Riconoscimento di volti

M. Maresca, V. Riccardi, M. Russo, A. Sposito

Progetto ESM – gruppo 11

(a.a. 2019-2020)

1. Introduzione

Il riconoscimento dei volti è il processo di identificazione di una o più persone in immagini o video tramite l'analisi e il confronto di pattern. In genere, gli algoritmi utilizzati estraggono dei descrittori locali dai volti e le confrontano con un database, che deve essere il più ampio possibile, per trovare la migliore corrispondenza. Questi descrittori sono in grado di descrivere il comportamento statistico di patch molto piccole in termini di istogrammi estratti dall'insieme delle patch dell'immagine. Gli istogrammi che rappresentano i vettori di feature sono poi usati per addestrare un classificatore.
Il problema principale per il riconoscimento dei volti è quello di individuare un descrittore compatto e discriminativo che risulti robusto ad ampie variazioni di illuminazione, posa, espressioni facciali, età, occlusioni parziali. Nella letteratura scientifica sono presenti numerosi studi sul riconoscimento dei volti, quest'ultimo è parte importante di molti sistemi di biometrica, sicurezza e sorveglianza, nonché in sistemi di indicizzazione di immagini e video.

2. Approccio implementato

Abbiamo implementato il lavoro proposto in [1], esso è suddiviso in tre fasi: pre-processing, estrazione delle feature, riconoscimento. Lo schema a blocchi del processo è rappresentato in figura 1.

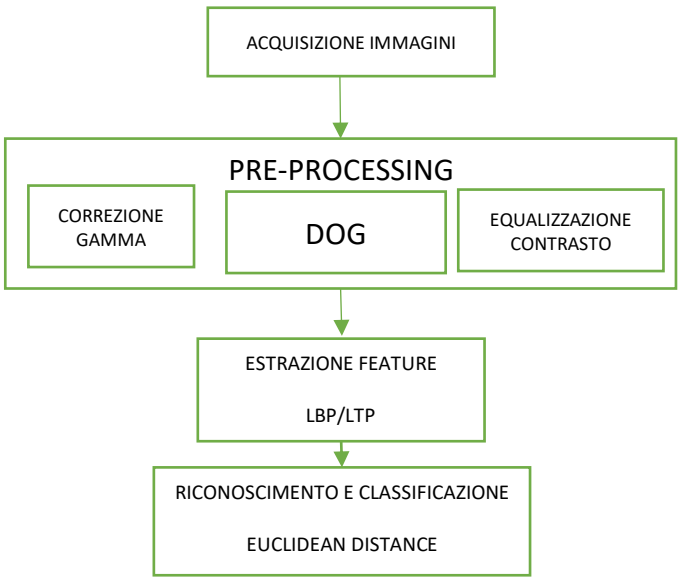


Figura 1.

2.1 Pre-processing

La fase di preelaborazione delle immagini ha lo scopo di contrastare gli effetti delle variazioni di illuminazione e dell'ombreggiatura locale, pur conservando gli elementi essenziali per il riconoscimento.

Nel dettaglio, i passaggi sono i seguenti: correzione gamma, differenza di gaussiane, equalizzazione del contrasto.

2.1.1 Correzione gamma.

Essa è una trasformazione non lineare che sostituisce il livello di grigio I con I^{γ} (per $\gamma > 0$) dove $\gamma \in [0,1]$ è un parametro definito dall'utente. Ha l'effetto di migliorare il contrasto dell'immagine in regioni scure o ombreggiate, mentre comprime la dinamica locale in regioni luminose.

Il principio di base è che l'intensità della luce riflessa da un oggetto è il prodotto dell'illuminazione in entrata L (che è per lo più smussata) e della riflettanza della superficie locale R (che trasporta informazioni dettagliate sull'aspetto dell'oggetto). Vogliamo recuperare le informazioni dell'oggetto indipendentemente dall'illuminazione, e usare il logaritmo semplifica il compito poiché converte il prodotto in una somma: per una costante illuminazione locale, un dato andamento di riflettanza produce un dato andamento nel log (I) indipendentemente dalla intensità effettiva dell'illuminazione. In pratica, però, una trasformazione con il logaritmo è spesso troppo forte, tendendo ad amplificare eccessivamente il rumore nelle regioni scure dell'immagine, ma una legge di potenza con esponente γ nell'intervallo [0, 0,5] è un buon compromesso. Qui usiamo $\gamma = 0.2$ come impostazione predefinita.

2.1.2 Differenza di gaussiane (DoG).

La correzione gamma non rimuove l'influenza dei gradienti di intensità generali come gli effetti di ombreggiatura. L'ombreggiatura indotta dalla struttura della superficie è potenzialmente un utile indizio visivo ma è principalmente un'informazione spaziale a bassa frequenza, che è difficile da separare dagli effetti causati dai gradienti di illuminazione. Il filtro passa-alto rimuove sia le informazioni utili che quelle accidentali, semplificando così il problema del riconoscimento e in molti casi aumentando le prestazioni complessive del sistema. Allo stesso modo, la soppressione delle frequenze spaziali più alte riduce aliasing e rumore, e, in pratica, spesso riesce a farlo senza distruggere troppo il segnale sottostante su cui deve essere basato il riconoscimento. Il filtro DoG è un modo conveniente per ottenere il comportamento passa-banda risultante. I dettagli spaziali fini sono di fondamentale importanza per il riconoscimento, quindi la gaussiana interna (più piccola) è in genere piuttosto stretta ($\sigma 0 \leq 1$ pixel), mentre quella esterna potrebbe avere $\sigma 1$ di 2-4 pixel o più, a seconda della frequenza spaziale alla quale l'informazione a bassa frequenza diventa fuorviante piuttosto che informativa.

Di seguito utilizziamo $\sigma 0 = 1.0$ e $\sigma 1 = 2.0$ per impostazione predefinita.

Se il DoG viene eseguito senza una normalizzazione gamma, le immagini risultanti mostrano chiaramente in che misura il contrasto locale (e quindi i dettagli visivi) è ridotto nelle regioni ombreggiate.

2.1.3 Equalizzazione del contrasto.

Il passaggio finale della preelaborazione dell'immagine ridimensiona globalmente le intensità dei livelli di grigio per standardizzare una solida misura del contrasto complessivo. È importante utilizzare uno stimatore robusto perché il segnale, in genere, contiene ancora una piccola mescolanza di valori estremi prodotti da luci, da imperfezioni ai bordi dell'immagine e da piccole regioni scure come le narici. Il processo è suddiviso in due stadi:

$$I(x,y) \leftarrow \frac{I(x,y)}{(mean(|I(x\prime,y\prime)|^a))^{1/a}}$$

$$I(x,y) \leftarrow \frac{I(x,y)}{(mean(min(\tau,|I(x',y')|)^a))^{1/a}}$$

Nei quali a è un esponente fortemente compressivo che riduce l'influenza dei valori grandi, τ è una soglia utilizzata per troncare i valori grandi dopo la prima fase di normalizzazione e la media è sull'intera immagine. Di default usiamo α = 0.1 e τ = 10.

L'immagine risultante è ora ben scalata ma può ancora contenere valori estremi. Per ridurre la loro influenza sulle fasi successive dell'elaborazione, applichiamo infine una funzione non lineare per comprimere valori troppo grandi. Usiamo la tangente iperbolica

$$I\,(x, y) \leftarrow \tau \tanh (I\,(x, y) / \tau)$$

limitando così I all'intervallo (−τ, τ).

2.2 Estrazione delle feature

Per l'estrazione delle feature abbiamo utilizzato due approcci per poi confrontarne i risultati. Nel primo abbiamo utilizzato l'operatore LBP (Local Binary Pattern). Nel secondo abbiamo utilizzato una variante del LBP, più robusta al rumore, chiamata LTP (Local Ternary Pattern).

Questi sono descrittori locali che codificano il segno della differenza tra un pixel e quelli vicini, essi sono molto semplici e riescono a catturare le microstrutture presenti in un'immagine, inoltre sono robusti a cambiamenti di illuminazione.

L'operatore LBP per ogni pixel x dell'immagine prende un vicinato locale 3x3 ed esegue la seguente elaborazione:

$$y = \sum_{i=0}^{7} u(x_i - x)\, 2^i$$

dove x_i è l'i-esimo pixel del vicinato di x, mentre u(x) = 1 quando $x \geq 0$ e 0 altrimenti.

$$\begin{bmatrix} x_0 & x_1 & x_2 \\ x_7 & x & x_3 \\ x_6 & x_5 & x_4 \end{bmatrix}$$

Il processo di codifica LBP è illustrato in figura 2. Per ogni patch dell'immagine si estrae la sequenza binaria che si ottiene confrontando un pixel con quelli del suo vicinato, quindi il numero binario si converte in numero decimale; si ottiene così l'immagine LBP.

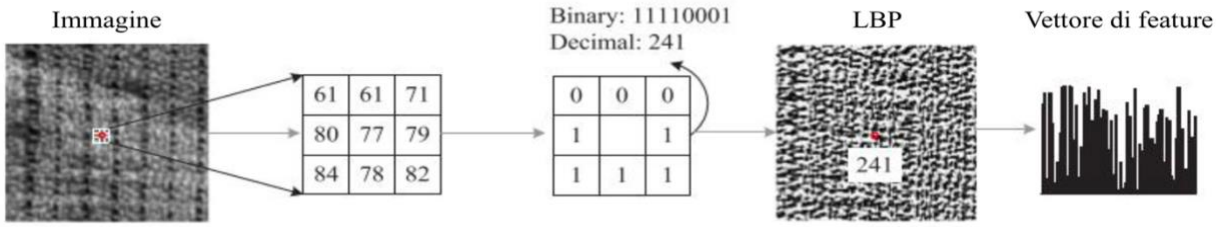


Figura 2.

Successivamente abbiamo fatto l'istogramma dell'immagine LBP, ottenendo, in questo modo, il vettore di feature dell'immagine.

Per ogni soggetto abbiamo eseguito questa operazione per tutte le 54 immagini del training set e abbiamo concatenato i vari istogrammi per ottenere un unico vettore di feature relativo al soggetto.

Poiché gli LBP eseguono un'operazione di soglia considerando il valore del pixel centrale, tendono ad essere sensibili al rumore, specialmente nelle regioni dell'immagine quasi uniformi. Dato che molte regioni facciali sono relativamente uniformi, abbiamo introdotto il Local Ternary Pattern per migliorare la robustezza dei descrittori in queste aree.

L'operatore LTP estrae un codice ternario, ovvero i livelli di grigio nella regione [-t, +t] vengono quantizzati a zero, quelli superiori a t si quantizzano con +1 e quelli inferiori con -1. In LTP la funzione u(x) precedente è sostituita con u'(x):

$$u'(x) = \begin{cases} 1, x >= t \\ 0, |x| < t \\ -1, x < -t \end{cases}$$

nella quale t è una soglia specificata dall'utente.
Il processo di codifica LTP è illustrato in figura 3a, nella quale abbiamo considerato t=5, quindi l'intervallo di tolleranza è [49,59]. Lo schema di codifica considerato prevede di dividere ogni pattern ternario nelle sue parti positive e negative come illustrato in figura 3b. Successivamente abbiamo calcolato gli istogrammi delle immagini LTP upper e LTP lower e li abbiamo concatenati ottenendo un unico vettore di feature.
Sia per l'estrazione del vettore di feature mediante LBP che mediante LTP abbiamo seguito due approcci:

· nel primo caso abbiamo estratto il vettore da tutta l'immagine come rappresentato in figura 4a;

· nel secondo caso abbiamo suddiviso l'immagine in blocchi disgiunti in modo tale da ottenere una rappresentazione del volto più efficace, poiché si conservano anche le informazioni spaziali. Abbiamo calcolato l'istogramma per ogni blocco e successivamente li abbiamo concatenati per ottenere un unico vettore di feature che ci fornisce una descrizione globale del viso. In figura 4b è rappresentato graficamente il processo descritto.

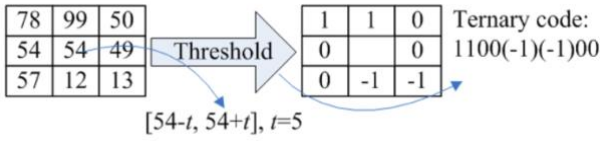


Figura 3a.

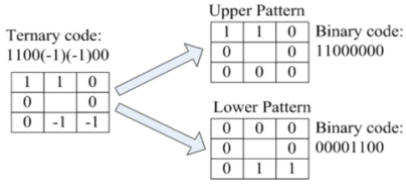


Figura 3b.

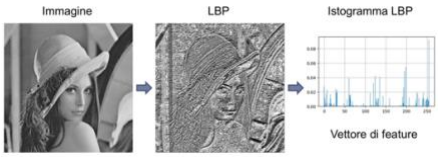


Figura 4a.

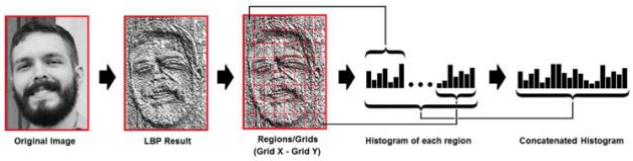


Figura 4b.

2.3 Riconoscimento

Per il riconoscimento abbiamo utilizzato un approccio nearest neighbor come suggerito in [4], in particolare abbiamo utilizzato la distanza euclidea

$$\chi(S, M) = \sqrt{\sum_i (Si - Mi)^2}$$

Nella quale S e M sono due vettori di feature, il primo è del soggetto memorizzato in fase di training e il secondo è dell'immagine da testare. Essa calcola la differenza tra ogni elemento dei due vettori di feature in questione. Questa operazione viene ripetuta confrontando il vettore dell'immagine di test con i vettori di feature di ognuno dei 10 soggetti considerati e associa l'immagine test al soggetto che minimizza la distanza χ.
Utilizzando la funzione Matlab fitcknn abbiamo ottenuto il modello di classificazione formato dai vettori di feature dei 10 soggetti, ognuno dei quali calcolato concatenando gli istogrammi delle 54 immagini per ogni soggetto. Abbiamo eseguito questa operazione utilizzando gli stessi quattro approcci analizzati in precedenza, LPB e LTP su tutta l'immagine, LBP e LTP sull'immagine suddivisa a blocchi.

3. Dataset

Il dataset Extended Yale Face Database B [3] contiene i volti di 38 soggetti con 9 diverse condizioni di posa e 64 illuminazioni diverse. Le immagini sono divise in 5 sottoinsiemi in base all'angolo formato tra la direzione della sorgente di luce e l'asse centrale della fotocamera (12°, 25°, 50°, 77°, 90°). Abbiamo considerato solo i primi due sottoinsiemi nei nostri esperimenti e solo 10 dei 38 soggetti. Abbiamo ottenuto, in questo modo, un dataset di 640 immagini, ovvero 64 immagini per ogni soggetto. Queste le abbiamo suddivise in due parti, 54 immagini per ogni soggetto sono state prelevate per formare il training set, ovvero l'insieme di immagini utilizzate nella fase di training, e la restante porzione l'abbiamo inclusa nel test set, ovvero l'insieme di immagini utilizzate per la fase di testing.

Abbiamo scaricato il dataset delle immagini cropped, il quale contiene una versione ritagliata delle immagini originali, in modo tale da includere solo il viso e ridurre al minimo la presenza di capelli e dello sfondo. Ogni immagine di posa frontale è allineata (ruotata e ridimensionata) in maniera tale che gli occhi giacciono in una posizione fissa rispetto alla finestra dell'immagine. Inoltre, il viso è centrato lungo una direzione verticale in maniera tale che le due linee orizzontali immaginarie che passano attraverso gli occhi e attraverso la bocca siano equidistanti dal centro della finestra ritagliata. Questo allineamento è stato eseguito per rimuovere eventuali distorsioni dai risultati del riconoscimento dovuti all'associazione di una particolare scala, posizione o orientamento per un viso particolare.

4. Valutazione delle prestazioni

Per la valutazione delle prestazioni abbiamo preferito confrontare i tempi di esecuzione per immagine e il tasso di successo degli algoritmi LBP e LTP, applicati sull'intera immagine o su una suddivisione della stessa in blocchi disgiunti di dimensioni: 2x2, 3x3 e 4x4.
In figura 5 facciamo riferimento alle tabelle dei tempi di esecuzione.

Figura 5.

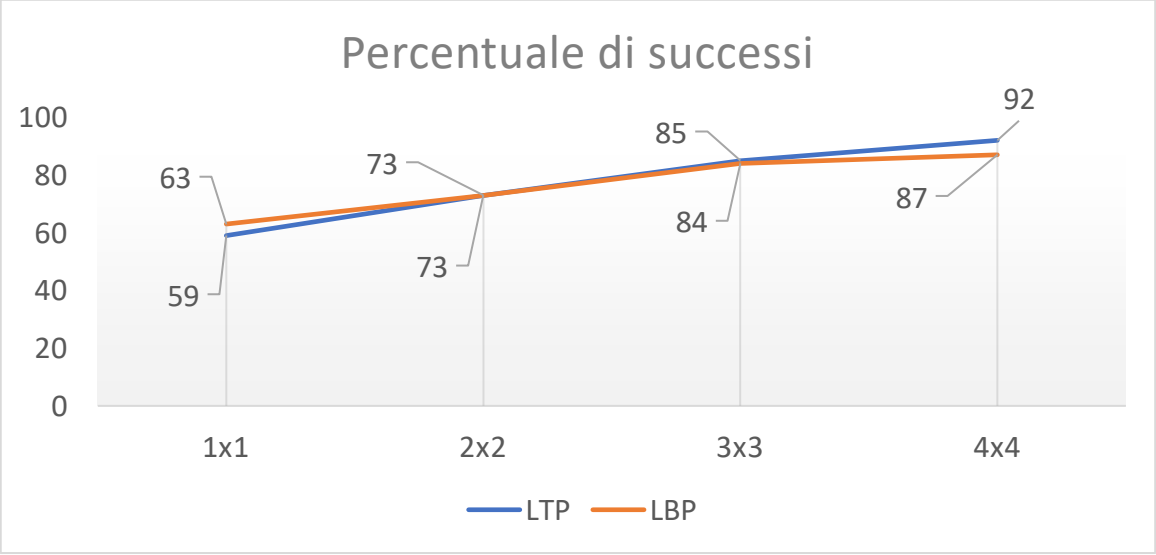
LTP	Enhancement	Feature extraction	Training	Testing
1x1	8.1	9	0.62	29.4
2x2	8.1	9	0.62	29.4
3x3	8.1	10	0.62	29.4
4x4	8.1	10	0.62	29.4

LBP	Enhancement	Feature extraction	Training	Testing
1x1	8.1	4.5	0.34	17.4
2x2	8.1	4.5	0.34	17.4
3x3	8.1	4.8	0.34	17.4
4x4	8.1	5.3	0.34	17.4

- **Nota bene**: tutti i tempi espressi nelle tabelle riguardano quelli di esecuzione di ogni fase, calcolati per ogni immagine. L'unità di misura adoperata è il millisecondo (ms).

In figura 6 valutiamo i grafici del success ratio. Dalle tabelle dei tempi notiamo che, definito un algoritmo, i tempi di esecuzione sono più lunghi all'aumentare della divisione in blocchi dell'immagine, questo perché si devono più volte iterare le operazioni di estrazione di feature, ed il vettore risultante è più lungo, in proporzione al numero di blocchi utilizzati.
La divisione in blocchi non costituisce un problema nonostante l'aumento dei tempi di esecuzione, poiché come si evince dal grafo della percentuale di successo più divisioni garantiscono esiti migliori, in quanto andremo a valutare in maniera specifica caratteristiche sempre più localizzate di un determinato soggetto in esame, garantendo in fase di testing un matching più spinto delle feature del soggetto.

Figura 6.



- **Nota bene:** il grafo esprime la percentuale di successo in base all'algoritmo usato, sull'asse verticale troviamo le percentuali di successo che vanno da zero a cento, sull'asse orizzontale troviamo il fattore che indica in quanti blocchi è stata divisa l'immagine.

Riferimenti bibliografici

[1] X. Tan and B. Triggs, “Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions,”International Workshop on Analysis and Modeling of Faces and Gestures, pp. 168–182, 2007.

[2] T. Ojala, M. Pietika¨inen and T. Ma¨enpa¨a¨, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002.

[3] A. Georghiades, P. Belhumeur and D. Kriegman, “From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 6, pp. 643–660, 2001.

[4] T. Ahonen, A. Hadid and M. Pietika¨inen, “Face recognition with local binary patterns,” European Conference on Computer Vision, pp. 469–481, 2004.