CNN for Chest X-ray Image Classification

G. Tittarelli

18 Settembre 2023

Abstract

L'applicazione dei metodi delle reti neurali, in particolare Convolutionary Neural Network (CNN), riveste un ruolo fondamentale nel campo medico migliorando l'accuratezza e l'efficienza della diagnosi. Lo scopo di questo progetto è di sviluppare un modello di CNN per la classificazione di immagini radiografiche del torace in due categorie: "normale" e "polmonite". Per l'addestramento e la convalidazione del modello è stato utilizzato un dataset che contiene una raccolta completa di casi normali e di polmonite. L'architettura della rete neurale convoluzionale è di tipo sequenziale e utilizza diversi strati convoluzionali per l'estrazione delle caratteristiche e strati di pooling per la riduzione delle dimensioni spaziali. La rete viene collegata ad uno strato Flatten connesso a sua volta ad una rete fully connected con output finale composto da due classi.

Per studiare la performance del modello viene studiata la metrica accuracy, mentre l'analisi qualitativa viene fatta attraverso la confusion matrix e la curva ROC. Nonostante il modello non riesca a generalizzare perfettamente su dati nuovi, questo risulta essere efficiente nella classificazione tra casi "normali" e "polmonite".

Obiettivi

Questo report presenta un progetto incentrato sulla creazione di un modello di rete neurale convoluzionale (CNN) per la classificazione delle immagini di radiografie del torace come "normali" o "polmonite". Lo scopo di questo progetto è sfruttare le tecniche di deep learning per diagnosticare i casi di polmonite analizzando le immagini cliniche. Avere diagnosi corrette è quindi lo scopo di questo progetto, perciò oltre alla metrica di accuratezza sono state studiate la anche la sensitività e la specificità con l'obiettivo di ottere per entrambe un risultato intorno al 90%.

Metodo

Per implementare il seguente modello di rete neurale convoluzionale è stato utilizzato il dataset "chest_xray" reperibile su Kaggle[1].
Il dataset, diviso in training, validation e test sets, contiene immagini di radiografie del torace divise in due classi: "Normal" e "Pneunomia".

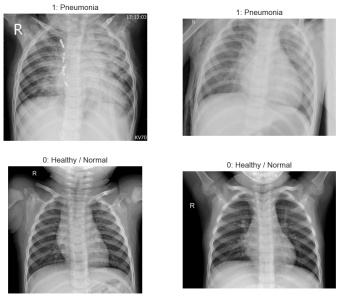

Figure 1: *Alcune immagini dal dataset*

[1] https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

In generale, la distribuzione di immagini "Healty/Normal" e "Pneunomia" non è omogenea, come visibile in *fig.2*. Inoltre anche il numero di elementi in entrambe le classi è piccolo rispetto agli standard per i progetti di reti neurali.

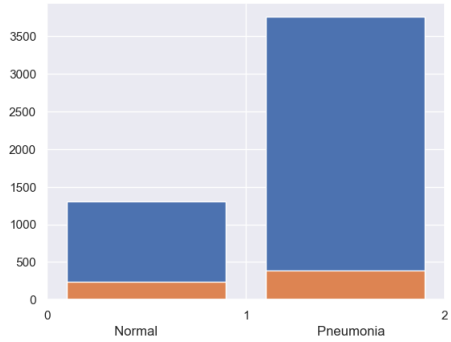

Figure 2: *Il grafico riporta la distribuzione di dati nei set di training (in blu) e di test (in arancione). Il primo ha* 1342 *elementi nella classe "Normal" e* 3.876 *nella classe "Pneumonia", mentre il set di test ha* 235 *immagini nella classe "Normal" e* 391 *nella classe "Pneumonia"*

Si osserva inoltre che il validation set fornito dal dateset contiene solo 16 elementi e per questo si è deciso di non prenderlo in considerazione. Al suo posto è stato utilizzato l' 1% del set di training.
La creazione del modello si divide in due parti: il preprocessamento dei dati e la costruzione dell'architettura della CNN.

Lavorazione dei dati

Abbiamo iniziato definendo i parametri principali del nostro processo di addestramento. Questi includono le dimensioni delle immagini, la dimensione dei batch, il numero di classi target e i percorsi delle directory contenenti i dati. Abbiamo impostato la dimensione delle immagini a 250×250

pixel e il batch size a 8. Il nostro dataset è composto da due classi target: "NORMAL" e "PNEUMONIA". Abbiamo creato tre liste separate per i dati di addestramento, test e validazione. Le immagini "sane" sono state etichettate come classe ”0” mentre le immagini di "polmonite" come classe ”1”. Abbiamo quindi mescolato casualmente le liste per garantire una distribuzione casuale dei dati.
Successivamente, per ridurre i costi computazionali, tutte le immagini sono state ridimensionatte a 68×68 pixel e convertite in scala di grigi mentre i valori dei pixel sono stati normalizzatiti nell'intervallo $[0, 1]$. Inoltre, per rendere il modello più generale è stato aumentato il numero dei dati nel set di training utilizzando tecniche come rotazione, zoom e spostamento. Per questo scopo è stato fatto uso dell'algoritmo *ImageDataGenerator* il quale ad ogni epoca produce dati nuovi[2] moltiplicando il batch size per il numero di step in un epoca.

Architettura CNN

Dopo una prima fase di lavorazione dei dati si è costruita la rete neurale convoluzionale schematizzata in *fig.3*.

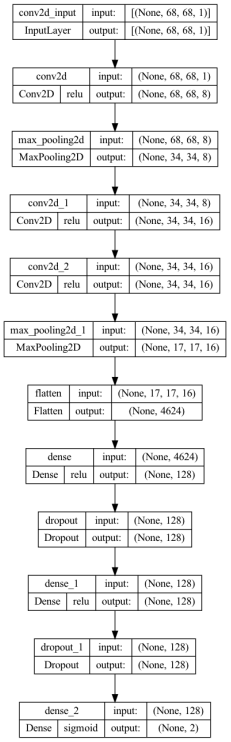

Figure 3: *Architettura CNN*

Il modello CNN creato per questo progetto è composto da diversi livelli convoluzionali e di pooling per estrarre caratteristiche gerarchiche dalle immagini in ingresso. Nello specifico:

- Uno strato Conv2d, con 8 filtri e dimensioni del kernel $(7, 7)$, seguito da uno strato MaxPooling2D, con dimensione 2×2.
- Due strati Conv2d, con 16 filtri e dimensioni del kernel $(5, 5)$, seguito da uno strato MaxPooling2D, con dimensione 2×2.

Ogni operazione di convoluzione viene eseguita con funzione di attivazione *ReLU* e con l'operazione di *padding* impostata al valore "same". Quest'ultimo è necessario per mantenere le dimensioni dell'input e dell'output simili.
Dopo gli strati convoluzionali, l'output viene appiattito e passato alla rete fully connected attraverso uno strato *Dense* con 128 unità, seguito da uno strato *Dropout* con un tasso del 20% per prevenire l'overfitting. Questi passaggi vengono ripetuti una seconda volta, aggiungendo allo strato *Dense* il regolarizzatore L^2 con parametro di kernel[3] impostato a 0.03. Il modello si conclude con lo strato di output finale, il quale ha 2 unità con funzione di attivazione data dalla sigmoide per la classificazione binaria.
Il modello è stato addestrato usando l'algoritmo Adam come ottimizzatore con learning rate di 0.0005 e con weight decay di 10^{-5} ed è stato implementato l'algoritmo di early stopping, con una pazienza di 6 epoche monitorando la funzione costo, al fine di prevenire l'overfitting.
Infine, tenendo in considerazione la dimensione ridotta del dataset, per il training sono stati fissati i seguenti iperparametri:

$$\text{Epochs} = 100 \quad \& \quad \text{Batch size} = 32 \tag{1}$$

Risultati

L'addestramente viene monitorato utilizzando l'*accuracy* e la *loss* come metriche primarie di valutazione, ottendendo i risultati in *fig.4*.

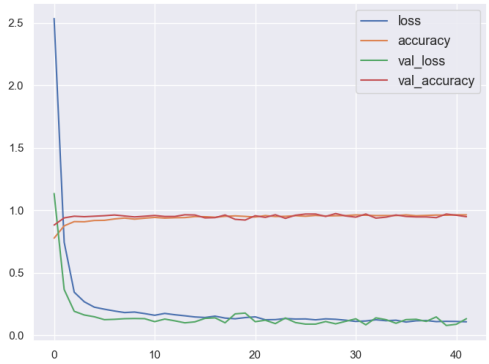

Figure 4: *Il grafico riporta l'andamento di learning curves per il modello CNN. Sia la curva di validation loss (in verde) che di validation accuracy (in rosso) presentano oscillazioni, ma seguono rispettivamente l'andamento della loss (in blu) e dell'accuracy (in arancione)*

.

All'epoca 42 il modello esce dall'addestramento per via dell'intervento dell'algoritmo *Early Stopping*. Osserviamo

[2] In questo caso i dati nuovi sono modifiche dei dati del set di training

[3] In letteratura questo parametro è identificato con λ

inoltre che il metodo di data augmentation utlizzato produce 192.085 utilizzati nella fase di training. Dopo essere stato addestrato, il modello è stato valutato su dati mai visti utilizzando il dataset di test.

$$\text{Test Loss} = \quad 0.286 \tag{2}$$

$$\text{Test Accuracy} = \quad 0.901 \tag{3}$$

Inoltre, è stata studiata la *confusion matrix* in *fig.5* per comprendere le prestazioni del modello in termini di previsioni vere positive TP, vere negative TN, false positive FP e false negative FN.

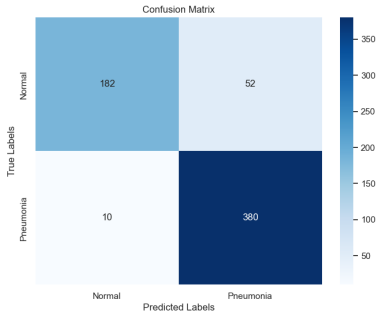

Figure 5: *I valori di veri positivi TP corrispondono ai "Healty/Normal" veri predetti mentre i veri negativi TN corrispondo ai casi di "Pneumonia" correttamente predetti. Di conseguenza i falsi positivi FP corrispondono ai casi di "Pneumonia" diagnosticati in modo errato, mentre i falsi negativi FN sono i casi "Healty/Normal" predetti in modo errato.*

Dalla matrice di confusione ricaviamo la sensitività

$$\text{Sensitivity} = \frac{TP}{TP + FN} = 0.948 \tag{4}$$

e la specificità

$$\text{Specificity} = \frac{TN}{TN + FP} = 0.880 \tag{5}$$

Per visualizzare il compromesso tra il tasso di veri positivi e il tasso di falsi positivi si è studiata la curva *Receiver Operating Characteristic* (ROC) visibile come segue in *fig.6*.

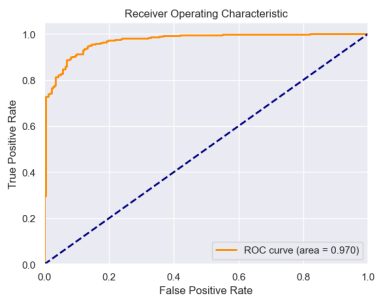

Figure 6: *Curva ROC*

L'area sotto la curva, in letteratura l'area *AUC*, vale:

$$AUC = 0.970 \tag{6}$$

Conclusioni

In generale, il modello riesce a completare il task di classificazione di radiografie del torace. La sfida principale è stata dover lavorare con un dataset contenente pochi elementi e con una distribuzione dei dati nelle classi non omogenea, come visibile in *fig.2*. Per questo motivo è stato fondamentale fare *data augmentation* del set di training al fine di prevenire overfitting, oltre ad utilizzare l'algoritmo di *early stopping* con pazienza di 6 epoche. Come mostrato in *fig.4*, l'early stopping ha avuto un impatto positivo sul modello, impedendo l'eccessiva adattabilità ai dati di addestramento. Inoltre, l'uso del regolarizzatore L^2 ha contribuito a ottenere buoni risultati, aiutando a controllare l'overfitting e a migliorare la generalizzazione del modello. Tuttavia, sempre in *fig.4* si osserva come il set di validation non riesca ad adattarsi perfettamente al modello, continuando ad oscillare sia per quanto riguarda l'accuracy che per la loss. Questa instabilità ha influito sulla generalizzazione del modello a nuovi dati. Nonostante ciò, raggiungere un'accuratezza (3) del 90% sul set di test rappresenta un risultato positivo poiché evidenzia la robustezza del modello nell'affrontare dati inediti. Questo aspetto emerge chiaramente dalla matrice di confusione riportata in *fig.5*, che ci consente di calcolare la sensitività (4) e la specificità (5). La sensitività ci dice che il modello è in grado di rilevare circa il 95% dei casi positivi ("Healty/Normal") in modo accurato, un risultato congruente con l'obiettivo del progetto, dato che in ambito medico si cerca di minimizzare il rischio di diagnosi errate. Tuttavia, il modello mostra una performance inferiore nella classificazione dei casi negativi ("Pneumonia"), evidenziata dalla specificità del 88%.
Nonostante la disparità tra queste due metriche, l'area sotto la curva ROC (6) in *fig.6* conferma l'efficacia del modello come classificatore binario. Con un AUC (6) di 0.970, il modello dimostra un'eccellente capacità di discriminazione tra le due classi.
In conclusione, i risultati ottenuti sono il frutto di iterazioni e adattamenti degli iperparametri del modello basati sull'osservazione diretta delle sue prestazioni. Purtroppo, non è stato trovata la fonte di rumore che ha influenzato le oscillazioni viste durante la fase di addestramento del modello, rovinando i risultati su dati nuovi. Il miglioramento delle performance potrebbe coinvolgere ulteriori esperimenti, come l'aumento della risoluzione delle immagini, a discapito di un tempo di addestramento più lungo. Inoltre, sarebbe interessante studiare il comportamento del modello presentato in un dataset con una distribuzione omogenea dei dati nelle classi.

References

[1] Goodfellow, I., Bengio, Y., & Courville A. (2016). Deep learning. MIT press.

[2] Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow.

[3] Tan, P. N., Steinbach, M., & Kumar, V. (2016). Introduction to data mining. Pearson Education India.

[4] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization.

[5] Yadav, S. S., & Jadhav, S. M. (2019). Deep convolutional neural network based medical image classification for disease diagnosis.

[6] Kumar, N., Gupta, M., Gupta, D., & Tiwari, S. (2023). Novel deep transfer learning model for COVID-19 patient detection using X-ray chest. images.