CNN PER RILEVAZIONE DEL GENERE

CONTENUTO DELL'ANALISI

Il seguente studio si propone di eseguire una analisi sugli aspetti implementativi di una rete neurale convoluzionale utilizzando python analizzando gli aspetti critici della rete, il grado di confidenza con il quale la rete ci fornisce le risposte e una valutazione pseudo real-time.

ESEMPI DI UTILIZZO

La rilevazione real-time mediante reti neurali può diventare di fondamentale importanza, basti pensare ai possibili campi di applicazione di sistemi di questo genere, come ad esempio nei negozi di abbigliamento dove vogliamo avere informazioni su quante persone entrano ed escono dal negozio, l'età, il genere, le preferenze e il grado di soddisfazione medio del cliente. Architetture di questo tipo sono in grado di raggiungere lo scopo cosi da dare dati statistici al fine di poter calibrare meglio l'acquisto di prodotti ed evitare incompatibilità tra richieste dei clienti e proposte del commerciante.

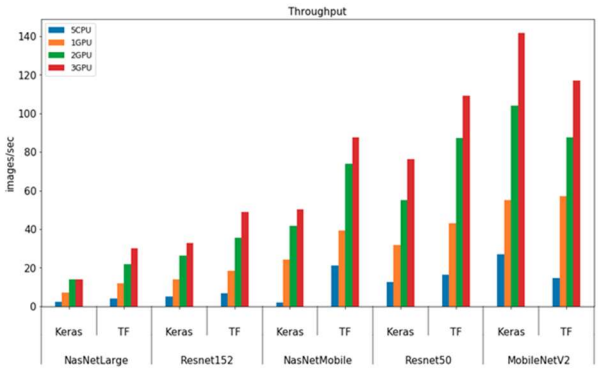
L'altro rilevante aspetto di utilizzo per quanto riguarda l'utilizzo di queste reti è quello riguardante i social media, è infatti possibile fornire dati di tipo statistico analizzando i vari interessi, e le caratteristiche delle persone basandosi sulle foto che pubblicano sui social, tutto in maniera automatica. Basti pensare a quanto massivamente sono utilizzati questi sistemi dalle grandi aziende per fini pubblicitari per dare idea dell'importanza dello sviluppo delle reti neurali.

SOFTWARE & HARDWARE

Per la costruzione della rete neurale sono stati necessari alcuni accorgimenti iniziali, infatti le tempistiche di training sono particolarmente influenzate dall'hardware utilizzato e dal software di basso livello.

Per il training della rete è stata utilizzata come support hardware una scheda grafica GTX780Ti, scelta giustificata dal fatto che il training effettuato su processore, risulta nettamente più lento a causa della serializzazione di operazione che esegue il processore, a differenza delle schede video in grado di processare dati adottando una forte serializzazione dei dati.

La scelta riduce notevolmente il tempo di training in rapporto circa 1: 10.



Pur essendo la struttura del problema particolarmente semplice il tempo di training non è trascurabile, La scelta effettuata ci fornisce la possibilità di effettuare un numero maggiore di test.

A livello di software di basso livello è stata utilizzata una libreria di primitive CUDNN fornite dalla Nvidia che si interfaccia con I driver CUDA per ottenere l'accelerazione del processo di training mediante scheda video.

Per l'implementazione effettiva della rete neurale è stata invece utilizzato il framework Tensorflow-gpu come backend per la libreria di alto livello Keras che ci permette di scrivere codice semplice per la creazione e il training. L'effettiva implementazione viene infatti eseguita mediante Keras, lasciando gli aspetti tecnico-computazioali a Tesnsorflow e Cuda.

DATASET UTILIZZATO

Il dataset utilizzato è stato fornito da Kaggle.

Le modalità di acquisizione dei dati è sconosciuta e poco documentata e conseguentemente non è possibile fare considerazioni preliminari a riguardo.

Il dataset è composto da 3 sezioni:

- Test
- Train
- Validation

Ognuno dei quali preposto alla funzione suggerita dal nome.

Ogni sezione è ulteriormente divisa in 2 parti:

- Man
- Woman

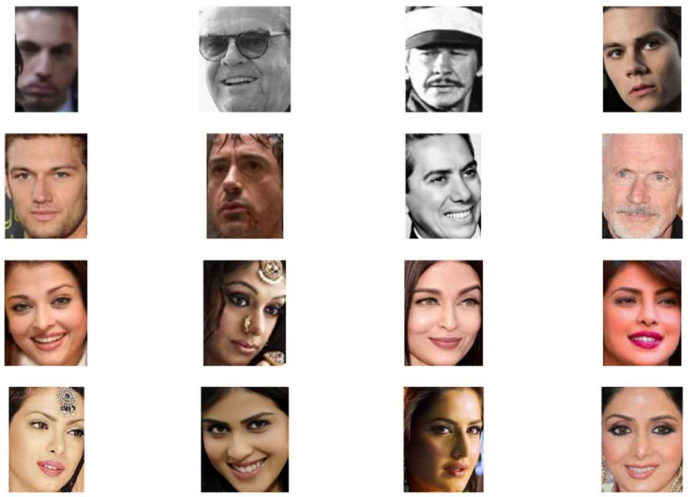
Esse contengono I dati necessari al training.

Il dataset che riguarda il training è composto da 800 foto di volti maschili e 800 foto di volti femminili.

Quello di test e validazione 170 per ogni classe.

Il dataset presenta delle criticità, ma dal numero di immagini possiamo concludere che è bilanciato.

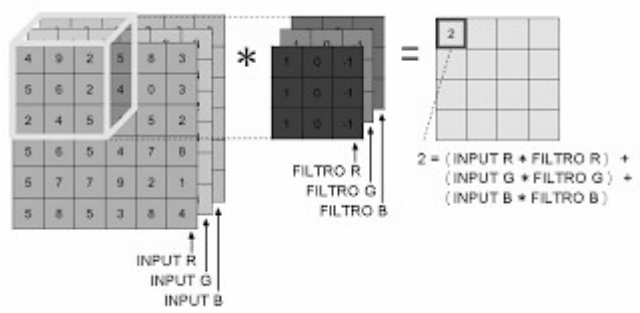
Come possiamo subito notare il dataset in questione non è molto ampio inoltre, by inspection vediamo la poca varietà di foto presenti. Da queste considerazioni è possibile subito individuare la principale criticità del progetto.

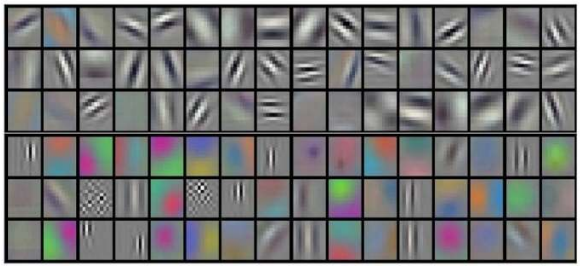


Nell'articolo di congronto "Gender Recognition Through Face Using Deep Learning" il dataset utilizzato conteneva 200 foto.

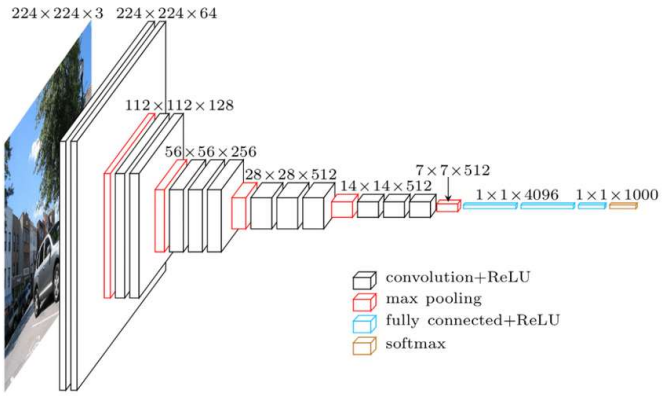
ARCHITETTURA DELLA RETE

Per ottenere buoni risultati mediante reti neurali dobbiamo utilizzare reti neurali di tipo convoluzionale in grado di trovare i pattern relativi all'identificazione uomo/donna all'interno delle singole foto. I layer convoluzionali, una volta addestrati si comportano da filtro per identificare il genere del soggetto.





La struttura utilizzata inizialmente era una struttura standard VGG-16 (Visual Geometry Group a 16 layer) così composta:



Questa architettura,diventata famosa grazie agli articoli di Simonyan e Zisserman nei paper pubblicati nel 2014 è uno standard per quanto riguarda le architetture di reti neurali convoluzionali per rilevamento di pattern all'interno di immagini.

Nell'articolo di confronto "Gender Recognition Through Face Using Deep Learning" è stata utilizzata prorpio questa architettura.

Si è voluto in questo studio mantenere una architettura più semplice possibile e limitare il numero di parametri da utilizzare per poter poi effettuare una valutazione sui tempo di risposta.

Per questo la rete è stata scelta in maniera più semplice: composta da soli 4 layer convoluzionali con attivazione ReLU abbinati ognuno a un layer di pooling 2x2, un layer flattern e un solo layer fully-connected di dimensione 512, infine un layer di uscita con attivazione sigmoid che ci dà una distribuzione di probabilità in output.

La rete ha un layer di input di (150x150x3) e un output singolo scalare.

Le dimensioni dei filtri convoluzionali e del layer fully connected sono state effettuate procedendo su 2 fronti paralleli:

- cercando di imitare reti neurali note
- effettuando test preliminari sulla architettura

L'architettura finale risultante risulta cosi' composta:

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 148, 148, 16)	448
max_pooling2d (MaxPooling2D)	(None, 74, 74, 16)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	4640
max_pooling2d_1 (MaxPooling2	(None, 36, 36, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 17, 17, 64)	0
conv2d_3 (Conv2D)	(None, 15, 15, 128)	73856
max_pooling2d_3 (MaxPooling2	(None, 7, 7, 128)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 512)	3211776
dense_1 (Dense)	(None, 1)	513

L'architettura, essendo più semplice, permette un training più veloce e, una volta che i pesi sono arrivati a convergenza mediante algoritmo di

backpropagation, ciò permette una valutazione più veloce del risultato, garantendo un calcolo veloce del risultato.

PREPROCESSING & NORMALIZZAZIONE

Le foto date in input alla rete, così come come i dati di test e di validazione devono essere scalati e normalizzati per combaciare con le dimensioni in input della rete.

Per fare ciò l'oggetto immagine è sata inizialmente scalata a dimensioni 150x150, dopodichè é stato inserito un filtro antialiasing.

L'immagine risultate è stata poi trasformata in un array multimensionale numpy di dimensioni (150,150,3).

I valori all'interno dell'array sono stati poi normalizzati: ogni pixel possiede le 3 coordinate RGB variabili tra 0 e 255, di conseguenza la scelta è stata quella di normalizzare tutti i valori di pixel dividendo l'array numpy per 255, ottendendo un array di valori compresi tra 0 e 1, più facilmente elaborabile dall'algoritmo di training.

L'implementazione di questo tipo di preprocessing è stato effettuato anche nella parte di evaluation della rete neurale. Quando forniamo nuovi dati alla rete è necessario richiamare una funzione che esegue quindi i passaggi sopra descritti prima di inserire l'array multimensionale nella rete.

PARAMETRI DELLA RETE

Durante la costruzione della rete, prima di effettuare il processo di training è stato necessario impostare alcuni parametri chiave:

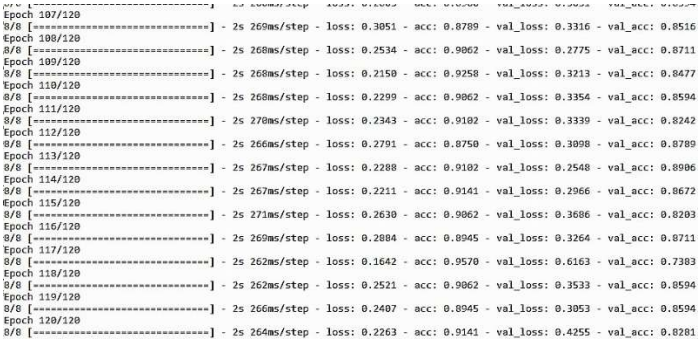
- epoche
- dimensione del batch
- learning rate

la dimensione del batch è stata mantenuta costante nei vari test effettuati con valore di 35, analogamente il learning rate con valore di 0.001.

Le epoche sono invece state fatte variare volutamente non implementando l'early stopping in modo da valutare l'overfitting del modello, in questo modo possiamo avere un'idea indicativa delle prestazioni sui dati di validazione nelle varie condizioni di prova.

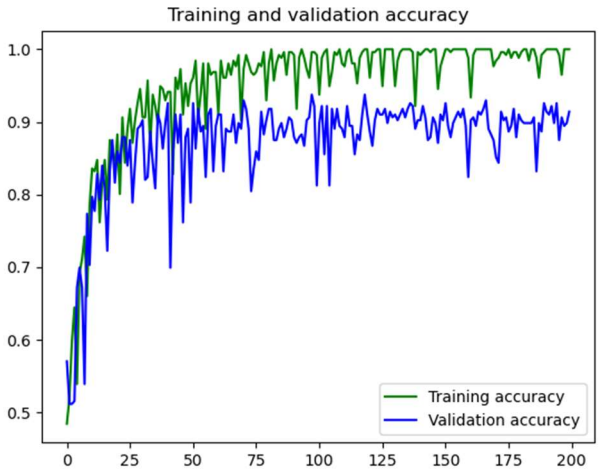
TRAINING & VALIDAZIONE

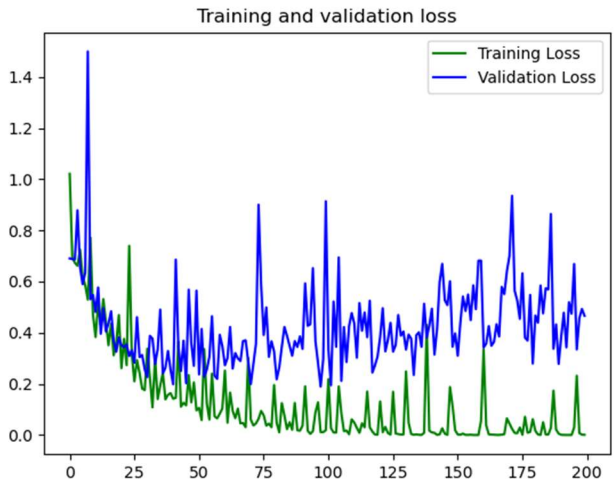
Una volta scelti tutti i parametri della rete andiamo ad effettuare il training richiamando la funzione model.fit(args**). Questa funzione inizializza l'immssione dei dati nella rete e applica l'algoritmo di backpropagation ad ogni iterazione.



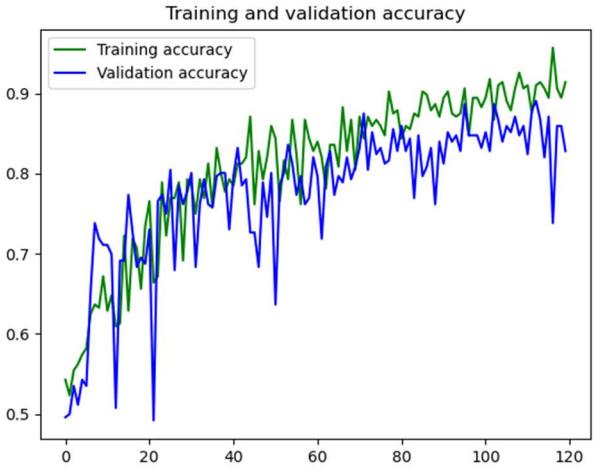
Terminato il processo di training della durata di circa 5 minuti, possiamo andare a effettuare il confronto in varie casistiche di training:

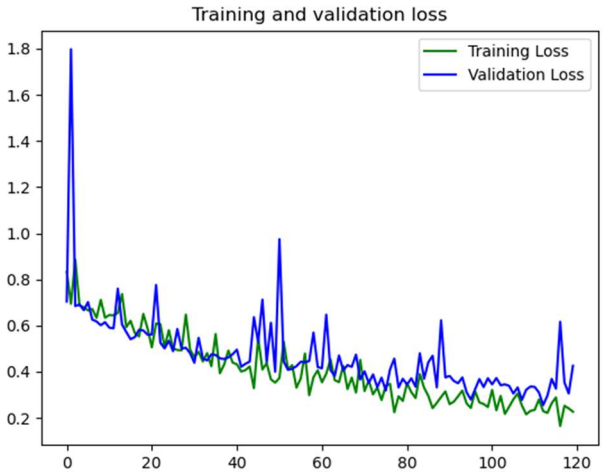
PRIMO TEST: 200 epoche, dataset standard



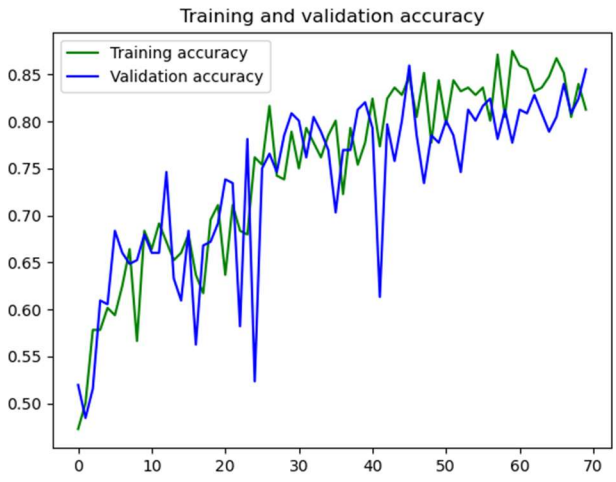


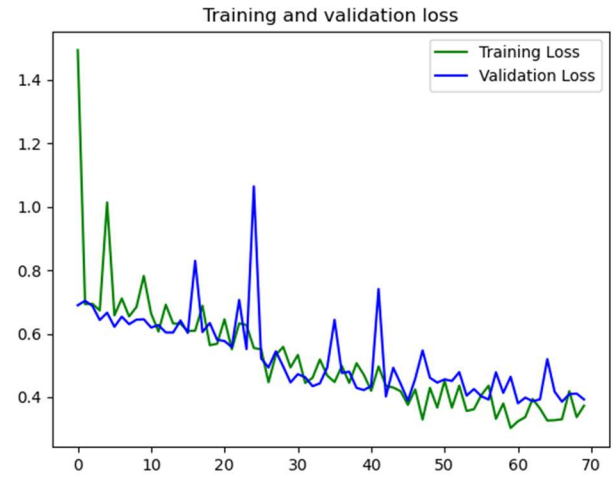
SECONDO TEST: 120 epoche , dataset aumentato





TERZO TEST: 70 epoche, dataset aumentato





Dal primo test effettuato vediamo che il sistema è in forte overfitting, la curva di accuratezza sui dati di training si discosta fortemente da quella sui dati di validazione. Ciò è anche a causa della ridotta quantità di elementi presenti nel dataset

Il secondo test cerca di aggiustare il tiro rispetto al secondo introducendo il dataset aumentato e riducendo il numero di epoche, vediamo che il sistema entra in overfit circa alla 70esima epoca.

Il terzo test, rappresentativo della architettura effettivamente implementata non entra in overfit e raggiunge una accuratezza sui dati di validazione dell'85%. Il livello di accuratezza ottenuto è ritenuto accettabile per quanto riguarda lo scopo non trattandosi di una identificazione di tipo critico.

DATASET AUMENTATO

Per ovviare al problema della bassa quantità di dati a disposizione si è deciso di utilizzare la tecnica del dataset aumentato, particolarmente efficacie per ovviare alla carenza di dati. In particolare sono stati settati i seguenti parametri:

- rotazione casuale dell'immagine fino a 20 gradi
- variazione casuale di larghezza 20%
- variazione casuale di altezza 20%
- immagine specchiata
- rumore di fondo

I primi 4 parametri aiutano a effettuare una diversificazione del dataset in modo da garantire il rilevamento in caso di diverse scale e angolazioni e sopperisce,almeno in parte, alla carenza di dati.

Il rumore di fondo è invece stato aggiunto per ottenere una caratteristica di robustezza della rete, infatti aggiungere rumore di fondo può aiutare a rendere la rete più efficacie nel rilevamento in casi in cui le informazioni derivanti dalle immagini sono incomplete o disturbate. Ad esempio, se la telecamera atta al rilevamento fosse sporca, questa tecnica permetterebbe di effettuare comunque una elaborazione corretta estrapolando, dalle informazioni parziali la risposta all'identificazione.

E' importante osservare che l'aumento del dataset è stato eseguito sia per i dati di training che per quelli di validazione, in maniera da avere stime quanto più corrette delle performance.

TEMPISTICHE DI RISPOSTA

Andiamo adesso a valutare le tempistiche di risposta della rete. Questa può importante specialmente per applicazioni nel caso di robotica, guida autonoma e per varie applicazioni critiche. Il tempo di risposta della rete ci dice quanto tempo intercorre tra l'immissione dei dati nella rete e la sua completa valutazione, ossia quanto tempo ci mette la rete, una volta allenata, a effettuare una valutazione del risultato.

E' evidente che più la dimensione della rete cresce, più crescono i parametri e conseguentemente anche le operazioni matematiche e computazionali che sono necessarie per produrre l'output.

Nel caso di applicazioni critiche dobbiamo andare a valutare il caso peggiore possibile di risposta della rete e non il tempo di risposta medio per poter garantire l'affidabilità del sistema.

Sono stati effettuati una decina di test (in realtà poco rilevanti a livello statistico) per andare a effettuare una analisi preliminare riguardante il response time del sistema utilizzando prima la struttura semplificata. Dalla analisi svolta, facendo valutare alla rete immagini nuove sono emersi i seguenti valori per la struttura semplificata:

- tempo di risposta medio: 0.015
- tempo di risposta worst case: 0.021s

Mentre sono emersi i seguenti per la struttura vgg-16 utilizzata nell'articolo di confronto:

- tempo di risposta medio: 0.017
- tempo di risposta worst case: 0.025s

Possiamo effettuare varie considerazioni a riguardo:

1. il tempo di risposta della rete, pur in un caso piuttosto semplice, richiede qualche millisecondo anche se il tempo di valutazione worst case è sicuramente più elevato di quello stimato, dato il basso numero di prove. In questa applicazione, non essendo critica, possiamo permetterci tempi di valutazione anche piuttosto elevati, ma per applicazioni più delicate (si pensi al rilevamento di persone nel caso di veicoli.
   autonomi) è necessario contenere la dimensione della rete ed effettuare una analisi approfondita sul tempo di risposta.
2. Si hanno forti variazioni a seconda dei dati inseriti nella rete.
3. Il tempo di risposta può variare anche utilizzando i medesimi dati.

La valutazione è stata effettuata considerando esclusivamente il tempo di valutazione della rete, senza considerare il tempo impiegato per il processing dei dati e la normalizzazione comunque trascurabili rispetto al tempo di risposta della rete.

L'hardware utilizzato è il medesimo usato per il training della rete.

CONCLUSIONI

Nonostante Il dataset ridotto e la semplice architettura si è ottenuta una accuratezza di rilevamento media dell'85%, questo risultato, pur non essendo ottimale, ci permette di fare considerazioni piuttosto importanti:

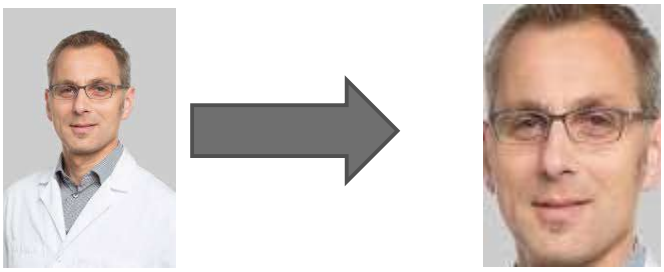
- la quantità di dati è molto rilevante, ma possiamo concludere che anche con pochi dati è possibile ottenerre buoni risultati,nell'articolo "Gender Recognition Through Face Using Deep Learning" sono stati infatti utilizzati solo 200 volti
- Le tecniche di aumenta del dataset per quanto riguarda le immagini può essere molto utile per sopperire alle carenze di dati
- Avendo molti dati possiamo rendere il sistema molto robusto per quanto riguarda dati incompleti o in presenza di disturbi
- Pur essendo una semplice architettura il tempo di risposta è elevato

CRITICITÀ RISCONTRATE

Durante la valutazione delle performance è stata riscontrata una criticità per quanto riguarda il tipo di dato fornito in input.

Le immagini fornite in input devono infatti essere esattamente immagini di volti senza includere parti circostanti a esso. La rete funziona anche nel caso la foto non sia esattamente nel formato sopra descritto ma il grado di confidenza sul risultato risulta essere decisamente basso, e quindi è molto più soggetta a errori.

Per risolvere questo problema è possibile in fase di preprocessing utilizzare un'altra rete neurale, prima per individuare il volto, e poi per scalare la foto facendo in modo che solo il volto del soggetto sia presente. Questo risultato è facilmente ottenibile utilizzando librerie di computer vision come opencv e haarcascade.



Possono essere presenti anche casistiche(in realtà dai test riscontrate difficilmente) in cui la rete non individua correttamente il soggetto.

MIGLIORAMENTI E SVILUPPI FUTURI

Un miglioramento significativo della accuratezza della rete potrebbe essere utilizzare metodi di transfer learning per poter utilizzare meglio i dati a disposizione, partendo da una situazione in cui i pesi sono più vicini a quelli desiderati per la corretta valutazione del risultato.

Sviluppi futuri, sarebbero interessanti da valutare nel caso di applicazioni real-time con vincoli restrittivi per quanto riguarda tempo di risposta per osservare e capire meglio come implementare metodi per avere garanzie in applicazione critiche.

Sarebbe inoltre interessante effettuare test circa la robustezza della rete per valutarne l'abilità di rispondere correttamente con dati incompleti effettuando un confronto di questo tipo con e senza il dataset aumentato.

RIFERIMENTI

"Gender Recognition Through Face Using Deep Learning", Amit Dhomne , Ranjit Kumar, Vijay Bhan, Procedia Computer Science 132 (2018) 2–10.

"Face recognition: a convolutiona neural-network approach", S. Lawrence ; C.L. Giles ; Ah Chung Tsoi ; A.D. Back, IEEE Transactions on Neural Networks (Volume: 8 , Issue: 1 , Jan 1997)

https://www.kaggle.com/gmlmrinalini/notebook

https://www.kaggle.com/

https://www.tensorflow.org/tutorials

https://keras.io/

https://github.com/opencv/opencv/tree/master/data/haarcascades

https://opencv.org/