UNIVERSITÀ DEGLI STUDI DI CAGLIARI

FACOLTÀ DI SCIENZE

Corso di Laurea Triennale in Informatica

A Matlab Application for Covid-19 Medical imaging

Relatori
Prof.ssa Cecilia Di Ruberto
Dott. Andrea Loddo

Studente
Luca Zedda
Matr. N. 65636

ANNO ACCADEMICO 2020/2021

Lo studio riportato all'interno di questa tesi fa riferimento all'analisi di immagini mediche, nello specifico, di tomografie computeritizzate riguardanti la regione polmonare mediante estrazione delle misure (features), mirata il riconoscimento della presenza di Covid-19 attraverso l'utilizzo di classificatori. Di seguito verranno esaminate le implementazioni di relativi test, la struttura del dataset utilizzato e i risultati di tale ricerca. L' intero studio è stato svolto mediante l'utilizzo di Matlab.

Indice

Capitolo 1

Introduzione

In tutto il mondo, giornalmente, migliaia di persone muoiono a causa del Covid-19. Risulta pertanto necessario utilizzare qualsiasi strumento a disposizione per salvaguardare il maggior numero di vite. Uno dei tanti strumenti che l'informatica fornisce per alleviare questa problematica è la **diagnostica per immagini**. L'indagine svolta in questo lavoro ricopre molteplici tipologie di feature e classificatori, con l'obiettivo primario di classificare delle immagini, riconoscendole come rappresentanti tomografie computerizzate (CT) di pazienti affetti da COVID-19. L'indagine svolta inoltre cerca di coprire un discreto numero di features e classificatori in modo da ottenere il migliore dei risultati combinando questi ultimi. Il dataset preso in considerazione è COVIDx CT-2A composto da 194,922 immagini [1] riguardanti le tomografie computerizzate di 3745 pazienti eventuali specifiche su features e classificatori utilizzati verranno discusse in seguito.
Gli obbiettivi ultimi di questo lavoro sono i seguenti:

1. Analisi delle features estratte in modo da comprendere quali siano le più adatte allo scopo del lavoro
2. Analisi dei classificatori allenati su le features analizzate nel punto precedente
3. Estensione all'utilizzo di features derivanti da Convolutional Neural Network
4. Analisi risultati ottenuti mediante ensamble
5. Analisi cross-dataset dei risultati ottenuti

Al termine della stesura verranno confrontati i risultati ottenuti dal sistema descritto in questa tesi con quelli presenti nelle seguenti pubblicazioni: "A light CNN for detecting COVID-19 from CT scans of the chest"[9] e "COVID-CT-MD,

[1] Nel nostro caso a livelli di grigio (grayscale)

COVID-19 computed tomography scan dataset applicable in machine learning and deep learning"[?].

Capitolo 2

Stato dell'arte

Lo studio del medical imaging in generale verge attualmente verso l'utilizzo di reti neurali convoluzionali (***Convolutional Neural Networks, CNN***), in quanto risultano molto efficienti nel raccogliere features di qualità in maniera automatica per poi classificare i dati in input. Conseguentemente, anche gli approcci riguardanti il Covid-19 si orientano in tal senso. Alcuni esempi di applicazioni vengono discussi e realizzati in [4] e [9].

Le CNN vennero introdotte per la prima volta da Yann LeCun nel 1980 e, a oggi risultano trovare applicazioni non solo nel medical imaging, ma vengono applicate in un ampia gamma di campi, ad esempio automobili autonome, droni, e così via.

Gli strati (**layer**) presenti in questo tipo di reti neurali sono generalamente di tre tipi:

- **Convoluzione**: scorre dei filtri sulle immagini, ciascuno dei quali attiva determinate features;
- **ReLu**: Mappa a zero i valori negativi raccolti durante la fase di convoluzione permettendo maggiore velocità e accuratezza;
- **Pooling**: Semplifica l'output dei layer precedenti permettendo di ridurre il carico computazionale.

Ciascuno di questi strati viene ripetuto fino a completare la raccolta delle features legate alle immagini. Terminato questo processo, viene eseguita la classificazione mediante un classificatore inserito in un livello ad hoc, che sfrutta le informazioni dal suo livello immediatamente precedente, completamente connesso rispetto ai livelli precedenti.

Capitolo 3

Metodi e Materiali

3.1 Descrizione dataset COVIDx CT-2A

Il dataset COVIDx CT-2A [3] comprende 194,922 immagini di tomografie computerizzate appartenti a 3,745 pazienti di età compresa tra 0 e 93 anni (età media 51 anni) supervisionate da esperti del settore. L'insieme dei casi descritti in questo documento sono stati raccolti dalle seguenti organizzazioni e pubblicazioni: [6],[14],[8],[19],[7],[16],[2]. Le immagini contenute nel dataset e i relativi pazienti sono distribibuite nei set di addestramento, validazione e test, come indicato nelle seguenti tabelle. Le classi riguardanti le immagini del dataset sono le seguenti:

- **Normale**: La classe delle immagini appartenenti a pazienti non infetti da covid-19 o polmonite;
- **Polmonite**: La classe delle immagini appartenti a pazienti affetti dalla pomonite;
- **Covid-19**: La classe delle immagini appartenti a pazienti affetti dal covid-19.

Tipo	Normale	Polmonite	COVID-19	Totale
Addestramento	35996	25496	82286	143778
Validazione	11842	7400	6244	25486
Test	12245	7395	6018	25658

Tabella 3.1: *Distribuzione delle immagini*

Tipo	Normale	Polmonite	COVID-19	Totale
Addestramento	321	558	1958	2837
Validazione	126	190	166	482
Test	126	125	175	426

Tabella 3.2: *Distribuzione dei pazienti*

3.2 Descrizione features

Le features utilizzate in questa tesi a differenza di quelle estratte in maniera automatica nelle reti neurali convoluzionali citate nel secondo capitolo [Stato dell'arte] fanno parte della famiglia cosiddetta "tradizionale". Il loro utilizzo prevede generalmente la presenza di un ingegnere delle features, il cui compito prevede di gestire la fase di estrazione e selezione delle features da impiegare nel processo di classificazione. L'utilizzo di features di nuova generazione viene attualmente affiancato a quelle tradizionali in modo da ottenere un'accuratezza maggiore. Alcuni lavori, ad esempio [12], parlano di questa dualità e inoltre cercano di portare alla luce i benefici di entrambi gli approcci. L'approccio legato al deep learning non necessita la figura di un'esperto nel campo delle features, seppur questo fatto possa far risaltare l' approccio deep learning come più comodo in quanto non è richiesto uno studio approfondito delle features da ricercare quest' ultimo risulta essere efficace perlopiù in problemi in cui la soluzione agli stessi è sempre una soltanto. A seguire verranno elencate le features prese in considerazione in questa tesi.

3.2.1 Haralick features

Le features legate alla matrice di co occorenza *(Haralick features)* vennero presentate per la prima volta nel 1973 nel documento "Textural Features for Image Classification" [15], in cui vennero proposte 28 fatures ma nello studio riportato in questa tesi vengono utilizzate solo le prime 13. La matrice di co occorrenza *GLCM* è un metodo statistico che permette di esaminare la tessitura *Texture* di un'immagine: essa rappresenta quante volte una detrminata coppia di pixel e contorno sono presenti in un'immagine e viene calcolata attraverso la seguente funzione.

$$P_{\triangle x,\triangle y}(i,j) = \sum_{x=1}^{n}\sum_{y=1}^{m}\begin{cases}1, \text{ se } I(x,y)=i \text{ e } I(x+\triangle x, y+\triangle y)=j\\ 0, \text{ altrimenti}\end{cases} \tag{3.1}$$

In cui: $(\triangle x, \triangle y)$ rappresenta il contorno. L'operazione viene ripetuta ruotando la matrice originale attorno alle diagonali e eseguendo la media delle matrici ottenute, gli angoli rappresentanti queste rotazioni sono i seguenti: $0, 45, 90, 135$.

Riprendendo il documento [15] le 13 su 28 features di Haralik prese in considerazione verrano elencate nelle seguenti righe ma prima è necessario stabilire le seguenti notazioni:

$p(i, j)$ = la (i, j)esima entrata in una matrice di co occorrenza a livelli di grigio normalizzata

$p_x(i)$ = la i-esima entrata della matrice di probabilità marginale ottenuta sommando le righe di $p(i, j)$ ovvero: = $\sum_{j=1}^{N_g} P(i, j)$.

N_g il numero di distinti livelli di grigio presenti nell'immagine

$\sum_j$ e $\sum_j$ corrispondono a $\sum_{i=1}^{N_g}$ e $\sum_{j=1}^{N_g}$

$p_y(j) = \sum_{i=1}^{N_g} p(i, j)$

$p_{x+y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g}{}_{i+j=k} p(i, j), k = 2, 3, ..., 2N_g.$

$p_{x-y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g}{}_{|i-j|=k} p(i, j), k = 0, 1, ..., N_g - 1.$

Stabilite queste premesse verranno ora elencate le features prese in considerazione

1. Angular Second Moment

$$f_1 = \sum_i \sum_j \{p(i, j)\}^2 \tag{3.2}$$

2. Contrasto

$$f_2 = \sum_{n=0}^{N_g-1} n^2 \{ \sum_{i=1}^{N_g} \sum_{j=1}^{N_g}{}_{|i-j|=k} p(i, j) \} \tag{3.3}$$

3. Correlazione

$$f_3 = \frac{\sum_i \sum_j (ij) p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \tag{3.4}$$

dove $\sigma_x, \sigma_y, \mu_x$ e μ_y corrispondono a media e deviazione standard di p_x e p_y

4. Varianza

$$f_4 = \sum_i \sum_j (i - \mu)^2 p(i,j) \tag{3.5}$$

5. Inverse Difference Moment

$$f_5 = \sum_i \sum_j \frac{1}{1 + (i-j)^2} p(i,j) \tag{3.6}$$

6. Somma della media

$$f_6 = \sum_{i=2}^{2N_g} i p_{x+y}(i) \tag{3.7}$$

7. Somma della varianza

$$f_7 = \sum_{i=2}^{2N_g} (i - f_8)^2 p_{x+y}(i) \tag{3.8}$$

8. Somma dell'entropia

$$f_8 = -\sum_{i=2}^{2N_g} p_{x+y}(i) log p_{x+y}(i) \tag{3.9}$$

9. Entropia

$$f_9 = -\sum_i \sum_j p(i,j) log(p(i,j)) \tag{3.10}$$

10. Differenza della varianza

$$f_{10} = \text{varianza di } p_{x-y} \tag{3.11}$$

11. Differenza dell'entropia

$$f_{11} = -\sum_{i=2}^{N_g - 1} p_{x+y}(i) log p_{x-y}(i) \tag{3.12}$$

12. Informazioni e misure della correlazione(1)

$$f_{12} = \frac{HXY - HXY1}{max\{HX, HY\}} \tag{3.13}$$

13. Informazioni e misure della correlazione(2)

$$f_{13} = 1 - exp[-2.0(HXY2 - HXY)]^{\frac{1}{2}} \tag{3.14}$$

Dove

$$HXY = -\sum_i \sum_j p(i,j) log(p(i,j)) \tag{3.15}$$

$$HXY1 = -\sum_i \sum_j p(i,j) log\{p_x(i)p_y(j)\} \tag{3.16}$$

$$HXY2 = -\sum_i \sum_j p_x(i)p_y(j) log\{p_x(i)p_y(j)\} \tag{3.17}$$

3.2.2 Local Binary Pattern

Il Local Binary Pattern *(LBP)* rappresenta uno dei descrittori più utilizzati e famosi nell'ambito della classificatione delle texture, viene utilizzato principalmente per la classificazione di oggetti, scene e volti. Il concetto di Local Binary Pattern venne discusso per la prima volta nel 1993 ma venne reso popolare nel 2003 dalla pubblicazione [13]. Il funzionamento dei LBP prevede una conversione preventiva dell'immagine, nel caso fosse necessario, in gray scale e, successivamente, l'immagine viene divisa in porzioni *(patches)*. Infine, si procede a calcolare, per ogni singolo pixel dell'immagine, la seguente funzione.

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) 2^P \tag{3.18}$$

$$s(x) = \begin{cases} 1, \text{ se } x >= 0 \\ 0, \text{ altrimenti} \end{cases} \tag{3.19}$$

Dove g_p rappresenta ilp-esimo elemento del contorno del pixel centrale: g_c, P e R rappresentano invece rispettivamente il numero di elementi del contorno e il raggio del contorno. Per poter visualizzare al meglio questa funzione è possibile prendere come riferimento la seguente immagine:

Per ogni elemento all'interno della circonferenza se il suo valore supera il valore dell'elemento centrale allora al valore dell'output per l'elemento di riferimento verrà sommato 2^{indice}.

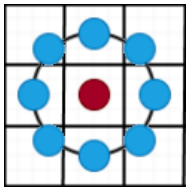

Figura 3.1: *Esempio circonferenza LBP*

In seguito viene calcolato l'istogramma della matrice ottenuta tramite la seguente funzione.

$$H_i = \sum_{x,y} I\{LBP(x,y) = i\}, i = 0, ..., n-1 \tag{3.20}$$

E successivamente nel caso in cui le porzioni di immagini abbiano dimensioni differenti, l'istogramma deve essere normalizzato mediante la seguente equazione:

$$N_i = \frac{H_i}{\sum_{j=0}^{n-1} H_j} \tag{3.21}$$

3.2.3 Hu Moments

I *momenti regolari* sono stati introdotti per la prima volta da Hu nel 1962 [5]. Presentano un insieme composto da 7 momenti invarianti a: traslazione, scalatura, rotazione e rifessione. In particolare, il settimo momento presenta un segno invertito nel caso di un eventuale riflessione dell'immagine
I sette momenti di Hu vengono definiti come segue.

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta 02)^2 + (2\eta_{11})^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} - \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12}^2) - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned} \tag{3.22}$$

Dove l'invariante di scalatura e traslazione η_{ij} viene costruito tramite la divi-

sione di un momento centrale con un momento centrale scalato sullo zero

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{1+\frac{i+j}{2}}} \tag{3.23}$$

Il momento centrale (μ_{ij}) è definito tramite la seguente funzione:

$$\mu_{pq} = \sum_x \sum_y (x - \hat{x})^p (y - \hat{y})^q f(x, y) \tag{3.24}$$

A sua volta $\hat{x}$ corrisponde al centroide dell' immagine o baricentro

$$\{\hat{x}\hat{y}\} = \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \tag{3.25}$$

Dove M_{pq} è un momento di ordine $p + q$ definito dalla seguente funzione

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \tag{3.26}$$

3.2.4 Zernike Moments

I **momenti di Zernike** sono momenti ortogonali, introdotti per la prima volta nel 1930 dal fisico vincitore del premio nobel Fritz Zernike per descrivere le aberrazioni ottiche. Questi ultimi risultano essere invarianti alla traslazione, rotazione e scalatura. I momenti di Zernike di basso ordine descrivono la forma in generale di un'immagine mentre all'aumentare dell'ordine si aggiunge complessità a questa descrizione. I polinomi di Zernike sono ortogonali all'interno di un cerchio unitario in coordinate polari:

$$x^2 + y^2 = 1 \tag{3.27}$$

Un set di polinomi di Zernike denominato $V_{nm}(x, y)$ viene definito nel seguente modo:

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) exp(jm\theta) \tag{3.28}$$

Il polinomio di Zernike è quindi diviso in due parti: R_{nm} è **il polinomio radiale** e $jm\theta$ la parte complessa. n è un numero positivo intero detto anche **ordine** e m un numero intero alternativamente chiamato **ripetizione**. Entrambi sono soggetti alle seguenti condizioni:

$$m \in 0, \pm 1, ..., \pm |n| \mid n|m|pari \tag{3.29}$$

Il polinomio radiale i è definito come:

$$R_{nm}(r) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2}-s)!(\frac{n-|m|}{2}-s)!} \rho^{n-2s} \tag{3.30}$$

La forma discreta dei momenti di Zernike per un'immagine digitale di dimensione M x N è definita come segue:

$$A_{nm} = \frac{p+1}{\pi} \sum_x \sum_y I(x,y) V_{n(-m)}(x,y) \tag{3.31}$$

Dove $I(x,y)$ corrisponde all'intensità dell'immagine in un pixel.

Avendo a disposizione tutti i momenti A_{mn} di una funzione dell'immagine $f(x,y)$ fino ad un ordine massimo definito n_{max} è possibile ricostruire la funzione $\hat{f}(x,y)$ nel seguente modo:

$$\hat{f}(x,y) \approx \sum_{n=0}^{n_{max}} \sum_m A_{nm} V_{nm}(\rho, \theta) \tag{3.32}$$

3.2.5 Legendre Moments

I **momenti di Legendre** sono anch'essi momenti ortogonali. Trovano ampio utilizzo non solo nel campo del riconoscimento delle immagini ma sono estensivamente utilizzati nell'ambito dell'ingegneria e della fisica, ad esempio risultano essere particolarmente efficaci nella determinazione della funzione d'onda degli elettroni attorno all'atomo.
I momenti di Legendre di ordine $(m+n)$ sono definiti come:

$$\lambda_{xy} = \frac{(2m+1)(2n+1)}{4} \sum_x \sum_y P_m(x) P_n(y) f(x,y) \tag{3.33}$$

dove $m, n = 0, 1, 2, ...$, e $P_{n,m}$ sono i polinomi di Legendre, i quali formano una base ortogonale nell'intervallo [-1,1], e sono definiti come:

$$P_n(x) = \sum_{j=0}^{n} a_{nj} x^j \tag{3.34}$$

Mentre a_{nj} sono i coefficienti di Legendre e sono definiti nel seguente modo:

$$a_{nj} = (-1)^{(n-j)/2} \frac{1}{2^n} \frac{(n+j)!}{\frac{(n-j)}{2}!(\frac{(n+j)}{2})!j!} \text{ dove } n-j \text{ restituisce un numero pari} \tag{3.35}$$

3.2.6 Chebyshev Moments

I momenti di Chebyshev vennero introdotti nella pubblicazione [11], essi sono basati sui polinomi di Chebyshev, descritti dalla seguente ricorrenza:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x) \end{aligned} \tag{3.36}$$

I momenti di Chebyshev risolvono le maggiori problematiche che vengono a presentarsi durante la computazione dei momenti di Legendre e Zernike. Infatti, questi utimi necessitano che le coordinate spaziali siano normalizzate ad un determinato intervallo, solitamente viene preso in considerazione l'intervallo $[-1, +1]$. Inoltre è necessario che gli integrali della funzione del momento debbano essere approsimati tramite delle sommatorie senza dover perdere le proprietà dell'ortogonalità.
I momenti di Chebyshev infatti sono ortogonali al dominio dello spazio delle cordinate dell'immagine, inoltre non necessitano di alcuna approssimazione. Proprio per questi motivi, infatti, i problemi legati ai momenti descritti in precedenza risultano essere completamente assenti, inoltre le immagini ricostruite tramite la trasformata inversa dei momenti di Chebyshev risultano essere migliori rispetto a quelle ottenute utilizzando i momenti di Legendre e Zernike. [11]
I momenti di Chebyshev vengono calcolati nel seguente modo. Data un'immagine $N \times N$, si cercano prima polinomi ortogonali discreti $t_n(x)$ che soddisfano la

condizione:

$$\sum_{x=0}^{N-1} t_m(x)t_n(x) = \rho(n,N)\delta_{mn}, \qquad m,n = 0,1,2,...,N-1 \tag{3.37}$$

dove $\rho(n,N)$ è la norma al quadrato dell'insieme polinomiale t_n. I classici polinomi discreti di Chebyshev soddisfano la proprietà di ortogonalità, con:

$$\rho(n,N) = \frac{N(N^2-1)(N^2-2^2)...(N^2-n^2)}{2n+1}, \quad n = 0,1,...,N-1 \tag{3.38}$$

e hanno la seguente relazione di ricorrenza:

$$(n+1)t_{n+1}(x) - (2n+1)(2x-N+1)t_n(x) + n(N^2-n^2)t_{n-1}(x) = 0, \quad n = 1,...,N-1 \tag{3.39}$$

3.3 Descrizione classificatori

TODO descrizione generale classificatori

3.3.1 Support Vector Machine

Il Support Vector Machine **SVM** è un modello di machine learning supervisionato, che si comporta particolarmente bene quando il numero di dati a disposizione per l'allenamento del modello risulta essere scarso. Il funzionamento del classificatore SVM permette di costruire uno o più iperpiani all'interno di uno spazio a più dimensioni.

La trasformazione dei dati su spazi a dimensione superiore risulterebbe molto dispendiosa a livello computazionale, per questo i dati vengono mappati in maniera implicita da funzioni denominate kernel. Tradizionalmente questo genere di approccio viene chiamato **kernel trick**.

3.3.2 Naive Bayes

Il classificatore Naive Bayes è un algoritmo di classificazione basato sul teorema di Bayes.

$$P(A_i|E) = \frac{P(E|A_i)P(A_i)}{\sum_{j=1}^{n} P(E|A_j)P(A_j)} \tag{3.40}$$

Questo algoritmo cerca di stimare le probabilità condizionali relative al problema, esse infatti non sono disponibili a priori. L' algoritmo esegue i seguenti passi:

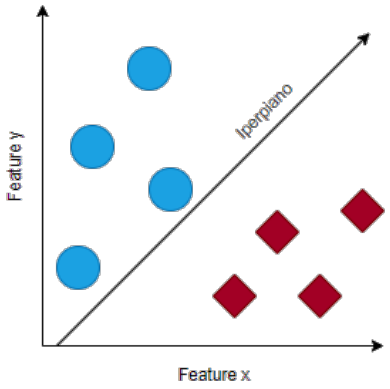

Figura 3.2: *Esempio funzionamento SVM*

1. Stima la denisità dei predittori per ogni classe
2. Calcola la probabilità a posteriori seguendo la seguente formula derivata dal teorema di Bayes

$$\hat{P}(Y = k|X_1, ..., X_p) = \frac{\pi(Y = k) \prod_{j=1}^{P} P(x_j|Y = k)}{\sum_{k=1}^{K} \pi(Y = k) \prod_{j=1}^{p} P(x_j|Y = k)} \tag{3.41}$$

Dove:

- Y è la variabile casuale corrispondente all'indice della classe in osservazione;
- $X_1, ..., X_p$ sono i predittori casuali dell'osservazione;
- $\pi(Y = k)$ è la probabilità a priosi della classe di indice k.

3.3.3 Reti Neurali

L'origine delle reti neurali artificiali **ANN** deriva dal documento [18] che presenta il concetto di neurone artificiale. Una rete neurale è un classificatore che basa il suo funzionamento su una struttura stratificata che ha come ispirazione una versione semplificata della struttura dei neuroni nel cervello umano. Questa struttura comprende strati composti da nodi connessi. A livello implementativo i layer di una rete neurale sono suddivisi in tre macrocategorie:

- Layer di input
- Layer nascosti
- Layer di output

I layer di input ricevono i dati da classificare, layer nascosti stabiliscono i pesi per ciascuna connessione mentre il layer di output contiene un nodo per classe di elemento da classificare, il nodo con il valore maggiore risulterà essere quello scelto come risultato della classificazione. Ogni nodo lavora in parallelo e trasmette il proprio output al nodo successivo il quale utilizzerà queste informazioni come input.

3.3.4 Ensemble

L'apprendimento d'insieme o **Ensemble** nell' ambito del machine learning consiste in una serie di modelli di insieme che utilizzano più modelli per ottenere una classificazione migliore rispetto ai singoli modelli costituenti. Le tipologie di Ensemble sono le seguenti:

- Bagging
- Boosting
- Stacking

La tipologia *Bagging* utilizza lo stesso peso per ciascun modello costituente, a differenza della tipologia *Boosting* che assegna un determinato peso a partire dall'accuratezza del modello. In entrambi i modelli l'output complessivo sarà stabilito mediante votazione. Nella terza tipologia, ovvero lo *Stacking* non viene utilizzata una votazione ma bensì viene utilizzato un meta-classificatore che utilizzerà le predizioni dei sotto-modelli per generare l'output finale. In particolare nello studio riportato in questa tesi vengono prese in considerazione le tipologie *Bagging* e

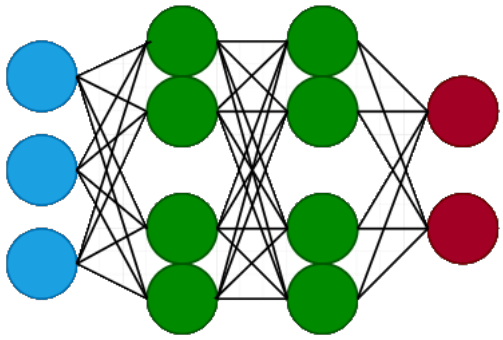

Figura 3.3: *Esempio di rete neurale*

Boosting entrambi utilizzando come sotto-modelli gli alberi decisionali. Il processo di classificazione avviene secondo la seguente descrizione

1. Per ogni classe $c \in C$ eper ogni albero $t = 1, ..., T$ viene calcolata la probabilità a posteriori della classe c data l'osservazione x utilizzando l'albero t: $\hat{P}_t(c|x)$
2. Viene calcolata la media pesata o aritmetica, a seconda della tipologia scelta, per la probabilità a posteriori di una determinata classe
3. Viene estratta come otput la classe con il valore maggiore

3.3.5 Alberi Decisionali

Gli alberi decisionali vengono utilizzati come metodo molto intuitivo per la classificazione di dati. Un albero decisionale predice la classe di un determinato elemento scorrendo i nodi fino alla radice, ciascun nodo effettua un controllo su determinate features prese in esame, a seconda del valore di queste features il cammino lungo l'albero prosegue in una determinata direzione. Terminati i controlli si arriva ad un nodo radice, sulla base di quest'ultimo nodo si ottiene il risultato della classificazione

3.3.6 K-nearest neighbors

L'algoritmo k-nearest neighbors **(k-NN)** venne sviluppato per la prima volta nel 1951 e discusso nel seguente articolo [17], il funzionamento del classificatore k-NN vine definito nel seguente modo: La classe di un oggetto da classificare è data dalla votazione dei k oggetti vicini, ovvero la classe più comune degli oggetti vicini.

3.4 Descrizione metriche

Per il calcolo delle metriche utilizzate è necessario dare delle precisazioni riguardo i temini utilizzati:

Popolazione totale (P+N)	Predizione "positività" (PP)	Predizione "nagatività" (PN)
Condizione realmente positiva (P)	Vera positività (TP)	Falsa negatività (FN)
Condizione realmente negativa (N)	Falsa positività (FP)	Vera negatività (TN)

Tabella 3.3: *Definizione metriche*

Per i risultati, è stato tenuto conto delle seguenti metriche:

- Accuracy: si tratta del rapporto tra in numero di veri positivi (TP) e negativi (TN) e il numero totale di predizioni

$$Accuracy = \frac{TP + TF}{TP + TF + FP + FN},$$

- Precision: si tratta del rapporto tra il numero dei veri positivi (TP) e la somma dei veri positivi e falsi positivi, ovvero (PP);

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

- Recall o Sensitivity si tratta del rapporto tra veri positivi (TP) e la somma di veri positivi (TP) e falsi negativi (FN), ovvero (P) [10];

$$Recall = \frac{TP}{TP + FN}.$$

- Specificity: si tratta del rapporto tra il numero di veri negativi e la somma di falsi positivi (FP) e veri negativi (TN), ovvero (N) [10];

$$Specificity = \frac{TN}{FP + TN},$$

- F-measure: si tratta di una media armonica dei valori di Precision e Recall;

$$F - measure = 2\frac{TP}{TP + FP + FN}$$

- MAvG: è la Macro-average geometric, definita come la media geometrica dell'Accuracy parziale di ciascuna classe. [1]

$$MAvG = (\prod_{i=1}^{J} Acc_i)^{\frac{1}{J}},$$

- MAvA: è la Macro-average arithmetic, definita come la media aritmetica dell'Accuracy parziale di ciascuna classe. [1]

$$MAvA = \frac{\sum_{i=1}^{J} Acc_i}{J}$$

Capitolo 4

Risultati sperimentali

Descrizione generale dei test fatti

4.1 Estrazione delle fetures

Features hand-crafted

La metodologia utilizzata per l'estrazione delle features hand-crafted è stata la seguente:

```
   Input: Il dataset
1  /* Lista contenente la stringhe relative alle features da
      estrarre.  Ad esempio nome_ord_rep, dove ord è l'ordine
      del momento e rep la ripetizione.                        */
2  lista_descrittori = [...]
3  for i ← 0 to lunghezza(lista_descrittori) do
4  |  vettore_features = []; vettore_labels = [];
5  |  for j ← 0 to lunghezza(dataset) do
6  |  |  features_img =
   |  |   estrai_features_img(dataset[j], lista_descrittori[i]);
   |  |   features_img = trasposta(features_img);
   |  |   label_img = dataset[j].label;
   |  |   vettore_features[j] = features_img;
   |  |   vettore_labels[j] = label_img;
7  |  end
8  |  matrice_labels_features =
   |   concatena_orizzontamente(vettore_labels =, vettore_features);
   |   salva_sul_disco(matrice_labels_features);
9  end
```

Algorithm 1: Estrazione features HC

In particolare il contenuto della lista dei descrittori nel lavoro svolto in questa tesi èstato il seguente:

- Momenti di Zernike per i seguenti ordini e ripetizioni:

  [(4,2) ; (4,4) ; (5,3) ; (5,5) ; (6,2) ; (6,4) ; (6,6) ; (7,3) ; (7,5) ; (7,7) ; (7,8) ; (7,9) ; (8,2) ; (8,4) ; (8,6) ; (8,8) ; (9,3) ; (9,5) ; (9,7) ; (9,9) ; (9,11) ; (10,2) ; (10,4) ; (10,6)]

- Momenti di Legendre integrando tramite la regola del trapezio per i seguenti ordini: [3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; 10]
- Momenti di Legendre integrando tramite la regola di Simpson per i seguenti ordini: [3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; 10]
- Momenti di Chebyshev per i seguenti ordini: [3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; 10]
- Momenti di Chebyshev del secondo tipo per i seguenti ordini: [3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; 10]
- Features di Haralik, le prime 13
- Local binary pattern di raggio 1 a 8 vicini

L'estrazione dei momenti di Zernike è stata velocizzata riscrivendo la funzione di estrazione in codice CUDA permettendo l'esecuzione su Scheda Video, riducendo le tempistiche di estrazione a $1/5$ delle tempistiche originali.

Features derivanti da CNN

La metodologia utilizzata per l'estrazione delle features derivanti da CNN è stata la seguente:

```
   Input: Il dataset
1  /* Lista contenente la stringhe relative al layer della
      rispettiva cnn da cui prelevare le attivazioni          */
2  lista_layer = [...]
3  /* Lista contenente le cnn addestrate sul dataset preso in
      considerazione                                          */
4  lista_cnn = [...]
5  for i ← 0 to lunghezza(lista_cnn) do
6  |  vettore_features = []; vettore_labels = [];
7  |  for j ← 0 to lunghezza(dataset) do
8  |  |  features_img =
   |  |   estrai_attivazioni_cnn(dataset[j], lista_layer[i], lista_cnn[i]);
   |  |   features_img = trasposta(features_img);
   |  |   label_img = dataset[j].label;
   |  |   vettore_features[j] = features_img;
   |  |   vettore_labels[j] = label_img;
9  |  end
10 |  matrice_labels_features =
   |   concatena_orizzontamente(vettore_labels =, vettore_features);
   |   salva_sul_disco(matrice_labels_features);
11 end
```

Algorithm 2: Estrazione features CNN

Le architetture CNN prese in considerazione nel lavoro svolto in questa tesi sono le seguenti: AlexNet, GoogleNet, Inceptionv3, MobileNetv2, ResNet18, ResNet50, ResNet101, ShuffleNet, Vgg16, Vgg19

4.2 Training dei classificatori

4.3 Test svolti

Capitolo 5

Conclusioni

Capitolo 6

Sviluppi futuri

Bibliografia

[1]

[2] Covid-19.

[3] Covidx ct-2a, https://www.kaggle.com/hgunraj/covidxct.

[4] Irmak E. Implementation of convolutional neural network approach for COVID-19 diseasedetection.

[5] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):179–187, 1962.

[6] et al. K. Zhang, X. Liu. Clinically applicable ai system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography. *Cell*, no. 6:1423–1433, 2020.

[7] et al. M. Jun, W. Yixin. Towards efficient covid-19 ct annotation: A benchmark for lung and infection segmentation. *arXiv preprint arXiv:2004.12537*, 2020.

[8] et al. M. Rahimzadeh, A. Attar. A fully automated deep learning-based network for detecting covid-19 from a new and large lung ct scan dataset, 2020.

[9] Giuseppe Placidi Matteo Polsinelli, Luigi Cinque. A light CNN for detecting COVID-19 from CT scans of the chest.

[10] Charles E. Metz. Basic principles of roc analysis. *Seminars in Nuclear Medicine*, 8(4):283–298, 1978.

[11] Ramakrishnan Mukundan, Seng-Huat Ong, and P.A. Lee. Discrete orthognal moment features using chebyshev polynomials. 01 2000.

[12] et al. Niall O' Mahony, Sean Campbell. Deep Learning vs. Traditional Computer Vision.

[13] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[14] et al. Peng An, Sheng Xu. "Ct images in covid-19 [data set], 2020.

[15] Its'Hak Dinstein Robert M. Haralik, K. Shanmugam. Textural Features for Image Classification. *IEEE Transactions on systems, man and cybernetics.*

[16] et al. S. Armato III, G. McLennan. Data from lidc-idri, 2015.

[17] B. W. Silverman and M. C. Jones. E. fix and j.l. hodges (1951): An important contribution to nonparametric discriminant analysis and density estimation: Commentary on fix and hodges (1951). *International Statistical Review / Revue Internationale de Statistique*, 57(3):233, 1989.

[18] W. Pitts W. McCulloch. A logical calculus of ideas immanent in nervous activity, bulletin of mathematical biophysics. *A Logical Calculus of the Ideas Immanent in Nervous Activity". With Walter Pitts. In: Bulletin of Mathematical Biophysics Vol 5, pp 115–133.*, 1943.

[19] J. Yang W. Ning, S. Lei. Open resource of clinical data from patients with pneumonia for the prediction of covid-19 outcomes via deep learning. *Nature Biomedical Engineering*, no. 4:1197–1207, 2020.

Elenco delle figure

Elenco delle tabelle

Ringraziamenti