

ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il Management

**COVID-19 X-RAY
ANALISI CON RETI NEURALI**

Relatore:
Chiar.ma Prof. ssa
ELENA LOLI PICCOLOMINI
Correlatore:
Chiar.mo Dott.
DAVIDE EVANGELISTA

Presentata da:
ANDREA ACCORNERO

Sessione II
Anno Accademico 2020/2021

Indice

Introduzione	4
1 Machine Learning	7
1.1 Machine Learning e Classificazione	7
1.2 Matrice di Confusione	8
1.3 Deep Learning	9
1.4 Reti Neurali Artificiali	10
1.5 Overfitting e Underfitting	11
2 Reti Neurali Convoluzionali	15
2.1 Layers	15
2.2 Livello Convoluzionale	17
2.2.1 Filtri	17
2.3 Livello ReLU	18
2.4 Livello Pool	19
2.5 Livello Fully Connected	19
3 Modello	21
3.1 Struttura della Rete Neurale	21

3.1.1	Funzioni di Attivazione	22
3.2	Parametri	24
3.2.1	Filtr	25
3.2.2	Padding	25
3.2.3	Kernel Regularizer	27
3.2.4	Dropout	27
4	Risultati	29
4.1	Introduzione ai dati	29
4.2	Andamento Dropout	30
4.3	Andamento Kernel_Regularizer	31
4.4	Matrice di Confusione	32
4.5	Osservazioni	33
Conclusioni		37
Ringraziamenti		37
Bibliografia		39

Introduzione

La velocità di trasmissione del COVID-19 dipende dalla nostra capacità di identificare in modo affidabile i pazienti infetti con un basso tasso di falsi negativi, ovvero test che forniscono risultati negativi (assenza di malattia o condizione) ma in realtà sono positivi (presenza di malattia o condizione).

Inoltre, è necessario un basso tasso di falsi positivi, ovvero test che forniscono risultati positivi (presenza di malattia o condizione) ma in realtà sono negativi (assenza di malattia o condizione), per evitare di aumentare ulteriormente l'onere per il sistema sanitario esponendo inutilmente i pazienti alla quarantena se ciò non è necessario. Insieme a un adeguato controllo delle infezioni, è evidente che il rilevamento tempestivo della malattia consentirebbe l'attuazione di tutte le cure di supporto richieste dai pazienti affetti da COVID-19.

Sia falsi negativi che positivi sono altamente indesiderati in medicina, soprattutto in campo diagnostico quando un dato test si ritiene essere capace di discriminare lo stato di malattia da quello di assenza di malattia.

Analizzeremo tali risultati tramite la *Matrice di Confusione*.

Il COVID-19 presenta diverse caratteristiche uniche. Sebbene la diagnosi sia confermata utilizzando la reazione a catena della polimerasi (PCR), i pazienti infetti con polmonite possono presentare radiografie del torace e immagini di tomografia computerizzata (TC) con uno schema poco interpretabile dall'occhio umano.

L'imaging radiologico non rappresenta un criterio diagnostico per l'infezione da Coronavirus (Sars-Cov2), ma è in grado di evidenziare l'eventuale polmonite che ad essa si può

associare, in tal caso infatti è possibile vedere alla radiografia o alla TC una opacità, definita addensamento. Questo è un segno di possibile polmonite Covid-19 dovuta all'infezione in corso che dovrà essere confermata con il tampone. Per questo motivo una RX o una TC del torace negativi, cioè senza un addensamento a livello polmonare, non possono escludere a priori l'infezione da Sars-Cov2.

Difatti l'analisi che verrà svolta, non mira ad evidenziare la presenza o no di infezione da Sars-Cov2, ma l'obbiettivo è riconoscere un RX che presenta Polmonite o no.

I dati medici sono difficilmente reperibili per questioni di Privacy, difatti il DataSet analizzato è di dimensioni ridotte, per questo il task affrontato non sarà una vera e propria classificazione, ma si analizzerà come cambiando dei parametri nella Rete Neurale l'addestramento perde molto a livello di accuratezza.

Capitolo 1

Machine Learning

Il Machine Learning è la scienza della branca dell'intelligenza artificiale, che permette ai computer di poter apprendere informazioni dai dati in modo non automatico. Vi sono vari campi di competenza in cui tale pratica è utilizzata.

Nell'ambito sanitario l'intelligenza artificiale trova largo impiego nell'imaging medico, in quanto consente una caratterizzazione migliore e un'identificazione più celere delle metastasi che possono migliorare gli esiti delle terapie. In particolare, le tecniche di deep learning vengono maggiormente utilizzate per gli aspetti legati all'imaging medico.

1.1 Machine Learning e Classificazione

Nel machine learning la *classificazione* è un problema che consiste nell'identificare a quale categoria appartiene un elemento in input, sulla base di un modello di classificazione ottenuto in apprendimento automatico.

Ogni oggetto esaminato viene detto istanza, l'algoritmo di ML si occuperà di analizzare ogni istanza ed etichettarla con una classe.

Quando le classi sono soltanto due (es. spam/no-spam se si tratta di email) si parla di classificazione binomiale o classificazione binaria. Se le classi sono più di due si parla di classificazione multiclasse

1.2 Matrice di Confusione

Una matrice di confusione è una tabella che viene utilizzata per descrivere le prestazioni di un modello di classificazione su un insieme di dati di prova per i quali sono noti i valori reali.

n=165	Predicted:		
	NO	YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
		55	110

Figura 1.1: Esempio Matrice di Confusione

In questo esempio su 165 istanze analizzate dove le possibili classi sono due SI/NO (Dove Si e No ad esempio prendendo il nostro caso può essere Covid e Non Covid), il modello aveva previsto 55 NO e 110 SI.

Di fatto però tra le 110 istanze classificate come SI, 10 hanno come valore reale NO. E analogamente per i 55 NO, 5 istanze hanno come valore reale SI.

Di fatto vi sono 4 tipi di dati evidenziati dalla Matrice di Confusione.

- **True Positive**, ovvero veri positivi, dove il modello ha predetto SI e il valore Reale corrisponde a SI, nel nostro esempio sono 100.
- **True Negative**, ovvero veri negativi, dove il modello ha predetto NO e il valore Reale corrisponde a NO, nel nostro esempio sono 50.

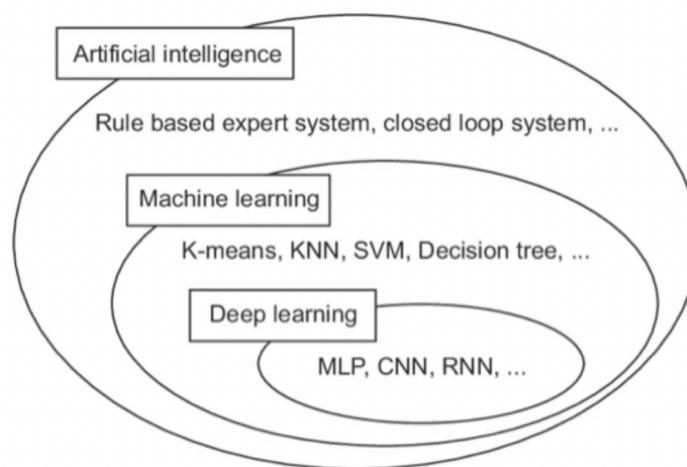
- **False Positive**, ovvero veri negativi, dove il modello ha predetto Si e il valore Reale corrisponde a No, nel nostro esempio sono 10.
- **False Negative**, ovvero veri negativi, dove il modello ha predetto No e il valore Reale corrisponde a Si, nel nostro esempio sono 5.

1.3 Deep Learning

Il Deep Learning è quel campo di ricerca dell'intelligenza artificiale che si basa su diversi livelli di rappresentazione, corrispondenti a gerarchie di caratteristiche di fattori o concetti, dove i concetti di alto livello sono definiti sulla base di quelli di basso.

In altre parole, per apprendimento profondo si intende un insieme di tecniche basate su reti neurali artificiali organizzate in diversi strati, dove ogni strato calcola i valori per quello successivo affinché l'informazione venga elaborata in maniera sempre più completa.

Tra i modelli più importanti del DL vi sono le Reti Neurali, in particolare verranno approfondite le Reti Neurali Convoluzionali(CNN).



1.4 Reti Neurali Artificiali

Un rete neurale artificiale è un modello di Deep Learning basato sulle reti biologiche dei nostri cervelli. L'obiettivo è cercare di raggiungere prestazioni cognitive che in qualche modo si avvicinino a quelle della mente umana.

Sono il vero centro del deep learning, grazie alla loro versatilità, potenza e scalabilità. Ideale per analizzare Dataset contenenti grandi quantità di dati.

Vi sono tre grandi paradigmi di apprendimento, ciascuno corrispondente ad un particolare compito astratto di apprendimento :

- **apprendimento supervisionato** quando si dispone di un insieme di dati per l'addestramento, in tal modo la rete può imparare le relazioni che lega valori di input con i valori di output. Successivamente, la rete è addestrata mediante un opportuno algoritmo, allo scopo di modificare i pesi e altri parametri della rete stessa in modo tale da minimizzare l'errore di previsione relativo all'insieme di addestramento. L'obiettivo finale dell'apprendimento supervisionato è la previsione del valore dell'uscita per ogni valore valido dell'ingresso, basandosi soltanto su un numero limitato di esempi di corrispondenza, per questo viene usato per problemi di regressione o classificazione.
- **apprendimento non supervisionato** basato su algoritmi di addestramento che modificano i pesi della rete facendo esclusivamente riferimento ad un insieme di dati che include le sole variabili di ingresso. Tali algoritmi tentano di raggruppare i dati di ingresso e di individuare pertanto degli opportuni cluster rappresentativi dei dati stessi, facendo uso tipicamente di metodi topologici o probabilistici. L'apprendimento non supervisionato è anche impiegato per sviluppare tecniche di compressione dei dati.
- **apprendimento per rinforzo** nel quale un opportuno algoritmo si prefigge lo scopo di individuare un certo modus operandi, a partire da un processo di osservazione dell'ambiente esterno; ogni azione ha un impatto sull'ambiente, e l'ambiente produce una retroazione che guida l'algoritmo stesso nel processo di apprendimento.

1.5 Overfitting e Underfitting

Spesso le reti neurali hanno migliaia di parametri, addirittura milioni. Questo aumenta moltissimo la flessibilità della rete ma pone anche il problema dell' *Overfitting*.

Nell'overfitting ci sono troppi parametri nel modello e un'elevata variabilità della classificazione. Il modello è troppo complesso e sensibile ai dati di training (high variance).

Per porre rimedio a questo problema vi sono una serie di tecniche di regolarizzazione, come ad esempio: *Batch Normalization*, *l1,l2 regularization*, *max-norm regularization* e *Dropout*.

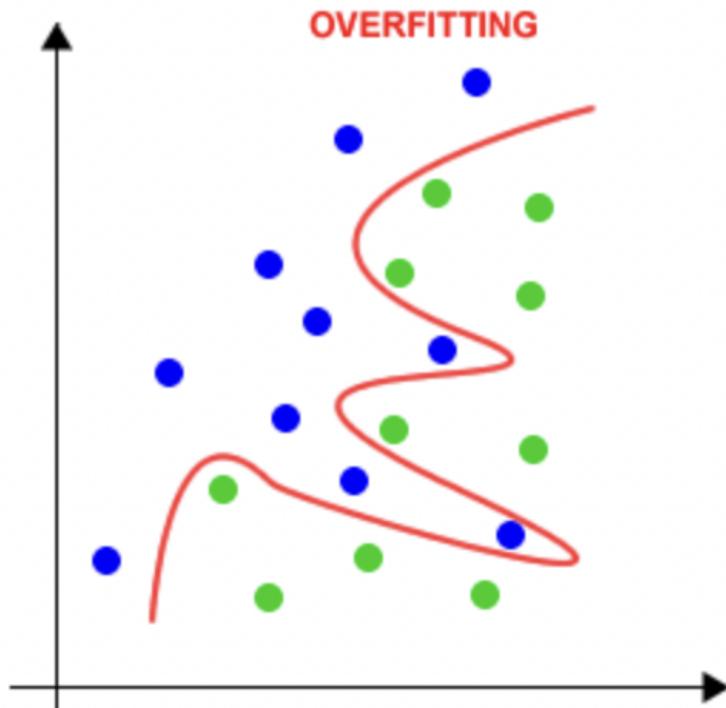


Figura 1.2: Overfitting

Il modello può soffrire anche del problema opposto ovvero l'*underfitting*. Si verifica quando il modello ha prestazioni scarse sui dati di addestramento e quindi un'elevata

discrepanza nella classificazione. Questo avviene perché il modello non è in grado di acquisire il rapporto tra gli esempi di input e i valori target.

Per porre rimedio generalmente bisogna aumentare il numero dei parametri dentro l'analisi.

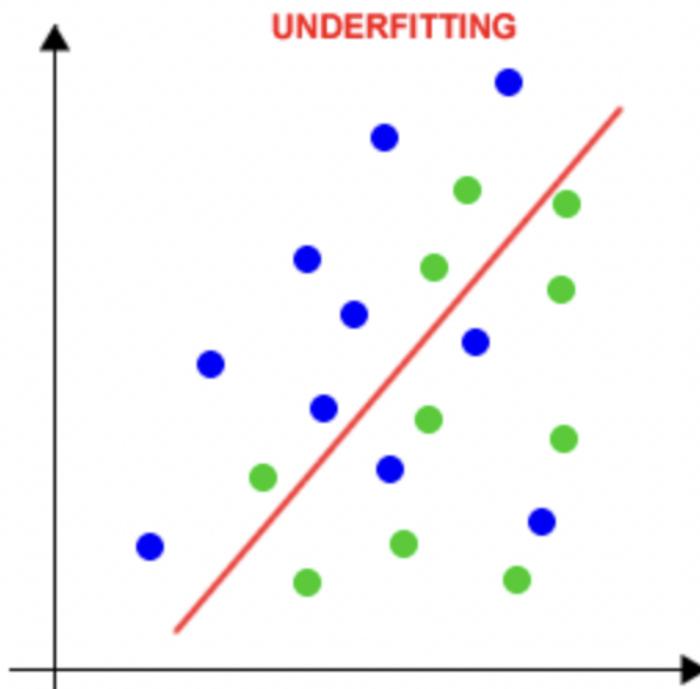


Figura 1.3: Underfitting

Per capire se vi è un problema di Overfitting o Underfitting suddivido il Dataset in training set e test set.

Poi estrapolo le predizioni e verifico l'accuratezza dei risultati sia sui dati di test (test set) che sui dati di addestramento (training set).



Figura 1.4:

- Se l'errore sui dati di training è elevato, c'è sicuramente un problema di underfitting. Il modello ha generalizzato troppo.
- Se l'errore sui dati di training è accettabile ma l'errore sui dati di test è elevato, c'è un problema di overfitting. Il modello non ha generalizzato abbastanza.

Capitolo 2

Reti Neurali Convoluzionali

Una rete neurale convoluzionale (CNN) è un’architettura di rete per il deep learning che apprende direttamente dai dati, eliminando la necessità di estrarre manualmente le feature, al giorno d’oggi è una delle architetture più utilizzate nell’ambito dell’apprendimento automatico.

Le CNN sono particolarmente utili per individuare pattern nelle immagini per il riconoscimento di oggetti, volti e scene, sono usate sin dagli anni 80 per il riconoscimento di immagini.

2.1 Layers

Le CNN sono composte da un livello (Layer) di Input iniziale e uno di output finale, ma ciò che è interessante sono i livelli intermedi nascosti dove verrà svolto il ”lavoro”.

Dopo aver appreso le feature in numerosi layer, l’architettura di una CNN passa alla classificazione.

Il penultimo layer è un layer completamente connesso che emette un vettore di dimensioni K dove K è il numero di classi che la rete sarà in grado di prevedere. Questo vettore contiene le probabilità per ciascuna classe di qualsiasi immagine classificata.

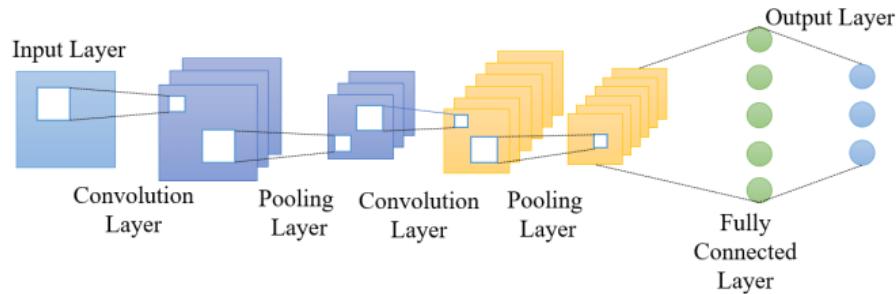


Figura 2.1: Esempio di Architettura CNN

Questi layer eseguono operazioni che alterano i dati al fine di apprendere le feature specifiche dei dati stessi. I layer più diffusi sono:

- La **Convoluzione** sottopone le immagini di input a una serie di filtri convoluzionali, ciascuno dei quali attiva determinate feature dalle immagini.
- L'unità lineare rettificata (**ReLU**) consente di eseguire un addestramento più rapido ed efficace mappando i valori negativi a zero e mantenendo quelli positivi. Questa operazione è talvolta definita attivazione, dal momento che solo le feature attivate vengono trasmesse al layer successivo.
- Il **pooling** semplifica l'output mediante l'esecuzione di un downsampling non lineare, riducendo in tal modo il numero di parametri che la rete deve apprendere.
- Il **Livello FC** (o Fully connected, completamente connesso): connette tutti i neuroni del livello precedente al fine di stabilire le varie classi identificative visualizzate nei precedenti livelli secondo una determinata probabilità. Ogni classe rappresenta una possibile risposta finale che il computer ti darà.

2.2 Livello Convoluzionale

E' il livello più importante della CNN. Nel primo livello *convoluzionale* i neuroni sono collegati solo ai pixel nella loro campi recettivi (vedi Figura 2).

A sua volta, ogni neurone nel secondo strato convoluzionale è connesso solo ai neuroni situati all'interno di una piccola sezione del primo livello.

Questa architettura consente alla rete di concentrarsi su piccole funzionalità di basso livello nel primo layer, quindi assemblarle in funzionalità di un layer superiore. Questa struttura gerarchica è comune nelle immagini del mondo reale, motivo per cui le CNN funzionano così bene per il riconoscimento delle immagini.

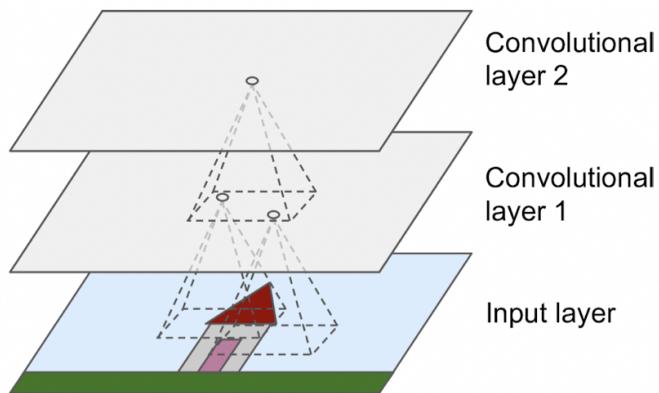


Figura 2.2: CNN Layers

2.2.1 Filtri

Per filtro generalmente, si intende una piccola matrice di poche righe e colonne che rappresenta una caratteristica (feature) che il livello convoluzionale vuole identificare, ad esempio le curve o una linea retta.

Per un livello convoluzionale il filtro identifierà le curve, per un altro linee orizzontali, per un altro ancora circonferenze, e così via negli ultimi livelli, fino a formare figure complesse che rappresenteranno oggetti più complicati. Identificata la caratteristica che

il filtro identifierà nel livello convoluzionale, si decide la dimensione del filtro e il numero di filtri da utilizzare nel livello.

Una volta stabilito cosa il filtro deve identificare si decide la dimensione del filtro e il numero di filtri da utilizzare nel livello.

Ci sono 4 parametri principali che influenzano il comportamento di un livello convoluzionale (passo, dimensione del filtro, numero e riempimento zero).

Per scegliere gli iperparametri di una rete neurale convoluzionale, non esiste uno standard stabilito che viene utilizzato da tutti i ricercatori, in quanto la rete dipende in gran parte dal tipo di dati a disposizione.

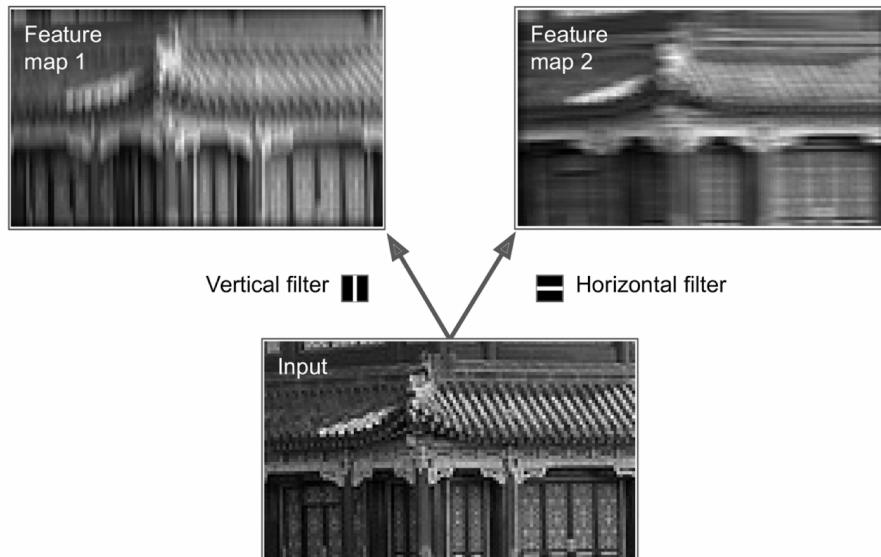


Figura 2.3: Applicazione di due differenti filtri su un unico input

2.3 Livello ReLU

Il Livello ReLU si pone l'obiettivo di annullare valori non utili ottenuti nei livelli precedenti ed è posto dopo i livelli convoluzionali, in quanto l'output del primo livello convoluzionale diventa l'input del secondo livello. Di conseguenza, l'output del livello convoluzionale diventa l'input del livello ReLU.

Il livello ReLU applica la funzione $f(x) = \max(0, x)$ a tutti i valori nel volume di input.

In parole semplici, questo livello annulla tutti i valori negativi, aumentando le proprietà non lineari del modello e della rete globale senza influenzare i campi ricettivi del livello convoluzionale.

2.4 Livello Pool

Il Pooling non ha parametri da settare, ma questo può essere di diversi tipi: Max, Sum, Average ecc..

Il Livello Pooling permette di identificare se la caratteristica di studio è presente nel livello precedente e rende più grezza l'immagine, mantenendo la caratteristica utilizzata dal livello convoluzionale. In altre parole il livello di pooling esegue un'aggregazione delle informazioni , generando feature map di dimensione inferiore.

2.5 Livello Fully Connected

Connette tutti i neuroni del livello precedente al fine di stabilire le varie classi identificative visualizzate nei precedenti livelli secondo una determinata probabilità.

Capitolo 3

Modello

Il dataset di Immagini preso in considerazione è composto da 188 immagini formattate manualmente secondo il formato 512x512.

```
1     for im in os.listdir(path):
2         img = image.load_img(os.path.join(path,im),
3                               target_size=((512,512)))
4         img_array = image.img_to_array(img)
5         image_data.append(img_array[:,:,:1])
6         labels.append(label_dict[ix])
```

3.1 Struttura della Rete Neurale

La Rete Neurale che andrà ad analizzare il nostro DataSet è costruita tramite la Function API di Keras.

```
1 x = Input(shape = (512,512,1))
```

Listing 3.1: Livello di Input

Come primo livello definiamo l'input, visto che le immagini prima di essere "addestrate" vengono ridimensionate secondo il formato 512x512, il parametro shape del livello di Input sarà tale.

Successivamente vi sono 4 livelli convoluzionali, per rendere più dinamico il codice, è stata definita una variabile `n_filters`, nei 4 livelli verrà richiamata questa variabile con un opportuno moltiplicatore.

Analogamente anche la variabile `n_kernel_regularizer` viene definita a inizio del codice, ma rimarrà invariata per tutti e 4 i livelli convoluzionali.

Al Parametro padding viene passata la stringa "same", che sta ad indicare la tecnica del **Zero Padding**.

La funzione ReLU è la più diffusa all'interno dei livelli nascosti,

```

1   h = Conv2D(filters=(n_filters * 2), kernel_size=(3,3),
2   activation='relu',kernel_regularizer=12(n_kernel_regularizer),
3   padding = "same")(h)
4   h = MaxPool2D(pool_size=(2, 2))(h)
```

Listing 3.2: Livello Convoluzionale

Successivamente è presente un livello di Dropout, ovvero una regolarizzazione dei parametri, verrà approfondito nel capitolo 4.2.4.

```
1   h = Dropout(0.50)(h)
```

Listing 3.3: Livello Dropout

```
1   y = Activation('sigmoid')(h)
```

Listing 3.4: Definizione finale modello

3.1.1 Funzioni di Attivazione

Una funzione di attivazione in una rete neurale definisce come la somma dell'input viene trasformata in un output da uno o più nodi in uno strato della rete.

Una rete normalmente ha tre tipi di livelli: livello di input che riceve dati non elaborati dal dominio, livelli nascosti che ricevono input da un altro livello e passano l'output a un altro livello e livello di output che effettua la previsione.

Tutti i livelli nascosti utilizzano in genere la stessa funzione di attivazione. Il livello di output in genere utilizzerà una funzione di attivazione diversa dai livelli nascosti e dipende dal tipo di previsione richiesta dal modello.

Livelli Nascosti

Vi sono 3 possibili funzioni di attivazione per i livelli nascosti

- Rectified Linear Activation (ReLU)
- Logistic (Sigmoid)
- Hyperbolic Tangent (Tanh)

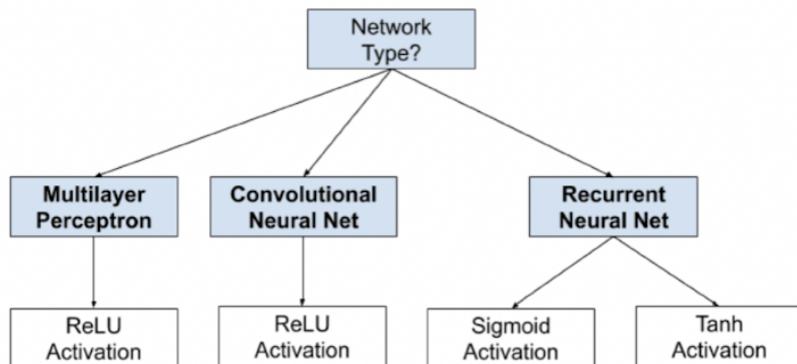


Figura 3.1: Activation Function for Hidden Layers

All'interno dei livelli nascosti della nostra rete neurale (*Hidden Layers*), useremo la funzione di attivazione di tipo "ReLU" (*rectified linear activation function*).

La funzione **ReLU** è la più diffusa all'interno dei livelli nascosti, in quanto di facile implementazione è performante.

La funzione ReLU è calcolata in questo modo:

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

Questo significa che se l'input x è negativo, tornerà un valore di tipo 0.0, altrimenti tornerà il valore stesso.

Livello Output

La scelta della funzione di attivazione per il Livello di Output è presa in base al tipo di problema di predizione che si affronta, in particolare in base al tipo di variabili che vengono previste.

Nel caso della rete costruita il miglior tipo di AF è "*sigmoid*", in quanto la classificazione predetta è di tipo Binario.

La funzione prende qualsiasi valore reale come input e restituisce valori nell'intervallo da 0 a 1. Più grande è l'input (più positivo), più il valore dell'output sarà vicino a 1.0, mentre più piccolo è l'input (più negativo), più vicino sarà l'output sarà a 0.0.

La funzione Sigmoid è calcolata in questo modo:

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

3.2 Parametri

Per vedere come la nostra analisi sul Dataset si comportasse a livello di accuratezza, la rete è stata addestrata più volte cambiando i parametri di regolarizzazione e il numero di filtri applicato.

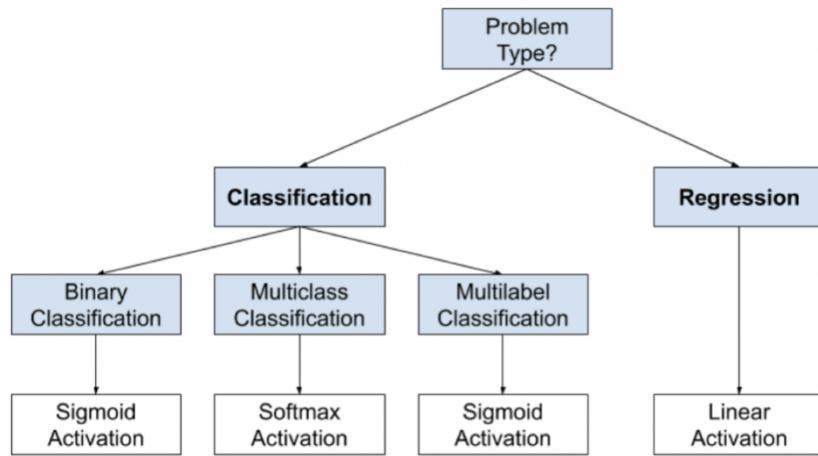


Figura 3.2: Activation Function for Output Layers

3.2.1 Filtri

Aumentando il numero di filtri ad ogni addestramento, aumenteranno anche il numero dei parametri analizzati. Nella nostra rete neurale vi sono 4 livelli convoluzionali, ad ogni livello il numero dei filtri verrà incrementato secondo il seguente schema : *1 *2 *3 *4, in questo modo partendo da un numero iniziale di 8 filtri al quarto livello verranno applicati 64 filtri.

3.2.2 Padding

Per non perdere informazioni da un livello a l’altro durante l’applicazione dei filtri, si può usare la tecnica del ”Zero Padding”, ovvero l’applicazione di un ”bordo” esterno all’immagine composto da 0. In caso applicassimo ZeroPadding con spessore 2 ad un’immagine 32x32x3 questa assumerebbe un valore di input per il livello di 36x36x3.

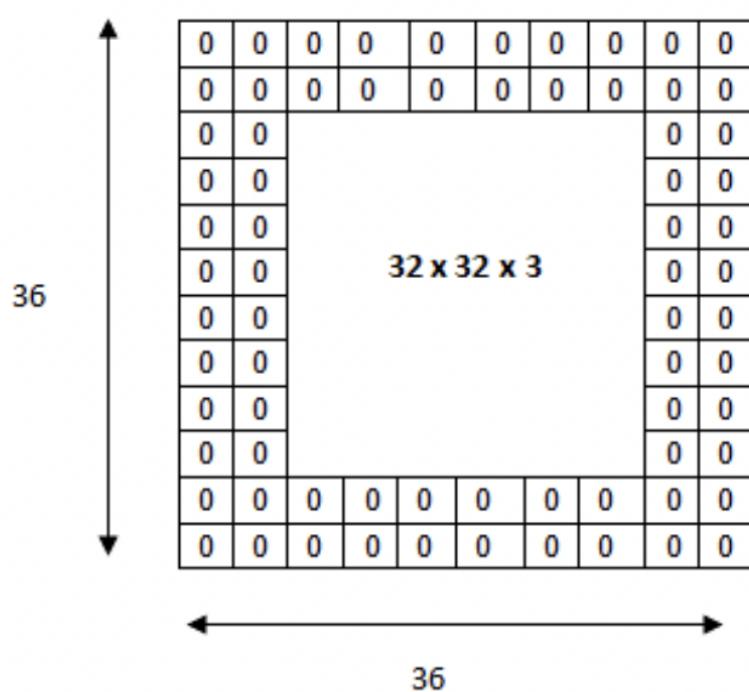


Figura 3.3: Applicazione Zero Padding

3.2.3 Kernel Regularizer

3.2.4 Dropout

Il *Dropout* è una delle tecniche di regolarizzazione più utilizzate.

Ad ogni step della fase di Training, ogni neurone della rete ha una probabilità $P(\text{Dropout Rate tra il } 10\% \text{ e il } 50\%)$ di essere temporaneamente "Scartato", cioè sarà ignorato durante questo step per poi magari essere riattivato a quello successivo.

"Ignorando" alcuno neuroni, lasciamo che gli altri rimasti apprendano più informazioni rendendoli meno sensibili e quindi riducendo il fenomeno dell'Overfitting.

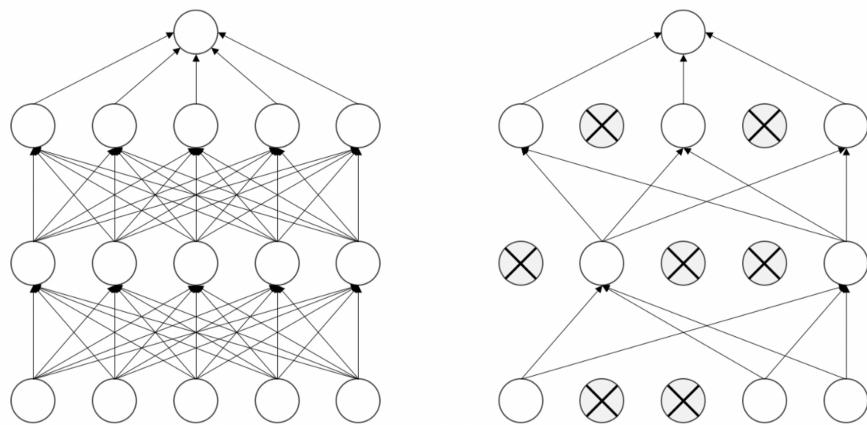


Figura 3.4: Rete neurale pre e post Dropout

Nell'analisi che andremo ad affrontare il Dropout varierà tra 0.25, 0.5 e 0.75.

Layer (type)	Output Shape	Param #
<hr/>		
input_34 (InputLayer)	[(None, 512, 512, 1)]	0
conv2d_132 (Conv2D)	(None, 512, 512, 40)	400
max_pooling2d_132 (MaxPooling2D)	(None, 256, 256, 40)	0
conv2d_133 (Conv2D)	(None, 256, 256, 80)	28880
max_pooling2d_133 (MaxPooling2D)	(None, 128, 128, 80)	0
conv2d_134 (Conv2D)	(None, 128, 128, 160)	115360
max_pooling2d_134 (MaxPooling2D)	(None, 64, 64, 160)	0
conv2d_135 (Conv2D)	(None, 64, 64, 320)	461120
max_pooling2d_135 (MaxPooling2D)	(None, 32, 32, 320)	0
flatten_33 (Flatten)	(None, 327680)	0
dense_66 (Dense)	(None, 224)	73400544
activation_66 (Activation)	(None, 224)	0
dropout_33 (Dropout)	(None, 224)	0
dense_67 (Dense)	(None, 2)	450
activation_67 (Activation)	(None, 2)	0
<hr/>		
Total params: 74,006,754		
Trainable params: 74,006,754		
Non-trainable params: 0		

Figura 3.5: Esempio di Training della Rete

Capitolo 4

Risultati

4.1 Introduzione ai dati

I dati presi in considerazione nell’analisi sono dati di tipo medico, in quanto trattiamo di radiografie del torace . A causa di questo il DataSet è molto ridotto(188 immagini), in quanto acquisire dati medici non è mai facile per questioni legate alla privacy. Questo comporta che il task di classificazione risulti molto semplice per una rete neurale, di fatto il focus dell’analisi è sull’andamento dell’accuratezza dell’addestramento modificando i principali parametri della rete neurale.

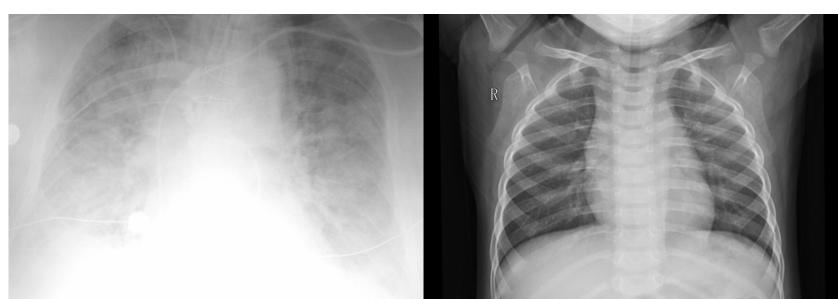
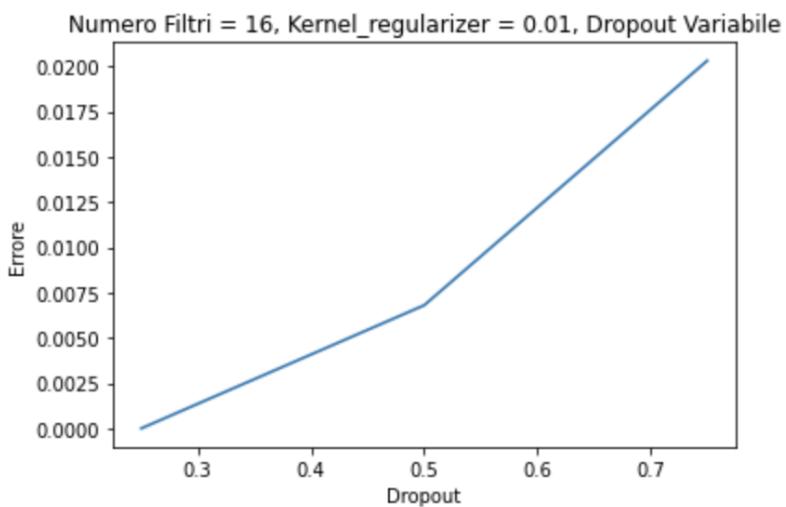
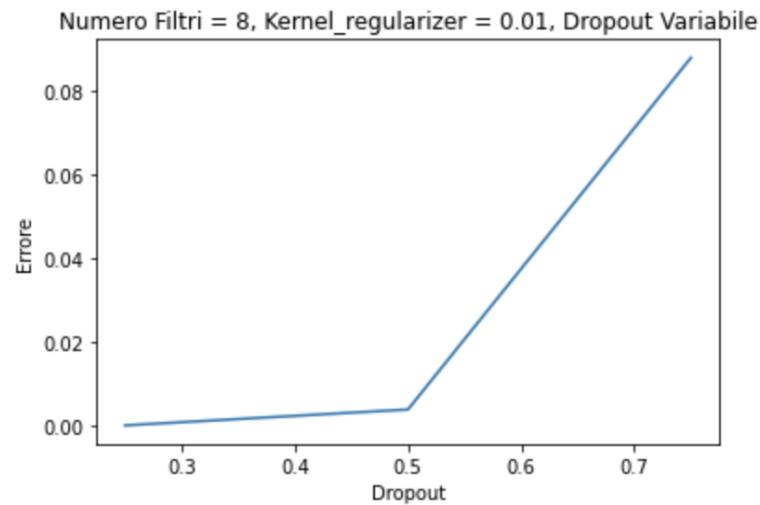


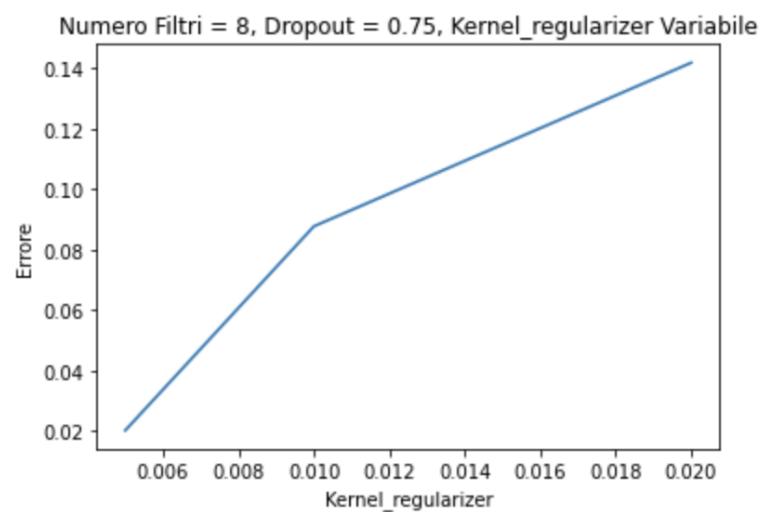
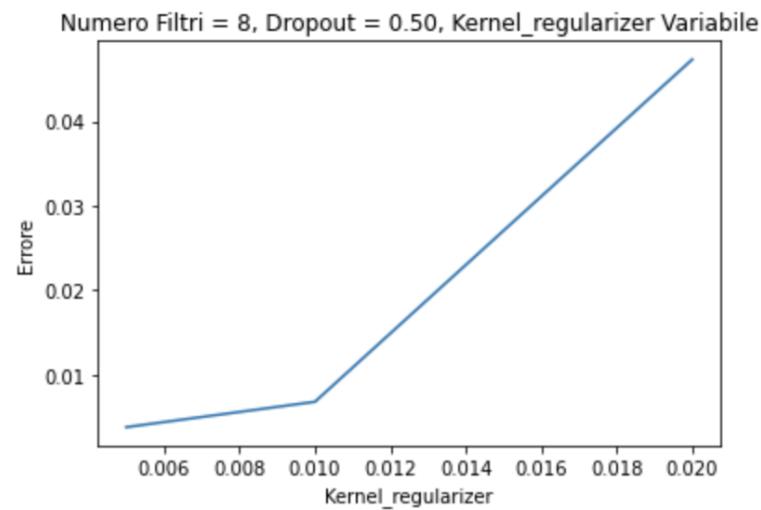
Figura 4.1: A sinistra una radiografia che presenta una polmonite ricollegabile al Covid19, a destra una radiografia di un paziente sano

4.2 Andamento Dropout



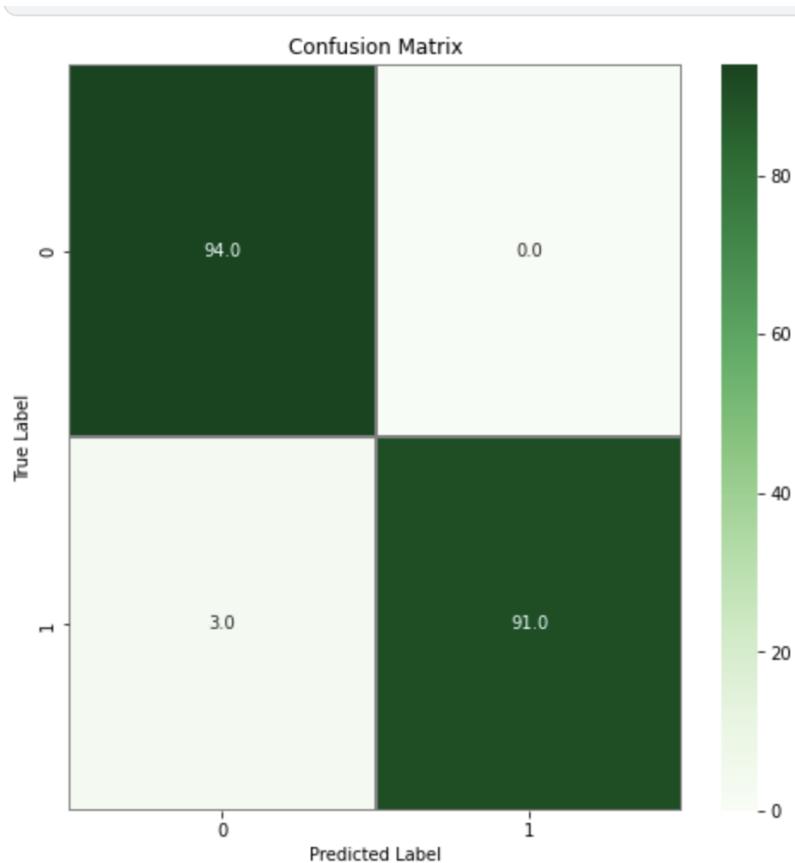
Tenendo fissi il numero dei Filtri e il Kernel-Regularizer, si può notare come all'aumentare del valore di DropOut l'accuratezza diminuisca e di conseguenza aumenta il "Misclassification Rate". Partendo con un Accuracy di 1 con Dropout uguale a 0,25 in entrambi i casi.

4.3 Andamento Kernel-Regularizer



Analogamente al caso del Dropout, anche aumentando il valore del Kernel-Regularizer l'accuratezza nel Train diminuisce, in particolare nel secondo caso tenendo il Dropout alto(0,75), il "Misclassification Rate" arriva a 0,14 con Kernel-Regularizer a 0.02, il peggior caso registrato durante i vari addestramenti.

4.4 Matrice di Confusione



Essendo le immagini a disposizione molto limitate, il test è stato eseguito su tutto il DataSet senza suddividere in Train Set e Test Set, il risultato come si può evidenziare è molto buono con solo 3 Falsi Negativi, e 185 istanze classificate in modo corretto.

Otteniamo quindi una **Precisione** del 0,984% ($185 / 188$ (Istanze corrette)/Totale delle istanze))

4.5 Osservazioni

Com'era prevedibile, la Matrice di Confusione evidenzia come una volta addestrata la Rete, la parte di Test risulti quasi perfetta, con solo 3 **Falsi negativi** su 188 istanze.

E' interessante osservare come al semplice variare dei parametri della Rete, il modello perde di Accuratezza e di conseguenza aumenta il "Misclassification Rate", di fatto si evidenzia l'importanza di impostare i parametri in modo adeguato per sfruttare al massimo la potenza delle CNN.

Conclusioni

In conclusione il problema analizzato risulta poco influente su un DataSet di così piccole dimensioni, rapportato però a un DataSet con migliaia o addirittura milioni di istanze, è evidente come questo possa impattare in modo molto più netto.

Nel caso analizzato, i test hanno sempre portato a ottimi risultati con analoghe Matrici di Confusione "perfette", ovvero senza Falsi Positivi/Falsi Negativi o in numero molto ridotto. Come anticipato questo è dovuto alla difficoltà di reperire dati Medici, in modo particolare radiografie del Torace.

Impiegando diversi algoritmi di deep learning, recenti studi hanno raggiunto risultati notevoli, dimostrando così l'efficacia di queste tecniche nei sistemi automatici di diagnosi. In futuro si prospettano ulteriori miglioramenti di performance e un sempre più diffuso ed efficace utilizzo. Grazie all'intelligenza artificiale è sempre più possibile automatizzare il rilevamento, il monitoraggio, la previsione della terapia e la risposta alla terapia stessa.

Ringraziamenti

In primis vorrei ringraziare la mia relatrice la Prof.ssa Piccolomini, per avermi accompagnato in questo percorso di Tesi con disponibilità e pazienza.

Inoltre vorrei ringraziare anche il dott. Evangelista, che mi è stato molto d'aiuto durante questo percorso con le sue competenze e la sua disponibilità.

Vorrei ringraziare la mia famiglia, che mi ha sempre sostenuto durante questi 3 anni e ha sempre creduto in me, in particolare mia mamma che è sempre stata la mia fan numero 1.

Vorrei ringraziare tutti i miei amici di Savona, che nonostante la distanza mi hanno sempre dato il loro supporto durante questo percorso.

Vorrei ringraziare tutte le persone che ho conosciuto in questa esperienza a Bologna, che mi hanno arricchito a livello umano e mi hanno fatto vivere questa prima esperienza fuori casa nel modo migliore possibile. In particolare vorrei ringraziare i coinquilini avuti durante questi 3 anni, che non mi hanno mai fatto sentire un "Fuorisede" ma sempre e solamente a casa.

Bibliografia

- [1] Aurèlien Geron. *Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow*
- [2] Fabian Vincenzi. *Reti Neurali Convoluzionali per il miglioramento di immagini tomografiche a angoli limitati*
- [3] Michele Bandirali, Luca Maria Sconfienza, Roberta Serra, Roberto Bremilla, Domenico Albano, Fabrizio Ernesto Pregliasco, and Carmelo Messina. *Chest Radiograph Findings in Asymptomatic and Minimally Symptomatic Quarantined Patients in Codogno, Italy during COVID-19 Pandemic*
- [4] Samir S. Yadav1 Shivajirao M. Jadhav, *Deep convolutional neural network based medical image classification for disease diagnosis*