

Corso di Laurea Triennale in Ingegneria e Scienze Informatiche

Analisi e sviluppo di algoritmi per la valutazione della qualità delle immagini del volto

Tesi di laurea in:
VISIONE ARTIFICIALE

Relatore

Franco Annalisa

Candidato

Senni Mattia

Correlatore

Borghi Guido

Abstract

La visione artificiale sta venendo sempre più utilizzata nel riconoscimento di un soggetto tramite immagini facciali, ambito in cui si stanno diffondendo sistemi di riconoscimento biometrico automatico. In questi contesti, è fondamentale garantire un'acquisizione controllata e standardizzata delle immagini facciali, al fine di ridurre errori di riconoscimento e vulnerabilità a minacce come il face morphing. Una cattiva acquisizione può compromettere l'intero processo biometrico, esponendo il sistema a minacce quali la falsificazioni e riducendone l'affidabilità. Questo progetto contribuisce allo sviluppo ed alla valutazione di un software che ha l'obiettivo di implementare un draft ISO, volto a valutare la qualità dell'acquisizione delle immagini facciali secondo diverse metriche.

Contents

CONTENTS

List of Figures

LIST OF FIGURES

List of Listings

listings/onnx_exporter.py	24
-------------------------------------	----

LIST OF LISTINGS

Chapter 1

Introduzione

1.1 Motivazioni

Nell'ambito della visione artificiale, e in particolare del riconoscimento facciale, si è ottenuta negli ultimi anni a una significativa riduzione dei tassi di errore grazie all'introduzione di algoritmi basati su tecniche di deep learning. Tuttavia, gli errori di riconoscimento rimangono ancora rilevanti e possono essere influenzati da numerosi fattori, tra cui le modalità di acquisizione dell'immagine, la cooperazione del soggetto durante la cattura biometrica, la specifica implementazione dell'algoritmo di confronto e la logica decisionale associata. In questo contesto, risulta evidente la necessità di un ammodernamento delle procedure di acquisizione, per evitare un peggioramento delle prestazioni dei sistemi biometrici a fronte dell'incremento dei volumi di dati da elaborare. L'urgenza di intervenire su questi aspetti si collega anche a una maggiore consapevolezza dell'importanza dell'usabilità dei sistemi biometrici, sia per gli utenti finali sia per gli operatori umani, i quali possono contribuire in modo significativo alla riduzione degli errori attraverso una migliore qualità nella fase di acquisizione. Spesso, infatti, un sistema progettato esclusivamente sulla base della tecnologia rischia di mostrare i propri limiti se non considera in modo integrato le dinamiche dell'interazione umana. Parallelamente, l'adozione crescente del riconoscimento facciale in contesti di ampia scala, come la gestione dei documenti di identità elettronici, le applicazioni commerciali e le attività di controllo delle forze dell'ordine, ha comportato la raccolta massiva di immagini

del volto, che fungono sia da campioni di riferimento in fase di registrazione sia da elementi di confronto successivo. Programmi di portata nazionale e internazionale, come quelli attivati in Cina, nell'Unione Europea, negli Stati Uniti e in India, testimoniano il crescente impiego di questa tecnologia in settori critici quali i trasporti, l'immigrazione e la verifica dell'identità. Tuttavia, molte delle immagini facciali oggi raccolte provengono da dispositivi di acquisizione non specificamente progettati per l'acquisizione di immagini del volto. Le immagini destinate a documenti d'identità o a database ufficiali vengono per lo più acquisite con telecamere configurate secondo specifiche documentarie, come lo standard ISO/IEC 39794-5, che ne disciplina l'inquadratura e le caratteristiche fotografiche. In assenza di strumenti automatici di valutazione della qualità, la verifica della conformità ai requisiti previsti dagli standard vigenti viene spesso affidata al fotografo. A questo si aggiunge il problema rappresentato da comportamenti involontari dei soggetti durante la cattura, portando ad una variazioni tra l'immagine di riferimento e quella di confronto, compromettendo l'efficacia del riconoscimento. È pertanto essenziale garantire la coerenza con la presentazione canonica prevista dagli standard, ossia un volto centrato e frontale, con espressione neutra, occhi aperti e privo di montature di occhiali che coprano gli occhi o troppo spesse. Considerando la grande eterogeneità degli utenti, per età, costituzione fisica, etnia, lingua, cultura, alfabetizzazione e familiarità con la tecnologia, è evidente l'importanza di un'attenta progettazione dal punto di vista dei fattori umani per migliorare l'acquisizione delle immagini facciali. Un'ulteriore criticità risiede nella separazione tra il processo di acquisizione dell'immagine e quello di valutazione della qualità, che spesso avviene solo in un secondo momento, quando la fotografia viene inviata a un server remoto. Se la qualità dell'immagine risulta inadeguata, si rende necessaria una nuova acquisizione aggravando tempi e costi. In questo scenario, il lavoro descritto da questa tesi si colloca nell'ambito della valutazione della qualità delle immagini del volto (ISO 29794-5), nell'ambito della specifica applicazione legata ai documenti di identità elettronici (ISO 39794-5).

1.2 Obbiettivo

Il progetto consiste in una valutazione del software OFIQ (Open Source Face Image Quality), strumento open source sviluppato per implementare le metriche descritte nel draft dello standard ISO precedentemente illustrato. Il lavoro consiste inoltre nella contribuzione allo sviluppo del software mediante l'implementazione di ulteriori metriche tra cui alcune suggerite dallo standard, ma non ancora integrate nel software. L'obiettivo principale è quello di verificare l'affidabilità di OFIQ per il controllo di qualità delle immagini facciali destinate all'identificazione biometrica, testandolo su esempi conformi alle specifiche e su esempi non conformi. Si vuole valutare se l'impiego del software durante la fase di acquisizione possa rappresentare uno strumento accurato per garantire la qualità e la conformità delle immagini biometriche.

1.3 Metodo

Per la realizzazione di questo progetto si è partiti con la comprensione delle varie metriche descritte nel draft ISO 29794-5:2024. Successivamente si è testata ogni singola metrica comparando diversi dataset divisi tra immagini che soddisfano i requisiti dello standard e immagini che non soddisfano la determinata metrica che si sta valutando, in seguito a queste valutazioni si guarda la distribuzione dei risultati.

Structure of the Thesis

Senni Mattia: At the end, describe the structure of the paper

Chapter 2

Implementazione della metrica per il rilevamento del difetto degli occhi rossi

2.1 Obiettivo della metrica

L'obiettivo di questa metrica è quello di rilevare il difetto degli occhi rossi all'interno di un viso. Il difetto degli occhi rossi nelle foto si verifica quando il flash della fotocamera illumina rapidamente la pupilla dell'occhio. La luce intensa penetra nella parte posteriore dell'occhio, raggiungendo la retina, che è ricca di vasi sanguigni. Poiché la luce viene riflessa direttamente verso la fotocamera dai capillari della retina, i quali sono rossi, l'effetto risultante nell'immagine è quello di pupille rosse anziché nere. Questo fenomeno è più comune in condizioni di scarsa illuminazione, quando le pupille sono più dilatate, permettendo a una maggiore quantità di luce di entrare e riflettersi. Dato che questo difetto noto altera il colore originale delle pupille è importante che un software per la valutazione di immagini di volti riesca a rilevarlo con una certa precisione. L'implementazione di questa metrica permette di dare un riscontro sotto forma di valutazione normalizzata da 0 a 100 sulla correttezza della foto rispetto a questo difetto.

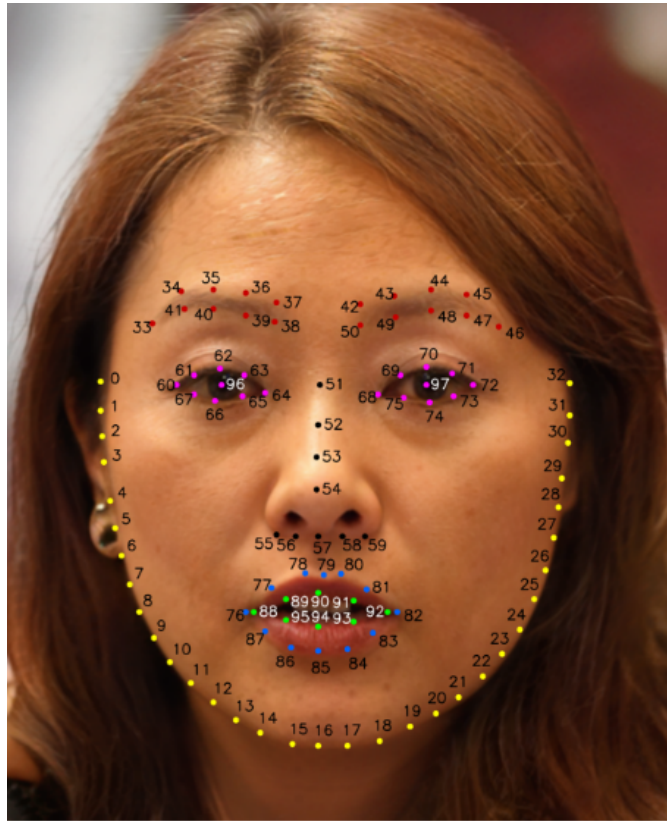


Figure 2.1: Immagine dei landmark estratti dalla rete ADNet

2.2 Requisiti preliminari

I requisiti preliminari di questo task sono i landmark del viso. Il draft della ISO utilizza come riferimento per l'estrazione dei landmark facciali la CNN ADNet allenata sul Wild dataset, prende in input un'immagine RGB e restituisce in output un set di 98 landmark ???. I landmark che interessano questa metrica sono i seguenti:

- dal 60 al 67: contorno occhio sinistro
- dal 68 al 75: contorno occhio destro
- 96: pupilla occhio sinistro
- 97: pupilla occhio destro

2.3 Implementazione della metrica

2.3.1 Descrizione dell'algoritmo

Viene inizialmente effettuata la segmentazione dell'occhio, l'algoritmo inizia isolando l'area dell'occhio dall'immagine completa utilizzando i landmarks forniti. Una volta estratta la regione dell'occhio, l'algoritmo crea una maschera specifica per l'iride, escludendo sia la pupilla centrale (che appare normalmente scura) sia le aree esterne all'occhio. Questa segmentazione geometrica si basa su proporzioni anatomiche standard. Lo scopo dell'algoritmo è ora quello di rilevare le zone dell'iride dove sono presenti pixel rosse, per eseguire questo compito è necessario convertire l'immagine dal tradizionale spazio colori RGB ad alcuni più utili all'isolamento di determinati colori all'interno dell'immagine quali :

- HSV: separa il colore dalla luminosità rendendo la misurazione meno sensibile alle variazioni di luce
- yCbCr: separa luminanza da cromaticanza garantendo una misurazione più robusta a luce ed ombre, inoltre, come illustrato dal paper "Face Detection in Color Images", questo spazio colore è ottimo per segmentare elementi facciali contraddistinti dal colore rosso.

Nel caso dello spazio colori HSV viene scelta una soglia di identificazione del colore rosso su tutti e tre i canali. Nel caso dello spazio colori yCbCr viene scelta una soglia minima per il canale cr, una massima per il canale cb ed inoltre vengono selezionati solamente i pixel con un valore di cr superiore alla media per evitare problemi di saturazione. In entrambi i casi le soglie sono state scelte in maniera sperimentale, cercando di rispettare un giusto connubio tra falsi positivi e falsi negativi. Infine l'algoritmo calcola il rapporto tra i pixel rossi rilevati nell'iride e l'area totale dell'iride stessa, fornendo un valore normalizzato tra 0 e 1 che quantifica l'intensità dell'effetto occhi rossi. La soglia di rosso è stata scelta in base ai risultati dei test, cercando di avere un giusto equilibrio tra falsi positivi e falsi negativi, come ad esempio le persone con gli occhi marrone o marrone chiaro.

2.3.2 Dati di Input

- immagine: immagine digitale a colori
- puntiOcchio: vettore di punti che definiscono il contorno dell'occhio
- cornea: punto centrale della cornea

2.3.3 Dati di Output

- rapportoRosso: valore decimale tra 0 e 1 che rappresenta la proporzione di pixel rossi nell'iride

2.3.4 Algoritmo

1. Creazione maschera occhio [Fig. eye mask]
 - Crea una maschera nera delle dimensioni dell'immagine
 - Riempie l'area delimitata dai punti dell'occhio con bianco
 - Calcola il rettangolo che racchiude l'occhio
2. Estrazione regione di interesse (ROI)
 - Estrae la porzione di immagine corrispondente al rettangolo dell'occhio [Fig. eye roi]
 - Estrae anche la corrispondente porzione di maschera dell'occhio [Fig. eye mask roi]
3. Calcolo parametri geometrici
 - Calcola raggio pupilla = $\max(2, \min(\text{larghezza}, \text{altezza}) / 8)$
 - Calcola raggio iride = $\text{altezza} / 2$
 - Calcola centro pupilla rispetto alla ROI
4. Creazione maschera iride [Fig. iris mask]
 - Crea maschera nera delle dimensioni della ROI

- Disegna cerchio bianco con raggio iride centrato sulla pupilla
- Disegna cerchio nero con raggio pupilla per escludere la pupilla centrale

5. Combinazione maschere [Fig. combined mask]

- Combina maschera occhio e maschera iride con operazione AND
- Risultato: maschera che isola solo la regione dell'iride

6. Rilevamento colore rosso [Fig. red mask]

- Versione con HSV
 - Converte l'immagine dell'occhio da BGR a HSV per migliore rilevamento colori
 - Definisce soglie HSV per il colore rosso:
 - * Gamma 1: H(0-10), S(100-255), V(50-255)
 - * Gamma 2: H(160-180), S(100-255), V(50-255)
 - Crea maschere separate per ciascuna gamma
 - Combina le maschere rosse con operazione OR
- Versione con yCbCr
 - Converte l'immagine dell'occhio da BGR a yCbCr
 - Separa i canali cb e cr
 - calcola media cr = la media dei valori della matrice cr
 - calcola la matrice valoriAltiCr = maschera sui valori di cr maggiori di 150
 - calcola la matrice valoriBassiCb = maschera sui valori di cb minori di 120
 - calcola la matrice valoriMaggioreMediaCr = maschera sui valori di cr maggiori alla media cr
 - calcola red mask = end logico bit a bit tra valoriAltiCr, valoriBassiCb, valoriMaggioreMediaCr

7. Isolamento pixel rossi nell'iride [Fig. red iris mask]

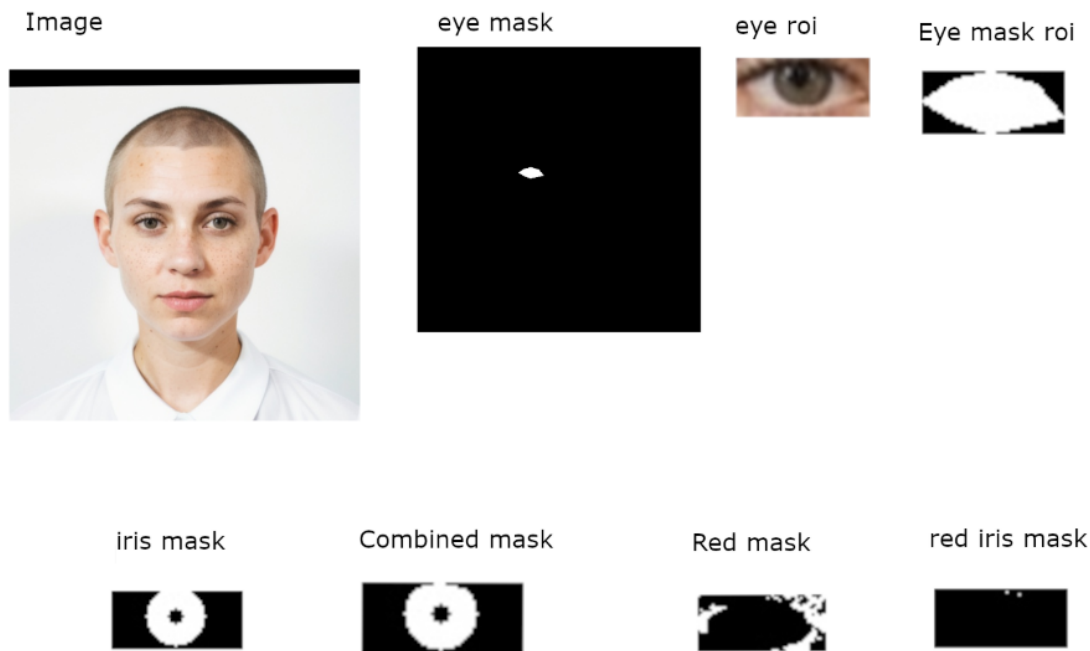


Figure 2.2: Immagine descrittiva del procedimento per il rilevamento del difetto degli occhi rossi

- Applica maschera rossa alla maschera iride combinata
- Risultato: pixel rossi presenti solo nell'area dell'iride

8. Calcolo rapporto finale

- Conta pixel rossi nell'iride
- Conta pixel totali nell'area iride
- Se $\text{area iride} < 0$: $\text{rapportoRosso} = \text{pixelRossi} / \text{pixelTotaliIride}$
- Altrimenti: $\text{rapportoRosso} = 0$

9. Ritorna rapportoRosso

Chapter 3

Implementazione della metrica per il rilevamento dello sguardo frontale

3.1 Obbiettivo della metrica

L'obiettivo della metrica è quello di assegnare uno score da 0 a 100 sul grado in cui un soggetto guarda in camera misura necessaria per garantire l'idoneità delle foto nei documenti di identità. Guardare dritto in camera è un requisito fondamentale per il riconoscimento facciale e per la conformità agli standard internazionali.

3.2 Requisiti preliminari

I requisiti preliminari di questo task sono i landmark del viso. Il draft della ISO utilizza come riferimento per l'estrazione dei landmark facciali la CNN ADNet allenata sul Wild dataset, prende in input un'immagine RGB e restituisce in output un set di 98 landmark ???. I landmark che interessano questa metrica sono i seguenti:

- 60, 64: angoli occhio sinistro
- 68, 72: angoli occhio destro



Figure 3.1: Immagine rappresentativa dei rapporti utilizzato dall’algoritmo di stima dello sguardo frontale manuale

- 96: pupilla occhio sinistro
- 97: pupilla occhio destro

3.3 Implementazione della metrica

3.3.1 Descrizione dell’algoritmo

L’algoritmo analizza la direzione dello sguardo misurando la posizione delle pupille all’interno degli occhi. Per ogni occhio, calcola la larghezza totale (distanza tra gli angoli interno ed esterno) e la distanza tra la pupilla e l’angolo interno. Dividendo queste due misure ottiene un rapporto che indica dove si trova la pupilla: un valore di 0.5 significa che la pupilla è perfettamente centrata, mentre valori minori o maggiori indicano sguardo verso sinistra o destra. L’algoritmo confronta entrambi gli occhi e sceglie quello con maggiore deviazione dal centro per determinare la direzione principale dello sguardo. Infine, applica una trasformazione parabolica (con parametri $\alpha = -400$ e $\beta = 100$) che mappa il risultato su una scala da 0 a 100, dove valori più alti indicano maggiore deviazione dalla posizione frontale. Questo approccio permette di quantificare oggettivamente quanto una persona stia guardando lateralmente rispetto alla fotocamera. Una visione grafica dei rapporti sopra elencati è disponibile

3.3.2 Dati di Input

- a: coordinate x ed y dell’angolo esterno dell’occhio sinistro
- b: coordinate x ed y dell’angolo interno dell’occhio sinistro

- c: coordinate x ed y dell'angolo esterno dell'occhio destro
- d: coordinate x ed y dell'angolo interno dell'occhio destro
- e: coordinate x ed y della pupilla dell'occhio sinistro
- f: coordinate x ed y della pupilla dell'occhio destro

3.3.3 Dati di Output

- valoreScalare: valore numerico che rappresenta la deviazione dello sguardo da 0 a 100

3.3.4 Algoritmo

1. Calcolo larghezza degli occhi

- $larghezzaSinistra = \sqrt{(b_x - a_x)^2 + (b_y - a_y)^2}$
- $larghezzaDestra = \sqrt{(d_x - c_x)^2 + (d_y - c_y)^2}$

2. Calcolo distanza pupilla-angolo interno

- $distanzaSinistra = \sqrt{(e_x - b_x)^2 + (e_y - b_y)^2}$
- $distanzaDestra = \sqrt{(f_x - d_x)^2 + (f_y - d_y)^2}$

3. Calcolo rapporti normalizzati

- $rapportoSinistro = \frac{distanzaSinistra}{larghezzaSinistra}$
- $rapportoDestro = \frac{distanzaDestra}{larghezzaDestra}$

4. Calcolo variazioni dal centro (*Nota: 0.5 rappresenta la posizione centrale ideale della pupilla*)

- $variazioneSinistra = |rapportoSinistro - 0.5|$
- $variazioneDestra = |rapportoDestro - 0.5|$

5. Selezione rapporto dominante

- Se $variazioneDestra > variazioneDestra$ allora: $punteggioGrezzo = rapportoSinistro$
- Altrimenti: $punteggioGrezzo = rapportoDestro$

6. Applicazione funzione di scaling

- $alpha = -400$
- $beta = 100$
- Calcola $valoreScalare = alpha \cdot (punteggioGrezzo - \frac{1}{2})^2 + beta$

Chapter 4

Rilevamento dello sguardo frontale tramite CNN

4.1 Selta del modello

Dopo un'attenta analisi sui modelli stato dell'arte nel task del rilevamento dello sguardo si è trovato come riferimento l'implementazione della rete neurale descritta all'interno del paper [?, L2CS-NET: FINE-GRAINED GAZE ESTIMATION IN UNCONSTRAINED ENVIRONMENTS]. Il modello descritto in questo paper si propone di prevedere lo sguardo in ambienti non vincolati, dando una stima degli angoli Pitch e Yaw del volto in radianti.

4.2 Utilizzo del modello

Il modello L2CS-NET viene distribuito sotto forma di pacchetto Python. Il pacchetto espone un API tramite la quale è possibile far processare un'immagine ed ottenere per ogni volto dell'immagine gli angoli Pitch e Yaw. Il modello L2CS-NET è costruito mediante il framework Pytorch ed è quindi un modello Pytorch, per poter utilizzare il modello all'interno del software OFIQ è necessario convertire il modello al formato Onnx, un formato open utilizzato per rappresentare i modelli di machine learning. Il formato Onnx permette di eseguire il modello all'interno del software OFIQ in C++ tramite la libreria OnnxRuntime disponi-

bile in C++. Inoltre il pacchetto Python di L2CS-NET esegue il preprocessa dell'immagine tramite una pipeline che effettua i seguenti step:

- Utilizza il modello RetinaFace (dalla libreria face detection) per trovare il bounding box dei vari volti all'interno dell'immagine
- Per ogni volto viene presa in considerazione solamente la parte dell'immagine compresa all'interno della bounding box (regione di interesse)
- Per ogni regione di interesse viene convertito lo spazio colori da BGR a RGB
- Ogni regione di interesse viene ridimensionato in un'immagine (224 * 224)
- Ogni regione di interesse viene passata al modello ResNet-50 seguito da 2 layer fully-connected che ritornano in output 90 classi rappresentanti intervalli discreti (sia per pitch che per yaw) con all'interno un valore rappresentante la probabilità che lo sguardo per il relativo angolo si trovi in quell'intervallo
- Per ogni valore ritornato da ResNet-50 viene applicata la funzione Softmax di modo da avere la probabilità in percentuale che lo sguardo si trovi all'interno di quell'intervallo
- Vengono moltiplicate le probabilità che lo sguardo si trovi in quell'intervallo per l'indice della probabilità nel vettore dei tensori
- Viene fatta la somma dei valori ottenuti precedentemente
- La somma viene moltiplicata per quattro ed il risultato sottratto di 180, in modo da convertire ogni probabilità ottenuta gradi.
- Viene eseguita la conversione da gradi a radianti

I passaggi che includono l'utilizzo di open cv sono facilmente replicabili all'interno del software OFIQ tramite la libreria open cv per cpp, mentre per trovare il bounding box del volto all'interno dell'immagine viene utilizzato il modello "SSD Face Detector CNN" che riporta risultati molto simili a quelli di RetinaFace, la scelta di utilizzare questo modello è dettata dal fatto di averlo già a disposizione all'interno del software in formato Onnx.

4.3 Implementazione della metrica

4.3.1 Dati di Input

- immagine: L'immagine del volto da valutare

4.3.2 Dati di Output

- valoreScalare: valore numerico che rappresenta la deviazione dello sguardo da 0 a 100

4.3.3 Algoritmo

1. Calcola *detectedFaces* tramite il modello SSD Face Detector CNN
2. Se *detectedFaces.length* < 1 allora: *valoreScalare* = 0. Fine.
3. *faceBoundingBox* = *detectedFaces*[0], solamente il primo volto trovato verrà valutato
4. Carica il modello in formato onnx tramite onnxRuntime
5. Calcola *immagineVolto* = la sezione di immagine all'interno di *faceBoundingBox*
6. Preprocessing per L2CS-NET
 - Definisce *inputSize* = 443
 - Calcola *immagineRidimensionata* ridimensionando con open cv *immagineVolto* alla grandezza (*inputSize* x *inputSize*)
 - Converte lo schema colore di *immagineRidimensionata* da BGR a RGB
 - Scala i valori dell'immagine dall'intervallo [0,255] a [0,1] convertendoli a float
 - Normalizza l'immagine
 - Definisce *mean* = [0.485*f*, 0.456*f*, 0.406*f*] (dal preprocessing della librerie L2CS-NET)

- Definisce $std = [0.229f, 0.224f, 0.225f]$ (dal preprocessing della libreria L2CS-NET)
 - Per ogni canale RGB dell'immagine: $channels[i] = \frac{(channels[i] - mean[i])}{std[i]}$ con $i = 0..3$
 - Definisce *immaginePreprocessata* la matrice formata dai canali normalizzati nei punti precedenti
 - Definisce $input_shape = [1, 3, inputSize, inputSize]$ (aggiunge la dimensione della batch (in questo caso 1 dato che viene processata solamente un'immagine))
 - Converte l'input in una matrice CHW:
 - Per ogni canale $c = 0..3$:
 - Per ogni riga dell'immagine $h = 0..inputSize$
 - Per ogni colonna dell'immagine $w = 0..inputSize$
 - $tensoreInput[c * inputSize * inputSize + h * inputSize + w] = immaginePreprocessata[h][w][c]$
7. Vengono usate le API offerte da *onnxRuntime* per eseguire il modello L2CS in formato Onnx.
 8. Vengono calcolati *pitch* e *yaw* sulla base dell'output del modello (il modello restituisce in output i valori per 90 classi sia per pitch che per yaw, i seguenti step vengono eseguiti per entrambi gli angoli)
 9.
 - Sia x il vettore con i 90 valori
 - $angoloInRadianti = \frac{\sum_i (e^{x_i * i})}{\sum_i (e^x)} * 4 - 180$
 10. $value = max(|pitch|, |yaw|)$ (Si usa il valore assoluto in quanto lo sguardo frontale viene classificato come angolo 0)
 11. $score = round((1 - (\frac{min(value, 45)}{45})) * 100)$ (Uso 45 come valore limite, tutti valore da 45 a 180 vengono classificati come punteggio 0)
 12. $value$ è il valore assoluto dell'angolo più lontano da 0 Gradi, mentre $score$ è la valutazione dell'immagine da 0 a 100

Chapter 5

Strumenti utilizzati per lo sviluppo delle metriche

5.1 Software OFIQ

Il software OFIQ, acronimo di *Open Source Face Image Quality*, è un progetto open source sviluppato per implementare le metriche descritte nel draft dello standard ISO 29794-5. È scritto in linguaggio C++ e dispone di file CMake per la compilazione su sistemi Linux, Windows e macOS. Al suo interno utilizza librerie come OpenCV per la gestione delle immagini e OnnxRuntime per l'esecuzione dei modelli Onnx. Inizialmente presentava un'implementazione fedele di tutte le metriche riportate nello standard ISO 29794-5, restituendo in output un valore grezzo ed un valore normalizzato tra 0 e 100 (utile come *score*) per ciascuna metrica. OFIQ include inoltre delle utility che permettono l'utilizzo dei seguenti modelli (in formato Onnx):

- SSD Face Detector CNN: modello per il rilevamento dei volti in un'immagine.
- ADNet: modello per l'estrazione dei landmark facciali.

Il software implementa le seguenti metriche:

- Quality Score: uno score generale dell'immagine ottenuto tramite il modello MagFace.

- Background uniformity: misura l'uniformità dello sfondo.
- Illumination uniformity: misura l'uniformità dell'illuminazione confrontando la parte destra e sinistra del volto.
- Luminance mean: verifica che l'immagine presenti un'illuminazione adeguata e uniforme.
- Luminance variance: misura i contrasti nell'immagine.
- Under-exposure prevention: verifica che l'immagine non contenga troppi pixel con bassa luminosità.
- Over-exposure prevention: verifica che l'immagine non contenga troppi pixel con alta luminosità.
- Dynamic range: misura la variazione di intensità luminosa nella regione del volto, assicurando che non sia completamente scura o chiara.
- Sharpness: valuta la nitidezza dell'immagine (corretta messa a fuoco).
- No compression artifacts: valuta la presenza di artefatti di compressione, assicurando che l'immagine non sia stata eccessivamente compressa.
- Natural colour: valuta la naturalezza del colore della pelle.
- Single face present: verifica la presenza di un singolo volto.
- Eyes open: controlla che entrambi gli occhi siano aperti in modo naturale.
- Mouth closed: verifica che la bocca sia chiusa.
- Eyes visible: verifica che siano visibili pupilla e iride in entrambi gli occhi.
- Mouth occlusion prevention: controlla che la bocca non sia occlusa.
- Face occlusion prevention: verifica che la regione del volto (dalla sommità del capo al mento e da orecchio a orecchio) sia chiaramente visibile.
- Inter-eye distance: misura in pixel la distanza tra gli occhi.

- Head size: valuta la dimensione del volto per evitare immagini troppo ravvicinate.
- Leftward crop of face: verifica che il volto non sia decentrato a sinistra.
- Rightward crop of face: verifica che il volto non sia decentrato a destra.
- Downward crop of face: verifica che il volto non sia decentrato in basso.
- Upward crop of face: verifica che il volto non sia decentrato in alto.
- Pose angle yaw frontal alignment: verifica che l'angolo *yaw* del volto sia inferiore a $\pm 5^\circ$ dal frontale.
- Pose angle pitch frontal alignment: verifica che l'angolo *pitch* del volto sia inferiore a $\pm 5^\circ$ dal frontale.
- Pose angle roll frontal alignment: verifica che l'angolo *roll* del volto sia inferiore a $\pm 8^\circ$ dal frontale.
- Expression neutrality: verifica che il volto presenti un'espressione neutrale.
- No head covering: verifica che il soggetto non indossi copricapi (ad esempio un cappello).

5.2 Dataset utilizzati per la valutazione

I seguenti dataset sono stati utilizzati per la valutazione delle metriche implementate.

5.2.1 ONOT

ONOT è un dataset introdotto nel paper [?, ONOT: a High-Quality ICAO-compliant Synthetic Mugshot Dataset]. È composto da immagini sintetiche di alta qualità conformi ai requisiti dello standard ISO/IEC 39794-5. Quest'ultimo definisce un formato per lo scambio di immagini facciali negli *electronic Machine-Readable*

Travel Documents (eMRTD), seguendo le linee guida dell'International Civil Aviation Organization (ICAO). Le immagini di ONOT includono volti di diverse etnie, età, generi e caratteristiche facciali, rendendo il dataset adatto alla valutazione delle metriche implementate. Per questa analisi sono state considerate le sole immagini del *Subset 1*, ICAO compliant.

5.2.2 TONO

TONO è un dataset introdotto nel paper [?, TONO: a synthetic dataset for face image compliance to ISO/ICAO standard]. È costituito da immagini sintetiche di volti ad alta qualità, create per sviluppare e valutare sistemi di verifica della conformità delle immagini facciali allo standard ISO/ICAO. Le immagini di TONO derivano dal dataset ONOT ([?]), con l'aggiunta di uno o più elementi non conformi agli standard. I difetti presenti in TONO sono i seguenti:

- Head and Shoulder Pose: assenza di posa frontale sia del volto che delle spalle.
- Gaze Direction: assenza di sguardo frontale.
- Expression: mancanza di espressione neutrale o presenza di denti visibili.
- Face Illumination: non uniformità dell'illuminazione del volto.
- Background: sfondo non uniforme.
- Head Coverings: presenza di copricapi.
- Eye Visibility: occhi chiusi, presenza di occhiali (da vista o da sole), make-up eccessivo.
- Photographic: difetti quali pixelazione, posterizzazione, sfocatura, sovraesposizione, sovrasaturazione.

Per questa casistica è stata utilizzata la versione di TONO in cui ogni immagine contiene un solo elemento in contrasto con i requisiti ISO/ICAO.

5.3 Onnx e OnnxRuntime

ONNX (*Open Neural Network Exchange*) è un formato open source per la rappresentazione di modelli di machine learning, progettato per garantire compatibilità tra diversi framework (PyTorch, TensorFlow, ecc.). Permette di salvare un modello in un file `.onnx` indipendente dall'ambiente di training. La libreria OnnxRuntime fornisce API utilizzabili da diversi linguaggi di programmazione e su diverse piattaforme (inclusi i web browser). Il progetto, sviluppato da Microsoft, è disponibile al seguente repository GitHub: <https://github.com/microsoft/onnxruntime>.

5.4 FVC Ongoing

FVC Ongoing ([?]) è una piattaforma web che consente la valutazione di algoritmi di vario tipo, tra cui il task *Face Image ISO Compliance Verification*. I test vengono condotti su diversi dataset e metriche note.

5.5 Framework PyTorch

Il linguaggio di programmazione Python ed il framework PyTorch sono comunemente utilizzati per l'addestramento e la distribuzione di modelli di Machine Learning. Nel progetto il loro impiego riguarda principalmente il testing e la conversione in formato Onnx del modello L2CS-Net, originariamente distribuito in PyTorch. In particolare, è stato utilizzato il modulo `ONNX exporter API`, che consente di esportare un modello PyTorch in formato Onnx, impiegato per la conversione di L2CS-Net tramite il seguente codice: ??

5.6 Jupyter Notebook, Numpy e Matplotlib

Jupyter Notebook, Numpy e Matplotlib sono stati utilizzati per creare documenti interattivi e rappresentare graficamente (tramite boxplot e istogrammi) le performance delle metriche di valutazione sui dataset ONOT, TONO e su un sottoinsieme di Biolab. Il linguaggio Python è stato inoltre impiegato per:

```
1 import torch
2 from l2cs import Pipeline
3 from pathlib import Path
4
5 def export_l2cs_to_onnx():
6
7     gaze_pipeline = Pipeline(
8         weights=Path("./L2CSNet_gaze360.pkl"), # pretrained model weight
9         arch='ResNet50',
10        device=torch.device('cpu')
11    )
12
13    model = gaze_pipeline.model
14
15    dummy_input = torch.randn(1, 3, 448, 448)
16
17    torch.onnx.export(
18        model,
19        dummy_input,
20        "l2cs_gaze.onnx",
21        export_params=True,
22        opset_version=11,
23        do_constant_folding=True,
24        input_names=['input'],
25        output_names=['pitch', 'yaw']
26    )
27    print("Model exported to l2cs_gaze.onnx")
28
29 if __name__ == "__main__":
30     export_l2cs_to_onnx()
```

- suddividere i dataset in base ai file di riferimento in formato `.txt`;
- elaborare le metriche per valutare il software OFIQ e gli algoritmi proposti.

Chapter 6

Valutazione delle metriche

6.1 Introduzione alle valutazioni

Per quanto riguarda le valutazioni del software OFIQ verranno riportate ed analizzate le sole valutazioni delle metriche implementate nelle loro diverse versioni. In particolare per ogni metrica verranno analizzati **Equal Error Rate** e **Distribuzione dei risultati**.

6.2 Valutazione su FVC-Ongoing

6.2.1 Metriche valutate

La piattaforma FVC-Ongoing valuta le seguenti 24 diverse metriche **??**. Nel caso di questo progetto le metriche da valutare saranno

- Looking Away: valuta se un soggetto sta guardando in camera
- Red Eyes: Valuta se un soggetto ha il difetto degli occhi rossi

6.2.2 Il protocollo

Per la sottomissione degli algoritmi sulla piattaforma FVC-Ongoing è necessario rispettare il seguente standard:

N°	Description of the test
Feature extraction accuracy tests	
1	Eye center location accuracy
Photographic and pose-specific tests	
2	Blurred
3	Looking away
4	Ink marked/creased
5	Unnatural skin tone
6	Too dark/light
7	Washed out
8	Pixelation
9	Hair across eyes
10	Eyes closed
11	Varied background
12	Roll/pitch/yaw \geq predefined thresholds
13	Flash reflection on skin
14	Red eyes
15	Shadows behind head
16	Shadows across face
17	Dark tinted lenses
18	Flash reflection on lenses
19	Frames too heavy
20	Frame covering eyes
21	Hat/cap
22	Veil over face
23	Mouth open
24	Other faces/toys too close

Table 6.1: Metriche valutate da FVC-Ongoing

- è necessario sottoporre una cartella in formato ZIP
- All'interno della cartella deve essere presente un file *Check.exe* che un eseguibile per Win 32 in formato console application
- la sintassi a linea di comando deve essere la seguente: ". /Check.exe <faceimagefile> <outputfile>" dove
 - faceimagefile: è il percorso dell'immagine del volto da valutare e può essere in formato BPM, JPG, PNG

- `outputfile`: è il percorso del file TXT di output dove dovrà essere scritta in modalità 'append' una stringa rappresentante l'esito dei test
- Ogni linea del `outputfile` dovrà contenere i campi seguenti separati da uno spazio vuoto:
 - `ImageName`: il nome file dell'immagine
 - `RetVal`: intero che indica se l'immagine può essere processata
 - * 1: immagine processabile
 - * 0: immagine non processabile
 - * -1: dimensione dell'immagine non supportata
 - * -2: formato immagine non supportato
 - * -3: contenuto dell'immagine non processabile
 - `LE_x`: intero, coordinata X del centro dell'occhio destro (in pixel)
 - `LE_y`: intero, coordinata Y del centro dell'occhio destro (in pixel)
 - `RE_x`: intero, coordinata X del centro dell'occhio sinistro (in pixel)
 - `RE_y`: intero, coordinata Y del centro dell'occhio sinistro (in pixel)
 - `Test_2`: intero da 0 a 100 indicante il grado di conformità dell'immagine di input rispetto al test 2. 0 significa non conformità, 100 significa massima complicità, è possibile rispondere anche con i seguenti caratteri:
 - * '-': se il programma non è in grado di valutare questa metrica
 - * '?': se il programma supporta la valutazione di questa metrica ma non è in grado di valutarla nell'immagine corrente per un determinato motivo
 - * '!': se il programma supporta la valutazione di questa metrica ma non è in grado di valutarla nell'immagine corrente per un motivo indefinito
 - ...
 - `Test_24`: analogo al test 2 ma relativo al 24
- L'eseguibile ha inoltre il permesso in scrittura solamente sul file di output e può caricare tutti i suoi file di configurazione in modalità lettura

Dopo un'analisi del protocollo richiesto dalla piattaforma FVC-Ongoing è stato necessario rendere il software OFIQ conforme al protocollo ed adattare le metriche di OFIQ a quelle richieste. Valutazione

6.2.3 Valutazione metrica per il rilevamento del difetto degli occhi rossi (versione HSV)

La valutazione della metrica del difetto degli occhi rossi è divisa in 2 parti, nella prima parte verranno analizzati i risultati ottenuti tramite l'algoritmo che utilizza lo spazio colore HSV per determinare la presenza degli occhi rossi, mentre nella seconda parte verrà analizzato quello che utilizza lo spazio colore yCbCr.

L'Equal Error Rate è 18.1%.

Mentre la distribuzione degli score è rappresentata nel grafico ??.

6.2.4 Valutazione metrica per il rilevamento del difetto degli occhi rossi (versione yCbCr)

La versione dell'algoritmo che rileva il difetto degli occhi rossi tramite l'utilizzo dello spazio colori yCbCr ottiene risultati migliori rispetto al precedente.

Equal Error Rate: 14.6%.

La distribuzione degli score è rappresentata nel grafico ??

6.2.5 Valutazione metrica per il rilevamento dello sguardo frontale (metodo algoritmico)

I risultati ottenuti nella valutazione di questo difetto tramite il metodo algoritmico rientrano nella media dei risultati ottenute dagli algoritmi pubblicati sulla piattaforma.

Equal Error Rate: 17.1%.

La distribuzione degli score è rappresentata nel grafico ??

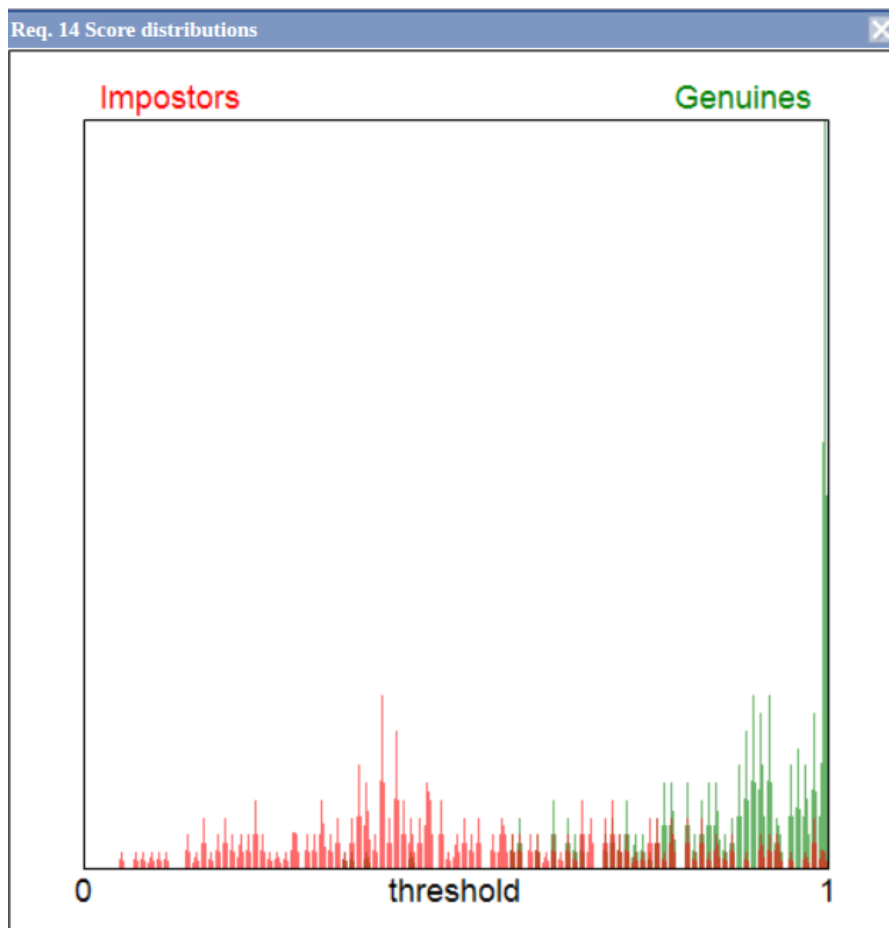


Figure 6.1: Distribuzione dei risultati metrica rilevamento del difetto degli occhi rossi, versione HSV

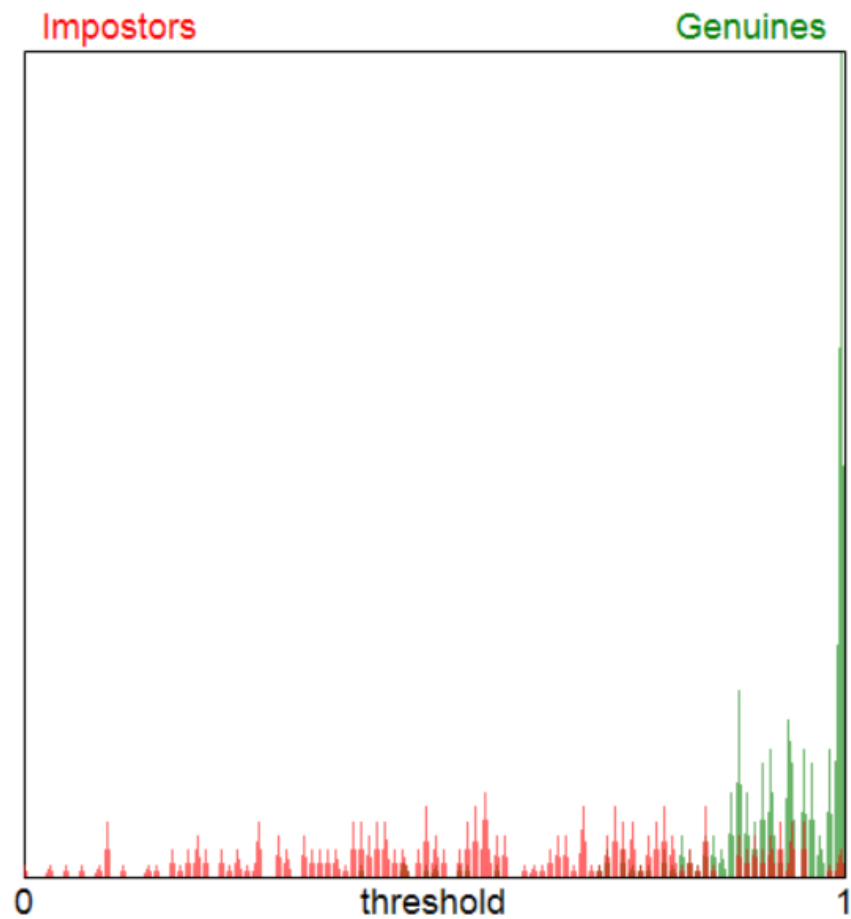


Figure 6.2: Distribuzione dei risultati metrica rilevamento del difetto degli occhi rossi, versione yCbCr

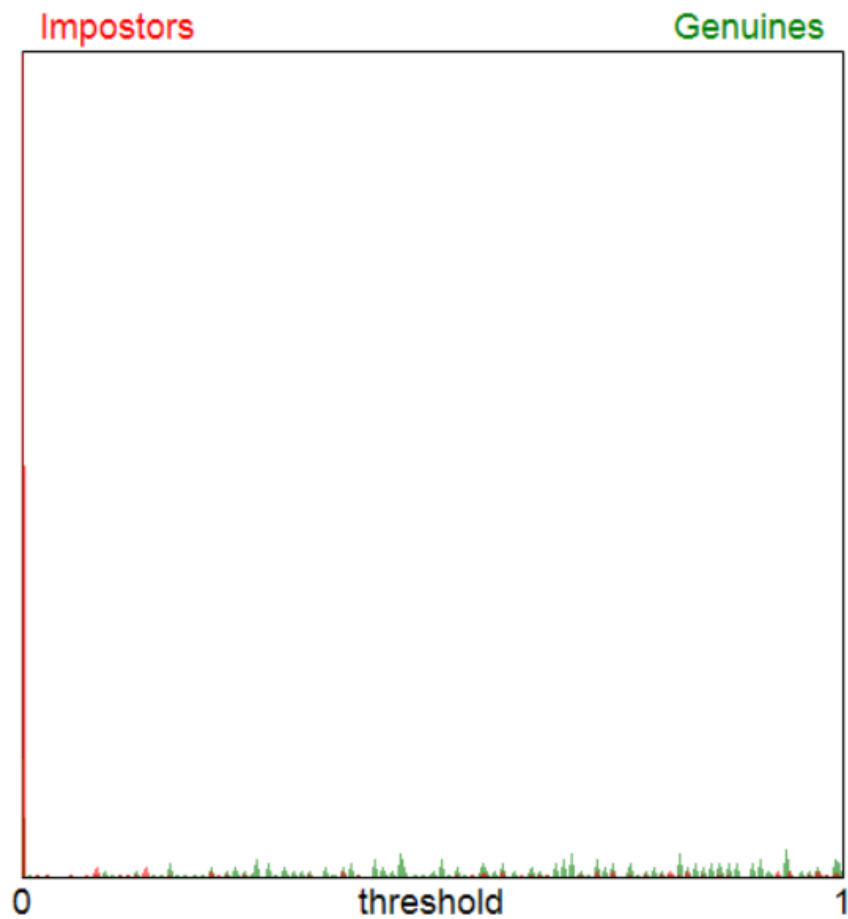


Figure 6.3: Distribuzione dei risultati metrica rilevamento del difetto dello sguardo frontale, versione algoritmica

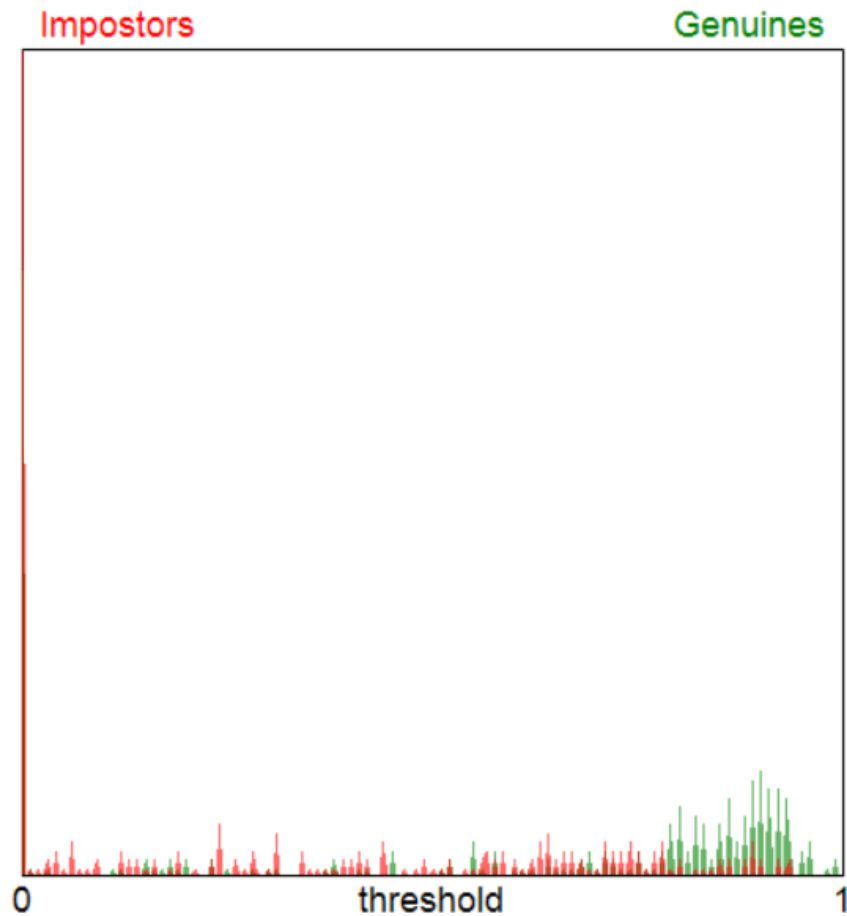


Figure 6.4: Distribuzione dei risultati metrica rilevamento del difetto dello sguardo frontale, versione L2CS-NET

6.2.6 Valutazione metrica per il rilevamento dello sguardo frontale (metodo L2CS-Net)

L'utilizzo di una CNN (L2CS-Net) in grado di predire l'angolatura dello sguardo di un volto per classificare se lo sguardo fosse frontale oppure no, si è rilevata molto più inefficiente del metodo algoritmico.

Equal Error Rate: 36.0%.

La distribuzione degli score è rappresenta nel grafico ??

6.3 Valutazione su dataset sintetici

6.3.1 Dataset utilizzati

I dataset utilizzati per questa valutazione sono ONOT e TONO, precedentemente introdotti, TONO verrà utilizzato per fornire immagini non compliant sul singolo difetto in atto di valutazione, mentre ONOT fornirà le immagini compliant.

6.3.2 Metriche valutate

Per quanto riguarda le metriche valutate su TONO e ONOT non è possibile la valutazione del difetto degli occhi rossi, causa l'assenza di immagini riportanti tale difetto nel dataset, verrà quindi riportata la sola valutazione sul difetto del rilevamento dello sguardo frontale.

6.3.3 Procedimento di valutazione

La valutazione delle metriche sui dataset ONOT e TONO ha seguito i seguenti procedimenti:

1. **Filtraggio delle immagine in ONOT:** tramite i file txt forniti insieme dal dataset ONOT sono state conservate le sole immagini aderenti allo standard ICAO
2. **Valutazione delle metriche di TONO:** ogni cartella di TONO e le rimanenti immagini di ONOT sono state valutate dal software OFIQ generando per ogni cartella valutata un file csv, contenente in ogni riga l'immagine valutata e nelle colonne gli score assengati per i vari difetti valutati
3. **Analisi dei risultati:** le valutazioni delle immagini di TONO sono state messe in confronto con quelle di ONOT, le valutazioni di ogni sottocartella di Tono nella metrica relativa al difetto rappresentato sono state messe in relazione (tramite grafici BoxPlot) con i risultati del dataset ONOT nella metrica corrispondente
4. **Visualizzazione dei risultati:** per ogni metrica valutata i risultati sono stati visualizzati tramite i seguenti metodi:

- **BoxPlot:** Uno per i risultati di TONO ed uno per quelli di ONOT, utilizzato per mettere in risalto la differenza della distribuzione degli score di valore. Il Box Plot va letto nel seguente modo: ogni metrica valutata ha due box plot, il primo (da sinistra verso destra) indica gli score di TONO per la metrica valutata, il secondo indica gli score di ONOT, così via per tutte le metriche valutate per il relativo difetto
- **Distribuzione degli score:** un istogramma che mette in confronto la distribuzione degli score, in rosso quelli di TONO ed in verde quelli di ONOT
- **Equal Error Rate:** metrica utilizzata per misurare quanto gli score sono bilanciati tra falsi positivi e falsi negativi

6.3.4 Valutazione metrica per il rilevamento dello sguardo frontale (metodo algoritmico)

Il software TONO ha 2 cartelle contenenti immagini che riportano il difetto dello sguardo frontale:

- *la_1*: Immagini di volti con lo sguardo rivolto verso destra
- *la_2*: Immagini di volti con lo sguardo rivolto verso sinistra

I risultati verranno riportati divisi per entrambi le cartelle.

Risultati per la cartella *la_1*

Equal Error Rate: 15.8%.

Distribuzione degli score: ??

Box Plot comparativo: ??

Risultati per la cartella *la_2*

Equal Error Rate: 11.1%.

Distribuzione degli score: ??

Box Plot comparativo: ??

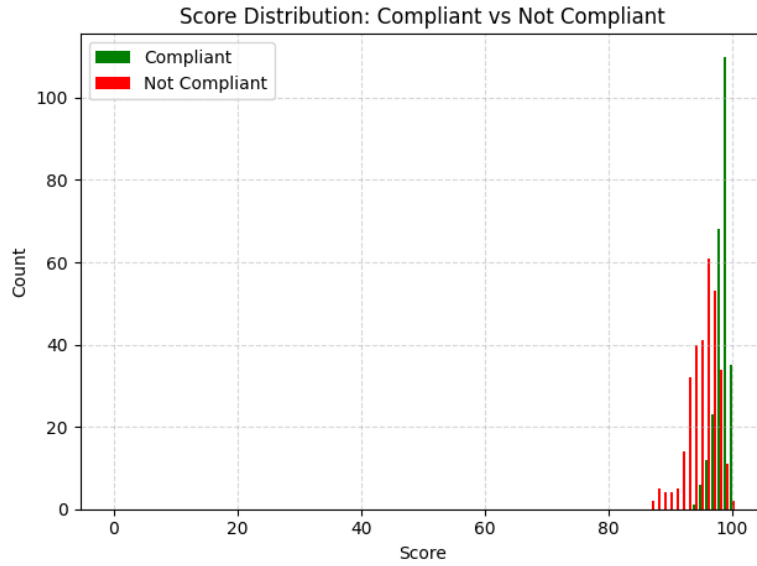


Figure 6.5: Distribuzione degli score per la metrica del rilevamento dello sguardo frontale (metodo algoritmico) valutata sui dataset TONO (cartella la_1) e ONOT

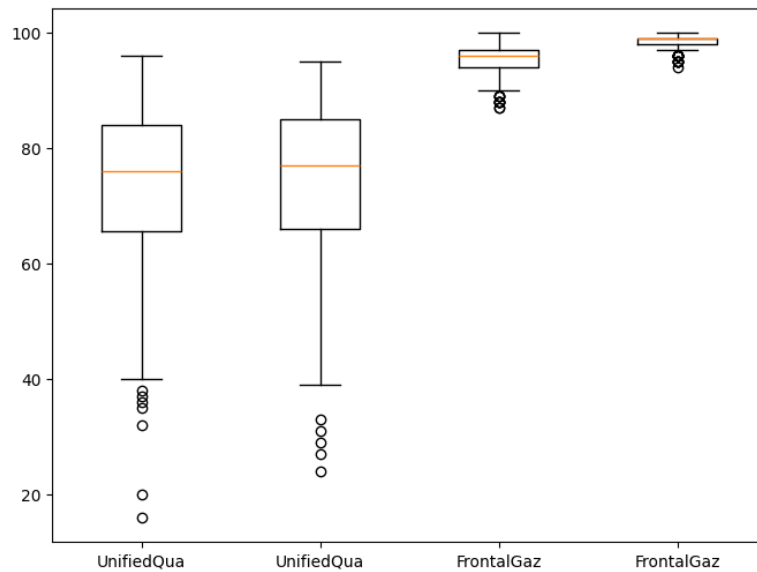


Figure 6.6: Box Plot con la metrica del rilevamento dello sguardo frontale (metodo algoritmico) valutata sui dataset TONO (cartella la_1) e ONOT, metriche valutate (da sinistra verso destra): Unified Quality Score e Frontal Gaze

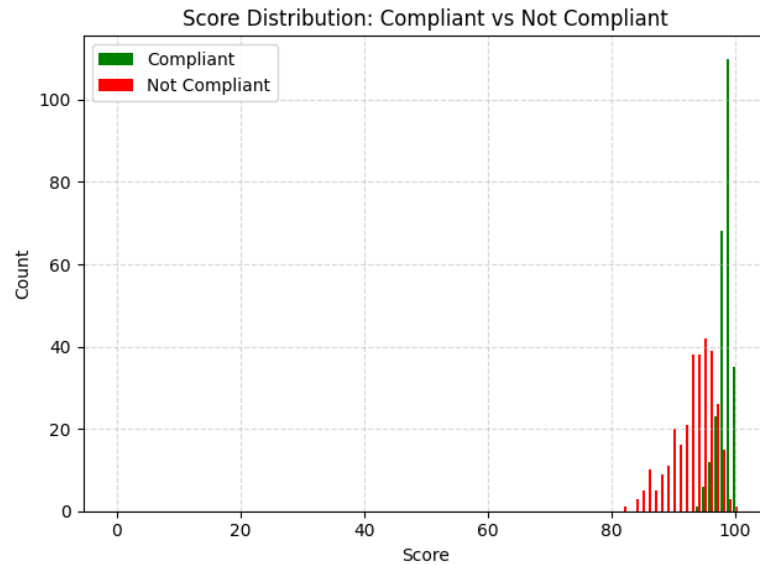


Figure 6.7: Distribuzione degli score per la metrica del rilevamento dello sguardo frontale (metodo algoritmico) valutata sui dataset TONO (cartella la_2) e ONOT

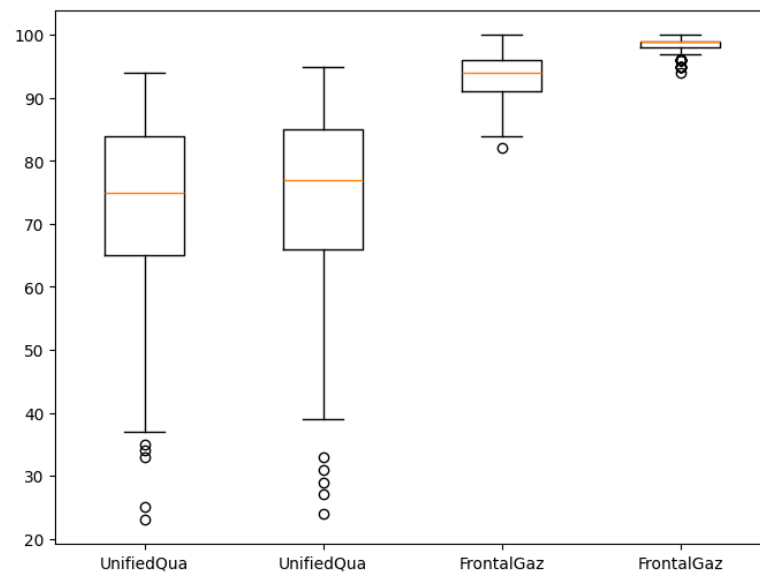


Figure 6.8: Box Plot con la metrica del rilevamento dello sguardo frontale (metodo algoritmico) valutata sui dataset TONO (cartella la_2) e ONOT, metriche valutate (da sinistra verso destra): Unified Quality Score e Frontal Gaze

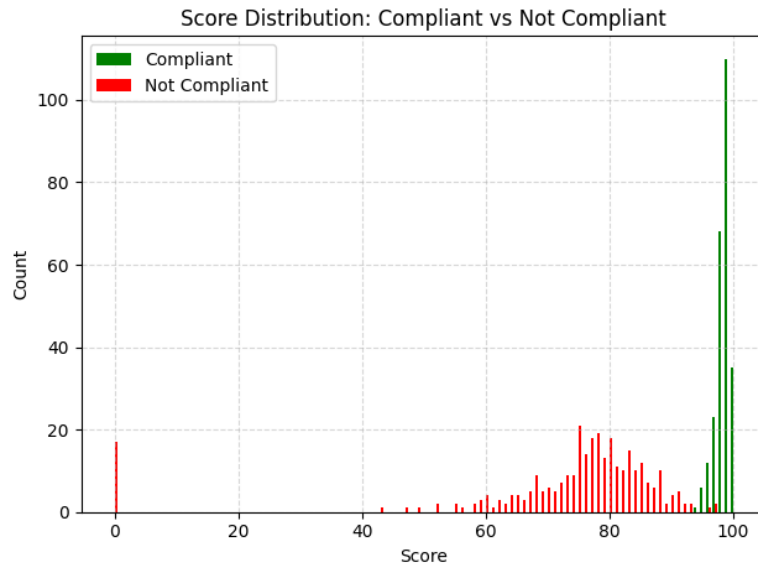


Figure 6.9: Distribuzione degli score per la metrica del rilevamento dello sguardo frontale (metodo L2CS-Net) valutata sui dataset TONO (cartella *la_1*) e ONOT

6.3.5 Valutazione metrica per il rilevamento dello sguardo frontale (metodo L2CS-Net)

Come introdotto precedentemente, i risultati verranno riportati sia per la cartella *la_1* che per la cartella *la_2*. Risultati per la cartella *la_1*

Equal Error Rate: 0.6%.

Distribuzione degli score: ??

Box Plot comparativo: ??

Risultati per la cartella *la_2*

Equal Error Rate: 3.1%.

Distribuzione degli score: ??

Box Plot comparativo: ??

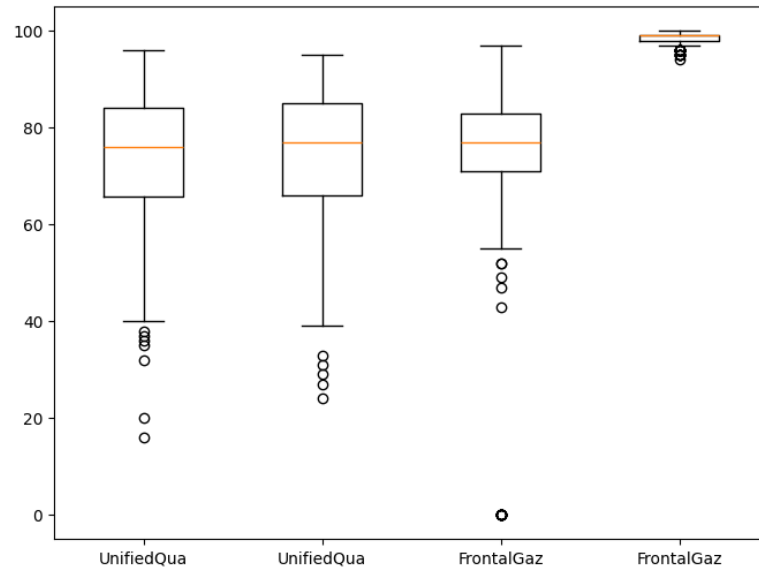


Figure 6.10: Box Plot con la metrica del rilevamento dello sguardo frontale (metodo L2CS-Net) valutata sui dataset TONO (cartella la_1) e ONOT, metriche valutate (da sinistra verso destra): Unified Quality Score e Frontal Gaze

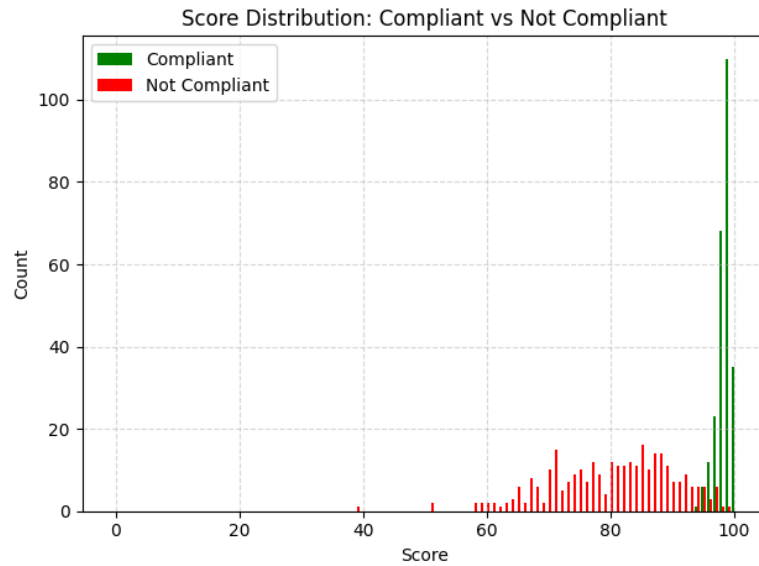


Figure 6.11: Distribuzione degli score per la metrica del rilevamento dello sguardo frontale (metodo L2CS-Net) valutata sui dataset TONO (cartella la_2) e ONOT

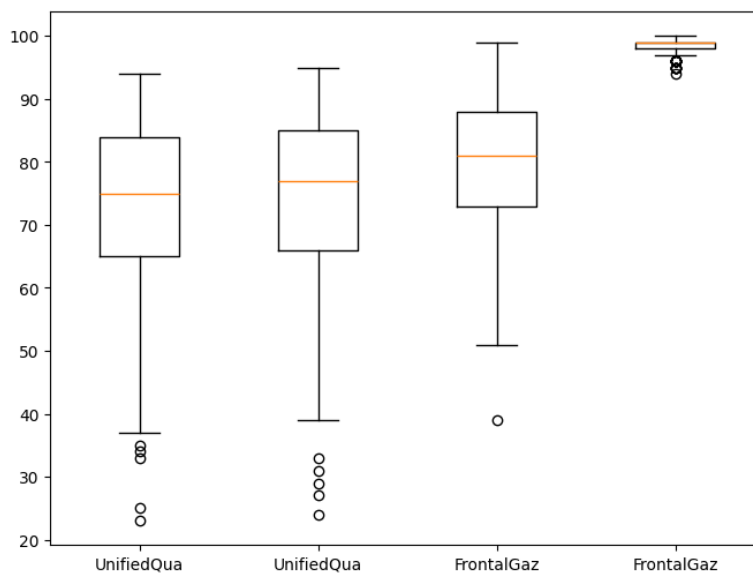


Figure 6.12: Box Plot con la metrica del rilevamento dello sguardo frontale (metodo L2CS-Net) valutata sui dataset TONO (cartella la_2) e ONOT, metriche valutate (da sinistra verso destra): Unified Quality Score e Frontal Gaze

Chapter 7

Contribution

7.1 Fancy formulas here

Bibliography

- [AHK⁺23] Ahmed A. Abdelrahman, Thorsten Hempel, Aly Khalifa, Ayoub Al-Hamadi, and Laslo Dinges. L2cs-net : Fine-grained gaze estimation in unconstrained environments. In *2023 8th International Conference on Frontiers of Signal Processing (ICFSP)*, pages 98–102, 2023.
- [BFDDM24] Guido Borghi, Annalisa Franco, Nicolò Di Domenico, and Davide Maltoni. Tono: a synthetic dataset for face image compliance to iso/icao standard. In *The 18th European Conference on Computer Vision Workshops 2024*, 2024.
- [DCF⁺09] B. Dorizzi, R. Cappelli, M. Ferrara, D. Maio, D. Maltoni, N. Houmani, S. Garcia-Salicetti, and A. Mayoue. Fingerprint and on-line signature verification competitions at icb 2009. In *Proceedings of the International Conference on Biometrics (ICB)*, pages 725–732, Alghero, Italy, 2009.
- [DDBF⁺24] Nicolò Di Domenico, Guido Borghi, Annalisa Franco, Davide Maltoni, et al. Onot: a high-quality icao-compliant synthetic mugshot dataset. In *The 18th IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pages 1–6, 2024.

Acknowledgements

Optional. Max 1 page.