\% =$$

$$\text{Precisione}_{\text{Very Mild Dementia}} = \frac{TP}{TP + FP} = \frac{811}{811 + 108 + 4} = \frac{811}{923} = 87.86\%$$

$$\text{Precisione}_{\text{Mild Dementia}} = \frac{TP}{TP + FP} = \frac{908}{908 + 10 + 16} = \frac{908}{934} = 97.21\%$$

$$\text{Precisione}_{\text{Moderate Dementia}} = \frac{TP}{TP + FP} = \frac{61}{61} = 100\%$$

$$\text{Precisione media} = \frac{\sum_{i=1}^n \text{Precisione}_i}{4} = 95.43\%$$

$$\text{Sensibilità}_{\text{Non Demented}} = \frac{TP}{TP + FN} = \frac{1157}{1157 + 108 + 16} = \frac{1157}{1281} = 90.32\%$$

$$\text{Sensibilità}_{\text{Very Mild Dementia}} = \frac{TP}{TP + FN} = \frac{811}{811 + 33 + 10} = \frac{811}{854} = 94.96\%$$

$$\text{Sensibilità}_{\text{Mild Dementia}} = \frac{TP}{TP + FN} = \frac{908}{908 + 4 + 3} = \frac{908}{915} = 99.23\%$$

$$\text{Sensibilità}_{\text{Moderate Dementia}} = \frac{TP}{TP + FN} = \frac{61}{61} = 100\%$$

$$\text{Sensibilità media} = \frac{\sum_{i=1}^n \text{Sensibilità}_i}{4} = 96.13\%$$

$$\text{Specificità}_{\text{Non Demented}} = \frac{TN}{TN + FP} = \frac{1794}{1794 + 33 + 3} = \frac{1794}{1830} = 98\%$$

$$\text{Specificità}_{\text{Very Mild Demented}} = \frac{TN}{TN + FP} = \frac{2145}{2145 + 108 + 4} = \frac{2145}{2257} = 95\%$$

$$\text{Specificità}_{\text{Mild Demented}} = \frac{TN}{TN + FP} = \frac{2170}{2170 + 16 + 10} = \frac{2170}{2196} = 98.81\%$$

$$\text{Specificità}_{\text{Moderate Demented}} = \frac{TN}{TN + FP} = \frac{3050}{3050} = 100\%$$

$$\text{Specificità media} = \frac{\sum_{i=1}^n \text{Specificità}_i}{4} = 98.95\%$$

$$F1 \text{ score medio} = 2 \cdot \frac{\text{Precisione media} \cdot \text{Sensibilità media}}{\text{Precisione media} + \text{Sensibilità media}} = 97.52\%$$

## **Capitolo 4 – Discussione**

La CNN in esame ha dunque presentato i seguenti risultati: un'accuratezza complessiva pari a 94.4%, una precisione media pari a 95.43%, una sensibilità media pari a 96.13%, una specificità media pari a 98.95%, un F1 score medio pari a 97.52%. Nel complesso il modello presenta delle metriche particolarmente buone. Esso è in grado di:

- classificare, in media, correttamente 94 immagini MRI su 100 (accuratezza);
- riconoscere in media 95 immagini MRI su 100 assegnate a una generica classe come realmente appartenenti a quella classe (precisione);
- classificare in media 96 immagini MRI su 100 realmente appartenenti ad una classe all'interno della classe corretta (sensibilità);
- classificare 99 immagini MRI su 100 appartenenti ad un'altra classe come realmente appartenenti a quella classe (specificità);

L'F1-score medio del 97,52% indica un ottimo equilibrio tra precisione e sensibilità, confermando la capacità del modello di mantenere buone performance anche nei casi in cui è importante minimizzare sia i falsi positivi che i falsi negativi.

Lo scopo di questo lavoro di tesi di laurea era quello di fornire un supporto all'attività del neurologo e del radiologo nel riconoscere gli stadi iniziali della malattia, in cui avvengono piccoli cambiamenti nella struttura cerebrale che possono essere notati da un occhio esperto attraverso il confronto tra immagini ottenute con una certa distanza temporale. Perciò, la qualità del modello, in questo contesto, non verrà valutata basandosi sulla sua capacità di classificare accuratamente gli stadi intermedi o avanzati della malattia. Invece, l'efficacia del modello sarà determinata dalla sua capacità di riconoscere e distinguere gli stadi preliminari della malattia, in particolare lo stadio di demenza molto lieve ( $CDR = 0.5$ ) e lo stadio lieve ( $CDR = 1$ ).

Per quanto riguarda lo stato molto lieve di demenza (*Very Mild Dementia*) sono stati ottenuti i seguenti risultati:

- Precisione pari a 87.86%;
- Sensibilità pari a 94.96%;
- Specificità pari a 95%;

Ovvero il modello risulta in grado di:

- riconoscere in media 87 immagini MRI su 100 assegnate alla classe *Very mild Dementia* come realmente appartenenti a quella classe (precisione)
- classificare in media 95 immagini MRI su 100 realmente appartenenti alla classe *Very Mild Dementia* all'interno di essa (sensibilità);
- classificare 95 immagini MRI su 100 appartenenti ad un'altra classe come non appartenenti alla classe *Very Mild Dementia* (specificità);

Le stime, dunque, per questa tipologia di classe sono più basse, in particolare l'accuratezza, in quanto i cambiamenti della struttura cerebrale sono così sottili da causare un numero maggiore di falsi positivi e negativi. Per quanto riguarda lo stato lieve di demenza (*Mild Dementia*) sono stati ottenuti i seguenti risultati:

- Precisione pari a 97.21%;
- Sensibilità pari a 99.23%;
- Specificità pari a 98.81%;

Ovvero il modello risulta in grado di:

- riconoscere in media 97 immagini MRI su 100 assegnate alla classe *Mild Dementia* come realmente appartenenti a quella classe (precisione);
- classificare in media 99 immagini MRI su 100 realmente appartenenti alla classe *Mild Dementia* all'interno di essa (sensibilità);
- classificare 98 immagini MRI su 100 appartenenti ad un'altra classe come non appartenenti alla classe *Mild Dementia* (specificità);

Le stime per questa tipologia di classe sono più alte rispetto alla precedente, poiché i cambiamenti della struttura cerebrale in questa fase della malattia sono più marcati e dunque più facilmente riconoscibili.

Inoltre, un'importanza particolare va riservata alla casistica di persone sane; il modello in esame deve cercare di minimizzare la percentuale di persone negative alla malattia che invece vengono classificate come malate.

Per quanto riguarda lo stato di assenza di demenza (*Non Demented*) sono stati ottenuti i seguenti risultati:

- Precisione pari a 96.65%;
- Sensibilità pari a 90.32%;
- Specificità pari a 98%;

Ovvero il modello risulta in grado di:

- riconoscere in media 96 immagini MRI su 100 di persone classificate come sane come realmente appartenenti alla classe *Non Demented* (precisione);
- classificare in media 90 immagini MRI su 100 realmente appartenenti alla classe *Non Demented* all'interno di essa (sensibilità);
- classificare 98 immagini MRI su 100 appartenenti ad un'altra classe come non appartenenti alla classe *Non Demented* (specificità);

Ciò implica che su 100 persone sane che adotteranno il test, 90 verranno riconosciute come sane e su 100 persone che verranno rilevate come sane dal test, 96 di questa lo saranno davvero. Mentre su 100 persone malate che adotteranno il test, 98 di queste non verranno assegnate alla classe caratterizzante l'assenza di demenza.

Per quanto riguarda la presenza di falsi positivi e negativi nelle classi Non Demented e Very Mild Demented è importante notare che l'87% dei falsi negativi della classe Non Demented vengono assegnati alla classe Very Mild Demented, mentre il 77% dei falsi negativi della classe Very Mild Demented vengono assegnati alla classe Non Demented. Questo poiché le differenze tra queste due classi sono così sottili da rendere la classificazione borderline. Questa problematica è in generale tutta l'attività di classificazione del modello

potrebbe essere migliorata in futuro attraverso l'implementazione di ulteriori dati relativi ai pazienti, come: età, sesso e storia clinica, mentre per ora la rete utilizza come unico elemento decisionale l'immagine di risonanza magnetica e dunque non è in grado di pesare differenze di età importanti e la presenza o assenza di patologie.

In futuro un modello così migliorato potrà essere implementato nei software di risonanza magnetica in maniera tale da poter rilevare automaticamente la probabile presenza di demenza nei pazienti, così da notificare prontamente il medico per eventuali approfondimenti.

Una diagnosi negli stadi preliminari della malattia permetterebbe la pronta attuazione di terapie tali da rallentarne il decorso, attuando interventi mirati per il decadimento cognitivo, il monitoraggio dell'evoluzione della malattia, con conseguente preservazione della condizione di autonomia del paziente e la possibilità per egli di pianificare il proprio futuro quando ancora in grado di prendere decisioni. Infine, i modelli di DL consentono la standardizzazione dell'analisi dei dati, soggetta a possibilità di errori di valutazione tra diversi operatori.

## **Conclusioni**

La malattia di Alzheimer è una condizione neurodegenerativa che mostra manifestazioni cliniche invalidanti come: deficit della memoria, disorientamento e deterioramento cognitivo. Ad oggi non esiste una cura ma una diagnosi precoce e un intervento tempestivo possono migliorare la qualità della vita dei pazienti e rallentare la progressione della malattia.

Le immagini MRI sono uno strumento comunemente impiegato per la diagnosi della malattia di Alzheimer. L'intelligenza artificiale rappresenta un approccio imparziale all'analisi multiparametrica e permette di analizzare grandi set di dati apportando un effetto favorevole in campo medico. Nel corso di questo progetto di tesi è stato realizzato un modello di DL, basato sul modello Inception V3, una CNN preaddestrata su milioni di classi di immagini. L'obiettivo era quello di trasferire la conoscenza accumulata dal modello preaddestrato, sottoforma di pesi, per ottimizzare il processo di estrazione delle features delle immagini MRI del dataset OASIS. Questo dataset contiene immagini cerebrali umane caratterizzate da diversi stadi della malattia di Alzheimer. Il modello, utilizzato per classificare le immagini MRI in quattro categorie, relative a quattro stadi di gravità della malattia di Alzheimer, ha mostrato un'accuratezza del 94,4%, una precisione media del 95,43%, una sensibilità media del 96,13% e una specificità media del 98,95%. Questi risultati indicano che il modello è in grado di discriminare con alta affidabilità tra le varie fasi della malattia, fornendo così uno strumento utile nella diagnosi precoce e nella gestione della patologia. I principali vantaggi dell'integrazione dell'intelligenza artificiale in campo medico potrebbero essere:

- L'automazione e il miglioramento dell'estrazione dei dati;
- L'individuazione di nuove caratteristiche predittive;
- La standardizzazione e l'oggettivazione dell'analisi dei dati, spesso esposta a un'enorme variabilità tra diversi centri e diversi operatori;
- La riduzione degli errori umani associati alla valutazione soggettiva delle immagini;
- La diminuzione del tempo necessario per la valutazione.

In definitiva, si può affermare che i modelli di DL rappresentano un progresso significativo nella diagnosi precoce della malattia di Alzheimer e che un loro utilizzo in ambito clinico potrebbe aiutare ricercatori e medici a classificare la malattia di Alzheimer con un miglioramento dei parametri di classificazione e interpretabilità al fine di poter agire tempestivamente e garantire un migliore trattamento della malattia.

## Bibliografia

### CAPITOLO 1

- [1] U.S. Food and Drug Administration. (2024, July 8). Artificial Intelligence and Machine Learning (AI/ML)-Enabled Medical Devices [Online]. Available: <https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-aiml-enabled-medical-devices>.
- [2] IRCSS Humanitas Research Hospital. Malattia di Alzheimer [Online]. Available: <https://www.humanitas.it/malattie/alzheimer/>.
- [3] J. C. Morris, "The Clinical Dementia Rating (CDR)," *Neurology*, vol. 43, no. 11, pp. 2412–2412-a, Nov. 1993.
- [4] Washington University School of Medicine in St. Louis. (2024). Knight Alzheimer Disease Research Center: CDR® Scoring Table [Online]. Available: <https://knightadrc.wustl.edu>.
- [5] Centro Alzheimer. (2016). Fattori di rischio e di protezione [Online]. Available: <https://www.centroalzheimer.org/area-familiari/la-malattia-di-alzheimer/malattia-di-alzheimer/fattori-di-rischio-e-di-protezione/#:~:text=Il%20diabete%2C%20l%27ipercolesterolemia%2C,cerebrali%2C%20patologie%20cerebrovascolari%2C%20vasculopatie>.
- [6] Alzheimer Italia. La malattia di Alzheimer [Online]. Available: <https://www.alzheimer.it/malattia.pdf>.
- [7] IRCCS Ospedale San Raffaele. (2023, Sep. 21). Malattia di Alzheimer: sintomi, diagnosi, terapie e il nuovo Centro dedicato del San Raffaele [Online]. Available: <https://www.hsr.it/news/2023/settembre/alzheimer-sintomi-diagnosi-terapie>.
- [8] Enfea Salute. (2020, Dec. 15). Come funziona la risonanza magnetica [Online]. Available: [https://www.enfeasalute.it/risonanza-magnetica-funzionamento/#:~:text=La%20risonanza%20magnetica%20sfrutta%20le,nella%20tomografia%20computerizzata%20\(TC\).](https://www.enfeasalute.it/risonanza-magnetica-funzionamento/#:~:text=La%20risonanza%20magnetica%20sfrutta%20le,nella%20tomografia%20computerizzata%20(TC).)
- [9] P. Hnilicova *et al.*, "Imaging Methods Applicable in the Diagnostics of Alzheimer's Disease, Considering the Involvement of Insulin Resistance," *International Journal of Molecular Sciences*, vol. 24, no. 4, p. 3325, Jan. 2023.
- [10] J. Jorge Cardete. (2024, Feb. 7). Convolutional Neural Networks: A Comprehensive Guide [Online]. Available: <https://medium.com/thedeephub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175>.
- [11] P. Piacenti. (2019, Oct. 18). Le 4 funzioni di attivazione neuronale più usate negli Artificial Neural Network [Online]. Available: <https://medium.com/@paolopiacenti1/le-4-funzioni-di-attivazione-neuronale-pi%C3%B9-usate-negli-artificial-neural-network-41096953b85c>.
- [12] H. H. Sultan, N. M. Salem, and W. Al-Atabany, "Multi-Classification of Brain Tumor Images Using Deep Neural Network," *IEEE Access*, vol. 7, pp. 69215–69225, 2019.
- [13] P. Saxena, A. Maheshwari, and S. Maheshwari, "Predictive modeling of brain tumor: a deep learning approach," *Innov. Comput. Intell. Comput. Vis.*, pp. 275–285, 2020.

- [14] W. Ayadi, W. Elhamzi, I. Charfi, and M. Atri, “Deep CNN for Brain Tumor Classification,” *Neural Process Lett.*, vol. 53, pp. 671–700, 2021.
- [15] D. AlSaeed and S. F. Omar, “Brain MRI Analysis for Alzheimer’s Disease Diagnosis Using CNN-Based Feature Extraction and Machine Learning,” *Sensors*, vol. 22, no. 8, p. 2911, Apr. 2022.

## CAPITOLO 2

- [16] D. S. Marcus, T. H. Wang, J. Parker, J. G. Csernansky, J. C. Morris, and R. L. Buckner, “Open Access Series of Imaging Studies (OASIS): Cross-Sectional MRI Data in Young, Middle Aged, Nondemented, and Demented Older Adults,” *J. Cogn. Neurosci.*, vol. 19, pp. 1498–1507, 2007.
- [17] N. Aithal. (2023). OASIS Alzheimer’s Detection [Online]. Available: <https://www.kaggle.com/datasets/ninadaithal/imagesoasis>.
- [18] V. Kurama. (2024, Oct. 10). A Review of Popular Deep Learning Architectures: ResNet, InceptionV3, and SqueezeNet [Online]. Available: <https://www.digitalocean.com/community/tutorials/popular-deep-learning-architectures-resnet-inceptionv3-squeezeon>.
- [19] S. H. Tsang. (2018, Sept. 10). Review: Inception-v3 — 1st Runner Up (Image Classification) in ILSVRC 2015 [Online]. Available: <https://sh-tsang.medium.com/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>.

## Appendice

```
# Installazione della libreria Kaggle per l'accesso ai dataset
!pip install kaggle

# Importazione della libreria per gestire Google Drive
from google.colab import drive

# Montaggio di Google Drive per accedere ai file
drive.mount('/content/drive')

# Creazione di una directory per le API di Kaggle
!mkdir ~/.kaggle

# Copia del file delle credenziali di Kaggle nel percorso appropriato
! cp /content/drive/MyDrive/KaggleAPI ~/.kaggle/

# Impostazione dei permessi per il file delle credenziali
! chmod 600 ~/.kaggle/kaggle.json

# Download del dataset specificato da Kaggle
! kaggle datasets download ninadaithal/imagesoasis

# Decompressione del file zip del dataset
! unzip imagesoasis.zip

# Importazione delle librerie necessarie
import os
import tensorflow as tf
import keras
from keras.utils import load_img
import numpy as np
import pandas as pd
from sklearn.model_selection import GroupShuffleSplit
from tqdm import tqdm
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from imblearn.over_sampling import SMOTE
from tensorflow.keras import layers, models, metrics
```

```

from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras import regularizers
from tensorflow.keras.models import Model
import matplotlib.pyplot as plt
import seaborn as sns

# Definizione del percorso del dataset
dataset_path = '/content/Data'

# Inizializzazione di una lista per raccogliere i dati delle immagini
data = []

# Ciclo per contare le immagini in ciascuna classe del dataset
for folder in os.listdir(dataset_path):
    folder_path = os.path.join(dataset_path, folder)
    num_images = len(os.listdir(folder_path))
    data.append((num_images, folder))

# Creazione di un DataFrame avente come colonne il numero di immagini e la classe di appartenenza
df = pd.DataFrame(data, columns=['Num of Images', 'Class'])

# Ordinamento delle classi per la visualizzazione
class_order = ['Non Demented', 'Very mild Dementia', 'Mild Dementia', 'Moderate Dementia']

# Definizione della paletta di colori per le barre
palettes = {
    'Non Demented': 'green',
    'Mild Dementia': 'blue',
    'Moderate Dementia': 'purple',
    'Very mild Dementia': 'red'
}

# Creazione di un grafico a barre per visualizzare la distribuzione delle immagini per classe
plt.figure(figsize=(10, 6))

```

```

ax = sns.barplot(x='Class', y='Num of Images', hue=df['Class'], data=df, palette=palette, order=class_order)

# Aggiunta all'istogramma del numero di immagini per ciascuna classe
for p in ax.patches:
    height = p.get_height()
    ax.annotate(f'{height}',

                (p.get_x() + p.get_width() / 2., height),
                ha='center', va='bottom',
                xytext=(0, 0),
                textcoords='offset points')

# Impostazione delle etichette e del titolo del grafico
plt.xlabel('Classi')
plt.ylabel('Numero di Immagini')
plt.title('Distribuzione delle Immagini del Dataset OASIS1 Prima dell\'undersampling')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# Definizione dei percorsi delle classi nel dataset
dataset_path = '/content/Data'
very_mild_demented_path = os.path.join(dataset_path, 'Very mild Dementia')
non_demented_path = os.path.join(dataset_path, 'Non Demented')

# Conteggio e visualizzazione del numero di immagini per classe prima dell'undersampling
num_elements = len(os.listdir(very_mild_demented_path))
print('Numero di elementi di classe Very Mild Demented prima dell\'undersampling: ' + str(num_elements))

num_elements = len(os.listdir(non_demented_path))
print('Numero di elementi di classe Non Demented prima dell\'undersampling: ' + str(num_elements))

# Rimozione di immagini in eccesso dalla classe "Very Mild Demented"
i = 0
for image in os.listdir(very_mild_demented_path):
    if i > 5001:
        image_path = os.path.join(very_mild_demented_path, image)
        os.remove(image_path)

```

```

    i += 1
else:
    i += 1

# Conteggio e visualizzazione del numero di immagini per classe dopo l'undersampling
num_elements = len(os.listdir(very_mild_demented_path))
print('Numero di elementi di classe Very Mild Demented a seguito dell'undersampling: ' +
str(num_elements))

i = 0
for image in os.listdir(non_demented_path):
    if i > 5001:
        image_path = os.path.join(non_demented_path, image)
        os.remove(image_path)
        i += 1
    else:
        i += 1

num_elements = len(os.listdir(non_demented_path))
print('Numero di elementi di classe Non Demented a seguito dell'undersampling: ' + str(num_elements))

# Creazione delle liste per immagini, etichette e gruppi
dataset_path = '/content/Data'
images = []
labels = []
groups = []

# Ciclo per raccogliere i percorsi delle immagini e le rispettive etichette
i = 0
j = 0
for folder in os.listdir(dataset_path):
    folder_path = os.path.join(dataset_path, folder)
    for image in os.listdir(folder_path):
        image_path = os.path.join(folder_path, image)
        images.append(image_path)
        labels.append(folder)
        if i == 60:

```

```

i = -1
groups.append(j)
j += 1
else:
    groups.append(j)
i += 1

# Conversione delle liste in array NumPy
images = np.array(images)
labels = np.array(labels)
groups = np.array(groups)

# Creazione di un DataFrame con le informazioni delle immagini
df = pd.DataFrame({'images': images, 'labels': labels, 'groups': groups})

# Creazione di un DataFrame per visualizzare il numero di immagini per classe dopo l'undersampling
dataset_path = '/content/Data'
data = []
for folder in os.listdir(dataset_path):
    folder_path = os.path.join(dataset_path, folder)
    num_images = len(os.listdir(folder_path))
    data.append((num_images, folder))

df = pd.DataFrame(data, columns=['Num of Images', 'Class'])

# Visualizzazione della distribuzione delle immagini per classe
plt.figure(figsize=(10, 6))
ax = sns.barplot(x='Class', y='Num of Images', hue=df['Class'], data=df, palette=palette, order=class_order)
for p in ax.patches:
    height = p.get_height()
    ax.annotate(f'{height}', (p.get_x() + p.get_width() / 2., height),
                ha='center', va='bottom',
                xytext=(0, 0),
                textcoords='offset points')
plt.xlabel('Classi')
plt.ylabel('Numero di Immagini')

```

```

plt.title('Distribuzione delle Immagini del Dataset OASIS1 dopo l\'undersampling')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# Divisione del dataset in set di addestramento e test utilizzando GroupShuffleSplit
gss = GroupShuffleSplit(n_splits=1, test_size=0.2, random_state=2)
train_index, test_index = next(gss.split(images, labels, groups=groups))

group_train = groups[train_index]
group_test = groups[test_index]

# Divisione del set di addestramento in set di addestramento e validazione
gss_val = GroupShuffleSplit(n_splits=1, test_size=0.25, random_state=2)
train_index_final, val_index = next(gss_val.split(images[train_index], labels[train_index],
groups=group_train))

train_index_final = train_index[train_index_final]
val_index = train_index[val_index]

group_train_final = groups[train_index_final]
group_val = groups[val_index]

# Creazione degli array per le immagini e le etichette del set di addestramento, validazione e test
X_train = images[train_index_final]
y_train = labels[train_index_final]
X_val = images[val_index]
y_val = labels[val_index]
X_test = images[test_index]
y_test = labels[test_index]

# Creazione di DataFrame per i set di addestramento, validazione e test
df_train = pd.DataFrame({'images': X_train, 'labels': y_train})
df_val = pd.DataFrame({'images': X_val, 'labels': y_val})
df_test = pd.DataFrame({'images': X_test, 'labels': y_test})

# Creazione di generatori di immagini per il set di addestramento e validazione

```

```

train_datagen = ImageDataGenerator(rescale=1./255)
val_datagen = ImageDataGenerator(rescale=1./255)

# Creazione dei generatori per i set di addestramento e validazione
train_generator = train_datagen.flow_from_dataframe(df_train, x_col='images', y_col='labels',
target_size=(299, 299), batch_size=32, class_mode='categorical', shuffle=True)
val_generator = val_datagen.flow_from_dataframe(df_val, x_col='images', y_col='labels', target_size=(299,
299), batch_size=32, class_mode='categorical', shuffle=True)

# Creazione del modello di rete neurale utilizzando l'architettura InceptionV3
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(299, 299, 3))

# Congelamento dei layer del modello base
for layer in base_model.layers:
    layer.trainable = False

# Creazione del modello finale
model = Sequential()
model.add(base_model)
model.add(GlobalAveragePooling2D())
model.add(Dropout(0.5))
model.add(Dense(4, activation='softmax'))

# Compilazione del modello
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Definizione dei callback per l'early stopping e il ridimensionamento del learning rate
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=1e-6)

# Addestramento del modello
history = model.fit(train_generator, validation_data=val_generator, epochs=30, callbacks=[early_stopping,
reduce_lr])

# Visualizzazione delle curve di apprendimento
plt.figure(figsize=(14, 5))
plt.subplot(1, 2, 1)

```

```

plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Curva di Apprendimento - Accuratezza')
plt.xlabel('Epochs')
plt.ylabel('Accuratezza')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Curva di Apprendimento - Perdita')
plt.xlabel('Epochs')
plt.ylabel('Perdita')
plt.legend()

plt.tight_layout()
plt.show()

# Valutazione del modello sul set di test
test_datagen = ImageDataGenerator(rescale=1./255)
test_generator = test_datagen.flow_from_dataframe(df_test, x_col='images', y_col='labels', target_size=(299, 299), batch_size=32, class_mode='categorical', shuffle=False)

# Valutazione del modello e visualizzazione dei risultati
loss, accuracy = model.evaluate(test_generator)
print('Loss sul set di test:', loss)
print('Accuratezza sul set di test:', accuracy)

# Previsione delle etichette per il set di test
predictions = model.predict(test_generator)
predicted_classes = np.argmax(predictions, axis=1)
true_classes = test_generator.classes

# Creazione di una matrice di confusione
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(true_classes, predicted_classes)

```

```
# Visualizzazione della matrice di confusione
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=test_generator.class_indices.keys(),
            yticklabels=test_generator.class_indices.keys())
plt.xlabel('Predicted Classes')
plt.ylabel('True Classes')
plt.title('Matrice di Confusione')
plt.show()
```