ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

Dipartimento di Informatica
Scienza e Ingegneria
DISI

**Corso di Laurea in Ingegneria Informatica**

# Title

Candidato:

*Gianmiriano Porrazzo*

Relatore:

*prof. Giuseppe Notarstefano*

Correlatore:

*Name Surname*

Anno Accademico
*2021–2022*

Sessione
*II*

# Indice

**Abstract** **3**

**Introduzione** **4**

**1 Strumenti usati** **5**

**2 Dataset** **6**

**3 Implementazione: CNN** **7**
3.1 Transfer Learning . . . . . 8
3.2 Fine Tuning . . . . . 10

**4 Gestione dati eterogenei con MLP** **13**

**A Appendice** **14**

**Bibliografia** **14**

# Abstract

L'uso di tecniche di machine learnig è molto

# Introduzione

Il machine learning ha vari campi applicativi. Uno di questi è quello di riconoscere dei pattern in modo che tali algoritmi possano apprendere e fare predizioni su un insieme di dati.

Un settore sul quale gli algoritmi di machine learning sono molto applicati è quello della medicina. In questo documnento si tratterà di come usare gli algoritmi di machine learning per fare predizioni su un dataset che contiene dati relativi a pazienti affetti da Covid-19.

Esistono vari modi per analizzare i dati e fare apprendere la rete neurale in modo che le predizioni da essa effettuate abbiano senso. Tali metodologie possono variare per via dei dati che compongono il dataset, ma anche dal tipo di previsioni che la rete deve effettuare.

L'algoritmo usato consiste in una rete neurale artificiale. Tale rete è composta da vari neuroni, i quali hanno il compito di prendere dei dati in input, apprenderne le caratteristiche principali e sulla base di ciò effettuare le previsioni.

I neuroni che si occupano dell'apprendimento sono organizzati in vari layer nascosti. Ogni layer è composto da uno o più neuroni che interagiscono tra di loro.

Come già accennato in base al tipo di dato da studiare si può prediligere una tipologia di rete ad un'altra. Nel caso di questa tesi si è scelto di usare una rete neurale convoluzionale (CNN-*Convolutional Neural Network*) per apprendere dalle immagini e una MLP ( *Multi-Layer Perceptron* ) per gestire i metedati relativi alle immagini.

# 1 Strumenti usati

Esistono vari framework per lo sviluppo di reti neurali, uno dei più usati è TensorFlow.
TensorFlow presenta modelli già creati e allenati, i cui pesi possono essere presi per sfruttare tale modello. Insieme all'uso di altre librerie come Keras l'implementazione di una rete neurale che gestisca i dati a disposizione è molto semplificato.
La rete di base usata per partire con l'analisi delle immagini è la MobileNetV2. Tale rete è una CNN che, prendendo come input delle immagini. A tale rete, tuttavia, sono state apportate delle modifiche per via delle dimensioni dell'input e della tipologia di dato che si vuole ottenere come previsione.
Oltre a questa tipologia di rete si è sfruttata anche una U-Net, al fine di omogeneizzare tutte le immagini che vengono date come input della MobileNetV2. La U-Net usata è EfficientNet, con dei pesi preallenati per riconoscere immagini simili a quelle presenti nel dataset di riferimento.

# 2 Dataset

*inserire link all'hackaton e immagini dataset*

Il dataset usato per effettuare lo studio è stato preso da un hackaton riguardante la creazione di una rete neurale in grado di apprendere da immagini relative a radiografie di pazienti affetti da covid in modo da predirre la gravità della prognosi.

Dunque il dataset usato è formato da immagini raffiguranti radiografie dei polmoni, le quali tuttavia non sono omogenee. Per tale motivo si è dovuto effettuare un passaggio preliminare in modo da rendere le immagini tutte della stessa dimensione, ma non distorcendole.

Oltre alle immagini sono presenti anche un insieme di metadati riguardanti lo stato in cui il paziente è entrato all'ospedale e l'anamnesi dello stesso. Tali metadati presentano descrivono varie informazioni relative al paziente, per tale motivo sono rappresentate anche in modo diverso: esistono dati di tipo categorico, sia con più categorie che con due sole categoria, dati che possono essere considerati come booleani e dati interi, come ad esempio l'età.

# 3 | Implementazione: CNN

Lo sviluppo della rete neurale convoluzionale, per effettuare previsioni partendo dalle immagini, è basata sulla rete MobileNetV2. L'uso di tale rete come base consente di usare dei pesi preallenati in modo da rendere il processo di training sul dataset più efficiente.

Tuttavia tale processo necessita di alcune fasi aggiuntive per fare in modo che la rete riesca a prendere i dati della dimensione esatta e restituire delle previsioni nell'insieme desiderato.

Tali operazioni sono:

- Inserire la dimensione delle immagini nel layer di input
- Effettuare fine tuning e transfer learning per poter usare i pesi preallenati
- Inserire dei layer finali per ottenere degli output significativi

La dimensione scelta delle immagini è (224x224), dunque la rete prende in ingresso degli array di dimensione 224x224x3 (dove quest'ultimo indica l'uso dei colori RGB). Per tale motivo il layer di input deve accettare tale dimensione.

I pesi preallenati che si sono scelti sono pesi allenati su ImageNet. ImageNet è un dataset di immagini suddivise in 1000 classi. Il dataset su cui si deve svolgere il training, tuttavia, possiede unicamente una classe, per via del fatto che abbiamo trasformato il valore della prognosi da una string ad un valore binario. Per cui l'immagine può appartenere o meno a tale classe.

Al fine di poter usare tali pesi per effettuare il training, si necessita dunque di ulteriori passaggi:

- Transfer Learning
- Fine tuning

## 3.1 Transfer Learning

Il transfer learning consente di sfruttare la rete già allenata a risolvere problemi diversi, ma comunque correlati con quello di interesse. Nel caso considerato tale tecnica permette di usare una rete allenata per prevedere l'appartenenza di una immagine ad una delle 1000 classi di ImageNet per creare una rete in grado di classificare le immagini del dataset in una unica classe.

Per sfruttare la rete allenata, congeliamo lo stato dei layer di classificazione, al fine di non alterarli, e settiamo la rete come non allenabile. In tal modo si è ottenuto un nuovo modello basato sulla MobileNetV2.

Ora si presenta un problema relativo alla classificazione. Per ovviare al fatto che tale operazione sarà fatta per una sola classe, si necessita dell'inserimento di altri layer alla fine del modello precedentemente ottenuto.

Tali layers sono:

- un GlobalAveragePooling2D()
- due Dense()

Il primo layer serve per via del fatto che allo stato attuale la rete produce un output multidimensionale e, per ottenere previsioni formate da un singolo vettore della dimensione prevista, usiamo tale layer, il quale genera previsioni basate sul blocco multidimensionale e facendone una media.

Il primo Dense layer viene usato per generare l'output prodotto dalle immagini dei polmoni. Tale layer è formato da 100 neuroni ed ha come funzione di attivazione ReLu.

Il secondo Dense layer svolge il lavoro di classificazione vero e proprio. Per via del fatto che si è scelto di usare una label binaria, ovvero classifichiamo su un'unica classe, tale layer è composto da un solo neurone, ed ha come funzione di attivazione Sigmoid.

Una volta creati i layers, per completare la fase di transfer learning, si deve compilare il modello finale, ottenuto aggiungendo i nuovi layers.

Ora la rete è pronta per una prima fase di training. In tale fase sono stati usati i seguenti parametri per il training:

- Loss function: Adam
- Metrica: accuracy

- Epoche 150
- Batch size: 30

Per quanto concerne il valore del learning rate sono si è deciso di effettuare prove sia con $10^{-3}$ che con $10^{-4}$. I risultati ottenuti dal training sono espressi nei seguenti grafici.

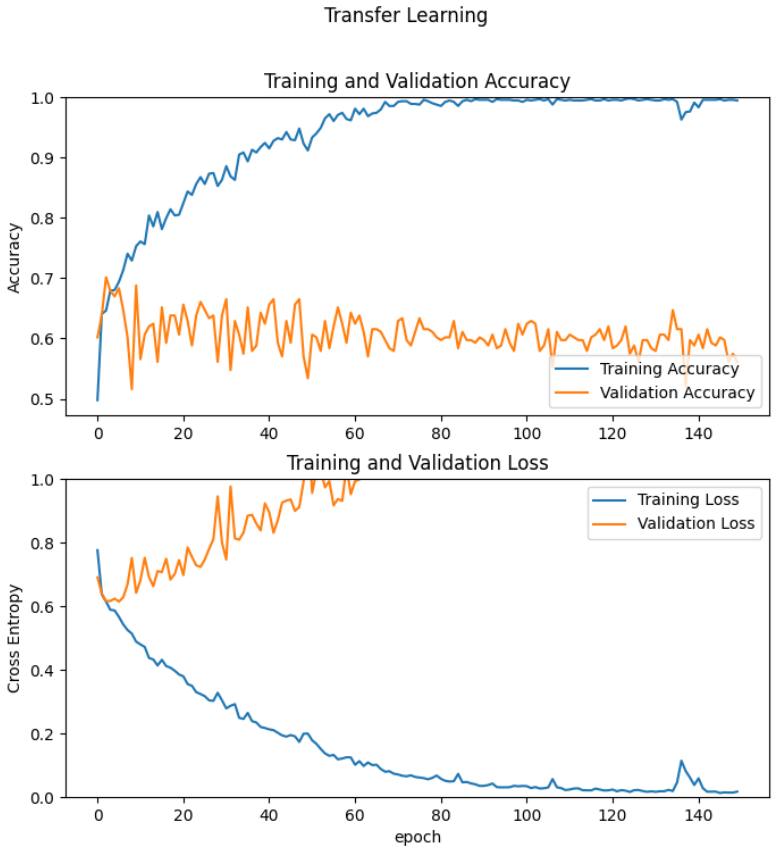


**Figura 3.1:** Test effettuato usando come learning rate $10^{-3}$

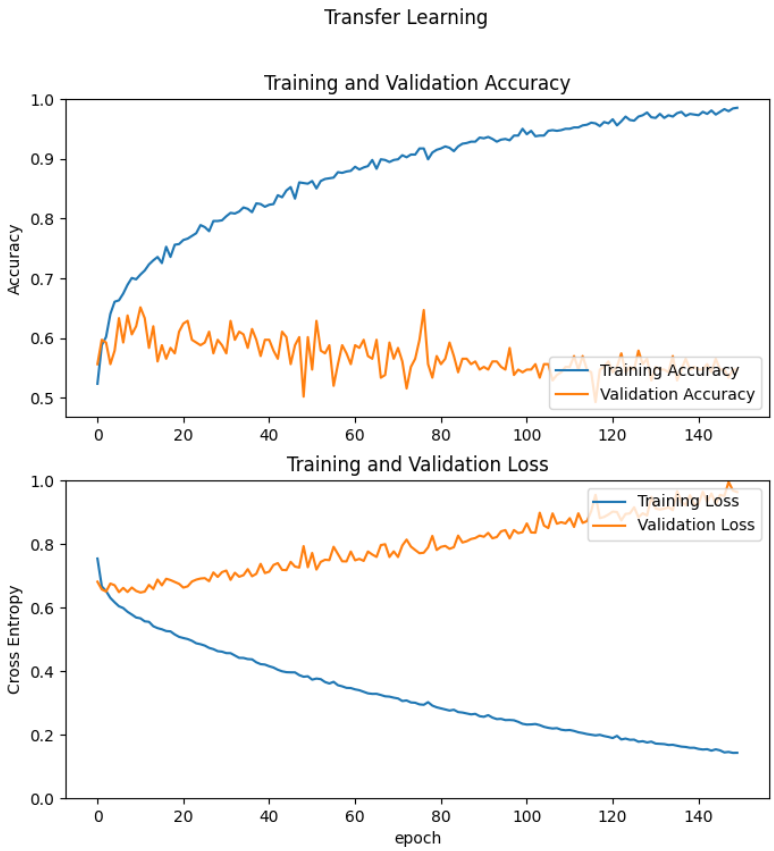


**Figura 3.2:** Test effettuato usando come learning rate $10^{-4}$

## 3.2 Fine Tuning

Per procedere con la fase di fine tuning, si deve rendere nuovamente trainabile il modello creato, in modo tale che i pesi possano essere allenati considerando i nuovi layers. Così facendo si permette ai pesi di regolarsi sul dataset d'interesse, partendo da quello su cui sono stati inizialmente allenati.

Essendo gli ultimi layers di una rete inutili in termini di classificazione, possiamo decidere di congelarli, ovvero non considerarli durante il training, in modo da risparmiare risorse. Effettuato tale passaggio si può procedere con la compilazione del modello ottenuto e procedere con il training.

I parametri relativi al training in questa fase sono:

- Loss function: Adam
- Metrica: accuracy

- Epoche 30
- Batch size: 30

Di seguito sono riportati i risultati ottenuti a seguito del transfer learning e fine tuning, sempre considerando i due valori scelti per il learning rate.

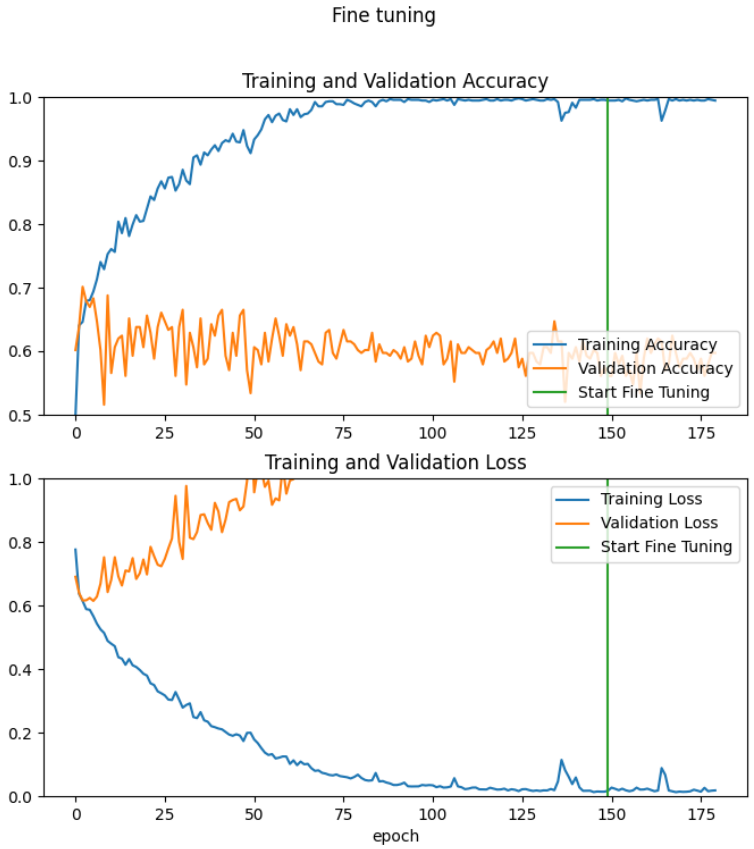


**Figura 3.3:** Test effettuato usando come learning rate $10^{-3}$

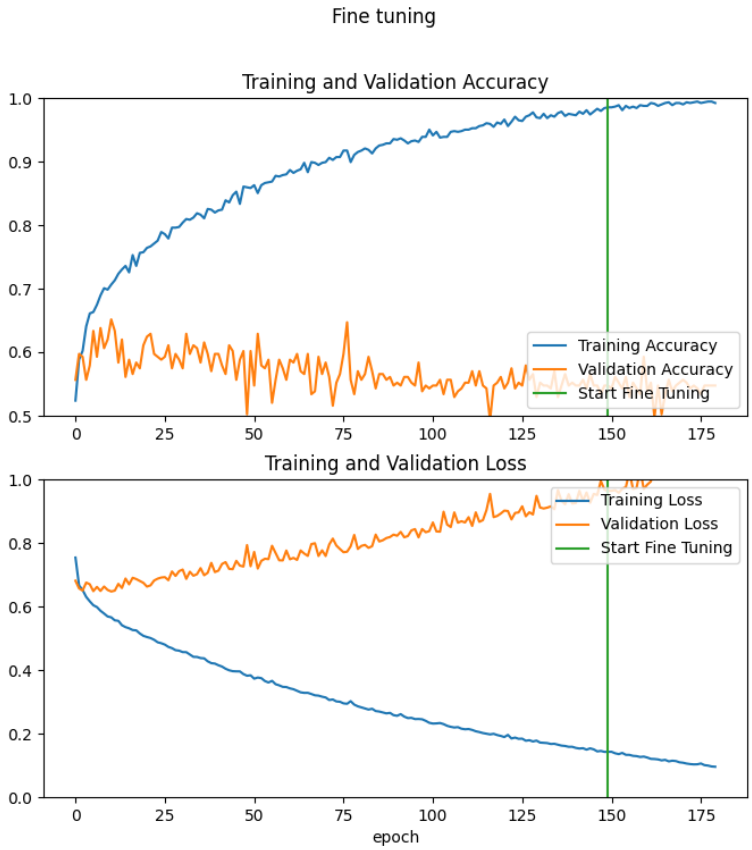


**Figura 3.4:** Test effettuato usando come learning rate $10^{-4}$

# 4 Gestione dati eterogenei con MLP

# A | Appendice