

Masked and UnMasked Face Similarity

Pasquale Matrone
Università degli Studi di Salerno
Dipartimento di Informatica

Sergio Del Sorbo
Università degli Studi di Salerno
Dipartimento di Informatica

Abstract

Dall'avvento della pandemia di coronavirus, una preoccupazione per gli utenti di iPhone è stata lo sblocco del proprio device mediante Face ID indossando una mascherina sanitaria. Il nostro lavoro è stato lo sviluppo di un tool [1] per quantificare, attraverso l'attribuzione di uno score, la differenza tra uno stesso volto che indossa o non indossa una mascherina, utilizzando un approccio diverso ai modelli di deep learning basati su Reti Siamesi. Il tool in Python sviluppato utilizza ResNet-50, una rete neurale convoluzionale preaddestrata e profonda 50 strati. I risultati ottenuti sono stati buoni, con un'accuracy nei test effettuati di oltre il 90%

1 INTRODUZIONE

Dall'avvento della pandemia di coronavirus, una preoccupazione per gli utenti di iPhone ed in generale per tutti gli utenti abituati a sbloccare il proprio device con il volto è stata relativa allo sblocco del proprio dispositivo mediante Face ID o similari indossando una mascherina sanitaria.

Il nostro lavoro è stato lo sviluppo di un tool in Python per quantificare, attraverso l'attribuzione di uno score, la differenza tra uno stesso volto che indossa o non indossa una mascherina.

Il metodo proposto in sede di discussione dei progetti per risolvere il problema è stato di utilizzare un modello di deep learning basato su Reti Siamesi. In effetti, come descritto in [2], le reti siamesi possono essere applicate a diversi casi d'uso, come il rilevamento di duplicati, la ricerca di anomalie e il riconoscimento dei volti. In breve, una rete neurale siamese (o SNN da Siamese Neural Network) è una classe di architetture di reti neurali che contengono due o più sotto-reti identiche. Con il termine "identiche" si intende che hanno la stessa configurazione con gli stessi parametri e pesi. L'aggiornamento dei parametri viene rispecchiato in entrambe le sotto-reti e utilizzato per trovare somiglianze tra gli input confrontando i relativi vettori di caratteristiche. Le SNN utilizzano una funzione di similarità per confrontare i due input e dare in output uno score di similarità

Una parte significativa del nostro lavoro è stata la ricerca di un metodo alternativo a quello proposto, finalizzato ad avere più soluzioni per lo stesso problema di partenza, essenziale per confrontare sistemi diversi nati per risolvere lo stesso problema.

Il tool Python sviluppato utilizza ResNet-50, una rete neurale convoluzionale preaddestrata e profonda 50 strati, per calcolare gli embeddings delle due immagini di input. In poche parole, un'embedding di un'immagine [3] è una rappresentazione vettoriale densa dell'immagine. Alla fine, dati i due vettori di attributi numerici che rappresentano gli embeddings delle due immagini di input, il livello di similarità tra di loro è espresso utilizzando la similarità del coseno. L'output della funzione di similarità del coseno è uno score: se lo score è inferiore

ad una certa soglia, i due volti sono etichettati come appartenenti alla stessa persona, se lo score è superiore alla soglia, allora i due volti sono etichettati come appartenenti a persone diverse.

Nella sezione successiva, faremo una breve rassegna di tutti gli argomenti trattati e utili per comprendere al meglio il lavoro svolto. Successivamente, parleremo in maniera approfondita del sistema proposto e mostreremo i risultati sperimentali ottenuti. L'ultimo paragrafo sarà dedicato alle conclusioni e considerazioni finali.

2 RELATED WORKS

In questa sezione, viene fatta una breve rassegna di tutti gli argomenti trattati e utili per comprendere al meglio il lavoro svolto:

- (1) filtro gaussiano
- (2) face detection e MTCNN
- (3) embedding di un'immagine
- (4) rete neurale convoluzionale e ResNet-50
- (5) image similarity e similarità del coseno
- (6) metriche di valutazione delle prestazioni

2.1 Filtro gaussiano

Il funzionamento del filtro gaussiano è simile al filtro mediano, ma si differenzia per i pesi della maschera. I pesi sono distribuiti in maniera significativa verso il centro e in maniera meno significativa verso la periferia. Dopo diversi esperimenti si è visto che il Gaussiano è quello che preserva di più le forme riducendo il rumore.

2.2 Face detection

Il primo passaggio in ogni algoritmo di analisi del volto (face alignment, face modelling, face relighting, face recognition, face verification/authentication, head pose tracking, facial expression tracking e tanti altri) è la face detection. Attraverso questa tecnica l'algoritmo è in grado di riconoscere all'interno dell'immagini il volto sulla quale dovrà elaborare le informazioni, obiettivo tutt'altro che semplice, basti pensare che il primo step è proprio quello di essere in grado di comprendere se nel campione è effettivamente presente un volto da analizzare oppure no. Per poter riconoscere un volto bisogna considerare un notevole numero di possibilità e di fattori che potrebbero influenzare la qualità dei risultati, come la posizione, la luce, l'espressione, l'occlusione, ecc. A causa dell'importanza del problema della face detection, sono numerosi gli studi e le librerie sull'argomento che hanno portato a diverse alternative e possibilità. Nel nostro tool in Python di face-similarity, abbiamo utilizzato MTCNN [4].

MTCNN è un'implementazione del rilevatore di volti MTCNN per Keras in Python3.4+. È stata scritta da zero, utilizzando come riferimento l'implementazione di MTCNN di David Sandberg (FaceNet's

MTCNN) in Facenet. Si basa sull'articolo di Zhang, K et al. (2016) [5].

2.3 Image embeddings

Un'embedding è una rappresentazione lower-dimensional dell'immagine. In altre parole, è una rappresentazione vettoriale densa dell'immagine che può essere utilizzata per molti compiti come la classificazione. Per costruire l'embedding di un'immagine, possiamo utilizzare una qualsiasi rete convoluzionaria come ResNet-50 o EfficientNet [6]. Nel nostro lavoro, abbiamo utilizzato proprio ResNet-50, di cui forniremo maggiori dettagli nel sottoparagrafo successivo.

2.4 Rete neurale convoluzionale

Nell'apprendimento automatico, una rete neurale convoluzionale (CNN o ConvNet dall'inglese convolutional neural network) è un tipo di rete neurale artificiale feed-forward che hanno diverse applicazioni nel riconoscimento di immagini e video, nei sistemi di raccomandazione, nell'elaborazione del linguaggio naturale e, recentemente, in bioinformatica.

ResNet-50 [7] è una rete neurale convoluzionale profonda 50 strati. È possibile caricare una versione pre-addestrata della rete, addestrata su oltre un milione di immagini del database ImageNet. La rete preaddestrata è in grado di classificare le immagini in 1000 categorie di oggetti, come tastiera, mouse, matita e molti animali. Di conseguenza, la rete ha appreso rappresentazioni ricche di caratteristiche per un'ampia gamma di immagini. La rete ha una dimensione di input dell'immagine di 224x224.

2.5 Image similarity

L'image similarity, in ambito computer vision, significa essere in grado di calcolare un valore che sia in grado di dimostrare quanto due o più immagini in esame si assomigliano, in questo modo è possibile trarre specifiche conclusioni come l'appartenenza allo stesso gruppo (istanze della stessa classe) o, nel nostro lavoro, autenticare il volto di una persona. Per poter calcolare questo valore possiamo applicare una funzione di similarità, nel nostro progetto abbiamo utilizzato la similarità del coseno.

La similarità del coseno [8], o cosine similarity, è una tecnica euristica per la misurazione della similitudine tra due vettori effettuata calcolando il coseno tra di loro. Dati due vettori di embeddings di immagini, A e B, il livello di similarità tra di loro è espresso utilizzando la formula:

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}.$$

Un altro modo di indicare la formula, del tutto equivalente, è:

$$\frac{\sum_{k=1}^n A(k)B(k)}{\sqrt{\sum_{k=1}^n A(k)^2} \sqrt{\sum_{k=1}^n B(k)^2}}$$

Il valore di similitudine così definito è compreso tra 0 e +1, dove 0 indica una corrispondenza opposta e +1 indica due vettori uguali. Infatti, la dissimilitudine tra i due vettori A e B è data da $1 - \text{Cos}(A, B)$.

2.6 Metriche di valutazione delle prestazioni

Una matrice di confusione è semplicemente una matrice quadrata che rileva il conteggio dei veri positivi e dei veri negativi, dei falsi positivi e dei falsi negativi, come illustrato dalla Figura 1. Le metriche di valutazione delle prestazioni utilizzate nel nostro lavoro sono le seguenti:

- (1) accuracy
- (2) recall
- (3) precision
- (4) F1-score

2.6.1 Accuracy. L'Accuracy, o accuratezza, è una metrica molto usata, immediatamente comprensibile e spesso valida. Essa indica la percentuale di classificazioni corrette.

2.6.2 Recall. La recall è una metrica molto usata che indica il rapporto di istanze positive correttamente individuate dal sistema.

2.6.3 Precision. La precision è una metrica molto usata e indica l'accuratezza con cui un sistema prevede le classi positive.

2.6.4 F1-score. In pratica, spesso viene utilizzata una combinazione di precisione e recall, il cosiddetto punteggio F1-score.

		Classe prevista	
		P	N
Classe effettiva	P	Veri positivi (TP)	Falsi negativi (FN)
	N	Falsi positivi (FP)	Veri negativi (TN)

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{F1} = 2 \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Figure 1: Metriche di valutazione delle prestazioni

3 SISTEMA PROPOSTO

Il nostro obiettivo è stato implementare un tool in Python per quantificare attraverso l'attribuzione di uno score la differenza tra uno stesso volto che indossa o non indossa una mascherina. In altre parole, la nostra applicazione, date due immagini in input, riconosce il grado di somiglianza dei volti dei soggetti rappresentati in modo da valutare se si tratta della stessa persona o di persone distinte. Le tecniche e la strategia utilizzate per lo sviluppo del progetto verranno illustrate in questa sezione (Figura 2).

3.1 Resize e Face Detection

Come già detto nelle sezioni precedenti, il modello di rete neurale utilizzato è stato ResNet-50. Poiché la rete ResNet-50 prende in

input immagini di taglia 224x224, il primo passo da eseguire è il resize delle immagini alla size desiderata.

Per ridurre il rumore preservando le forme, è possibile applicare un filtro gaussiano alle immagini in input.

Il passaggio successivo, se le immagini in input non sono solamente volti, consiste nella detection del volto tramite l'algoritmo Multi-Task Cascaded Convolutional Networks (MTCNN) che restituisce in output i valori dei pixel (rettangolo) della porzione di immagine contenente il volto del soggetto. In altre parole, MTCNN restituisce le coordinate del volto all'interno dell'immagine. Esistono diversi face detector, come DLIB, RetinaFace, MediaPipe, ognuno con le proprie caratteristiche. La nostra scelta in questo lavoro è ricaduta su MTCNN perchè è in grado di performare in tempi accettabili anche senza sfruttare la GPU. Successivamente, le immagini del volto sono passate al nostro modello di CNN.

3.2 Calcolo embeddings delle immagini

Come già detto, un'embedding è una rappresentazione lower-dimensional di un'immagine. In altri termini, un'embedding di un'immagine è una rappresentazione vettoriale densa dell'immagine stessa. Per costruire l'embedding di un'immagine, nel nostro lavoro abbiamo utilizzato ResNet-50. ResNet-50 è una rete neurale convoluzionale pre-addestrata su oltre un milione di immagini del database ImageNet. ResNet-50, sfruttando cinquanta layer, fornisce dei risultati accurati. In sostanza, prese in input due immagini di volto, il nostro modello di rete convoluzionale ResNet-50 restituisce in output gli embeddings delle stesse.

3.3 Image similarity

L'ultimo passaggio consiste nel passare in input alla funzione di similarità del coseno i due embeddings rappresentanti i due volti. Come già detto, la funzione coseno restituirà uno score compreso tra 0 e +1, dove 0 indica una corrispondenza opposta e +1 indica due vettori uguali. Infatti, la dissimilarità tra i due embeddings (vettori) A e B è data da $1 - \text{Cos}(A, B)$. Nel nostro lavoro, abbiamo impostato una soglia a 0,5 e se

- (1) score <= soglia, si è verificato un match tra i due volti
- (2) non è un match, altrimenti

Un semplice esempio di applicazione della funzione di similarità del coseno per trovare la somiglianza tra due vettori A e B è mostrato nella Figura 2 [9].

DATASETS

Per verificare le prestazioni del modello di face similarity proposto in questo lavoro, sono stati utilizzati i dataset MASKED AND UNMASKED [10] e M2FREDFACE [11]. Il dataset MASKED AND UNMASKED contiene 6363 immagini di volti di persone senza mascherina e 6363 immagini di volti delle stesse persone ma con una mascherina aggiunta in digitale. I volti presentano un'elevata variabilità di occlusioni e mascherine di colori e stampe diverse. M2FREDFACE contiene 16 video (in un caso solo 12) in condizioni di posa e sfondo diversi di 43 persone senza mascherina e 16 video delle stesse persone nella stessa condizione di posa e sfondo ma con la mascherina per un totale di 1368 video. Per ogni persona, abbiamo recuperato due frame diversi per posa e sfondo con e senza mascherina.

$$A = \{ 3, 2, 0, 5 \}$$

$$B = \{ 1, 0, 0, 0 \}$$

$$\frac{\sum_{k=1}^n A(k)B(k)}{\sqrt{\sum_{k=1}^n A(k)^2} \sqrt{\sum_{k=1}^n B(k)^2}}$$

$$A \cdot B = 3*1 + 2*0 + 0*0 + 5*0 = 3$$

$$||A|| = \sqrt{(3)^2 + (2)^2 + (0)^2 + (5)^2} = 6.16$$

$$||B|| = \sqrt{(1)^2 + (0)^2 + (0)^2 + (0)^2} = 1$$

$$\diamond \quad \text{Cos}(A, B) = 3 / (6.16 * 1) = 0.49$$

La dissimilarità tra i due vettori A e B è data da:

$$\diamond \quad \text{Dis}(A, B) = 1 - \text{Cos}(A, B) = 1 - 0.49 = 0.51$$

Figure 2: Esempio funzione di similarità del coseno

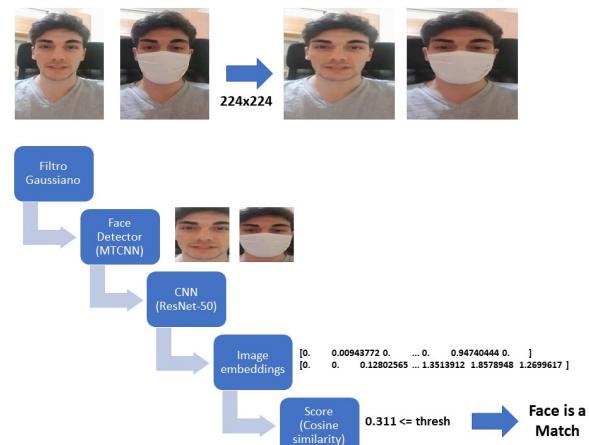


Figure 3: Processo di Face-Similarity

		CLASSI PREVISTE	
		SI	NO
CLASSI EFFETTIVE	SI	Stessa persona classificata correttamente (TP)	Diversa persona classificata scorrettamente (FN)
	NO	Stessa persona classificata scorrettamente (FP)	Diversa persona classificata correttamente (TN)

Figure 4: Matrice di Confusione

4 RISULTATI Sperimentali

In questa sezione vengono presentati i risultati raggiunti in termini di metriche come accuracy, precision, recall e F1-score. La Figura 4 mostra la matrice di confusione per il nostro modello.

4.1 Esperimento 1

Il primo set di test è stato condotto sul dataset MASKED AND UNMASKED. Questa scelta è motivata dalla possibilità di testare l'algoritmo sviluppato su un numero notevole di immagini che non presentano particolari differenze visive: per ogni individuo, è presente un'immagine senza mascherina ed un'altra con una mascherina aggiunta in digitale. Per questo motivo, tra le due immagini di volto della stessa persona senza e con mascherina non risultano notevoli differenze dovute all'illuminazione, alla postura, all'espressione, ecc. Per questo particolare dataset, data la caratteristica delle immagini che lo compongono, prive di sfondo ma già concentrate sul volto del soggetto, si è deciso di non ricorrere alla face detection in quanto avrebbe solo rischiato di compromettere i risultati qualora il detector non fosse in grado di riconoscerlo. Ogni volto di persona con mascherina è stato confrontato sia con il volto della stessa persona senza mascherina sia con il volto di una persona diversa senza mascherina.

Il set di test sono organizzati in due sessioni

- (1) le immagini in input **non** sono pre-elaborate tramite un filtro gaussiano (E1-S1)
- (2) le immagini in input sono pre-elaborate tramite un filtro gaussiano (E1-S2)

Il grafico rappresentante l'esito dei confronti ed i risultati delle metriche di valutazione delle prestazioni per E1-S1 e E1-S2 sono mostrati nella sezione successiva.



Figure 5: Esempio E1

4.2 Esperimento 2

Il secondo test effettuato ha utilizzato le immagini presenti nel dataset M2FREDFACE per confermare l'efficacia del sistema anche in contesti in cui sono presenti variazioni tra le immagini date dalla postura, l'illuminazione, l'espressione etc. In questo caso è imperativo l'impiego della face detection per poter escludere tutti gli elementi non rilevanti come lo sfondo, che avrebbe inciso negativamente sul valore di similarità calcolato. Prima di procedere all'effettivo impiego del tool per riconoscere il soggetto sulla base di due immagini con e senza mascherina, si è preferito osservare i risultati mantenendo una condizione favorevole per il sistema: le immagini confrontate prevedono solo soggetti privi di mascherina o solo soggetti che indossano una mascherina. Per tale motivo i test eseguiti sono i seguenti:

- (1) le immagini in input **non** presentano alcuna mascherina (E2-S1)
- (2) le immagini in input prevedono una mascherina correttamente indossata (E2-S2)

Il grafico rappresentante l'esito dei confronti ed i risultati delle metriche di valutazione delle prestazioni per E2-S1 e E2-S2 sono mostrati nella sezione successiva.



Figure 6: Esempio E2

4.3 Esperimento 3

Confermato il funzionamento del tool, il terzo test prevede di raggiungere l'obiettivo preposto: confrontare due immagini raffigurante la stessa persona (o diversa) con e senza la mascherina, ottenendo un risultato che confermi o meno l'identità. Anche in questo caso l'input è dato dal Dataset M2FREDFACE ma le immagini utilizzate raffigurano il soggetto con e senza la mascherina in un contesto molto simile: le immagini sono state catturate nello stesso momento, prima senza alcuna mascherina, dopo indossandone una; questo comporta che le immagini rappresentano un livello molto simile non solo di luminosità e fattori ambientali, ma anche la postura e l'angolazione del soggetto è invariata. Questo test è stato diviso in quattro sessioni, annotando per ognuna i risultati ottenuti così da poterli confrontare.

- (1) le immagini in input sono prive da qualunque pre-elaborazione o modifica (E3-S1)
- (2) le immagini in input sono pre-elaborate mediante un filtro gaussiano (E3-S2)

- (3) le immagini in input sono pre-elaborate riducendo le informazioni alla sola zona periocularare (E3-S3)
- (4) le immagini in input hanno subito una pre-elaborazione che ha ridotto le informazioni da elaborare alla sola zona periocularare e in seguito è stato applicato un filtro gaussiano(E3-S4)

In seguito ai risultati ottenuti, un ulteriore test è stato eseguito (E3-S5) aumentando il threshold a 0.6.

Il grafico rappresentante l'esito dei confronti ed i risultati delle metriche di valutazione delle prestazioni per E3-S1, E3-S2, E3-S3, E3-S4, E3-S5 sono mostrati nella sezione successiva.



Figure 7: Esempio E3

4.4 Esperimento 4

L'ultimo set di esperimenti eseguiti in questo lavoro riguarda le immagini del Dataset M2FREDFACE, scegliendo soggetti con e senza la mascherina e in diversi momenti della giornata (frame diversi del video), in questo modo da differenziarsi dal precedente esperimento, aggiungendo elementi diversi dati dall'ambiente e dalla postura del soggetto. Anche in questo caso sono state eseguite quattro diverse sessioni per osservare il comportamento e i risultati del tool su immagini diversamente pre-elaborate.

- (1) le immagini in input sono prive da qualunque pre-elaborazione o modifica (E4-S1)
- (2) le immagini in input sono pre-elaborate mediante un filtro gaussiano (E4-S2)
- (3) le immagini in input sono pre-elaborate riducendo le informazioni alla sola zona periocularare (E4-S3)
- (4) le immagini in input hanno subito una pre-elaborazione che ha ridotto le informazioni da elaborare alla sola zona periocularare e in seguito è stato applicato un filtro gaussiano(E4-S4)

Anche per questo esperimento è stata aggiunta una quinta sessione (E4-S5) aumentando il valore di threshold a 0.6 .

Il grafico rappresentante l'esito dei confronti ed i risultati delle metriche di valutazione delle prestazioni per E4-S1, E4-S2, E4-S3, E4-S4, E4-S5 sono mostrati nella sezione successiva.

5 DISCUSSIONE DEI RISULTATI

In questo capitolo vengono mostrati tutti i risultati ottenuti illustrando per ogni test il proprio grafico e i valori delle metriche di valutazione delle prestazioni calcolati. In particolare il grafico presenta sull'asse delle ascisse (x) il passo di iterazione dei confronti e



Figure 8: Esempio E4

sull'asse delle ordinate (y) il valore dello score di similarity ottenuto. Viene utilizzato un pallino **verde** per il confronto tra immagini raffigurante la stessa persona e un pallino **rosso** per il confronto tra immagini raffiguranti persone diverse. Il valore di threshold (la soglia che sancisce se lo score è da considerarsi un match oppure no) è impostato a 0.5 ed è raffigurato da una linea che divide il grafico a metà. Il capitolo è diviso in sotto-sezioni, così come il capitolo precedente, e per ognuno si analizzano i risultati ottenuti sulla base dell'accuracy, al termine di questo è illustrata una tabella che racchiude anche tutti gli altri parametri calcolati(True Positive, False Positive, True Negative, False Negative, Accuracy, Recall, Precision, F1-Score).

5.1 Esperimento 1

La prima sessione di questo esperimento (E1-S1) ha mostrato un valore di accuracy pari al **91.7%**. La seconda sessione (E1-S2) invece, mostra un valore di accuracy pari al **92%**. Significa dire che l'applicazione di un filtro guassiano non ha portato notevoli differenze, probabilmente a causa della natura delle immagini che soddisfa la qualità richiesta per un adeguato confronto.

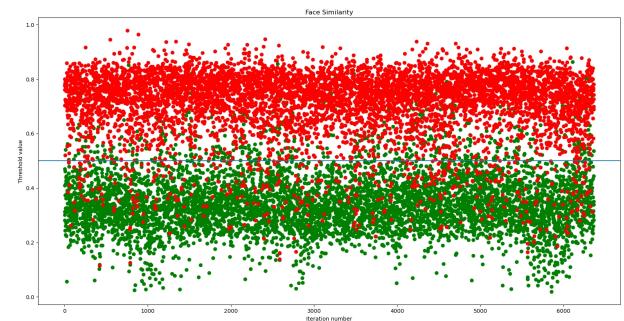


Figure 9: E1-S1

5.2 Esperimento 2

La prima sessione di questo esperimento (E2-S2) ha mostrato un valore di accuracy pari al **90.0%**. La seconda sessione (E2-S2) invece, mostra un valore di accuracy pari al **83.7%**. E' facile quindi notare come la presenza o meno della mascherina incida sul risultato finale. La causa è da attribuirsi alle diverse mascherine che i

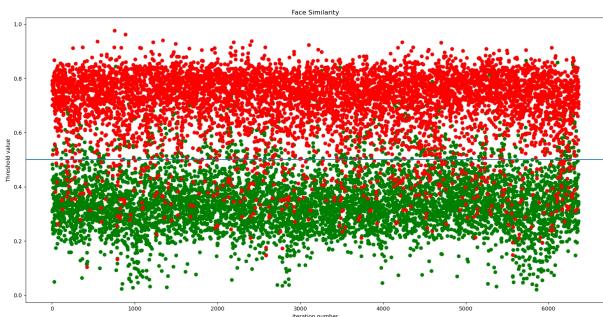


Figure 10: E1-S2

soggetti possono indossare quando avviene il confronto: trattandosi di un confronto fra immagini, ogni elemento rilevato come appartenente al volto di fatto diventa una caratteristica distintiva che può incrementare oppure diminuire il valore di similarità calcolato.

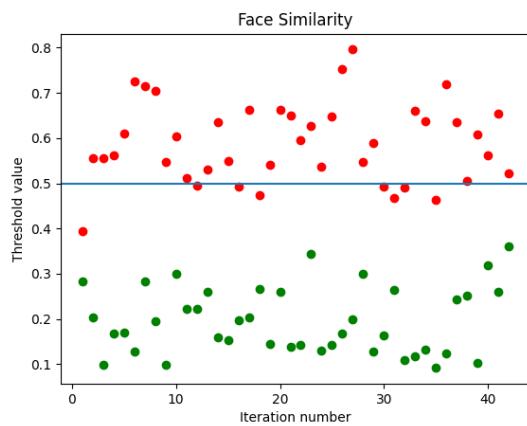


Figure 11: E2-S1

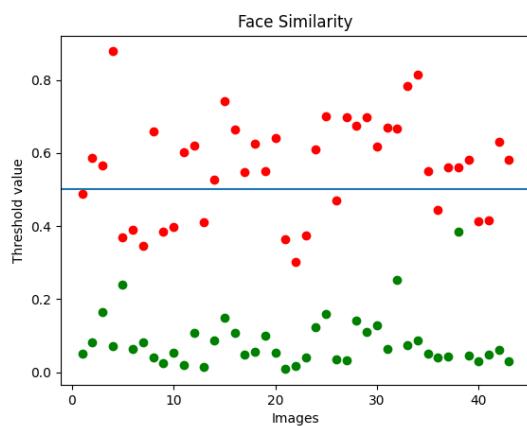


Figure 12: E2-S2

5.3 Esperimento 3

La prima sessione di questo esperimento (E3-S1) ha mostrato un valore di accuracy pari al **68.6%**. La seconda sessione (E3-S2) mostra un valore di accuracy pari al **77.9%**. La terza sessione (E3-S3) mostra un valore di accuracy pari al **94.1%**. La quarta sessione (E3-S4) mostra un valore di accuracy pari al **94.1%**. In questo caso il tool è stato testato valutando diverse possibilità. Si nota immediatamente come il solo confronto tra le immagini, prive da ogni pre-elaborazioni, producano dei risultati decisamente inferiori ai precedenti test. Tuttavia, applicare un filtro gaussiano permette di migliorare, anche se non considerevolmente, l'accuracy ottenuta, segno di una dipendenza alla qualità dell'immagine. Il miglior risultato è stato rilevato attraverso una riduzione delle informazioni analizzate: scegliendo di considerare la sola zona perioculare di ogni soggetto sono diminuiti gli elementi confrontabili, migliorando la qualità dei risultati. Infine si nota come l'impiego di entrambe le strategie: zona perioculare e filtro gaussiano, non abbia prodotto rilevanti differenze nei risultati. Il quinto test di questo esperimento ha prodotto un accuracy pari a **86.0%**, come prova del fatto che in alcuni contesti il valore di similarità calcolato non sia di molto superiore alla soglia standard di 0.5 stabilita e che quindi, in precise occasioni e per precise immagini, è possibile anche impostare una soglia più alta così da ottenere risultati migliori.

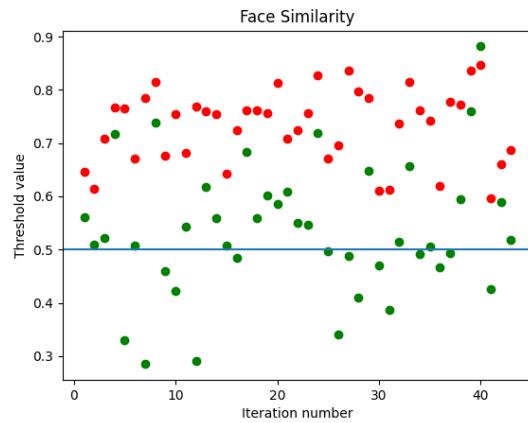


Figure 13: E3-S1

5.4 Esperimento 4

La prima sessione di questo esperimento (E4-S1) ha mostrato un valore di accuracy pari al **61.9%**. La seconda sessione (E4-S2) mostra un valore di accuracy pari al **63.0%**. La terza sessione (E4-S3) mostra un valore di accuracy pari al **91.6%**. La quarta sessione (E4-S4) mostra un valore di accuracy pari al **90.4%**. I risultati di questo esperimento confermano che il tool produce risultati positivi anche in contesti poco favorevoli, anche senza necessariamente applicare un filtro gaussiano prima del confronto fra le immagini. Anche in questo caso la scelta di confrontare solo la parte perioculare di ogni soggetto mostra un esito migliore rispetto all'interezza del volto, inoltre il filtro gaussiano rischia di peggiorare quello che è l'esito finale del confronto. La quinta sessione (E4-S5) mostra un'accuracy

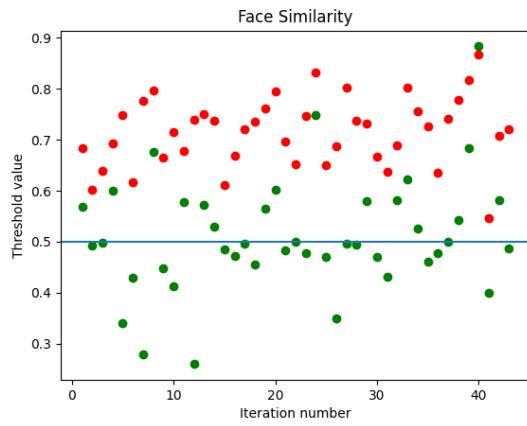


Figure 14: E3-S2

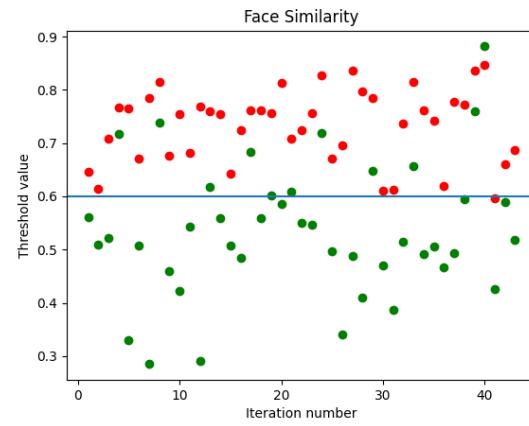


Figure 17: E3-S5

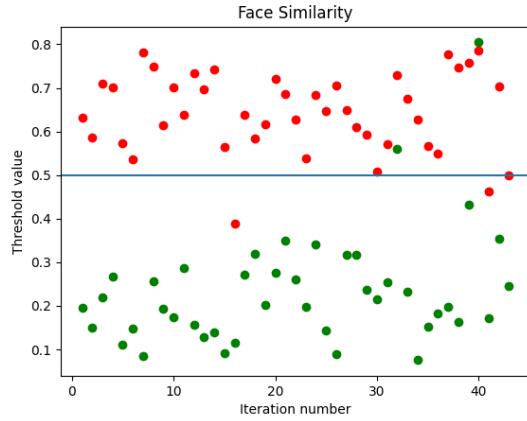


Figure 15: E3-S3

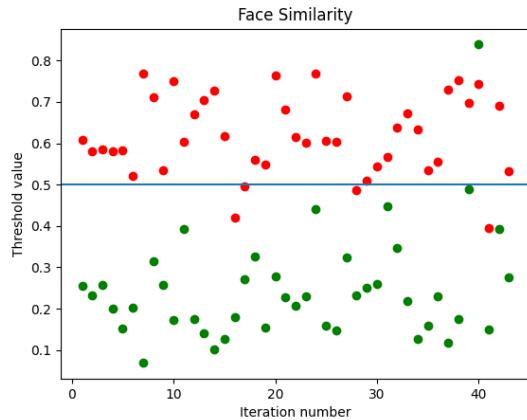


Figure 16: E3-S4

pari al **78.5%**, che confrontato al risultato ottenuto durante la prima sessione di questo esperimento, conferma nuovamente come sia possibile cambiare il valore di treshold per ottenere dei risultati migliore, accettando ovviamente il rischio dell'aumento dei valori di False Positive.

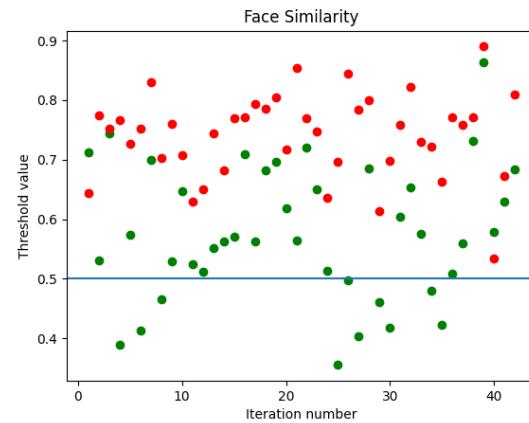


Figure 18: E4-S1

6 CONCLUSIONI

La pandemia di coronavirus ha evidenziato come sia importante quantificare la differenza tra uno stesso volto che indossa o non indossa una mascherina. Per risolvere il problema utilizzando un approccio differente da un modello di deep learning basato su reti siamesi, il presente lavoro si è concentrato sulla ricerca di un metodo alternativo a quello proposto in sede di presentazione dei progetti, finalizzato ad avere più soluzioni per lo stesso problema di partenza, essenziale per confrontare sistemi diversi nati per risolvere lo stesso problema. Il nostro algoritmo utilizza la rete neurale convoluzionale pre-addestrata ResNet-50 per calcolare gli embeddings delle due

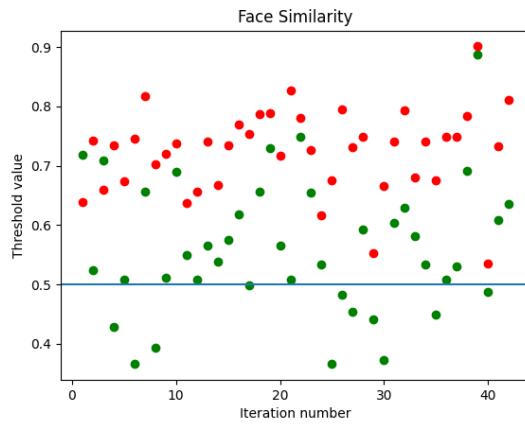


Figure 19: E4-S2



Figure 22: E4-S5

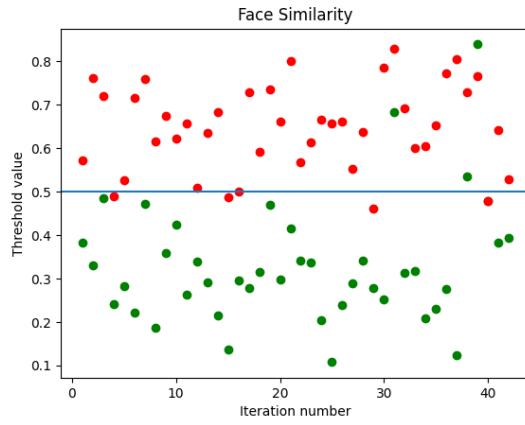


Figure 20: E4-S3

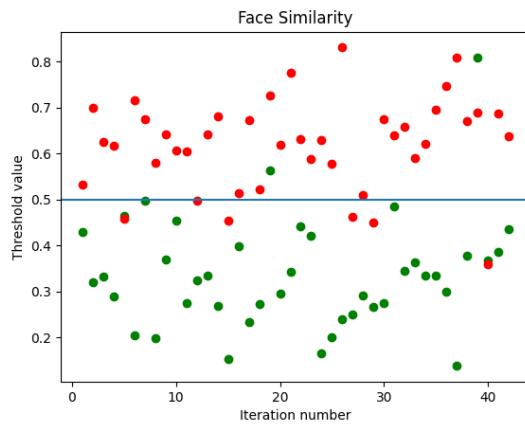


Figure 21: E4-S4

S	TP	FP	TN	FN	Acc	Recall	Prec	F1
E1-S1	5753	610	5922	441	0,917	0,928	0,904	0,916
E1-S2	5798	565	5912	451	0,920	0,927	0,911	0,919
E2-S1	42	0	34	8	0,904	0,840	1,0	0,860
E2-S2	43	0	29	14	0,837	0,754	1,0	0,913
E3-S1	16	27	43	0	0,686	1,0	0,372	0,542
E3-S2	24	19	43	0	0,779	1,0	0,558	0,716
E3-S3	41	2	40	3	0,941	0,931	0,953	0,942
E3-S4	42	1	39	4	0,941	0,913	0,976	0,943
E3-S5	32	11	42	1	0,860	0,969	0,744	0,842
E4-S1	10	32	42	0	0,619	1,0	0,238	0,384
E4-S2	11	31	42	0	0,630	1,0	0,261	0,415
E4-S3	39	3	38	4	0,916	0,906	0,928	0,917
E4-S4	40	2	36	6	0,904	0,869	0,952	0,909
E4-S5	25	17	41	1	0,785	0,961	0,595	0,735

Table 1: Metriche di prestazione

immagini di volto in input e successivamente la similarità del coseno per calcolare il livello di similarità tra di loro.

I risultati sperimentali verificano che utilizzando una soglia di 0,5, l'accuratezza nello stabilire se due volti appartengono o meno alla stessa persona è superiore al 91%. Inoltre, la velocità dell'algoritmo proposto in questo lavoro per solo due immagini di input è accettabile.

Sarebbe interessante utilizzare per generare gli embeddings dell'immagine un modello differente da ResNet-50, in particolare la rete neurale convoluzionale a 16 layers VGG-16 o la rete neurale convoluzionale EfficientNet B0, quest'ultima particolarmente veloce pur mantenendo una discreta precisione.

REFERENCES

- [1] Pasquale Matrone e Sergio Del Sorbo. 2022. *Face-Similarity*. <https://github.com/pasqualematrone/Face-Similarity>, (May 2022).
- [2] Hazem Essam and Santiago L. Valdarrama. 2021. *Image similarity estimation using a Siamese Network with a triplet loss*.

- [https://keras.io/examples/vision/siamese_{network}/](https://keras.io/examples/vision/siamese_network/), (March 2021).
- [3] Margaux Masson-Forsythe. 2021. *Generate image embeddings using a pre-trained CNN and store them in Hub*. <https://www.activeloop.ai/hub/margauxmassonforsythe/generate-image-embeddings> (September 2021).
- [4] Justin Shenk. 2022. *Implementation of the MTCNN face detector for Keras in Python3.4+*. <https://pypi.org/project/mtcnn/>.
- [5] Zhang Z. Li Z. Zhang K. and Y. Qiao. 2016. *Joint face detection and alignment using multitask cascaded convolutional networks*. IEEE Signal Processing Letters, 23(10):1499–1503.
- [6] Romain Beaumont. 2020. *Image embeddings*. <https://rom1504.medium.com/>, (July 2020).
- [7] MathWorks Inc. 2022. *ResNet-50*. <https://it.mathworks.com>.
- [8] Wikipedia. 2022. *Cosine similarity*. <https://en.wikipedia.org>.
- [9] Sharada Rao. 2020. *Cosine similarity example*. <https://www.geeksforgeeks.org/cosine-similarity-example/>.
- [10] Muhammed Dalkiran. 2020. *Masked - Unmasked Face Recognition*. Kaggle.
- [11] Unisa. 2020. *M2FRED_{FACE}*.