UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA TRIENNALE IN

INGEGNERIA INFORMATICA

Transfer learning per selezionare gli informative-frame in una laringoscopia usando learned features

Relatore:

PROF. NANNI LORIS

Laureando:

BORTOLIN SIMONE

1188553

Anno Accademico 2020/2021

Abstract

In questa tesi si introducono alcuni metodi per il preprocessing di immagini e il data augmentation per creare nuove ensemble a partire da quelle esistenti in combinazione con un metodo di addestramento di tipo two round tuning al fine di massimizzare le prestazioni di una convolutional neural network (CNN).

In particolare useremo il transfer learning riutilizzando una CNN pre addestrata per effettuare una selezione degli informative-frame in una laringoscopia. Una buona selezione degli informative-frame permette di ottimizzare le elaborazioni e la visione da parte dei medici e tecnici competenti.

Tutto questo viene fatto attraverso una rete CNN di tipo deep pre addestrata con l'uso del trasnfer learning, data augmentation e preprocessing.

In particolare oltre ai classici metodi di rotazione e mirroring dell'immagine viene utilizzato pure una DCT e varie tecniche di preprocessing per dare in pasto alla rete una immagine più dove è maggiormente analizzabile.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

La laringoscopia è una procedura medica utilizzata per ispezionare la laringe e per diagnosticare, ed eventualmente curare, i disturbi della laringe e delle corde vocali.

In particolare ci sono due tipi di laringoscopia, quella indiretta e quella diretta. La prima fa uso di un semplice specchio e di una fonte luminosa, la seconda richiede l'uso di un laringoscopio e una anestesia.

Il laringoscopio è uno strumento che grazie a una fibra ottica, una sorgente luminosa e una telecamera permette di osservare nei minimi dettagli la laringe e tutti gli elementi che la costituiscono come la epiglottide e le corde vocali[4]. In particolare permette di aiutare a diagnosticare malattie come la laryngeal squamous cell carcinoma (SCC), la cui diagnosi precoce riduce la mortalità del paziente[10].

L'obiettivo di questa tesi è quello di usare algoritmi di apprendimento automatico basati sul transfer learning e sul preprocessing di immagini per la classificazione dei frame di una laringoscopia, in particolare la selezione degli informative-frame e lo scarto di tutti i frame non utili come frame pieni di saliva o non a fuoco. In fig. 1.2 alcuni esempi di frame da tenere estratti da una laringoscopia.

1.1 Descrizione dei dataset

Le prestazioni delle caratteristiche estratte dalla CNN sono state valutate sul dataset NBI-InfFrames che è stato costruito da 18 video endoscopici NBI, riferiti a 18 diversi pazienti affetti da carcinoma a cellule squamose (SCC). Tutti i video sono stati acquisiti con un sistema endoscopico NBI (processore video Olympus Visera Elite S190 e un videoscopio rino-laringo ENF-VH) con frame rate di 25 fps e dimensioni dell'immagine di 1920×1072 pixel.

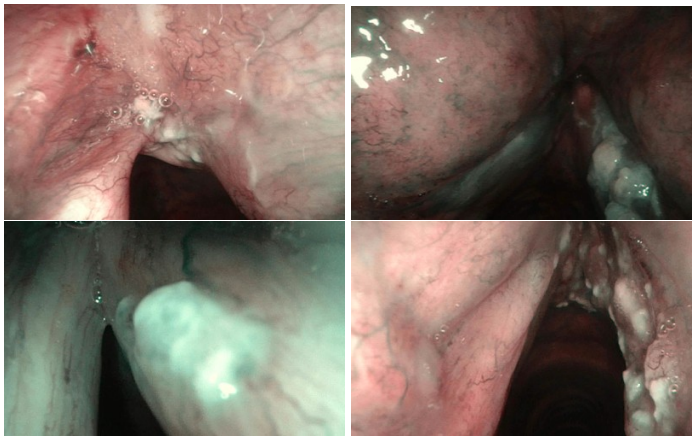


Figura 1.1: Esempio di fotogrammi endoscopici laringei

Il dataset NBI-InfFrames consiste in un totale di 720 frame, che sono equamente divisi in quattro classi: informative (I), sfuocato (B), con saliva o riflessi speculari (S) e underexposed (U).

Per ciascuno dei 18 video endoscopici NBI, i fotogrammi video sono stati estratti a caso e presentati prima a due valutatori umani. Poi, ai due valutatori è stato chiesto di etichettare i fotogrammi. Nel caso in cui i due valutatori non fossero d'accordo sulla classe, a un terzo valutatore è stato chiesto di scegliere la classe definitiva tra le due proposte dai due valutatori. Questo processo è stato ripetuto fino a quando tutti i 720 fotogrammi sono stati estratti dai video.

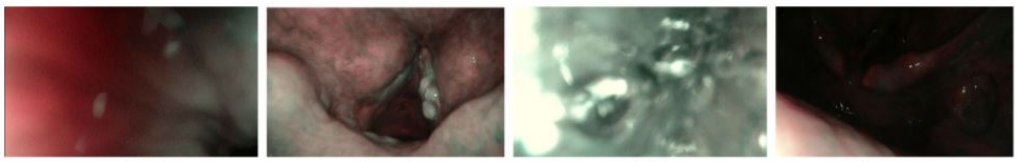


Figura 1.2: Esempio di fotogrammi endoscopici laringei di NBI-InfFrames divisi per I: informative, sfuocato: B, S: con saliva o riflessi speculari, U: underexposed

Il secondo dataset è un dataset di alcuni pixel in cui è stato effettuato un cutting a mano partendo dai frame di un video endoscopico NBI frammenti di laringe, il procedimento è mostrato nella fig. 1.3 e in fig. 1.4 sono mostrati alcuni frame catalogati il dataset 2 è catalogato in base allo stato del tessuto, in particolare Blu: tessuto con intraepithelial papillary capillary loop-like vessels; Giallo: tessuto con Leukoplakia; Verde: tessuto sano; Rosso: tessuto con hypertrophic vessels[10].

1.2 Lavori precedenti

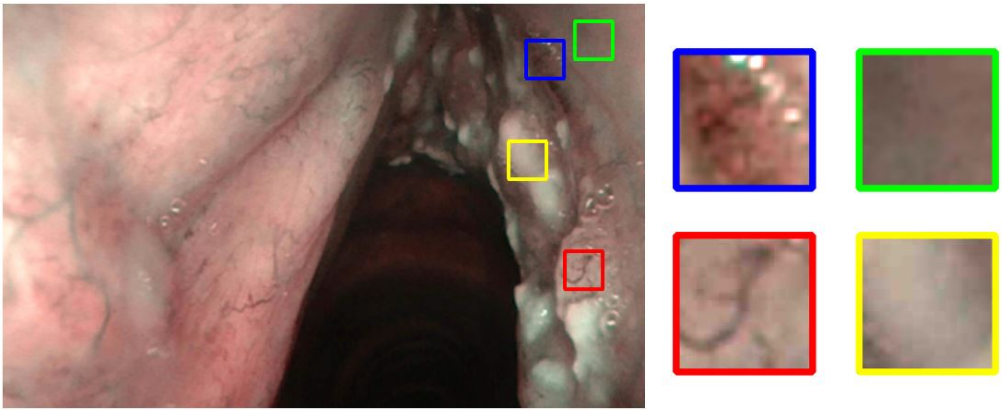


Figura 1.3: Esempio di come sono state estratte le immagini del dataset 2

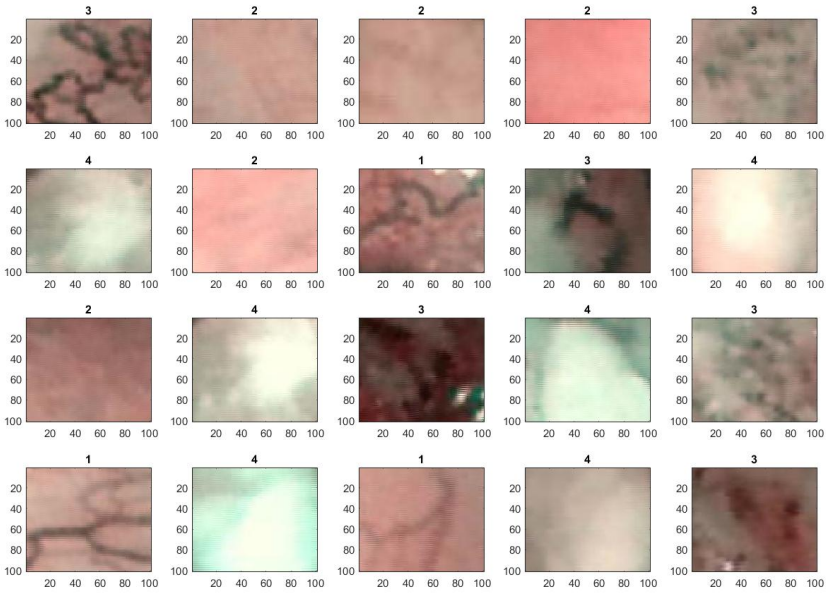


Figura 1.4: Esempio di immagini del dataset 2, le varie classi sono suddivise in base allo stato del tessuto in relazione alla SCC.

Capitolo 2

Transfer Learning

Con Transfer Learning si intente l'azione di riutilizzare una rete pre addestrata come una CNN addestrata con un milione di immagini in mille classi, per altri scopi senza dover riaddestrare la CNN ri addrestrare la rete da zero. Senza nessuna azione di riaddestramento della rete la classificazione di una immagine non darà il risultato sperato, quindi almeno in parte la rete è da riaddestrare in quanto la rete classicifica per ciò che è stata allenata. Però la struttura di una rete è tale da poter riutilizzare senza problemi.

2.1 Riutilizzo di una CNN preaddestrata

Per riutilizzare una rete preaddestrata si parte dalla rete e si rimuovono gli ultimi livelli, cioè quelli di classificazione finale e si lasciano quelli di feacture extractor, la rete senza i livelli finali è può essere usata come feacture extractor. Per esempio si può usare una SVM per la classificazione finale, oppure si possono riaggiungere layer dello stesso tipo di quelli originali ovviamente riadattati al numero di classi del programma, a differenza dei layer originali essi hanno i pesi non inizializzati, questa tecnica si chiama fine-tuning. È una tecnica molto usata in quanto il training di CNN complesse (es. AlexNet) su dataset di grandi dimensioni (es. ImageNet) può richiedere giorni/settimane di elaborazione su un computer, anche se eseguito su hardware dedicato. Una volta che la rete è stata addestrata, il tempo richiesto per la classificazione di un pattern (propagazione forward o inference) è in genere inferiore[19][9].

In genere è da sottolineare che addestrare da zero una CNN porta a risultati migliori, ma richiede una potenza di calcolo molto più elevata, oltre che dei dataset molto grandi, per evitare l'overfitting. L'addestramento di una CNN funziona molto bene su grandi insiemi di dati, tuttavia su piccoli insiemi di dati non riesce a raggiungere guadagni significativi e proprio in questo caso viene utilizzato il transfer learning[3].

Il transfer learning è un metodo molto utilizzato nella computer vision, in quanto permette l'uso della conoscenza acquisita per un compito, come riconoscere le immagini del dataset ImageNet, per risolvere quelli correlati, come riconoscere immagini di un altro dataset. Si basa sul fatto che funzionalità riguardanti l'estrazione di caratteristiche di basso livello (per esempio, bordi, forme e angoli) possono essere condivise tra i vari compiti di una rete, nel nostro caso: classificare le immagini in 1000 classi o classificare immagini sfuocate e non sfuocate[14].

2.2 Fine-Tuning

Nell'approccio fine-tuning del Transfer Learning si parte con una rete pre addestrata su un problema simile e:

1. Si rimpiazza il livello di output con un nuovo livello di output softmax (adeguando il numero di classi) ed altri livelli finali, come i fully connected.

2. Come valori iniziali dei pesi si utilizzano quelli della rete pre-trained, tranne che per le connessioni tra il penultimo e ultimo livello i cui pesi sono inizializzati random.

3. Si eseguono nuove iterazioni di addestramento per ottimizzare i pesi rispetto alle peculiarità del nuovo dataset (non è necessario che il dataset sia di grandi dimensioni, ma per migliorare le prestazioni si usa data augmentation e preprocessing).

In generale, tutti i pesi sono allenabili, cioè in ogni nuovo allenamento essi possono cambiare normalmente attraverso l'algoritmo di ottimizzazione gradient descent (GD). In alcuni lavori essi vengono tenuti fissi durante la fase di fine-tuning. L'unico livello integrato che ha pesi non allenabili è il livello di batch normalization[16][8].

In particolare si possono individuare numerosi approcci per il fine-tuning, tra cui il classico one round:

One round tuning (1R) È l'approccio standard per il fine-tuning di una rete pre addestrata: la rete viene inizializzata secondo i pesi ottenuti dall'allenamento con il dataset ImageNet e riaddestrata utilizzando il training set del problema target. I pesi degli strati di classificazione vengono azzerati, i pesi dei primi stati non vengono forzati a rimanere stabili e infine viene usato lo stesso learning rate per tutti gli strati[8].

Two rounds tuning (2R) Questa strategia prevede un primo round di fine-tuning in un dataset simile a quello target e un secondo round utilizzando il training set del problema target. Il primo passo consiste nel fine-tuning della rete, pre-inizializzata secondo i pesi di ImageNet che include

immagini di classi non incorporate nel problema target. Il secondo passo è un one round tuning eseguito usando un dataset esterno.

L'idea alla base di questo metodo è di insegnare in primo luogo alla rete a riconoscere le laringi, che sono molto diverse dalle immagini nel dataset ImageNet, poi il secondo round viene utilizzato per regolare i pesi di classificazione in base al problema di destinazione. A causa della possibilità di overfitting, aumentare il numero di round turning non garantisce un aumento delle prestazioni, sopratutto nel caso di un numero basso di immagini in addestramento[8]. Nella fig. 2.1 è riportato uno schema dei due metodi, dove ogni colore è relativo a un approccio separato.

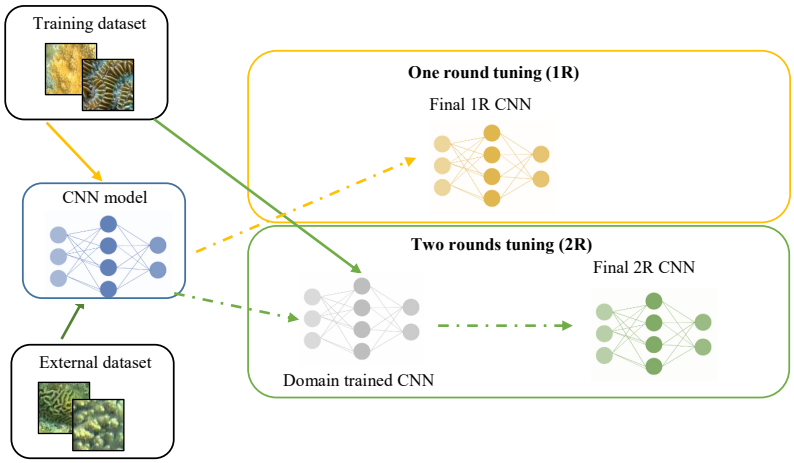


Figura 2.1: Uno schema dei due approcci: In giallo il One round turing (1R), in verde il Two round tuning. Le frecce piene denotano l'input per l'addestramento, le frecce tratteggiate denotano i flussi di output (modelli addestrati).

In particolare per ottenere buoni risultati con il fine-tuning è importante diminuire la velocità di apprendimento (cioè il fattore learning rate che modifica i pesi), e se necessario congelare alcuni livelli preaddestrati che potrebbero essere alterati in maniera sostanziale abbassando le performance.

Se la velocità di apprendimento è troppo alta i risultati peggiorano a casua del gradient descent, in particolare questo può portare a uno stato in cui la rete neurale non riesce a trovare il minimo globale ma solo locale[2][3].

2.3 Data preprocessing e data augmentation

La fase di pre-elaborazione è essenziale nelle immagini delle laringi per rimuovere diversi tipi di rumori, questo vale ancora di più nel caso di transfer learning in quanto la dimensione del dataset di allenamento è in genere molto inferiore a di quello originale. E che numerosi classi hanno un significato meno generale o più generale[3]. In particolare i metodi di preprocessing di immagini sono trattati in capitolo 5.

Per ottenere prestazioni più elevate nell'accuratezza, la CNN richiede grandi insiemi di dati in quanto le performance della CNN con piccoli set di dati si deteriora a causa dell'overfitting[3] in particolare i metodi di data augmentation di immagini sono trattati in capitolo 4.

Capitolo 3

Frame

Il laringoscopio fornisce una serie di immagini (detti anche frame) in sequenza che formano un video, non tutti i frame che escono dal laringoscopio sono utili a capire lo stato di salute della laringe di una persona in quanto non sono a fuoco, o sono sfuocate o contengono

Capitolo 4

Data augmentation

Con Data augmentation si intende la tecnica usata per la generazione di nuovi dati per aumentare artificialmente il numero di campioni di allenamento ed è diventato una pratica comune nell'area del deep learning.

È noto che l'aumento dei dati può aumentare l'invarianza di un classificatore e può agire come un regolatore nel prevenire l'overfitting nelle reti neurali, come in AlexNet (la rete utilizzata in questa tesi). Gli autori del noto AlexNet hanno dichiarato che senza l'aumento dei dati la loro rete avrebbe sofferto di un sostanziale overfitting, che li avrebbe costretti a usare reti meno profonde[7]. Ma non solo le reti profonde, anche quelle meno profonde possono essere notevolmente migliorate adottando l'aumento dei dati, tipicamente utilizzato nel deep learning[18].

4.1 Metodi comuni di Data augmentation

Le principali operazioni sono per esempio il ritaglio di immagini per l'addestramento in posizioni diverse, la rotazione delle immagini (sia leggere di qualche grado che di 90, 180, 240 gradi), modificare l'illuminazione ed il contrasto, una DA basica da applicare ad un rete neurale è la seguente[11]:

- L'immagine si riflette nella direzione sinistra-destra con il 50% di probabilità.
- L'immagine è riflessa in modo casuale in entrambe le direzioni sinistra-destra e in alto-basso.
- Dalla trasformazione precedente l'immagine viene scalata lungo entrambi gli assi di due diversi fattori che sono campionati casualmente dalla distribuzione uniforme tra [1...2].
- Combina tutte le trasformazioni precedenti e aggiunge la rotazione e la traslazione dell'immagine in entrambe le direzioni. La rotazione viene effettuata utilizzando un angolo che

viene campionato casualmente dall'intervallo [−10...10], mentre la traslazione consiste nello spostare l'immagine di un certo numero di pixel campionati casualmente da [0...5].

4.2 Data augmentation con DCT

In aggiunta alle trasforomazioni standard precedenti si usa anche l'approccio trattato in [11] attraverso le modifiche nei sottospazi ottenuti attraverso le trasformazioni Discrete Cosine Transform (DCT). La trasformazione è basata sulla proiezione dell'immagine originale sul sottospazio DCT, sulla perturbazione del sotto spazio e della proiezione dell'immagine perturbata dal sottospazio al dominio immagine, come mostrato in fig. 4.1.

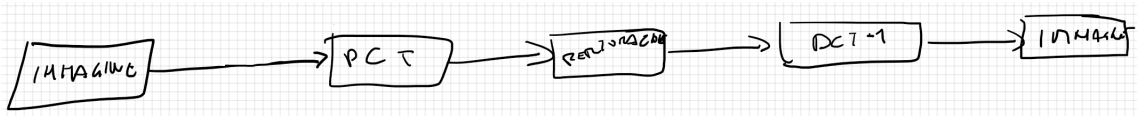


Figura 4.1: Architettura di una trasfromazione via DCT

4.3 Discrete Cosine Transform (DCT)

La Discrete Cosine Transform è una delle più importanti trasformate nell'ambito della computer vision, in particolare è in grado di rilevare le variazioni di informazione tra un'area e quella contigua di un'immagine raster trascurando le ripetizioni, viene usata in numerosi ambiti dalla compressione delle immmagini all'elaborazione delle immagini. In particolare è simile alla trasformata discreta di Fourier (DFT), ma fa uso solo di numeri reali[6].

Ci sono numerose varianti della DTC, in particolare la DTC tipo II, che spesso è chiamata DTC e la sua inversa, la DTC tipo III che viene chiamata spesso come la IDCT. In linguaggio matematico la formula della DCT tipo II è la seguente[6][11]:

$$\text{DCTimage}(x,y) = \sum_{p,q=1}^{n} a_p a_q \,\text{Image}(p,q) \cos\left(\frac{2p-1}{2n}\right) \cos\left(\frac{2q-1}{2n}\right) \quad \text{where}\, a_p = \begin{cases} \sqrt{\frac{1}{n}}, n = 1 \\ \sqrt{\frac{2}{n}}, n > 1 \end{cases}$$

In particolare si individuano 3 possibili algoritmi di preprocessing DTC:

...

Capitolo 5

Preprocessing

Le immagini raw nei dataset sono sufficienti ad una CNN per ottenere buoni risultati, ma è possibile migliorare i risultati attraverso una pre-eleaborazione delle immagini, in particolare la rimozione di artefatti, del rumore e migliorare la qualità dell'immagine in cui le feacture possono essere estratte correttamente[15]

5.1 Contrasto

Il contrasto è la differenza tra la massima e la minima intensità dei pixel. Si può modificare il contrasto all'immagine per migliorarla secondo determinati valori statici, oppure secondo tecniche più raffinate, per esempio attraverso la deviazione standard e la media.

Un metodo più raffinato è la stima del MSE e PSNR. Nel processo di miglioramento del contrasto, i pixel con un valore di colore più basso di un valore specifico vengono visualizzati come nero, mentre i pixel con un valore di colore più alto vengono visualizzati come bianco, e i pixel che hanno un valore di colore tra questi due valori vengono visualizzati come una tinta di grigio. Il risultato di questo processo è una mappatura lineare di un sottoinsieme di valori che coprono l'intera gamma di grigi da nero al bianco, creando un'immagine di maggiore contrasto.

L'algoritmo per l'ottimizzazione del contrasto è usato per modificare la gamma dei valori di colore per usare tutti i valori possibili per migliorare il contrasto. Per preservare l'accurata proporzione del colore quando viene usato un algoritmo per la stiratura del contrasto, viene applicata una stiratura simile per lo stretching di tutti i canali.

Nel caso di una immagine a colori, per esempio se i canali rosso e verde sono bilanciati, ma non il canale blu viene allungato l'istogramma del canale blu su entrambi i lati per ottenere istogramma ben distribuito.

L'equalizzazione dell'istogramma si può basare, per esempio sul peak signal-to-noise ratio (PSNR), è un indicatore di qualità dell'immagine è calcolato usando l'equazione:

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{\mathrm{MAX}_1^2}{\mathrm{MSE}}\right)$$

Il PSNR è ben definito semplicemente attraverso l'errore quadratico medio (MSE). Le immagini monocromatiche $m \times n$ prive di rumore I e la sua approssimazione rumorosa K è definita dal MSE:

$$\mathrm{MSE} = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

Una volta determinato il PSNR è possibile effettuare una equalizzazione dell'istogramma per aggiustare il contrasto come mostrato in fig. 5.1[13][15][5].

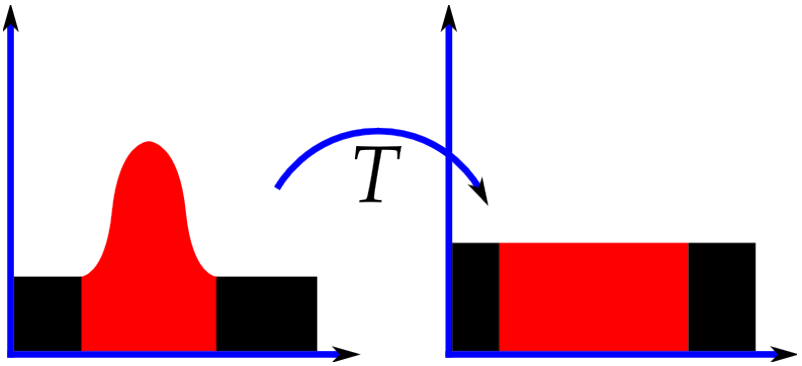


Figura 5.1: Metodo dell'equalizzazione dell'istogramma

CLAHE Una tecnica più avanzata per migliorare il contrasto nelle immagini è la contrast limited adaptive histogram equalization (CLAHE) che a differenza della classica equalizzazione dell'istogramma divide ogni immagini in tante sotto immagini e per ogni sotto immagine calcola l'istogramma ed effettua il miglioramento del contrasto, come mostrato in fig. 7.7[5].

5.2 Gamma

La correzione gamma è una correzione non lineare che viene applicata all'immagine per far apparire all'occhio umano le sfumature catturate dalle fotocamere digitali. La correzione gamma si basa sul fato che i nostri occhi non percepiscono la luce come fanno le fotocamere, l'occhio è molto più sensibile ai cambiamenti tra i toni scuri mentre il sensore della fotocamera è molto più lineare, come mostrato in fig. 5.3. L'informazione sulla gamma è semplicemente incorporata nei metadata del file e l'immagine viene codificata in maniera lineare, rendendo necessario applicare il filtro

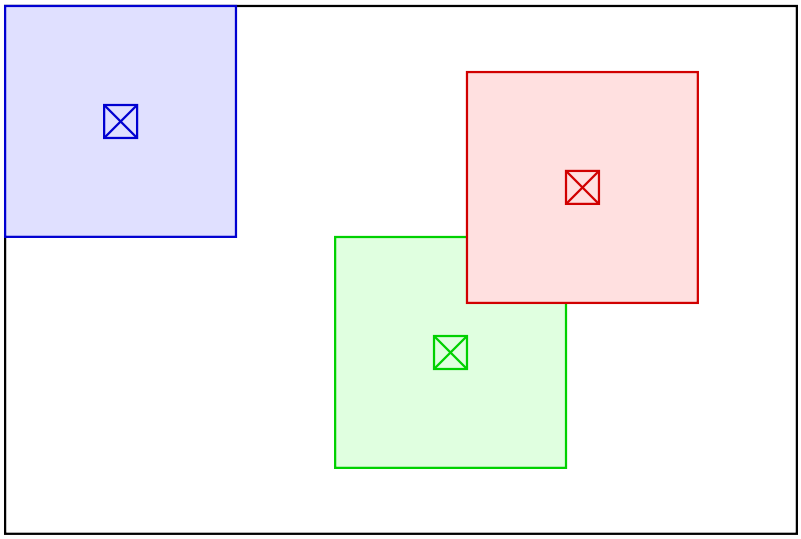


Figura 5.2: Metodo dell'equalizzazione dell'istogramma

gamma in fase di decodifica, è plausibile ipotizzare che pure una CNN funzioni meglio con una immagine pre elaborata con un filtro per la gamma[17].

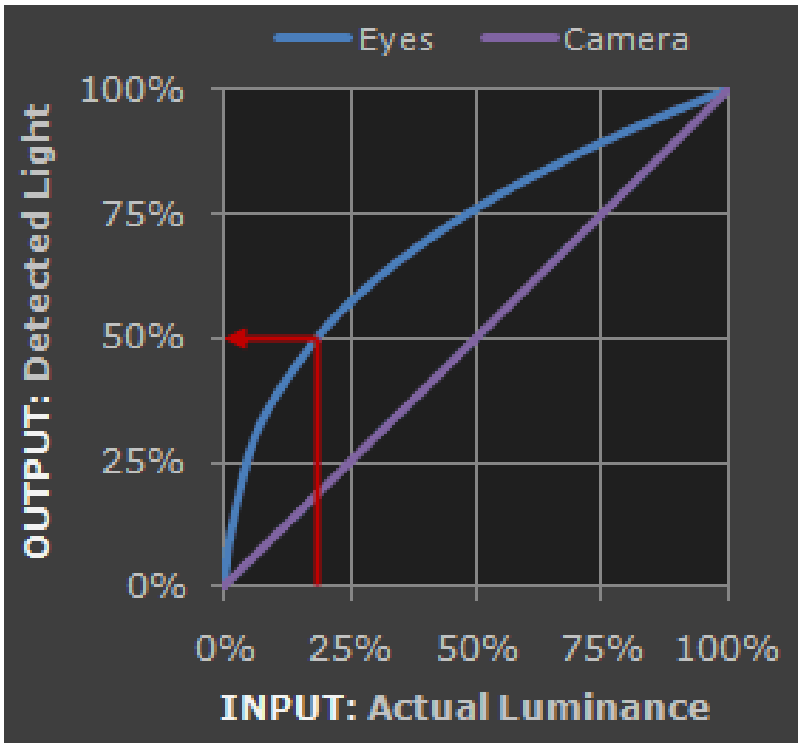


Figura 5.3: Luminanza dell'obbiettivo fotografico e dell'occhio umano

Capitolo 6

Ensemble di classificatori

L'utilizzo di più classificatori per creare un ensemble (insieme) è frequentemente utilizzato nel Deep Learning, si basa sull'utilizzo di più reti, o modelli, per risolvere un problema di classificazione. L'idea di base è che utilizzando più punti di vista diversi, uniti secondo una regola, si possano raggiungere accuratezze più alte ed errori meno frequenti.

Ogni singolo classificatore fornisce in output la propria decisione che consiste nella classe cui ha assegnato il pattern.[9][20].

6.1 Majority Rule

Il più semplice modo per combinare più classificatori è a livello di decisione, con questa metodologia, ogni classificatore fornisce in output la classe alla quale ha segnato il Pattern. Con n classificatori: ognuno vota una classe per il dato pattern, e semplicemente assegno il Pattern (cioè classifico il Pattern) come la classe maggiormente votata.

Più formalmente. Se $\{C_1, C_2, ... C_{nc}\}$ è un insieme di nc classificatori, e

$$\theta_{ij} = \begin{cases} 1 & \text{se } i \text{ è la classe votata da } C_j \\ 0 & \text{altrimenti} \end{cases} \qquad (1 \leq i \leq s, 1 \leq j \leq nc)$$

Allora il pattern è assegnato alla classe t tale che:

$$t = \underset{i=1...s}{\text{argmax}} \left\{ \sum_{j=1...nc} \theta_{ij} \right\}$$

Capitolo 7

Addestramento rete neurale

Come già menzionato nei capitoli precedenti l'idea è quella di allenare una CNN per la classificazione dei frame di una laringoscopia attraverso il transfer learning impiegando la tecnica del fine-tuning nelle due varianti (One Round Tuning e Two Round Tuning) su una rete pre-allenata chiamata AlexNet.

7.1 AlexNet

AlexNet è la rete che ha rivoluzionato la computer vision nel 2012 essa è una CNN sviluppata da Goffrey Hinton e Alex Krizhevsky dell'università di Toronto vince, con ampio margine, l'ImageNet challenge: object classification and detection su milioni di immagini e 1000 classi.

La rete AlexNet è organizzata nel seguente modo: costituita da 8 livelli di cui 5 di convolution layer e 3 fully connected, ed infine la funzione di attivazione rectified linear (ReLu), la dimensione delle immagini in input è di 227×227. L'architettura completa con le dimensioni dei layer di convoluzione è illustrata in fig. 7.1[7].

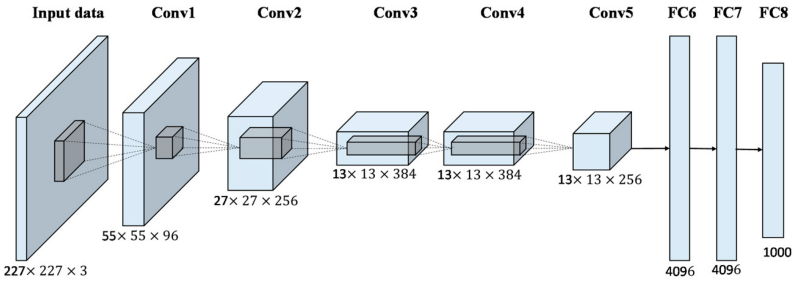


Figura 7.1: Architettura della CNN AlexNet

La rete AlexNet utilizzata è già allenata per il riconoscimento di 1000 classi di soggetti (mele, gatti, tastiere, telefoni, ecc.) su un dataset di oltre 1 milione di pattern presi dal database ImageNet[7].

Viene usato un allenamento k-fold, in particolare con $k = 3$, in ogni fold i pattern vengono elaborati con un ordine diverso e si utilizza circa il 20% di essi come test set.

7.2 Allenamento 1R

La sperimentazione inizia con l'allenamento della CNN con un fine-tuning standard e classico (denominato anche One Round Tuning, descritto in sezione 2.2), nel senso che si rimpiazzano gli ultimi 3 layer e poi si riallena la rete. Come dataset è stato usato il dataset descritto NBI-InfFrames in sezione 1.1. È da prestare attenzione ai vari iperparametri, come Learning rate e Mini-batch size. In particolare è stato usato un learning rate di 0.0001 e un mini batch size di 30, il metodo di ottimizzazione è lo stochastic gradient descent (SGD), come data augmentation sono stati usati i valori descritti nel capitolo 4, in questo caso non è stata usata la discrete cousene transform. La confusion matrix di questo allenamento è mostrato in fig. 7.2.

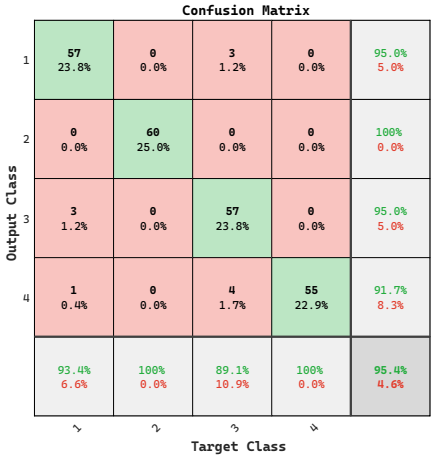


Figura 7.2: Confusion matrix dell'allenamento con TL 1R senza preprocessing e con una semplice data augmentation

7.3 Allenamento 2R

L'allenamento 2R descritto in sezione 2.2 è composto da una prima fase di allenamento con un dataset similare e poi il dataset NBI-InfFrames, entrambi descritti in sezione 1.1. In questo caso gli iperparametri sono: learning rate di 0.0001 e un mini batch size di 30, il metodo di ottimizzazione è il SGD, come data argumentation sono stati usati i valori descritti nel capitolo 4 e non è stata usata la discrete cousene transform. La confusion matrix di questo allenamento è mostrato in fig. 7.3.

Per effettuare un allenamento 2R è sufficiente duplicare il codice di un TL facendo attenzione a usare la stessa rete.

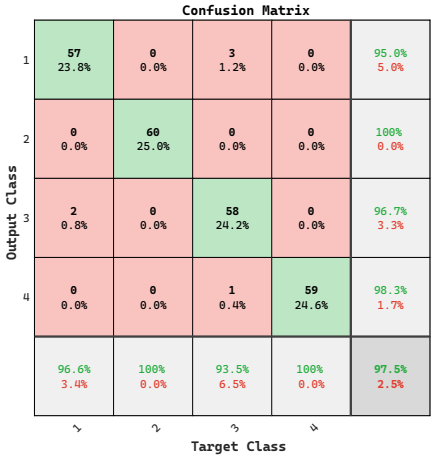


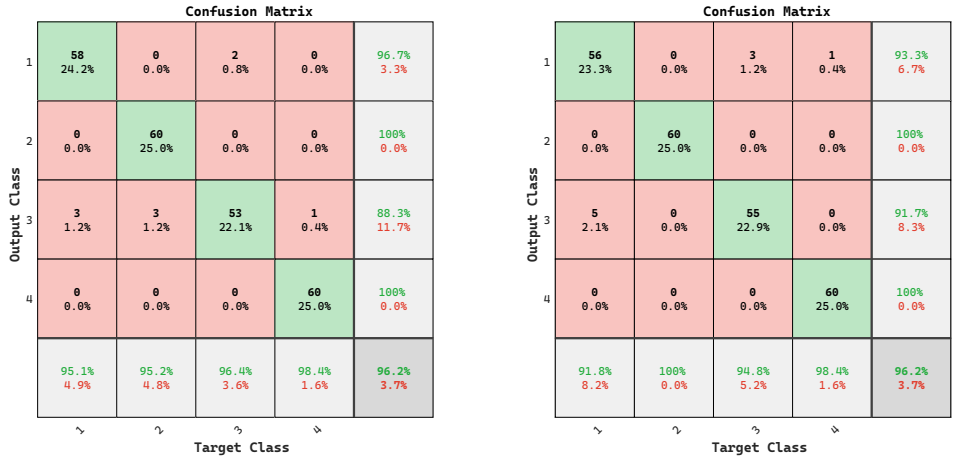
Figura 7.3: Confusion matrix dell'allenamento con TL 2R senza preprocessing e con una semplice data augmentation

7.4 Image preprocessing

Un primo filtro di image preprocessing è modificare il contrasto dell'immagine, come noto il contrasto di una immagine influenza molto la vista umana, e dato che le CNN sono molto simili al modello di connettività dei neuroni nel cervello umano è plausibile che modifiche al contrasto influenzano i risultati ottenuti. Un semplice filtro per il contrasto si ottiene con:

```
IM=imadjust(IM,[.2 .3 0; .6 .7 1],[]);
```

In figura fig. 7.4 sono presenti le confusion matrix dei risultati ottenuti con la modifica del contrasto.



(a) Allenamento 1R

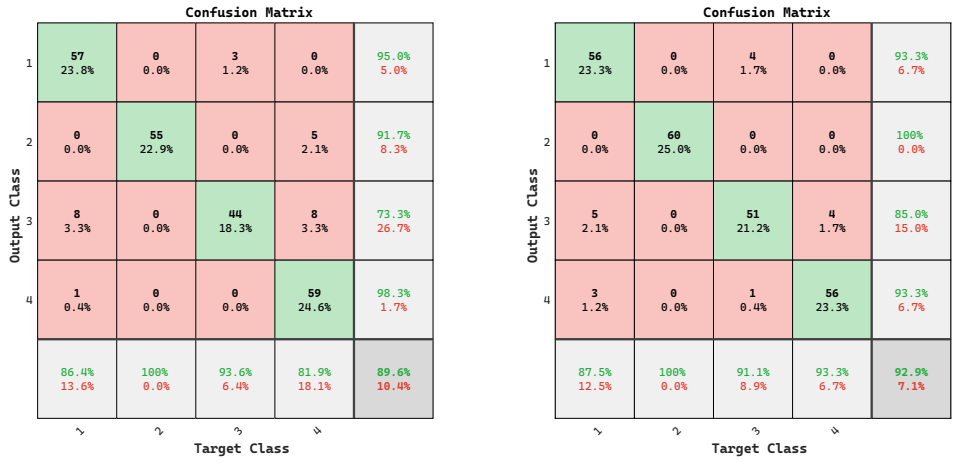
(b) Allenamento 2R

Figura 7.4: Confusion matrix dell'allenamento con TL 1R e 2R con correzzione contrasto hardcoded

Dato che nella preelaborazione precedente si usa un filtro hardcoded, lo step successivo è mo-

dificare il contrasto in base alla media e alla derivazione standard dell'immagine, con n un iperparametro che indica quanto modificare il contrasto. In fig. 7.5 la confusion matrix con i risulati ottenuti con una correzione contrasto dinamica.

```
n = 0.5;
Idouble = im2double(IM);
avg = mean2(Idouble);
sigma = std2(Idouble);
IM = imadjust(IM,[avg-n*sigma avg+n*sigma],[]);
```



(a) Allenamento 1R

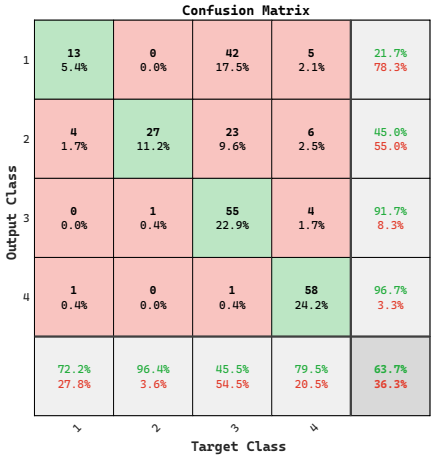
(b) Allenamento 2R

Figura 7.5: Confusion matrix dell'allenamento con TL 1R e 2R con correzione contrasto dinamica

Una versione più avanzata per l'elaborazione del contrasto è il CLAHE, il codice è descritto nel listato successivo e i risultati in fig. 7.7.

```
[X, MAP] = rgb2ind(IM, 65536);
RGB = ind2rgb(X,MAP);
LAB = rgb2lab(RGB);
L = LAB(:,:,1)/100;
L = adapthisteq(L,'NumTiles',[8 8],'ClipLimit',0.005);
LAB(:,:,1) = L*100;
IM = lab2rgb(LAB);
```

Per effettuare un miglioramento della Gamma (descritto in sezione 5.2) dell'immagine e renderla più simile a quella dell'occhio umano si può usare il seguente codice, i risultati sono in fig. 5.3.

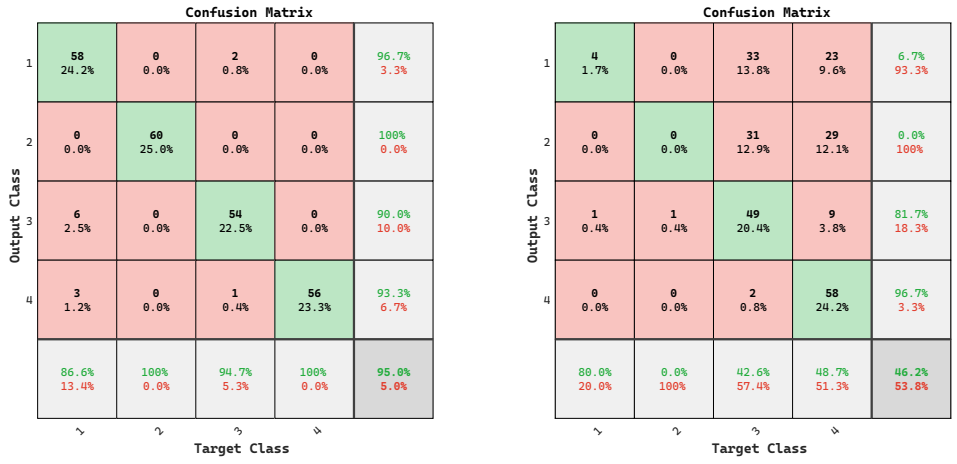


(a) Allenamento 1R

Figura 7.6: Confusion matrix dell'allenamento con CLAHE

```
hgamma = vision.GammaCorrector(2.0,'Correction','De-gamma');
```

```
IM= hgamma(IM);
```



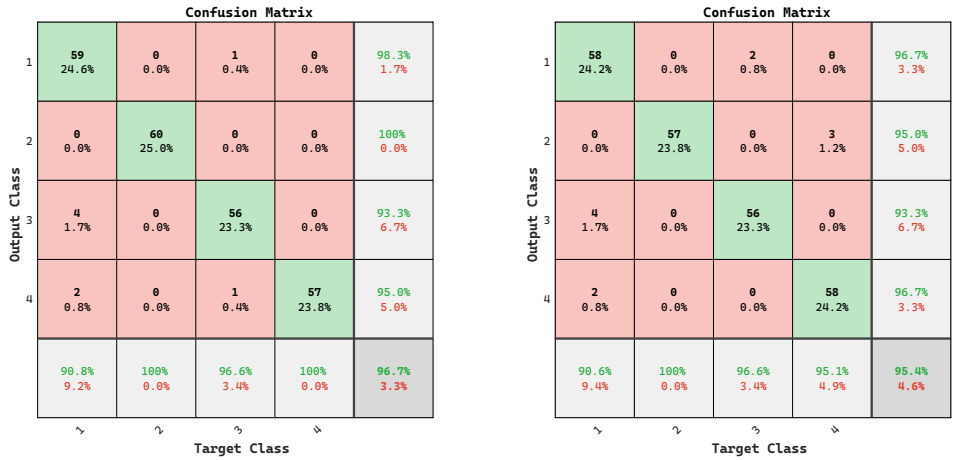
(a) Allenamento senza CLAHE

(b) Allenamento con CLAHE

Figura 7.7: Confusion matrix dell'allenamento con ottimizzazione gamma

Per aggiungere rumore artificiale all'immagine ed evitare che l'immagine si concentra su feature secondarie che potrebbero determinare basse prestazioni nei test set, i risultati sono in fig. 7.8.

```
IM = imnoise(IM,'salt & pepper',0.02);
```



(a) Allenamento 1R

(b) Allenamento 2R

Figura 7.8: Confusion matrix dell'allenamento con TL 1R e 2R con aggiunta di rumore artificiale

7.5 Data argumentation con DCT

Ci sono numerosi metodi per eseguire la DCT ed la relativa data augmentation, in particolare in fig. 7.9 sono illustrati due algoritmi, il primo che per ogni frequenza la annulla con probabilità 0.5 e la seconda che la attenua con probabilità $z \sim U\left(-\frac{1}{2}, \frac{1}{2}\right)$[11].

I risultati con DTC e Noise sono illustrati nella fig. 7.10.

7.6 Ensemble di classificatori

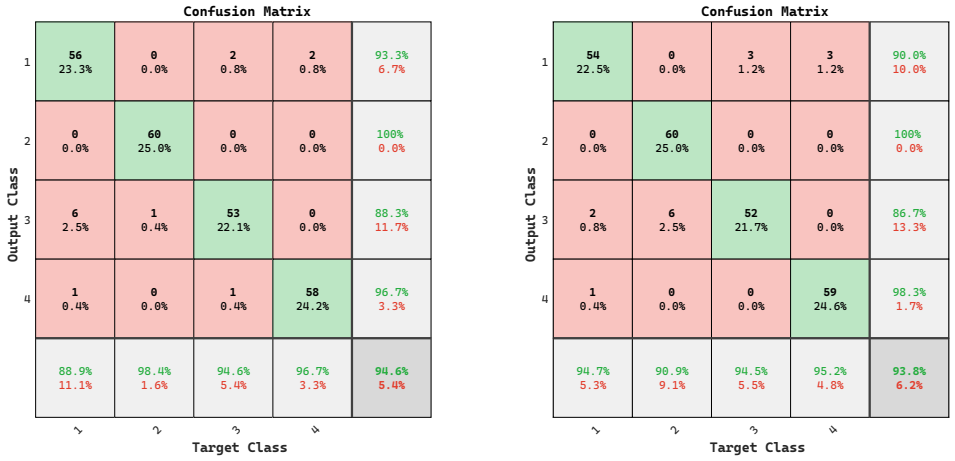
Algorithm MethodOne

Input: Image: tensor $n\ x\ n\ x\ 3$
Output: NewImage: tensor $n\ x\ n\ x\ 3$ channel <- 1; **for** every channel **do**
#DCTimage is a matrix of dimension n x n
DCTimage <- calculateDCT(Image(:, :, channel)); *# see Eq. 1* **for** row,col in DCTimage **do** with probability 0.5 **do**
DCTimage(row, col) = 0; #except DCTimage(1,1) that cannot be reset
end
end
#inverse of the perturbated image
NewImage(:, :, channel) <- inverseDCT(DCTimage);
end

Algorithm MethodTwo

Input: Image, tensor $n\ x\ n\ x\ 3$
Output: NewImage, tensor $n\ x\ n\ x\ 3$ channel <- 1; **for** every channel **do**
#DCTimage is a matrix of dimension n x n
DCTimage <- calculateDCT(Image(:, :, channel)); *# see Eq. 1*
Sigma = standardDeviation(Image)/2; **for** row,col in DCTimage **do**
DCTimage(row, col) += sigma * random number $z \sim U(-\frac{1}{2}, \frac{1}{2})$;
except DCTimage(1,1) that cannot be modified
end
#inverse of the perturbated image
NewImage(:, :, channel) <- inverseDCT(DCTimage);
end

Figura 7.9: Due algoritmi di data augmentation con DCT



(a) Allenamento 1R

(b) Allenamento 2R

Figura 7.10: Confusion matrix dell'allenamento con TL 1R e 2R con DCT e noise

Capitolo 8

Conclusioni

Si nota come un semplice doppio allenamento 2R migliora le performance generali della rete, in particolare si arriva a un 97.5% di performance generali con un data augmentation basico. Nonostante ciò con l'inserimento di rumore le performance calano, lasciando intuire che la rete in generale prende in considerazioni alcune feature secondarie, che con il noise verrebbero meno.

	1R		2R	
	(1))	(2)	(1)	(2)
Semplice DA	95.4%	100%	97.5%	100%
Filtro Contrasto Semplice	96.2%	100%	96.2%	100%
Filtro Contrasto con media e STD	89.6%	91.7%	92.9%	100%
CLAHE	63.7%	45.8%		
Correzione Gamma	95.0%	100%		
Correzione Gamma e CLAHE	46.0%	0%		
Noise	96.7%	100%	95.4%	95.0%
DCT e Noise	94.6%	100%	93.8%	100%

Tabella 8.1: Specchio riassuntivo delle performance singole dei vari metodi analizzati, (1): Divisione corretta nelle 4 classi, (2): Riconoscimento dei frame Informative

Glossario

rectified linear (ReLu) È una funzione di attivazione nata per risolvere alcuni problemi delle funzioni di attivazione usate nelle reti MLP, come la sigmoide, in quanto essa da problemi nella retropropagazione del gradiente. La funzione è definita come

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

La funzione restituisce 0 se il valore è minore di zero altrimenti restituisce il numero stesso. 19

convolutional neural network (CNN) Una rete neurale convoluzionale (ConvNet/CNN) è un algoritmo di Deep Learning che prende in input un'immagine, ai vari input vengono assegnati i pesi e un bias in modo che la rete sia in grado di cogliere gli aspetti e gli oggetti dell'immagine ed essere in grado di differenziare gli uni dagli altri. Come si deduce dal nome le CNN fa ampio uso della convoluzione

La pre-elaborazione richiesta in una CNN è molto più bassa rispetto ad altri algoritmi di classificazione. Mentre nei metodi primitivi i filtri sono costruiti a mano, con abbastanza addestramento, le CNN hanno la capacità di imparare questi filtri.

L'architettura di una CNN è molto simile a quella del modello di connettività dei neuroni nel cervello umano ed è stata ispirata all'organizzazione della corteccia visiva. I singoli neuroni rispondono agli stimoli solo in una regione ristretta del campo visivo noto come campo recettivo. Un insieme di tali campi si sovrappone per coprire l'intera area visiva.

La prima rete deep a funzionare era AlexNet nel 2012, ma già nel 1998 le CNN ottengono buone prestazioni in problemi di piccole dimensioni (es. riconoscimento caratteri, riconoscimento oggetti a bassa risoluzione), ma bisogna attendere il 2012 (AlexNet) per un radicale cambio di passo.

I problemi di una CNN nel 1998 non erano tanto in ambito matematico, problema che è stato risolto in maniera semplice, bensì nella disponibilità di dataset e potenza di calcolo per gestire ed elaborare i dataset.

In particolare nelle reti CNN, a differenza di quelle classiche l'accuratezza aumenta all'aumentare del dataset, e a differenza di una rete shallow dove il plaeau è facilmente raggiungibile in una deep ci sono vari studi che indicano che non si raggiunge e la rete migliora sempre (un asintoto orizzontale che determina la impossibilità di allenare meglio la rete).

In particolare il problema matematico era relativo al problema della Vanishing (or exploding) gradient che impediva l'allenamento di reti profonde. Infatti se io moltiplico tante volte un numero raggiungo un valore prossimo allo zero il risultato non cambia e la rete neurale non modifica i relativi pesi una volta raggiunto il minimo la rete neurale esce. Ma la rete neurale a ogni addestramento deve modificare i proprio pesi, perciò non si riesce ad avere un addestramento stabile o ad avere un minimo, per esempio dopo aver raggiunto il minimo è possibile che il valore si modifichi andando a peggiorare le performance.

Le reti CNN hanno diverse applicazioni nel riconoscimento di immagini, video e audio, nei sistemi di raccomandazione, nell'elaborazione del linguaggio naturale e, recentemente, in bioinformatica. i, iii, v, 1, 5, 8, 13, 19–21

gradient descent (GD) È il principale algoritmo di error backpropagation, la minimizzazione dell'errore avviene attraverso passi in direzione opposta al gradiente.

6, 30, 32

stochastic gradient descent (SGD) È un metodo iterativo per l'ottimizzazione di funzioni differenziabili, è l'approssimazione stocastica del metodo di GD quando la funzione costo è una somma. È ampiamente usato nell'allenamento dell'intelligenza artificiale 20

batch normalization È una trasformazione che mantiene l'output medio vicino a 0 e la deviazione standard dell'output vicino a 1. Ciò ha l'effetto di stabilizzare il processo di apprendimento e di ridurre drasticamente il numero di periodi di addestramento necessari per addestrare reti profonde. 6

convolution layer È un layer che prende in input una immagine e ne torna fuori un'altra immagine dopo aver effettuato una convoluzione. In particolare prende in input una immagine $IH \times IW \times IC$ e torna una feacture map definita da $FH \times FW \times FC$ che rappresenta il risultato all'interno del livello.

Le operazioni all'interno del livello vengono determinate da filtri digitali $L \times L$, che vengono fatti scorrere attraverso ogni pixel di input, calcolandone il prodotto scalare. Si possono determinare diversi parametri, la dimensione del filtro, l'utilizzo di padding e di stride, per ottenere la miglior rappresentazione dell'immagine. 19, 31

convoluzione È un operatore matematico che a partire da funzioni $f(t)$ e $g(t)$, restituendone una terza che rappresenta come la forma di una funzione una influisce sull'altra. È una operazione che gode della proprietà commutativa:

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(t)g(\tau - t)d\tau = \int_{-\infty}^{\infty} g(t)f(\tau - t)d\tau$$

L'operazione è tipica dell'elaborazione dei segnali e più in generale delle immagini, nel caso di un una immagine discreta è:

$$\mathbf{I} * \mathbf{F}[i,j] = \sum_{y=1}^{m}\sum_{x=1}^{m}\left(\mathbf{I}\left[i + \left\lceil\frac{m}{2}\right\rceil - y, j + \left\lceil\frac{m}{2}\right\rceil - x\right] \cdot \mathbf{F}[y,x]\right)$$

Dove $\mathbf{F}$ è il filtro definito sulla griglia $m \times m$, $\mathbf{I}$ è la nostra matrice immagine, e $\mathbf{I} * \mathbf{F}$ sono i nostri pixel filtrati.

La Complessità computazionale della convoluzione è piuttosto elevata: data un'immagine di $n \times n$ pixel e un filtro di $m \times m$ elementi, la convoluzione richiede m^2n^2 moltiplicazioni e altrettante somme.

La soluzione è lavorare nello spazio di Fourier, ricordando che la FFT (Fast Fourier Transform) si ottiene con un procedimento $O(n \log n)$. Sapendo che date due funzioni f e g la cui convoluzione è indicata da $f * g$, sia $\mathcal{F}$ l'operatore trasformata di Fourier, $\mathcal{F}\{f\}$ e $\mathcal{F}\{g\}$ sono le trasformate di f e g rispettivamente. Allora: $\mathcal{F}\{f * g\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{g\}$. 19, 29, 30

fully connected I livelli completamente connessi in una rete neurale sono quei livelli in cui tutti gli input di un livello sono collegati a ogni unità di attivazione del livello successivo. Normalmente nei modelli di apprendimento automatico più diffusi, gli ultimi livelli sono livelli completamente connessi che compilano i dati estratti dai livelli precedenti per formare l'output finale. È il secondo livello più dispendioso in termini di tempo, dopo il convolution layer. 6, 19, 33

learning rate Detta anche velocità di apprendimento è un iperparametro in un algoritmo di ottimizzazione, normalmente il GD. Determina quanto modificare i pesi in base all'attuale errore. La scelta del tasso di apprendimento è impegnativa in quanto un valore troppo piccolo può comportare un lungo processo di allenamento che potrebbe bloccarsi, mentre un valore troppo grande può comportare l'apprendimento di una serie di pesi non ottimali o un processo di allenamento instabile.

La velocità di apprendimento può essere l'iperparametro più importante durante la configurazione della rete neurale. Pertanto è fondamentale sapere come indagare gli effetti del tasso di apprendimento sulle prestazioni del modello e costruire un'intuizione sulla dinamica del tasso di apprendimento sul comportamento del modello. 20

mini-batch size La dimensione del batch definisce il numero di campioni che verranno analizzati attraverso la rete in un turno.

Ad esempio, supponiamo che tu abbia 1050 campioni di addestramento e desideri impostare un valore batch size uguale a 100. L'algoritmo prende i primi 100 campioni (dal 1° al 100°) dal set di dati di addestramento e addestra la rete. Successivamente, prende i secondi 100 campioni (dal 101° al 200°) e addestra nuovamente la rete. Possiamo continuare a eseguire questa procedura finché non abbiamo propagato tutti i campioni attraverso la rete. Il problema potrebbe verificarsi con l'ultima serie di campioni. Nel nostro esempio, abbiamo usato 1050 che non è divisibile per 100 senza resto. La soluzione più semplice è ottenere gli ultimi 50 campioni e addestrare la rete. Si parla di Mini-batch quando la dimensione del batch è molto piccola.

I vantaggi dell'uso di una batch size minore al numero totale dei campioni:

- Poiché si addestra la rete utilizzando meno campioni, la procedura di addestramento complessiva richiede meno memoria. Ciò è particolarmente importante se non sei in grado di inserire l'intero set di dati nella memoria della tua macchina.
- In genere le reti si addestrano più velocemente con i mini-batch. Questo perché aggiorniamo i pesi dopo ogni propagazione. Nel nostro esempio abbiamo propagato 11 batch (10 di loro avevano 100 campioni e 1 aveva 50 campioni) e dopo ciascuno di essi abbiamo aggiornato i parametri della nostra rete. Se usassimo tutti i campioni durante la propagazione, faremmo solo un aggiornamento per il parametro di rete.

È da sottolineare che più piccolo è il batch, meno accurata sarà la stima del gradiente. 20

overfitting Succede quando il sistema tende a identificare relazioni nell'insieme di addestramento che non valgono in generale cioè che il sistema impara a memoria, questo porta a ottimi risultati nel data set e pessimi risultati nel test set o negli usi reali. 5, 7, 8

padding Per evitare che i pixel laterali abbiano una minore influenza sull'output di una convoluzione è tipico aggiungere uno zero-padding (un bordo di pixel con valore zero) per poter filtrare anche i pixel laterali dell'immagine, che altrimenti verrebbero analizzati da un numero inferiore di filtri in quanto i filtri non possono uscire dalla matrice, causando un shrinking (riduzione di dimensione) dell'output e quindi a una perdita di informazioni. 31

propagazione forward Detta anche inference, è la propagazione delle informazioni in avanti: dal livello di input a quello di output. Una volta addestrata (cioè fissati i pesi w_i), una rete neurale può semplicemente processare pattern attraverso forward propagation.

Cioè si salvano i vari pesi w_i della rete in memoria e ogni volta che arriva un pattern la rete fa calcoli interni e fornisce l'output. Dopo aver addestrato la rete, il sistema è abbastanza veloce. Per esempio le reti deep sono estremamente complesse, ci vuole un elevato tempo per addestrarle, ma dopo averle addestrate hanno tempi velocissimi di esecuzione.

In una rete a 3 livelli ($d : n_H : s$): input d neuroni, livello nascosto (hidden) n_H neuroni, output s neuroni.

Il numero totale di pesi (o parametri) è: $d \times n_H + n_H \times s + n_H + s$ in quanto ogni neurone collegato con tutti gli altri neuroni del livello successivo, e gli ultimi due termini corrispondono ai pesi dei bias.

Il k-esimo valore di output può essere calcolato come:

$$z_k = f\left(\sum_{j=1\ldots n_H} w_{jk} \cdot y_j + w_{0k}\right) = f\left(\sum_{j=1\ldots n_H} w_{jk} \cdot f\left(\sum_{i=1\ldots d} w_{ij} \cdot x_i + w_{0j}\right) + w_{0k}\right)$$

Si nota come una volta noti i vari w_i le operazioni da fare per calcolare l'output sono eseguibili in un tempo proporzionale al numero di pesi e di livelli. 5

softmax Alla fine della rete neurale profonda c'è un livello finale softmax, che effettua una vera e propria classificazione: consiste in n neuroni, uno per ogni classe, ciascuno connesso a tutti i neuroni del livello precedente (fully connected). Essenzialmente è simile ai neuroni della MLP.

Il livello di attivazione net_k dei singoli neuroni si calcola nel modo consueto, ma come funzione di attivazione per il neurone k-esimo (invece di Sigmoide o ReLu) si utilizza:

$$\mathbf{z}_k = f(\text{net}_k) = \frac{e^{\text{net}_k}}{\sum_{c=1\ldots s} e^{\text{net}_c}}$$

dove i valori z_k prodotti possono essere interpretati come probabilità: appartengono a [0...1] e la loro somma è 1. Il livello SoftMax, quindi traduce i valori di net prodotti dall'ultimo livello della rete in probabilità.

È importante sottolineare che funzioni come Tanh, Sigmnoid, ReLu non forniscono probabilità in quanto:

$$\sum_{c=1\ldots s} z_c \neq 1$$

6

stride Questo parametro regola la dimensione tra pixel di ingresso e pixel in uscita, in particolare quando un filtro viene fatto scorrere sul volume di input, invece di spostarsi con passi unitari (di 1 neurone) si può utilizzare un passo (o stride) maggiore. Questa operazione riduce la dimensione delle feature map nel volume di output e conseguentemente il numero di connessioni. Lo stride è spesso impostato su un numero intero, anziché su una frazione o un decimale per facilitare i calcoli.

Sui primi livelli di una CNN con uno piccolo stride maggiore di uno, è possibile ottenere un elevato guadagno nell'efficienza con una piccola penalizzazione nella accuratezza. 31

Bibliografia

[1] Corina Barbalata e Leonardo S. Mattos. “Laryngeal Tumor Detection and Classification in Endoscopic Video”. In: *IEEE Journal of Biomedical and Health Informatics* 20.1 (2016), pp. 322–332.

[2] Joel Carneiro. *Fine Tuning vs. Transfer learning vs. Learning from scratch*.

[3] Joel Carneiro et al. “A novel deep learning based framework for the detection and classification of breast cancer using transfer learning”. In: *Pattern Recognition Letters* 125 (2019), pp. 1–6.

[4] Sperati Giorgio, Casolino Delfo e Salsi Daria. “Cenni storici. La videolaringoscopia.” In: *XXXII Convegno Nazionale di Aggiornamento 2008*. Pollenzo (Cuneo), 2008.

[5] Robert Hummel. “Image enhancement by histogram transformation”. In: *Computer Graphics and Image Processing* 6.2 (1977), pp. 184–195.

[6] Syed Ali Khayam. *The Discrete Cosine Transform (DCT): Theory and Application*. Michigan State University, 2003.

[7] Alex Krizhevsky, Ilya Sutskever e Geoffrey Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Neural Information Processing Systems* 25 (gen. 2012).

[8] Alessandra Lumini, Loris Nanni e Gianluca Maguolo. “Deep learning for Plankton and Coral Classification”. In: (ago. 2019).

[9] Davide Maltoni. “Deep Learning”. In: *Machine Learning - Materiale di Supporto al Corso* (2021).

[10] Sara Moccia et al. “Confident texture-based laryngeal tissue classification for early stage diagnosis support”. In: *Journal of Medical Imaging* 4 (set. 2017), p. 1.

[11] Loris Nanni et al. *General Purpose (GenP) Bioimage Ensemble of Handcrafted and Learned Features with Data Augmentation*. Apr. 2019.

[12] Sinno Jialin Pan e Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359.

[13] Pratibha Pandey, Kranti Kumar Dewangan e Deepak Kumar Dewangan. “Enhancing the quality of satellite images by preprocessing and contrast enhancement”. In: (2017), pp. 56–60.

[14] Ilaria Patrini et al. “Transfer learning for informative-frame selection in laryngoscopic videos through learned features”. In: *Medical & Biological Engineering & Computing* 58 (mar. 2020).

[15] S. Perumal e Velmurugan T. “Preprocessing by contrast enhancement techniques for medical images”. In: *International Journal of Pure and Applied Mathematics* 118 (gen. 2018), pp. 3681–3688.

[16] Keras Team. *Keras documentation: Transfer learning & fine-tuning*. en. URL: `https://keras.io/guides/transfer_learning/` (visitato il 28/06/2021).

[17] *Understanding Gamma Correction*. URL: `https://www.cambridgeincolour.com/tutorials/gamma-correction.htm` (visitato il 05/07/2021).

[18] Georg Wimmer, Andreas Uhl e Andreas Vécsei. *Evaluation of domain specific data augmentation techniques for the classification of celiac disease using endoscopic imagery*. Ott. 2017.

[19] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *Advances in Neural Information Processing Systems (NIPS)* 27 (nov. 2014).

[20] Zhi-Hua Zhou, Jianxin Wu e Wei Tang. “Ensembling neural networks: Many could be better than all”. In: *Artificial Intelligence* 137.1 (2002), pp. 239–263.