UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione

# Valutazione di algoritmi per il riconoscimento facciale con diversi sottoinsiemi di foto segnaletiche dalla procedura di foto-segnalazione

**Evaluating Deep Neural Networks for Face Recognition with Different Subsets of Mugshots from the Photo-Signaling Procedure**

Candidato:
**Nicolò Rossini**

Relatore:
**Chiar.mo Prof. Aldo Franco Dragoni**

Correlatore:
**Dott. Paolo Sernani**

Anno Accademico 2022-2023

# Sommario

Il riconoscimento facciale è una tecnica che permette di identificare una persona tramite la valutazione del suo volto all'interno di una o più immagini che lo ritraggono. L'avvento dell'intelligenza artificiale, in particolare con l'utilizzo di algoritmi di deep learning, ha reso questo campo più sofisticato, rendendo l'identificazione di soggetti sempre più accurata.
Questa tecnica viene utilizzata in molti campi. Uno di questi è l'uso, da parte della Polizia di Stato, di tecniche di fotosegnalamento per il riconoscimento di persone in diverse situazioni. In questa tesi sono state analizzate le performance di varie reti neurali riguardo a compiti di identificazione e verifica del volto, in modo da determinare se sia vantaggioso un fotosegnalamento, diverso dall'attuale standard italiano, composto da foto frontale e profilo sinistro. Per far ciò, la Polizia di Stato e l'Università Politecnica delle Marche hanno collaborato per lo sviluppo di un database chiamato FRMDB (*Face Recognition from Mugshots Database*). Questo dataset è composto da immagini di fotosegnalamento e videosorveglianza, prese in diverse angolazioni, di un determinato numero di soggetti: sulla base dei risultati ottenuti, si andrà a valutare quale rete neurale ha avuto un'accuratezza migliore.

# Indice

# 1. Introduzione

Il riconoscimento facciale è una tecnica biometrica che permette di riconoscere ed autenticare l'identità di una persona basandosi su immagini o video che la ritraggono. Al giorno d'oggi, questo sistema viene utilizzato in vari settori quali quello militare, della finanza e della sicurezza pubblica, oltre ad essere presente in numerose situazioni di vita quotidiana come, per esempio, lo sblocco di un dispositivo utilizzando il viso. Il riconoscimento facciale comprende due operazioni:

- `La verifica`: Una procedura con la quale si cerca di determinare l'appartenenza di un volto ad una certa identità. Un esempio di tale tecnica è la verifica del volto per sbloccare il telefono.
- `L'identificazione`: Una procedura che permette di individuare una persona tra più possibili identità. Un suo possibile utilizzo riguarda l'analisi, da parte delle forze dell'ordine, di un volto su un dataset per poterlo identificare.

A tal proposito, in questa tesi si cerca di dar continuità allo studio fatto dall'Università Politecnica delle Marche insieme alla Polizia di Stato riguardo l'applicazione di algoritmi di deep-learning per il riconoscimento facciale.
La Polizia di Stato ha già in uso un sistema di riconoscimento facciale chiamato SARI (*Sistema Automatico di Riconoscimento Immagini*) che permette di comparare le immagini riprese dalle videocamere di sorveglianza con le immagini contenute nella Banca Dati A.F.I.S. (acronimo di *Automated Fingerprint Identification System*). Durante la procedura di fotosegnalamento, le forze di polizia nazionali raccolgono regolarmente due foto, ovvero quella frontale e quella del profilo destro (comunemente dette mugshots), insieme alle impronte digitali e alle informazioni personali di un soggetto. L'obbiettivo di questa ricerca è quello di comprendere se, acquisendo delle foto di un soggetto in diverse angolazioni, si può migliorare l'accuratezza e l'efficienza del riconoscimento facciale. Per far ciò, l'Università Politecnica delle Marche ha sviluppato un database costituito da foto e video di vari soggetti presi in diversi punti di vista, chiamato FRMDB [1]. In questa tesi mi sono concentrato nell'andare ad eseguire dei test sul database prima citato, utilizzando tre modelli di reti neurali convoluzionali già pre-addestrate. In particolare, le tre reti sono state: la VGG16 [2], la ResNet50 [3] e la SE-ResNet50 [4]. I risultati prodotti hanno descritto l'accuratezza e le performance di questi tre modelli nel riconoscimento facciale. Per concludere, è stata identificata quale delle tre reti ha generato risultati migliori.

## 1.1 Storia del riconoscimento facciale

La tecnica del riconoscimento facciale risale agli anni '60 quando Woody Bledsoe sviluppò un sistema nel quale un operatore andava a misurare le distanze tra i diversi tratti del viso (come, per esempio, gli estremi della bocca) nelle fotografie stampate e le inseriva in un software che le confrontava con tutte le immagini presenti in un database in modo da identificare la persona [5]. Negli anni '90 uscì una pubblicazione che introduceva il modello Eigenface [6]. Tale modello estraeva le varie caratteristiche di un volto, utilizzando la *Principal component analysis* (PCA), le codificava e le comparava con le altre caratteristiche contenute in un database. Purtroppo, l'estrazione delle caratteristiche veniva fatta in maniera globale, tralasciando aspetti facciali più dettagliati. Infatti, questo modello era molto interessante dal punto di vista tecnico ma presentava delle difficoltà quando erano presenti variazioni facciali differenti rispetto alle informazioni acquisite. Proprio per questo motivo, negli anni successivi, si sviluppò un nuovo tipo di approccio basato sull'uso di descrittori di features raccolte dalle immagini. Alcune tecniche che prendono spunto da questo modello sono state Gabor e LPB (*Local Binary Patterns*): queste andavano ad acquisire manualmente le varie feature e questo comportava ad una scarsa generalità del sistema. Negli anni successivi, questa tecnica venne modificata in modo da far costruire i descrittori delle features da dei modelli, le cosiddette tecniche di apprendimento automatico. Questo ha permesso di avere un sistema con una maggiore precisione e una miglior ottimizzazione.
Nel 2012 venne introdotto un nuovo metodo che si basava su tecniche di deep learning per il riconoscimento facciale: questo modello venne definito AlexNet [7] e comprendeva una rete neurale convoluzionale profonda 8 strati (5 di convoluzione e 3 completamente connessi), dei quali ognuno apprendeva livelli di rappresentazione differenti, andando a garantire una grande stabilità alle variazioni presenti tra le immagini. Infatti, questo sistema presentava un tasso di errore nelle migliori 5 predizioni del 15.3%. Questo modello ha dato inizio allo sviluppo di tecniche per il riconoscimento facciale basate su algoritmi di deep learning per l'estrazione di features dalle immagini. Un esempio è stato la proposta da parte di ricercatori di Facebook del loro sistema chiamato DeepFace [8]: un modello di rete neurale convoluzionale costituito da 9 strati e addestrato su oltre 4 milioni di immagini che ha raggiunto il miglior livello di accuratezza sul dataset LFW (*Labeled Faces in the Wild*) [9] pari al 97,35%.
Nel 2015, il Visual Geometry Group dell'Università di Oxford ha sviluppato il sistema VGGFace [10] grazie alla raccolta di 2.6 milioni di immagini appartenenti a 2600 soggetti. Tale modello ha livelli di accuratezza ancora superiori al DeepFace (siamo attorno al 99%).
Con l'avanzare della tecnologia si sono sviluppati, e si andranno a sviluppare, dei modelli sempre più efficienti dal punto di vista computazionale e con un livello di accuratezza maggiore.

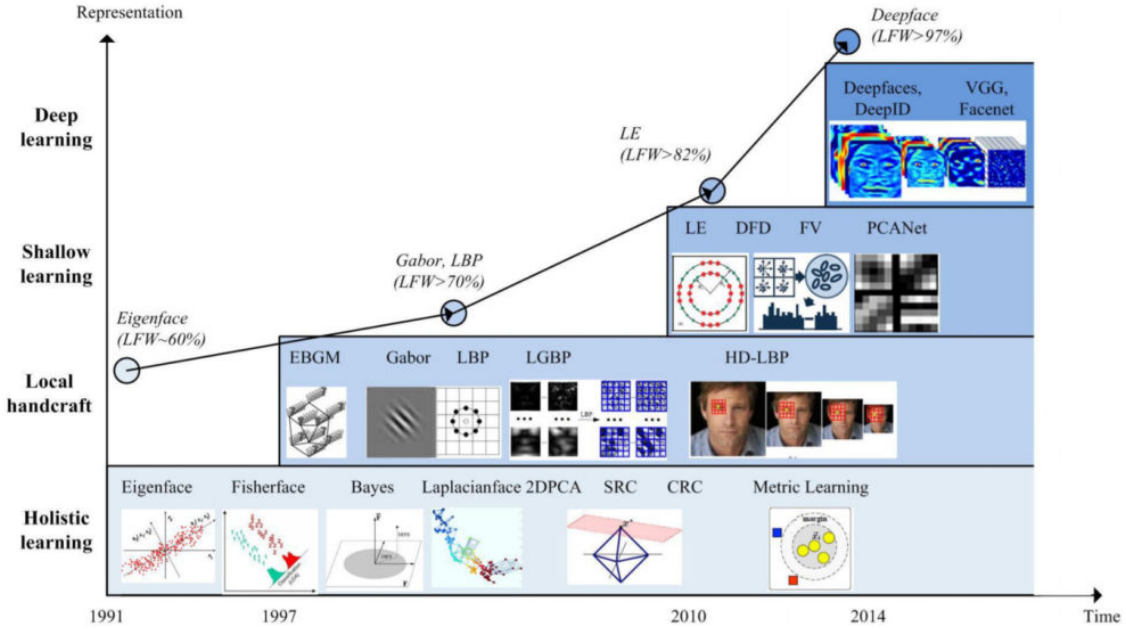



Figura 1.1: Rappresentazione di tutti i modelli sviluppati fino ad oggi.

# 2. Reti neurali convoluzionali

Le reti neurali convoluzionali (CNN, *Convolutional Neural Network*) sono una tipologia di reti neurali utilizzate per l'elaborazione delle immagini: vengono adottate nel campo della computer vision per la classificazione delle immagini, il rilevamento e la localizzazione di oggetti.
Le CNN, essendo reti neurali, sono organizzate in neuroni e questi compongono i vari strati dell'architettura (Figura 2.1). Questi ultimi sono collegati tra loro e ad ognuno viene associato un determinato peso: la complessità della rete è data proprio dal numero di livelli presenti.
Un neurone non è altro che una funzione matematica che riceve un input, elabora l'informazione e restituisce un output.
É bene precisare che in queste tipologie di reti, sia l'input che l'output, sono rappresentate su tre dimensioni: `Larghezza`, `Altezza` e `Profondità`. Tutte e tre vanno a costituire il volume di un oggetto.
Osservando un'immagine è possibile notare come questa sia composta da una serie di pixel che formano una matrice. Anch'essa viene rappresentata su tre dimensioni all'interno di una rete convoluzionale: in particolare, larghezza e altezza sono descritte dal numero di pixel che l'immagine rappresenta in queste due dimensioni, mentre la profondità è data dal numero di canali (tre se l'immagine è RGB).
Di seguito vengono illustrati i differenti strati che compongono una rete neurale convoluzionale.

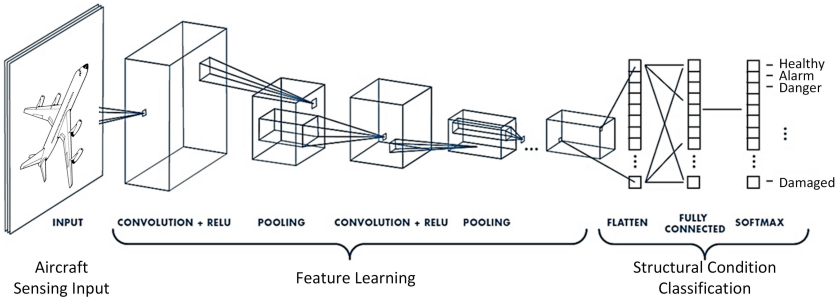



Figura 2.1: Composizione di una rete neurale convoluzionale. [11]

## 2.1 Strato convoluzionale

Lo strato convoluzionale è l'elemento principale di una CNN ed è incaricato di estrarre determinate caratteristiche da un'immagine presa in input. Per far ciò, si utilizzano particolari filtri detti *Kernel*.
Il Kernel è una matrice, di dimensioni minori rispetto all'immagine considerata, che viene fatta scorrere per l'intero volume di input e, tramite un'operazione di convoluzione, restituisce una matrice di valori chiamata *feature map* o *activation map*.
In altri termini, questo processo consiste nel far scorrere il filtro lungo la superficie dell'input, con un certo passo (o *stride*), e per ogni regione analizzata effettuare il prodotto scalare tra i valori del Kernel e quelli della regione di input, per ogni livello di profondità (Figura 2.2).

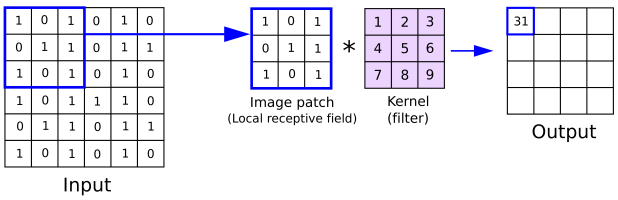



Figura 2.2: Operazione di convoluzione. [12]

Per ogni filtro applicato si produce una feature map: da questo ricaviamo che il volume dell'output prodotto avrà profodità pari al numero di filtri utilizzati. In alcuni casi, si ricorre anche all'uso del *padding*, ovvero l'introduzione di valori nulli all'interno della matrice di input. Questi vengono applicati quando si vuole ottenere un volume di output di una dimensione spaziale precisa.
Prendendo in esempio un input di dimensioni $W_i \times H_i \times D_i$, un numero di filtri di dimensioni F con passo S e si applica un padding P, si ottiene un output di dimensioni $W_o \times H_o \times D_o$, con

$$W_o = \frac{W_i - F + 2P}{S} + 1$$

$$H_o = \frac{H_i - F + 2P}{S} + 1$$

$$D_o = K$$

Notiamo che la profondità del volume di output è uguale al numero di filtri applicati K.

## 2.2 Strato ReLu

Lo strato ReLu (*Rectifier Linear Unit*) è situato dopo quello di convoluzione e viene utilizzato per introdurre una non linearità nel modello. Più precisamente, ReLu è un'unità che applica una funzione di attivazione $f(x) = \max(0, x)$ (dove x è la variabile di input) alla feature map. Questa funzione, come notiamo nella Figura 2.3, ha lo scopo di porre a 0 tutti i valori negativi della mappa e lasciare inalterati tutti gli altri. Questo procedimento non varia le dimensioni del volume.

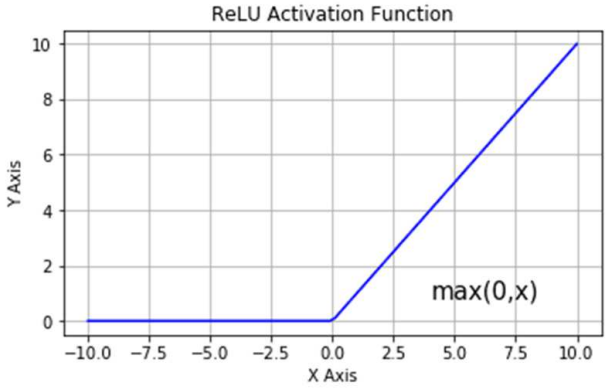



Figura 2.3: Funzione dell'unità ReLu. [13]

## 2.3 Strato di pooling

Il livello di pooling viene utilizzato per ridurre le dimensioni delle *feature map*. In modo molto simile allo strato di convoluzione, l'operazione di pooling applica un filtro su una porzione della mappa e produce un unico output, che dipende dal tipo di filtro applicato. Lo stesso procedimento viene ripetuto per l'intera feature map. Il risultato finale è una matrice di dimensioni minori rispetto a quella iniziale (Figura 2.4). Quest'ultima ha dimensione $W_o \times H_o \times D_o$, dove

$$W_o = \frac{W_i - F}{S} + 1$$

$$H_o = \frac{H_i - F}{S} + 1$$

$$D_o = D_i$$

con $W_i \times H_i \times D_i$ la dimensione della matrice di input.
Questo comporta una perdita di informazioni, ma, d'altra parte, offre una migliore efficienza alla CNN e ne riduce la complessità.
I filtri applicabili possono essere:

- `Max pooling`: estrae il valore più grande nella regione di mappa selezionata;
- `Average pooling`: estrae il valore medio dei termini considerati;
- `L2 pooling`: fornisce la radice della somma dei quadrati dei valori;

Prima di applicare il pooling, bisogna definire la dimensione F della sottomatrice sulla quale viene applicato ed il passo (*stride*) che ci indica lo spostamento del filtro sulla feature map.

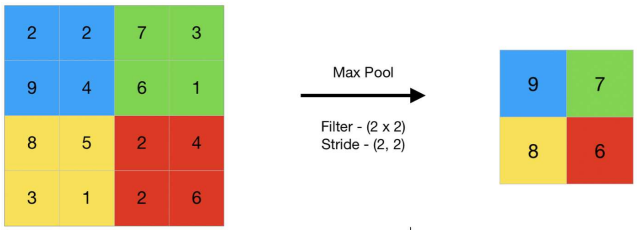



Figura 2.4: Strato di Max pooling applicato alla feature map. Filtro di dimensione 2x2 e passo 2. [14]

## 2.4 Strato completamente connesso

Gli strati completamente connnessi (*Fully connected*) sono disposti generalmente dopo gli strati di convoluzione e di pooling.
Questi "collegano" i loro neuroni con quelli degli strati precedenti: ogni neurone è associato ad un numero di pesi pari al numero di neuroni dello strato precedente e ad un termine bias.
Questo livello esegue l'attività di classificazione in base alle funzioni estratte tramite i livelli precedenti e i loro diversi filtri.

## 2.5 Strato di loss

L'ultimo strato della CNN è quello di *loss*. A questo livello viene definita la metodologia di addestramento dell'intera rete, utilizzano una funzione di attivazione. La più diffusa è chiamata *Softmax*:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e_j^z} \quad \text{per } i = 1, \ldots, K \text{ e } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

Questa funzione prende come input un vettore $z$ di K numeri reali e lo normalizza in una distribuzione di probabilità. Il risultato di tale funzione è un vettore in cui ogni elemento sarà compreso in un intervallo tra (0, 1) e la loro somma sarà uguale a 1, in modo che possano essere interpretati come probabilità. Inoltre, le componenti di input più grandi corrisponderanno a probabilità maggiori.

# 3. Modelli utilizzati

Le due reti che sono state utilizzate in questo studio sono state preaddestrate sulla VGGFace [10] per la VGG16 e sulla VGGFace2 [15] per la ResNet-50.
Non sono stati utilizzati altri modelli in quanto VGG16 e ResNet-50 hanno riportato risultati molto più accurati.

## 3.1 VGGFace

Con il termine VGGFace [10] si fa riferimento ad una serie di modelli per il riconoscimento facciale sviluppati dal Visual Geometry Group dell'Università di Oxford nel 2015. La particolarità di questo studio è stata quella di riuscire a raccogliere una grande quantità di campioni in modo da costruire un dataset molto esteso (2,6 milioni di immagini, 2600 identità).
Questi modelli hanno riportato livelli di accuratezza particolarmente elevati e questo è dovuto, soprattutto, ad una fase di addestramento ben realizzata (grazie anche alle dimensioni del dataset). In seguito, lo stesso gruppo di ricerca ha pubblicato un aggiornamento del VGGFace, chiamato VGGFace2 [15]. Quest'ultimo introduce un aumento dei campioni all'interno del dataset, passando così a 3,31 milioni di immagini con 9131 identità.

### 3.1.1 VGG16

La VGG-16 è un'architettura sviluppata nel 2014 da Karen Simonyan e Andrew Zisserman [2]. Questa rete è costituita da 13 livelli convoluzionali, 5 livelli di MaxPooling, 3 strati completamente connessi ed infine viene applicata una funzione di attivazione softmax (Figura 3.1).
Analizzando meglio la figura sottostante, possiamo notare che la VGG-16 accetta un'immagine con input di dimensione 224x224x3. A questa vengono applicati una serie di strati convoluzionali con filtri 3x3 e passo unitario che hanno il compito di estrarre le caratteristiche principali dell'input. Durante la convoluzione viene applicato un padding in modo tale da preservare la dimensione spaziale. Successivamente, ad ogni strato di convoluzione è presente una funzione ReLu e dopo un certo numero di livelli convoluzionali ne troviamo uno di MaxPooling con una dimensione 2x2 e un passo 2.

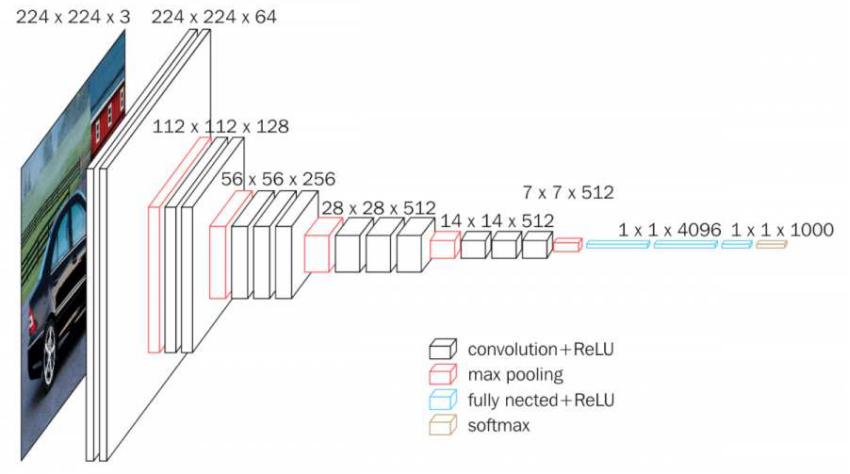



Figura 3.1: Architettura della rete VGG-16. [2]

Per addestrare questa rete viene utilizzata una funzione di *triplet loss* con l'aggiunta dell'algoritmo *SGD* (Stochastic Gradient Descent) per l'ottimizzazione.
La funzione triplet loss ((Figura 3.2) fa in modo che immagini corrispondenti ad una stessa identità, indipendentemente dalle loro condizioni, vengano rappresentate con embedding caratterizzati da una piccola distanza che li separa, mentre le coppie di immagini appartenenti ad identità differenti vengono raffigurate con embedding separati da un maggiore divario. Inoltre, tale funzione impone un margine tra le immagini di una stessa identità, in modo da evitare che esse vengano mappate in uno stesso punto dello spazio euclideo. Supponendo di indicare con $x$ una immagine input e con $f(x) \in \mathbb{R}^d$ l'embedding che rappresenta l'immagine nello spazio euclideo d-dimensionale. Il risultato che si vuole ottenere è che, data un immagine àncora $x_i^a$, tutte le immagini positive $x_i^p$ (ovvero quelle appartenenti alla stessa identità) siano localizzate ad una distanza minore rispetto alle immagini negative $x_i^n$ (quelle appartenenti ad identità differenti). Questo è descritto dalla

$$||f(x_i^a) - f(x_i^p)||_2^2 + \alpha < ||f(x_i^a) - f(x_i^n)||_2^2 \forall f(x_i^a), f(x_i^p), f(x_i^n) \in T$$

dove $\alpha$ è il margine e $T$ è l'insieme di tutti i possibili terzetti di immagini nel set di addestramento. Di conseguenza, la funzione di errore da minimizzare è

$$L = \sum_i^N [||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha]_+$$

In realtà, tra tutti i possibili terzetti di immagini ottenibili da un dataset ve ne sono molti che già in partenza soddisfano la condizione descritta dalla disequazione precedentemente mostrata e che, di conseguenza, contribuiscono poco all'apprendimento da parte della rete, rallentandone la convergenza. Per questa ragione è necessario, al

fine di addestrare velocemente la rete, selezionare i terzetti di immagini che violano la condizione in maniera più netta. Idealmente, fissata un àncora $x_i^a$, bisognerebbe individuare l'immagine positiva $x_i^p$ che massimizzi $||f(x_i^a) - f(x_i^p)||_2^2$ (*hard positive*) e l'immagine negativa che minimizzi $||f(x_i^a) - f(x_i^n)||_2^2$ (*hard negative*). Nella pratica, applicare questa tecnica all'intero dataset di addestramento non è computazionalmente realizzabile. La soluzione individuata è stata quella di selezionare un campione negativo in modo casuale tra quelli che violano la condizione di vicinanza. In questo modo, pur garantendo dei terzetti che contribuiscano all'apprendimento della rete, si riduce la complessità computazionale delle operazioni di ricerca.

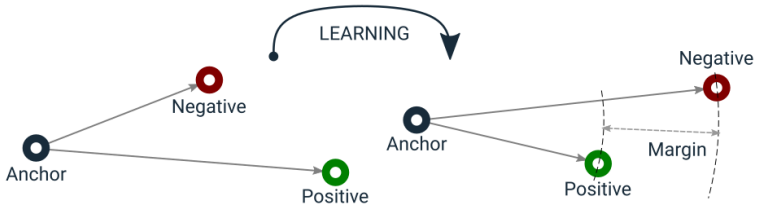



Figura 3.2: Funzione triplet loss. [16]

### 3.1.2 ResNet50

La rete ResNet [3] ha avuto molta importanza nell'epoca della sua pubblicazione. In quei anni le architetture che venivano progettate erano composte da un gran numero di strati in modo da evitare i fenomeni di decadimento del gradiente che si incontravano durante l'addestramento delle reti. Più precisamente, il gradiente è quel componente che determina il tasso di apprendimento di una rete, e più questo ha un valore elevato, più la rete sarà efficiente; viceversa, più questo tenderà a zero, più il sistema avrà difficoltà nell'apprendimento fino a raggiungere un limite di saturazione nel quale smette di apprendere.
Questa discrepanza è stata risolta dagli sviluppatori della ResNet introducendo una tecnica di apprendimento residuale: piuttosto che far apprendere ad un certo numero di strati una funzione $H(x)$, si cerca di trovare il suo residuale $F(x) = H(x) - x$, cosicchè la funzione originale diventi $H(x) = F(x) + x$. Questa operazione viene ottenuta dalle cosiddette *shortcut connection* (anche dette *skip connection*), ovvero connessioni che permettono di collegare i strati precedenti di una rete a quelli successivi, consentendo alle informazioni di bypassare uno o più livelli. La cosa importante è che l'output risultante non presenta parametri aggiuntivi e quindi la complessità computazionale della rete non aumenta. Il blocco così composto prende il nome di *blocco residuale* (Figura 3.3).
La situazione descritta sopra è dettata dal fatto che $F(x)$ e $x$ hanno la stessa dimensione. Non sempre questo accade e allora occorre applicare una proiezione nello shortcut del tipo $y = F(x, W_i) + W_s x$, dove $W_s$ è una matrice quadrata.

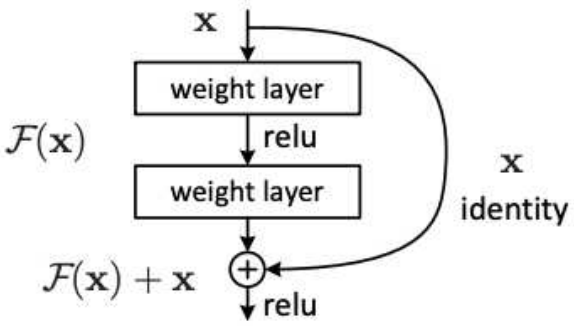



Figura 3.3: Blocco residuale. [3]

Esistono varie tipologie di ResNet: quella utilizzata nel nostro studio è la ResNet-50 in cui sono presenti 50 strati (48 strati convoluzionali, 1 di MaxPooling e 1 di AveragePooling) (Figura 3.4).
Gli strati convoluzionali sono composti da filtri 3x3 che seguono due linee conduttive: se le feature map prodotte per ogni livello hanno le stesse dimensioni, ogni strato avrà lo stesso numero di filtri. Al contrario, se le feature map hanno dimensioni dimezzate ogni strato avrà il doppio dei filtri. Dopo i blocchi convoluzionali sono presenti i blocchi identità che rappresentano i sistemi residuali descritti in precedenza.
I blocchi identità vengono utilizzati quando la dimensione dell'input è uguale a quella dell'output e, di conseguenza, i due possono essere sommati. Quando i due hanno dimensione diversa si adottano due strade: la prima riguarda l'introduzione di zeri all'interno dell'input per aumentarne le dimensioni (questa opzione non introduce parametri aggiuntivi), la seconda consiste nell'utilizzo della proiezione lineare vista nell'equazione sopra (questa viene fatta da una convoluzione 1x1).

### 3.1.3 SE-ResNet50

L'architettura SE-ResNet50 (*Squeeze and Excitation-ResNet50*) si basa sulla rete ResNet50 vista in precedenza. In questo modello, però, vengono aggiunti dei blocchi SE (*Squeeze and Excitation*).
In un tipico strato convoluzionale di una CNN, i pesi per ciascun canale sono uniformi quando si calcola l'output, trattandoli così tutti allo stesso modo e con la stessa importanza. Il blocco SE introduce un approccio adattivo in cui l'importanza di ciascun canale viene valutata individualmente in base al suo contesto. In termini più semplici, il blocco SE tiene conto della rilevanza di ciascun canale nel calcolo dell'output.
Il blocco SE è composto da due fasi: la fase di Squeeze (*compressione*) consiste nella compressione delle feature map prodotte da una convoluzione riducendole ad un singolo valore numerico. In questo modo si ottiene un vettore di n elementi, dove n è il numero di canali dello strato di convoluzione. Nella fase di Excitation (*eccitazione*),

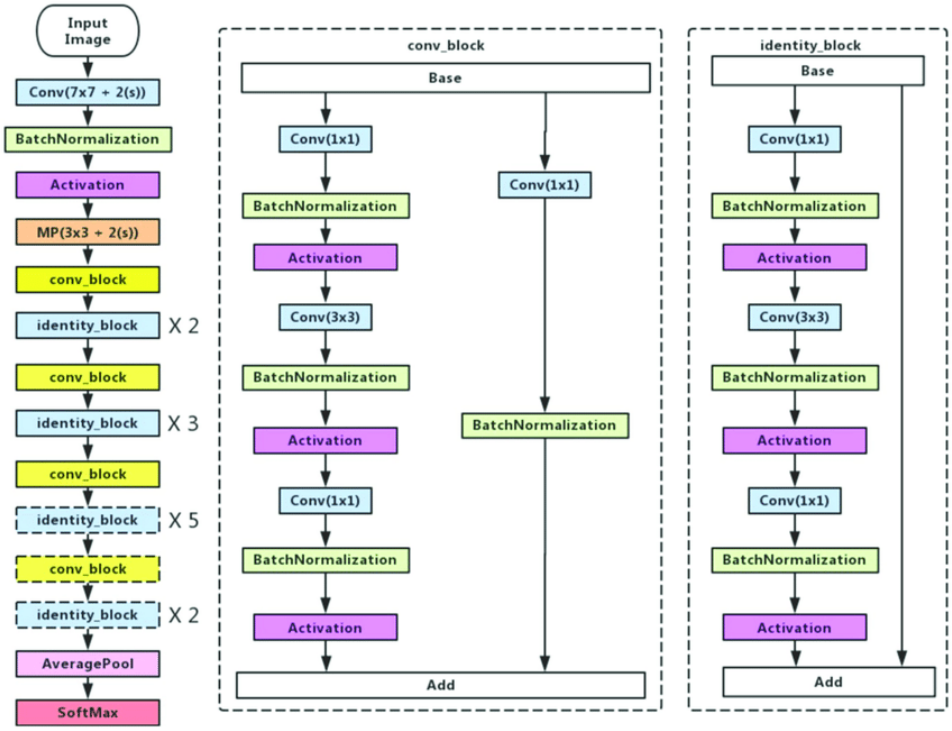



Figura 3.4: Architettura ResNet-50. [3]

si applica tale vettore a due strati completamente connessi, ottenendo come risultato un vettore di uguale dimensione, il quale viene utilizzato come collezione di pesi da applicare alla feature map originaria.
In particolare, un blocco SE si forma a partire da una trasformazione $F_{tr}$ che mappa l'input $X \in \mathbb{R}^{H' \times W' \times C'}$ nella future map $U \in \mathbb{R}^{H \times W \times C}$. Considerando $F_{tr}$ un'operazione di convoluzione e sia $V = [v_1, v_2, v_3, ..., v_C]$ un vettore che definisce l'insieme di filtri, l'output prodotto è $U = [u_1, u_2, u_3, ..., u_C]$, dove il c-esimo valore è calcolato come:

$$u_c = v_c * X = \sum_{s=1}^{C'} v_c^s * x^s$$

Con * indichiamo l'operazione di convoluzione, $v_c = [v_c^1, v_c^2, v_c^3, ..., v_c^{C'}]$, $X = [x^1, x^2, x^3, ..., x^{C'}]$ e $u_c \in \mathbb{R}^{H \times W}$. Non sono presenti i termini bias perchè si vuole semplificare la notazione.
Nella fase di Squeeze la compressione dei C canali della feature map è realizzata tramite Average-Pooling. Il vettore di descrittori $z \in \mathbb{R}^C$ è generato contraendo $U$ nelle sue dimensioni spaziali H e W in modo tale che il suo c-esimo elemento sia calcolato come:

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{K} u_c(i, j)$$

Successivamente, nella fase di Excitation, viene aggiunta una non linearità grazie ad uno strato completamente connesso, seguito da una funzione ReLu $\delta$ ed infine viene

unito un altro strato completamente connesso seguito da una funzione di attivazione Sigmoide:

$$s = F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2\delta(W_1 z)$$

con $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}$ e $W_1 \in \mathbb{R}^{C \times \frac{C}{r}}$. Più precisamente, il primo di questi realizza una riduzione dimensionale basata sul fattore di riduzione r (un possibile valore può essere $r = 16$), mentre il secondo opera un incremento dimensionale. L'output del blocco $\tilde{X} = [\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, ..., \tilde{v}_C]$ è calcolato scalando il vettore U con s, il c-esimo elemento è calcolato come:

$$\tilde{x}_c = F_{scale}(u_c, s_c) = s_c u_c$$

Come vediamo in Figura 3.5, il blocco SE viene inserito prima di ogni blocco residuale e questo ha permesso la creazione dell'architettura SE-ResNet50.

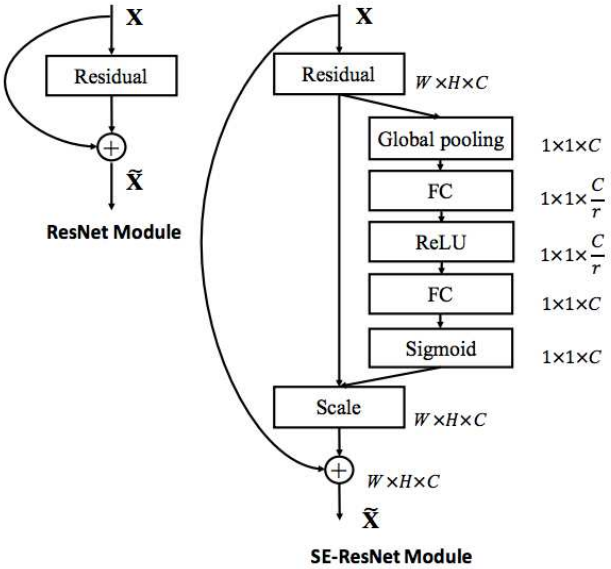



Figura 3.5: Blocco SE a confronto con il blocco residuale. [4]

# 4. Face detection

La Face detection è una tecnica che permette di identificare e localizzare i volti umani presenti all'interno delle immagini e dei video grazie all'utilizzo di algoritmi di intelligenza artificiale.
Per questo studio sono stati utilizzati due algoritmi di face detection chiamati *MTCNN* e *Viola and Jones* (anche detto *Haar Cascades*)

## 4.1 MTCNN

L' MTCNN(*Multi-Task Cascaded Convolutional Neural Networks*) è costituito da tre reti neurali (P-Net, R-Net e O-Net) collegate in cascata (Figura 4.1).

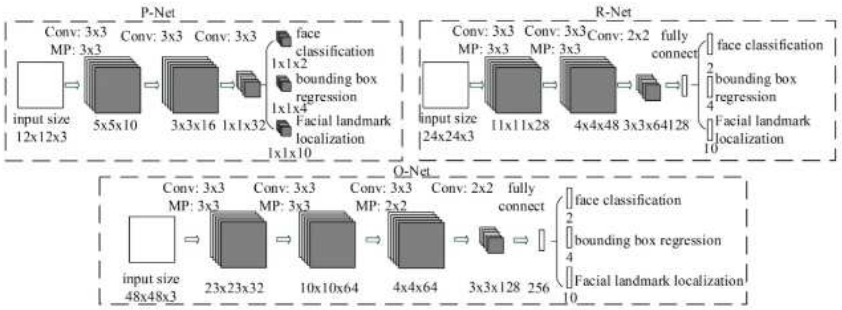



Figura 4.1: Struttura del MTCNN. [17]

La prima operazione eseguita sull'input è quella di costruire una piramide d'immagine, ovvero un'insieme di copie di varie dimensioni della stessa immagine. Questo viene fatto principalmente per rilevare volti di diverse dimensioni all'interno della stessa figura (Figura 4.2).
Una volta ottenuta la piramide ogni copia viene scansionata, partendo dall'angolo in alto a sinistra fino all'angolo in basso a destra, tramite un filtro 12x12 con passo 2 (il passo mi definisce di quanti pixel il filtro si muove ogni volta).

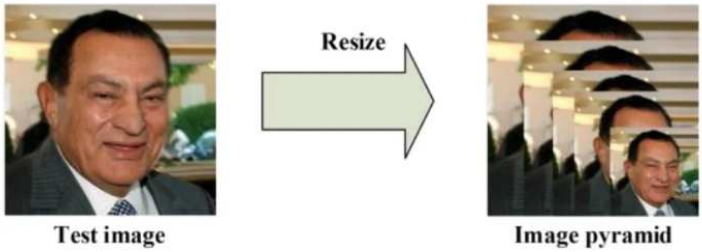



Figura 4.2: Prima fase dell'MTCNN. Formazione della piramide d'immagine [17]

Ciascuna delle porzioni 12x12 di ogni copia viene inviata alla prima rete neurale, la P-Net.
Questa particolare rete ha il compito di produrre un rettangolo (o bounding-box) attorno ad un probabile volto e un livello di confidenza ad esso associato. Se il livello di confidenza di quel specifico rettangolo è basso, allora viene eliminato. Alla fine si otterranno un certo numero di rettangoli di selezione con le rispettive coordinate relative all'immagine originale (Figura 4.3).

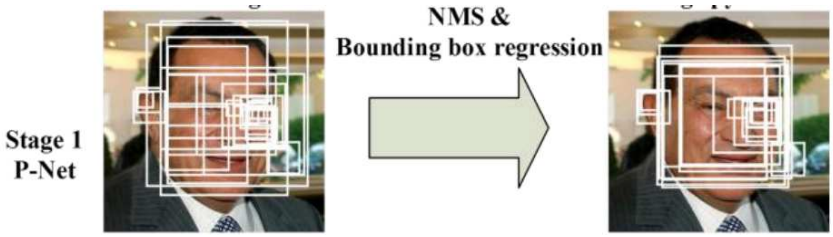



Figura 4.3: Rete neurale P-Net. L'immagine sulla destra è l'output della P-Net [17]

Può capitare di avere dei bounding-box sovrapposti ad altri e per ridurli viene utilizzata una tecnica chiamata *NMS* (Non-Maximum Suppression).
Come prima cosa, l'NMS, ordina tutti i rettangoli in base al livello di confidenza: successivamente, viene calcolata l'area di ciascun rettangolo, nonchè l'area di sovrapposizione tra ciascun rettangolo e quello con la confidenza più alta. I bounding-box che presentano un'area molto simile a quella del bounding-box "master" vengono eliminati (Figura 4.4).
Detto ciò, ogni rettangolo rimasto viene ridimensionato ad una dimensione 24x24, normalizzato a valori di -1 e 1 ed infine inviato alla R-Net.

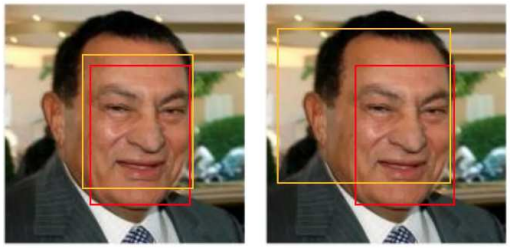


Figura 4.4: Funzionamento dell'NMS. Nell'immagine di sinistra il riquadro giallo viene eliminato perchè la sua area coincide molto con quello rosso. Mentre nella seconda immagine i due rimangono perchè hanno aree diverse. [17]

L'output della rete R-Net (Refine Network) è uguale a quello della P-Net. Infatti, nel secondo stadio si applica lo stesso procedimento visto nel primo, ovvero si vanno ad eliminare tutti i bounding-box con un basso livello di confidenza, si applica l'NMS e i delimitatori rimasti vengono ridimensionati ad una dimensione 48x48.
Nel risultato finale, si otterranno dei rettangoli con un livello di confidenza maggiore rispetto a quelli del primo stadio (Figura 4.5). Questi verranno inviati all'ultima rete neurale, la O-Net.

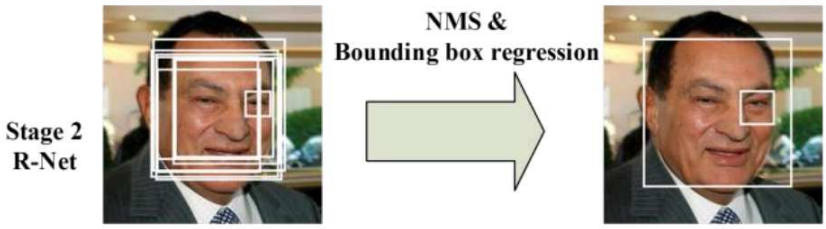



Figura 4.5: Output R-Net. [17]

L'obbiettivo della O-Net (Output Network) è quello di ripetere ancora una volta l'operazione di rifinitura dei bounding-box. Una volta fatto ciò, si otterrà un delimitatore per ogni viso presente nell'immagine ed, inoltre, ad ogni rettangolo vengono associati cinque punti chiave: due per gli occhi, due per la bocca ed uno per il naso (Figura 4.6).

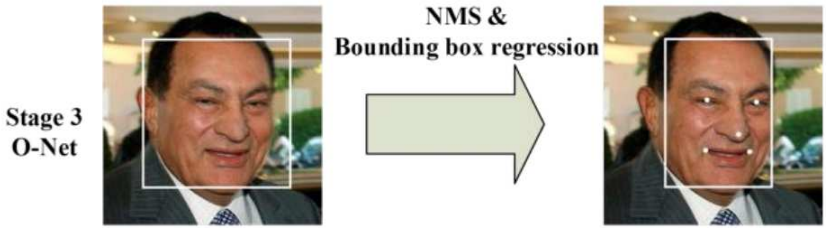



Figura 4.6: Output O-Net. I cinque punti chiave possono essere utilizzati per operazioni di allineamento. [17]

## 4.2 Viola-Jones framework

L'algoritmo *Viola-Jones*, anche detto Haar Cascade, viene utilizzato per la face detection. All'interno di questa tesi viene impiegato assieme all'MTCNN per avere un livello di accuratezza maggiore.
L'Haar Cascade introduce dei filtri (Figura 4.7) per la ricerca di bordi e linee all'interno dell'immagine o per la selezione di aree in cui si verifica un improvviso cambiamento nell'intensità dei pixel.

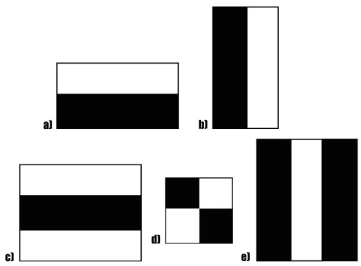


Figura 4.7: Filtri dell'Haar Cascade. Ogni filtro ha il compito di estrapolare caratteristiche differenti dall'immagine. [18]

Questi filtri sono costituiti da parti scure, in cui tutti i pixel sono a 1, e parti chiare, in cui tutti i pixel sono a 0. Questi vengono fatti scorrere su tutta l'immagine, partendo dallo spigolo in alto a sinistra ed arrivando allo spigolo in basso a destra, andando ad estrapolare tutte le caratteristiche di quest'ultima e trovando infine i volti.
Gli autori si sono accorti che non era necessario applicare tutti i filtri per ogni porzione di immagine che si andava ad analizzare in quanto questo rendeva l'algoritmo molto costoso dal punto di vista computazionale. A questo proposito hanno idealizzato una tecnica chiamata *Attentional Cascade*, dove l'idea alla base è il non utilizzo di tutti i filtri quando si esamina la porzione dell'immagine. Se un filtro non trova nessuna caratteristica su quella zona, allora quest'ultima viene automaticamente scartata e si passa alla sezione successiva.

# 5. Classificazione

Una volta estratti gli embedding si possono utilizzare per il processo di identificazione (o classificazione), ovvero quell'operazione che, dato un certo numero di ingressi (in questo caso gli embedding), mi permette di distribuirli in più classi. Come risultato si ottengono dei gruppi aventi all'interno soggetti con le stesse caratteristiche.
In questa tesi andiamo ad esaminare due algoritmi classificatori molto utilizzati noti come l'SVM (Support Vector Machine) e il k-NN (K-Nearest Neighbors).

## 5.1 SVM

L'SVM (*Support Vector Machine*) è un algoritmo di apprendimento supervisionato che può essere utilizzato sia per la classificazione sia per la regressione. Questo algoritmo è molto efficace per la classificazione dei problemi binari.
Il Support Vector Machine si basa sull'idea di trovare un iperpiano che divida al meglio un set di dati che forniamo in ingresso in due classi. Il migliore iperpiano trovato sarà quello che ha il margine tra le due classi maggiore (Figura 5.1).
Come possiamo notare dalla figura 4.1, i campioni più vicini all'iperpiano prendono il nome di *vettori di supporto* ed è da essi che dipende la posizione dell'iperpiano.

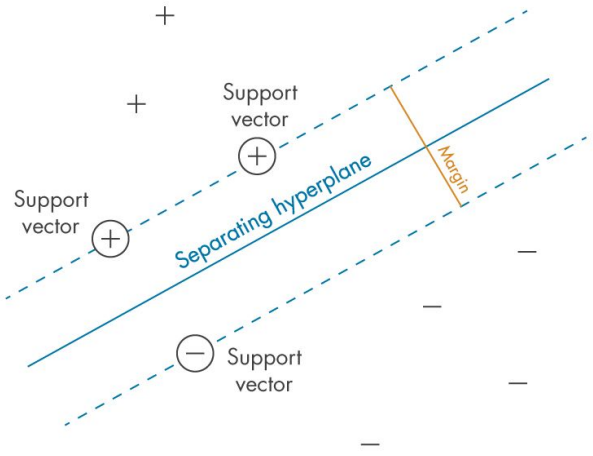



Figura 5.1: Rappresentazione iperpiano. [19]

Per la classificazione di un nuovo campione si andrà a vedere la posizione in cui esso sarà rispetto all'iperpiano: in base a quest'ultima, il campione verrà assegnato ad una delle due classi.
Tuttavia, possono verificarsi casi in cui i dati non sono separabili linearmente e, a questo proposito, ci si può aiutare con il `"kernel trick"`, o trucco del kernel. Questa tecnica consiste nel passare ad uno spazio dimensionale superiore per andare ad eseguire la suddivisione. Questo è reso possibile grazie alla definizione di una funzione kernel, che mappa i vettori dallo spazio originale a quello superiore (Figura 5.2).

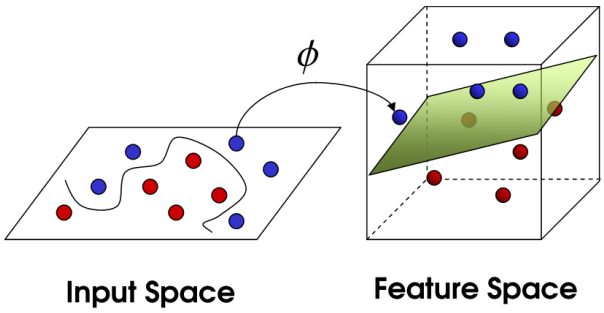



Figura 5.2: Rappresentazione del kernel trick. [20]

## 5.2 k-NN

Il k-NN (*k-Nearest-Neighbors*) è un classificatore di apprendimento supervisionato non parametrico. Come per l'SVM, anche lui è utilizzato in problemi di classificazione e regressione.
Dato un insieme di campioni rappresentati come vettori in uno spazio multi-dimensionale, l'algoritmo prevede, nella fase iniziale, la sola memorizzazione di tali vettori unitamente alle rispettive etichette di classe. Un nuovo vettore viene classificato semplicemente assegnandolo alla classe che presenta il maggior numero di occorrenze tra i k campioni di addestramento più vicini ad esso (Figura 5.3).
Il valore k definisce quanti "vicini" verranno controllati per determinare la classificazione del nuovo campione. Ad esempio, se k=1 la classificazione verrà fatta prendendo la distanza (nella maggior parte dei casi è quella Euclidea) di un singolo campione per classe.

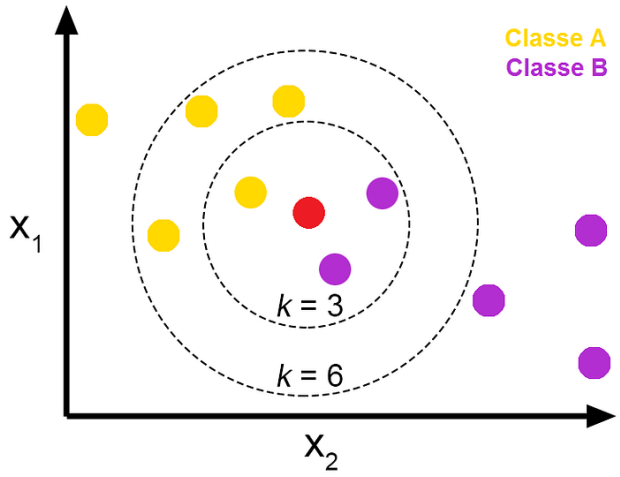



Figura 5.3: Algoritmo k-NN. Nell'esempio sono stati presi valori per k=3 e k=6. Nel primo caso il nuovo campione viene assegnato nella classe viola, mentre nel secondo caso a quella gialla. [21]

# 6. The Face Recognition from Mugshots Database (FRMDB)

Il Face Recognition from Mugshots Database [1] è un database sviluppato dall'Università Politecnica delle Marche appositamente per questo progetto di ricerca. Al suo interno troviamo 39 soggetti, 17 femmine e 22 maschi, di diverse età. Per ogni individuo abbiamo:

- 28 mugshots, presi in diversi punti. Ogni mugshot ha una risoluzione di 972x544.
- 5 video, presi da 5 posizioni differenti. I video sono codificati in H.264 e con un frame rate pari a 60fps. I frame presi dai video hanno una risoluzione di 352x288.

Come abbiamo accennato precedentemente, per ogni soggetto abbiamo un totale di 28 mugshots. In particolare, sono state scattate foto da 7 angolazioni sul piano orizzontale e 4 angolazioni sul piano verticale (Figura 6.1) utilizzando un braccio robotico con 4 fotocamere, che ruotava attorno alla persona.

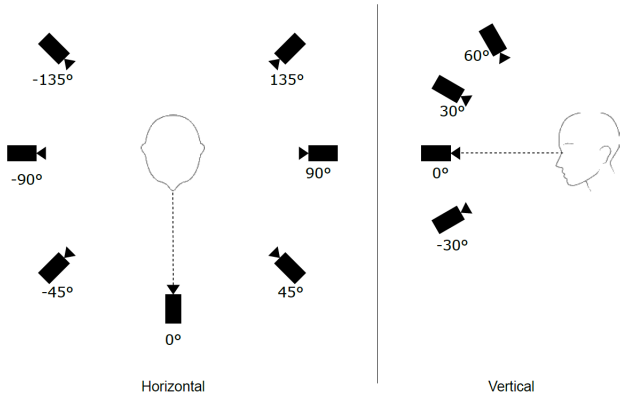



Figura 6.1: Schema acquisizione mugshot. [1]

Come si nota in figura (Figura 6.1), sul piano orizzontale si parte da un angolo di $+135°$ e si arriva a $-135°$, con step di $45°$, mentre sul piano verticale abbiamo un range che va da $-30°$ a $+60°$, con l'angolo $0°$ posto in corrispondenza degli occhi del soggetto.

Ogni immagine acquisita viene nominata nel seguente modo: `Img_XY.jpg`; dove la X indica l'angolo sul piano orizzontale (da 0 = +135° a 6 = −135°) e la Y indica l'angolo sul piano verticale (da 1 = +60° a 4 = −30°). (Figura 6.2)


(a) (b)

Figura 6.2: Esempio mugshot. [1]

La struttura del database è molto semplice (Figura 6.3): abbiamo due directory `original` e `cropped`. All'interno della cartella original troviamo diverse sottocartelle, una per ogni soggetto (39 in totale). Più precisamente ogni directory include:

- una cartella con tutti i 28 mugshots del soggetto;
- una cartella contenente i 5 video si sorveglianza;
- una cartella con i 5 frame presi dai video;
- un immagine che rappresenta il mugshot frontale;
- un file di testo dove sono indicate tutte le informazioni della persona (etá, sesso e se indossa occhiali).

Mentre nella cartella cropped troviamo due sottocartelle:

- `mugshots` suddivisa ulteriormente in 39 directory, una per ogni soggetto. In ciascuna cartella troviamo i 28 mugshots croppati;
- `surveillance` anch'essa suddivisa in 39 directory, dove al loro interno troveremo i 5 frame croppati.

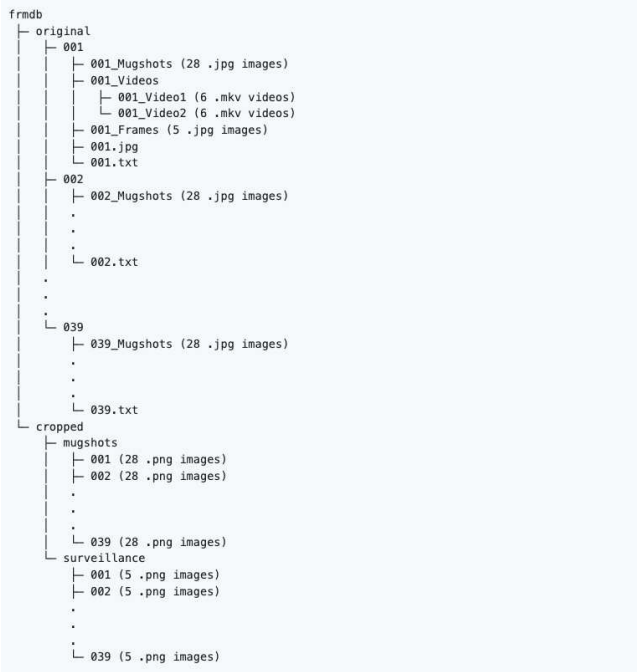



Figura 6.3: Struttura database. [1]

Oltre a questi 39 campioni, per questo studio, sono stati presi altri 28 nuovi soggetti. In particolare, questi ultimi sono strutturati nello stesso modo dei vecchi, con qualche piccola differenza. Ovvero per ogni soggetto sono presenti 28 mugshots e 3 video di sorveglianza presi in differenti angolazioni. Questi video hanno una qualità molto maggiore (1920x1080) rispetto a quelli dei 39 campioni e questo ha permesso di avere dei risultati migliori. Come vedremo nel prossimo capitolo, sono stati studiati i 28 nuovi campioni ed in seguito il database al completo, con un totale di 67 soggetti.

# 7. Test effettuati

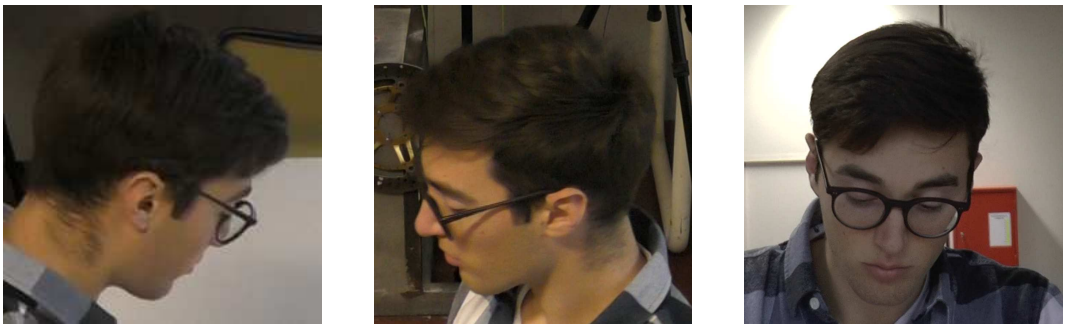
Per testare le performance dei tre modelli utilizzati per il riconoscimento facciale sono stati comparati gli embedding di ogni soggetto, presi dai video di sorveglianza, con i vari mugshots di ogni identità. Questo processo è stato fatto utilizzando la distanza euclidea.
Sono stati eseguiti 8 differenti test prendendo differenti configurazioni:

- Test F, usando il mugshot frontale.
- Test F-L1-R1, usando il mugshot frontale più il mugshot leggermente inclinato a destra e a sinistra.
- Test 1, usando il mugshot frontale più quello del profilo destro.
- Test 2, usando il mugshot frontale più il mugshot di profilo destro e sinistro.
- Test 3, unendo i test F-L1-R1 e il test 2.
- Test 4, si utilizza il test 3 con l'aggiunta delle immagini con angolazioni più elevate sia a destra che a sinistra.
- Test 5, viene aggiunto al test 4 le immagini con angolazioni a 30°.
- Test 6, si utilizza tutti i 28 mugshots di ogni soggetto.

| Test | Mugshots |
|---|---|
| Test F | (0°, 0°) |
| Test F-L1-R1 | (0°, 0°), (-45°, 0°), (45°, 0°) |
| Test 1 | (0°, 0°), (90°, 0°) |
| Test 2 | (0°, 0°), (90°, 0°), (-90°, 0°) |
| Test 3 | (0°, 0°), (90°, 0°), (-90°, 0°), (45°, 0°), (45°, 0°) |
| Test 4 | (0°, 0°), (135°, 0°), (-135°, 0°), (90°, 0°), (-90°, 0°), (45°, 0°), (45°, 0°) |
| Test 5 | (0°, 0°), (135°, 0°), (-135°, 0°), (90°, 0°), (-90°, 0°), (45°, 0°), (45°, 0°), (0°, 30°), (135°, 30°), (-135°, 30°), (90°, 30°), (-90°, 30°), (45°, 30°), (45°, 30°) |
| Test 6 | (0°, 0°), (135°, 0°), (-135°, 0°), (90°, 0°), (-90°, 0°), (45°, 0°), (45°, 0°), (0°, 30°), (135°, 30°), (-135°, 30°), (90°, 30°), (-90°, 30°), (45°, 30°), (45°, 30°), (0°, 60°), (135°, 60°), (-135°, 60°), (90°, 60°), (-90°, 60°), (45°, 60°), (45°, 60°), (0°, -30°), (135°, -30°), (-135°, -30°), (90°, -30°), (-90°, -30°), (45°, -30°), (45°, -30°) |

Figura 7.1: Angolazioni delle immagini per ogni test. [1]

In particolare, per ogni test è stata valutata la capacità delle reti VGG16, ResNet50 e Se-ResNet50 di riconoscere un determinato soggetto. Per analizzare i risultati di tali test sono state utilizzate due metriche: la prima comprendeva la percentuale di soggetti correttamente individuati nelle top-1, top-3, top-5 e top-10 foto segnaletiche e la seconda rappresentava la percentuale di soggetti correttamente individuati nelle top-1, top-3, top-5 e top-10 identità (ex. se prendiamo la top-1 si intende la capacità della rete di riconoscere il soggetto al primo tentativo).
Lo scopo di questo studio è stata quella di effettuare i test descritti in precedenza su un nuovo set di campioni: più precisamente, all'interno del dataset FRMDB erano presenti un totale di 39 campioni a cui sono stati aggiunti altri 28 soggetti. La differenza sta nel fatto che i vecchi campioni avevano 5 video di sorveglianza per ogni individuo, girati in bassa risoluzione, mentre i nuovi soggetti hanno 3 video con un risoluzione maggiore (1920x1080). Il numero di mugshots acquisiti invece, rimane invariato, ovvero 28. Per effettuare i test i video di sorveglianza sono stati manualmente croppatti in modo da ottenere cinque immagini con differenti angolazioni (Figura 7.2).
Come vedremo in seguito, il fatto di avere dei video girati in alta qualità ha prodotto risultati con un indice di accuratezza migliore.
Dopo aver tratto le varie conclusioni sulla rete con il miglior esito, ho accorpato i nuovi campioni con il database già esistente e ho effettuato gli stessi test sull'intero dataset.



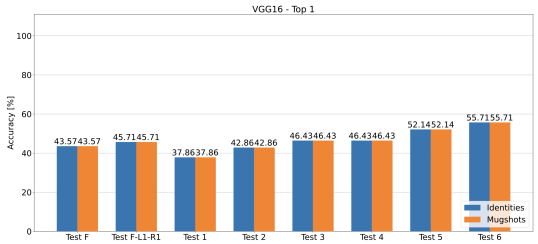
(a) (b) (c)

(d) (e)

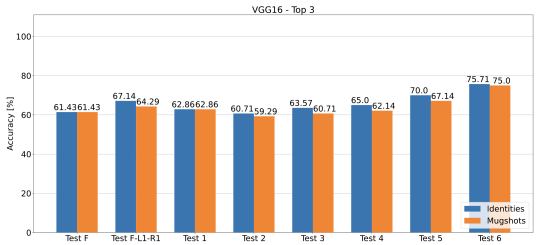
Figura 7.2: Le 5 foto raccolta dei video di sorveglianza.

## 7.1 Risultati

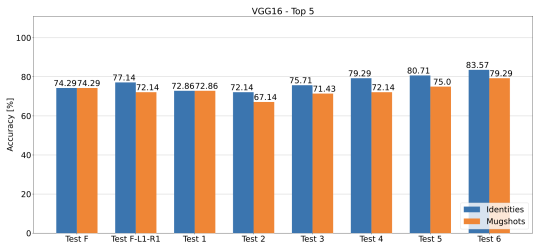
La prima analisi è stata condotta sul dataset composto dai campioni più recenti utilizzando i modelli VGG16, ResNet50 e SE-ResNet50. Ricordiamo che tutti e tre i modelli accettano in input immagini aventi una risoluzione di 224x224, di conseguenza è stato fatto un cropping che ha portato ad abbassare la qualità di queste ultime. Per eseguire il riconoscimento facciale e verificare l'accuratezza prodotta sono stati confrontati gli embeddings generati dai video di sorveglianza con i sottoinsiemi di mugshots visti nel Capitolo 6.



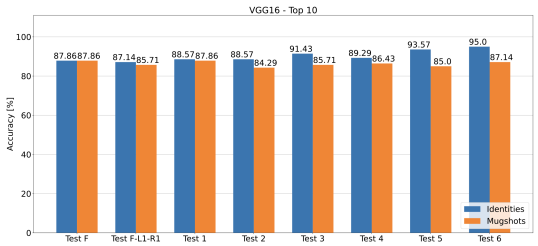

(a)




(b)

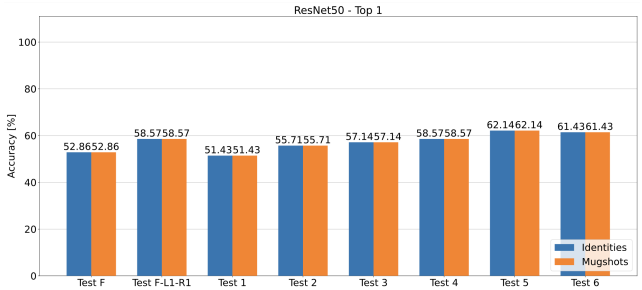



(c)

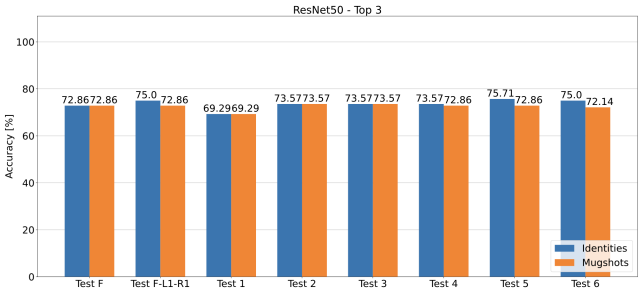



(d)

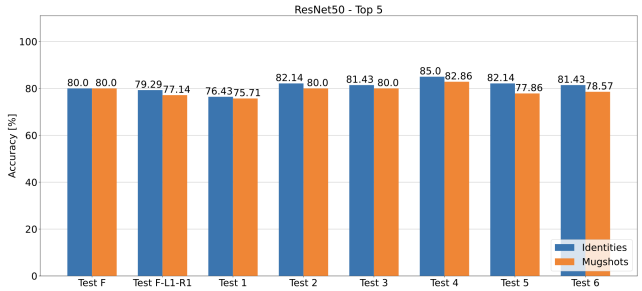
Figura 7.3: Risultati test con rete VGG16. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).



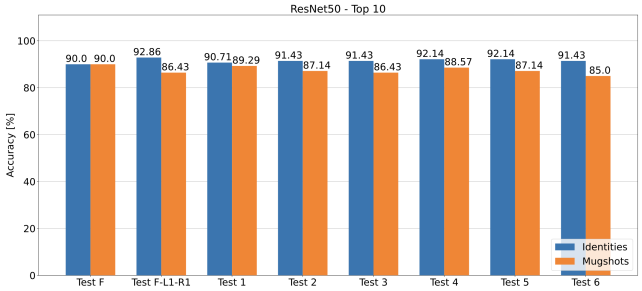

(a)




(b)

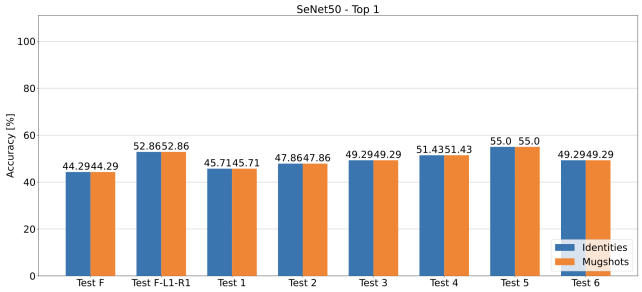



(c)

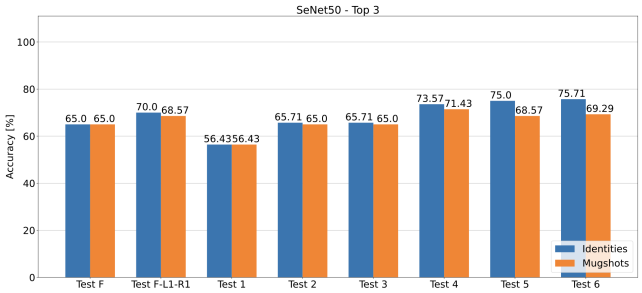



(d)

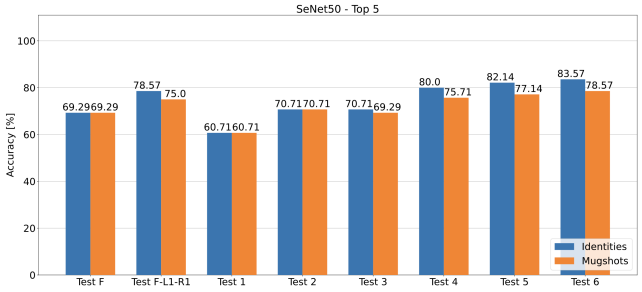
Figura 7.4: Risultati test con rete ResNet50. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).



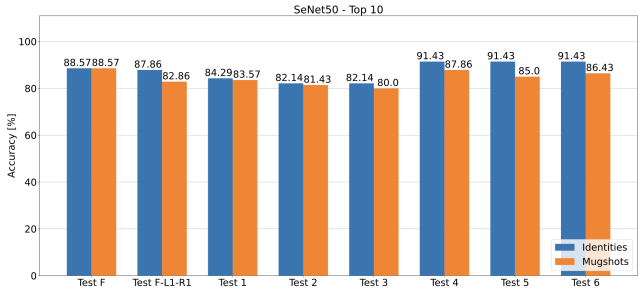

(a)




(b)



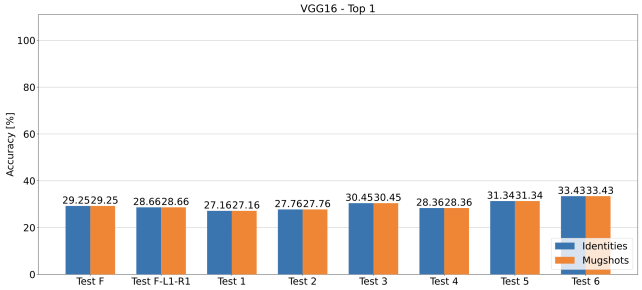

(c)



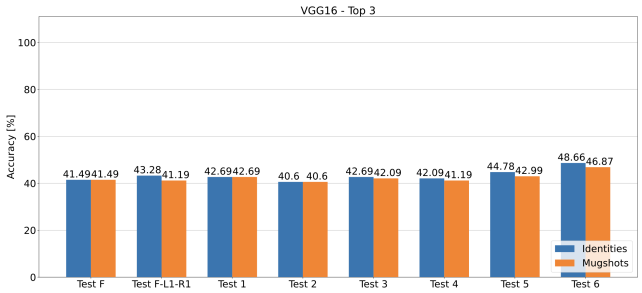

(d)

Figura 7.5: Risultati test con rete SE-ResNet50. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).

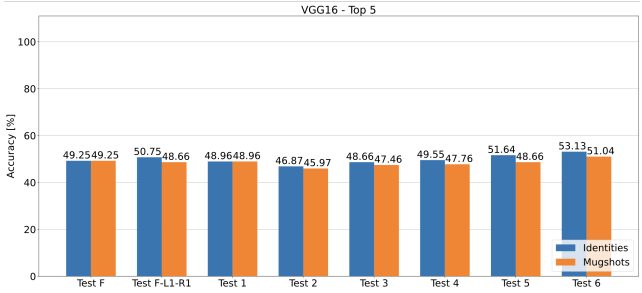
Il primo esito rilevabile è l'ottenimento di risultato quando si hanno sottoinsiemi composti da più mugshots, ovvero tra tutti i test si può vedere come i "Test-5" e "Test-6" producano risultati migliori per entrambe le due metriche di giudizio. Se prendiamo il "Test-6", ovvero quello che comprende tutti i 28 mugshots per ogni soggetto, si registra il risultato più alto con la rete VGG16, nonostante quest'ultima abbia un livello di accuratezza minore in tutti gli altri test, rispetto al ResNet50.
In quanto al "Test-1" (mugshot frontale + profilo destro), che indica il set attualmente usato dalla Polizia di Stato per la procedura di fotosegnalamento, si può evidenziare come la SE-ResNet50 produca un risultato peggiore nella Top-10 e nella Top-3 e la VGG16 abbia un indice di accuratezza basso nella Top-1 e nella Top-5. La ResNet-50 ha un trend in linea con la VGG16, ovvero si hanno i peggiori risultati nella Top-1 e nella Top-5. Tuttavia, è quella che registra la miglior accuratezza delle tre nella Top 10 (90% circa per entrambe le metriche). Per riassumere, avendo dei video con una qualità maggiore, i singoli frame croppati riescono a fornire maggiori dettagli. Notiamo dai test che siamo vicini ad un'accuratezza del 100% per quanto riguarda i risultati Top-10. Per quanto riguarda la Top-1, come si pensava, l'accuratezza sta attorno al 40-50%, che risulta essere superiore ai risultati Top-1 del vecchio database. Ora possiamo passare al dataset completo e vedere cosa le reti neurali hanno prodotto.



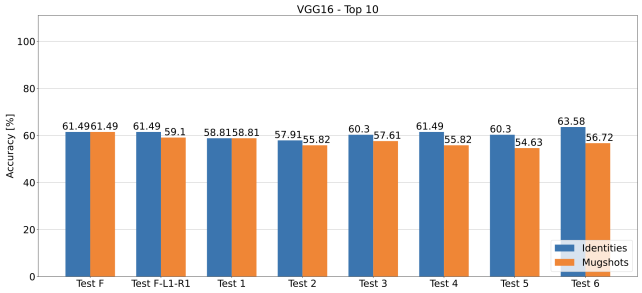

(a)




(b)

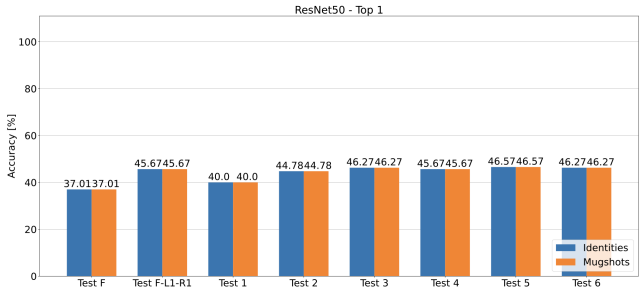



(c)

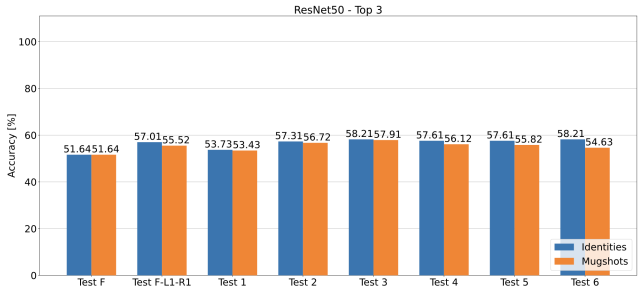



(d)

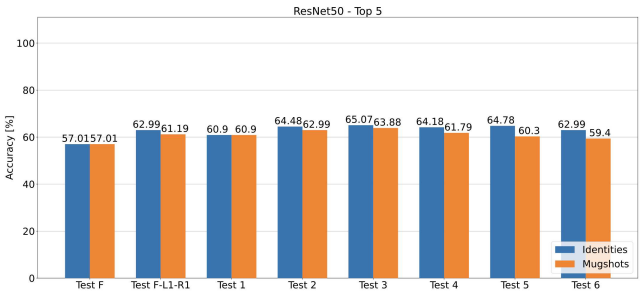
Figura 7.6: Risultati test con rete VGG16 usando l'intero dataset. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).



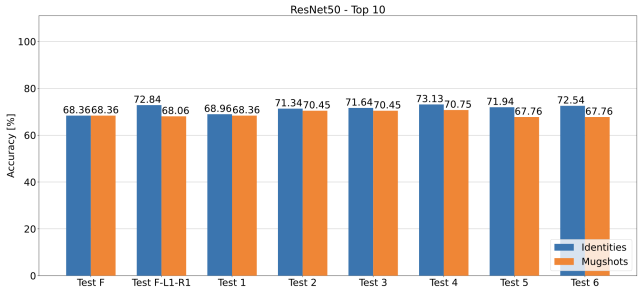

(a)




(b)

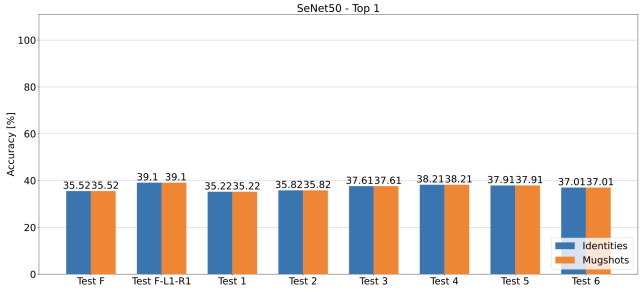



(c)

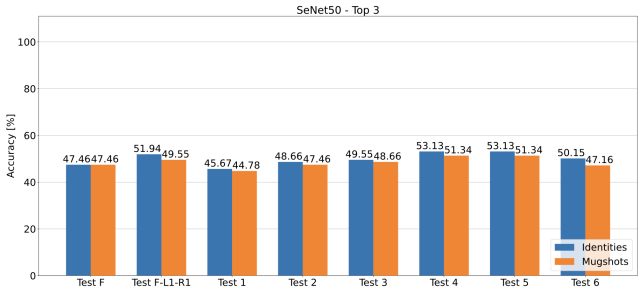



(d)

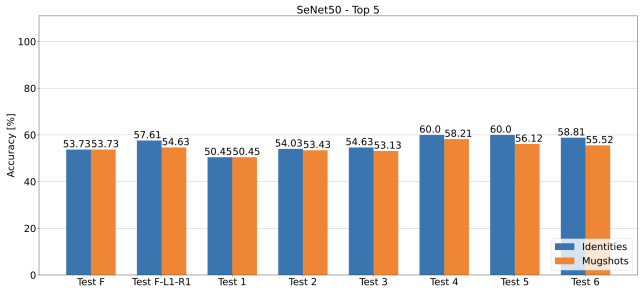
Figura 7.7: Risultati test con rete ResNet50 usando l'intero dataset. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).



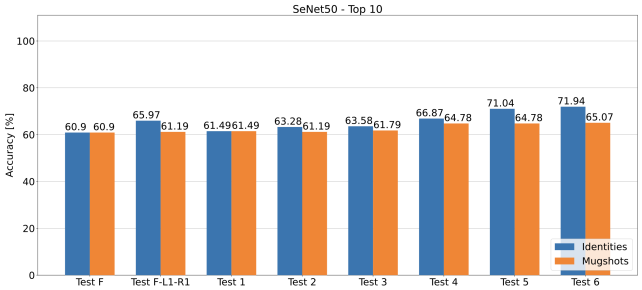

(a)




(b)




(c)




(d)

Figura 7.8: Risultati test con rete SE-ResNet50 usando l'intero dataset. Top-1 (a), Top-3 (b), Top-5 (c) e Top-10 (c).

Come possiamo notare dai grafici sopra esposti l'accuratezza prodotta da tutte e tre le reti è decisamente più bassa rispetto ai risultati visti utilizzando solo i nuovi campioni. Ciò si verifica perchè all'interno del dataset sono presenti 39 campioni aventi i video di sorveglianza con una bassa risoluzione.
Osserviamo che la VGG16 fornisce dei risultati peggiori in tutti i test presi in considerazione. È presente una netta differenza di accuratezza tra la VGG16 e la ResNet50. Paragonando i grafici delle due reti possiamo riscontrare una differenza del 10%-15% per ciascun test. Prendendo in esempio il Test-6 della Top-10 vediamo che la VGG16 ha fornito un 63.58%, mentre la ResNet50 ha un 72.54%: la Se-ResNet50 è in linea con i risultati della ResNet50, anche se presenta un'accuratezza leggermente inferiore.
Come accadeva nei test dei nuovi campioni, anche in questi abbiamo riscontrato che il Test-6 presenta una miglior accuratezza tra tutti i risultati. Un' interessante caratteristica che possiamo rilevare è la minima differenza di accuratezza tra il Test-6 e il Test-5 in tutte e tre le reti, nonostante nell'ultimo test si usino 28 mugshots rispetto ai 14 del Test-5. Si verifica quindi un leggero incremento dell'accuratezza con il doppio dei mugshots, il che rende poco conveniente un aumento dei mugshots per una differenza così minima. Se prendiamo la Top-5 della ResNet50 si ha un'accuratezza del 62.99% in termini di identità più vicine e 59.4% in termini di mugshots più vicini per il Test-6, mentre un 64.78% e 60.3% per il Test-5.
Per riassumere, un aumento dei campioni ha permesso di avere un accuratezza migliore rispetto ai test che si erano fatti in passato. Detto questo, il dataset al completo presenta ancora poche identità rispetto ai dataset oggi utilizzati. Tuttavia, l'obbiettivo di questo studio è quello di valutare le tecniche di riconoscimento facciale utilizzando dei mugshots su diverse angolazioni. Pertanto, il FRMDB è più adatto a scopi di test piuttosto che scopi di addestramento delle reti.

# 8. Conclusioni

In questa tesi sono state valutate le performance delle reti neurali per il riconoscimento facciale. In particolare sono stati utilizzati tre modelli: ResNet50, SeNet50 e VGG16. Il database su cui sono stati eseguiti questi test è stato sviluppato dall'Università Politecnica delle Marche, chiamato FRMDB. Più precisamente, al vecchio dataset sono stati aggiunti altri 28 nuovi campioni (39 soggetti aventi video di sorveglianza a bassa risoluzione più i 28 con video ad alta risoluzione).
Sono stati studiati gli indici di accuratezza prodotti prendendo in esame prima i 28 campioni e poi il dataset al completo. Si è osservato che i test prodotti con l'intero dataset hanno riportato un'accuratezza peggiore, per ogni rete, rispetto ai test fatti con i soli 28 campioni. Questo esito era stato previsto in quanto il database precedente ha video di sorveglianza con qualità nettamente inferiore e ciò ha provocato una riduzione del livello di accuratezza. Pertanto, si può evidenziare che i migliori livelli di accuratezza si ottengono nel Test-5 e nel Test-6, ovvero quelli dove si vanno ad utilizzare la metà o tutti i 28 mugshot per ogni campione. D'altra parte, i risultati peggiori si sono riscontrati utilizzando sia il mugshot frontale unito a quello del profilo sinistro (questa configurazione è attualmente utilizzata dalla Polizia per il fotosegnalamento) sia solamente con il mugshot frontale.
Questi test possono costituire la base di futuri studi, in particolare si potrebbe aumentare il numero di campioni del dataset FRMDB per osservare il cambiamento dell'accuratezza. Avendo un certo numero di pose per ogni soggetto si può andare a costatare quale tra queste produce un risultato migliore ed eliminare quelle che generano risultati peggiori. Una tecnica inerente a questo studio si chiama Pose-Invariant-Face-Recognition (PIFR) [22] e permette di valutare le pose migliori per identificare una persona. Grazie a questo studio si potrebbe alleggerire il database in quanto a spazio occupato.

# Bibliografia


[1] Tomassini S. Falcionelli N. Martarelli M. Castellini P. Dragoni. F Contardo P., Sernani P. The face recognition from mugshots database. 2023.

[2] Andrew Zisserman Karen Simonyan. Very deep convolutional networks for large-scale image recognition. *arxiv*, 2014.

[3] Shaoqing Ren Jian Sun Kaiming He, Xiangyu Zhang. Deep residual learning for image recognition. *arxiv*, 2015.

[4] E. Wu S. Albanie J. Hu, L. Shen e G. Sun. Squeeze-and-excitation networks. *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[5] Raviv Shaun. The secret history of facial recognition. *Wired*, 2023. URL `https://www.wired.com/story/secret-history-facial-recognition/`.

[6] A. Pentland M. Turk. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 1991.

[7] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 2012.

[8] Marc'Aurelio Ranzato Lior Wolf Yaniv Taigman, Ming Yang. Deepface: Closing the gap to human-level performance in face verification. *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[9] Tamara Berg Erik Learned-Miller Gary B. Huang, Manu Ramesh. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *University of Massachusetts*, 2007.

[10] Andrew Zisserman Omkar M. Parkhi, Andrea Vedaldi. Deep face recognition. *British Machine Vision Association*, 2015.

[11] Mayank Mishra. Convolutional neural networks, explained. *Towards Data Science*, 2020. URL `https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939`.

[12] Anh H. Reynolds. Convolutional neural networks (cnns). URL `https://anhreynolds.com/blogs/cnn.html`.

[13] Adithya Challa. What is a relu layer. *Educative*. URL `https://www.educative.io/answers/what-is-a-relu-layer`.

[14] Savyak Hosla. Introduction to pooling layer. *GeeksforGeeks*, 2023. URL `https://www.educative.io/answers/what-is-a-relu-layer`.

[15] Weidi Xie Omkar M. Parkhi Qiong Cao, Li Shen and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. *13th IEEE International Conference on Automatic Face  Gesture Recognition*, 2017.

[16] Yusuf Sarıgöz. Triplet loss — advanced intro. *Towards Data Science*. URL `https://towardsdatascience.com/triplet-loss-advanced-intro-49a07b7d8905`.

[17] Zhifeng Li Yu Qiao Kaipeng Zhang, Zhanpeng Zhang. Joint face detection and alignment using multi-task cascaded convolutional networks. *IEEE*, 2016.

[18] Michael Jones Paul Viola. Rapid object detection using a boosted cascade of simple features. *Conference on Computer Vision and Pattern Recognition*, 2001.

[19] Support vector machine (svm). *MathWorks*, 2017. URL `https://it.mathworks.com/discovery/support-vector-machine.html`.

[20] D. Wilimitis. The kernel trick in support vector classification. *Towards Data Science*, 2018. URL `https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f`.

[21] I. José. Knn (k-nearest neighbors) 1. *Towards Data Science*, 2017. URL `https://towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d`.

[22] Dacheng Tao Changxing Ding. A comprehensive survey on pose-invariant face recognition. *ACM Transactions on Intelligent Systems and Technology*, 2016.